SPPH 604 001 Lab Exercise: Prediction with machine learning

14 October, 2024

Contents

Problem Statement

We will revisit the article by Flegal et al. (2016). We will use the same dataset as in the previous lab exercise on survey data analysis, with some additional predictors in predicting obesity.

Our primary aim is to predict **grade 3 obesity** with the following predictors:

- Age: Age in years at screening
- Gender
- Race: Race/ethnicity
- Education: Education level
- Smoking: Smoking status
- Physical activity: Level of vigorous work activity
- Sleep: Hours of sleep
- High blood pressure: Ever doctor told a high blood pressure
- General health: General health condition

Question 1: Creating data

1(a) Downloading the datasets

You can see how datasets are downloaded and merged:

```
library(nhanesA)
library(SASxport)
library(plyr)

# Demographic data
demo <- nhanes('DEMO_H') # Both males and females: 0 - 150 YEARS
demo1 <- demo[c("SEQN", # Respondent sequence number
                "RIDAGEYR", # Age in years at screening
                "RIAGENDR", # gender
                "DMDEDUC2", # Education level - Adults 20+
                "RIDRETH3", # Race/Hispanic origin w/ NH Asian
                "RIDEXPRG", # Pregnancy status at exam
                "WTINT2YR", #  Full sample 2 year weights
                "SDMVPSU", # Masked variance pseudo-PSU
                "SDMVSTRA")] # Masked variance pseudo-stratum
demo_vars <- names(demo1)
demo2 <- nhanesTranslate('DEMO_H', demo_vars, data = demo1)

# BMI
bmx <- nhanes('BMX_H')
bmx1 <- bmx[c("SEQN", # Respondent sequence number
              "BMXBMI")] # Body Mass Index (kg/m**2): 2 YEARS - 150 YEARS
bmx_vars <- names(bmx1)
bmx2 <- nhanesTranslate('BMX_H', bmx_vars, data = bmx1)

# Smoking
smq <- nhanes('SMQ_H')
smq1 <- smq[c("SEQN", # Respondent sequence number
              "SMQ020", # Smoked at least 100 cigarettes in life
              "SMQ040")] # Do you now smoke cigarettes?: 18 YEARS - 150 YEARS
smq_vars <- names(smq1)
smq2 <- nhanesTranslate('SMQ_H', smq_vars, data = smq1)

# Physical activity
paq <- nhanes('PAQ_H')
paq1 <- paq[c("SEQN", # Respondent sequence number
```

```
              "PAQ605")] # Vigorous work activity
paq_vars <- names(paq1)
paq2 <- nhanesTranslate('PAQ_H', paq_vars, data = paq1)

# Sleep
slq <- nhanes('SLQ_H')
slq1 <- slq[c("SEQN", # Respondent sequence number
              "SLD010H")] # Hours of sleep
slq_vars <- names(slq1)
slq2 <- nhanesTranslate('SLQ_H', slq_vars, data = slq1)

# High blood pressure
bpq <- nhanes('BPQ_H')
bpq1 <- bpq[c("SEQN", # Respondent sequence number
              "BPQ020")] # Ever told you had high blood pressure
bpq_vars <- names(bpq1)
bpq2 <- nhanesTranslate('BPQ_H', bpq_vars, data = bpq1)

# General health condition
huq <- nhanes('HUQ_H')
huq1 <- huq[c("SEQN", # Respondent sequence number
              "HUQ010")] # General health condition
huq_vars <- names(huq1)
huq2 <- nhanesTranslate('HUQ_H', huq_vars, data = huq1)

# Combined data
dat.full <- join_all(list(demo2, bmx2, smq2, paq2, slq2, bpq2, huq2), by = "SEQN",
                     type='full')
dim(dat.full) # N = 10,175
```

1(b) Recoding

Let us recode the outcome and predictors to make them suitable for analysis:

```
# Survey design
dat.full$survey.weight <- dat.full$WTINT2YR
dat.full$psu <- dat.full$SDMVPSU
dat.full$strata <- dat.full$SDMVSTRA

# Class 3 obesity - BMI >= 40 kg/m^2
summary(dat.full$BMXBMI)
dat.full$obesity <- ifelse(dat.full$BMXBMI >= 40, 1, 0)
table(dat.full$obesity, useNA = "always")

# Age
dat.full$age.cat <- cut(dat.full$RIDAGEYR, c(20, 40, 60, Inf), right = FALSE)
table(dat.full$age.cat, useNA = "always")

# Gender
dat.full$gender <- dat.full$RIAGENDR
table(dat.full$gender, useNA = "always")
```

```r
# Race/Hispanic origin group
dat.full$race <- dat.full$RIDRETH3
table(dat.full$age.cat, dat.full$race, useNA = "always")
dat.full$race <- car::recode(dat.full$race,
                             " 'Non-Hispanic White'='White';
                             'Non-Hispanic Black' = 'Black';
                             c('Mexican American','Other Hispanic')='Hispanic';
                             c('Non-Hispanic Asian',
                             'Other Race - Including Multi-Rac')= 'Other';
                             else=NA",
                             levels = c("White", "Black", "Hispanic", "Other"))
table(dat.full$race, useNA = "always")

# Education
dat.full$education <- dat.full$DMDEDUC2
dat.full$education <- car::recode(dat.full$education,
                                  " c('Some college or AA degree',
                                  'College graduate or above') = '>High school';
                                  'High school graduate/GED or equi' = 'High school';
                                  c('Less than 9th grade',
                                  '9-11th grade (Includes 12th grad') =
                                  '<High school';
                                  else = NA",
                                  levels = c("<High school", "High school",
                                             ">High school"))
table(dat.full$education, useNA = "always")

# Smoking status
dat.full$smoking <- dat.full$SMQ020
table(dat.full$smoking, useNA = "always")
dat.full$smoking <- car::recode(dat.full$smoking, " 'Yes'='Current smoker';
                                'No'='Never smoker'; else=NA  ",
                                levels = c("Never smoker", "Former smoker",
                                           "Current smoker"))
dat.full$smoking[dat.full$SMQ040 == "Not at all"] <- "Former smoker"
table(dat.full$smoking, useNA = "always")

# Physical activity
dat.full$physical.activity <- dat.full$PAQ605
table(dat.full$physical.activity, useNA = "always")
dat.full$physical.activity <- car::recode(dat.full$physical.activity,
                                          " 'Yes'='Yes'; 'No'='No'; else=NA ",
                                          levels = c("No", "Yes"))
table(dat.full$physical.activity, useNA = "always")

# Sleep
dat.full$sleep <- dat.full$SLD010H
dat.full$sleep <- car::recode(dat.full$sleep, " 1:6 = 'Less than 7'; 7:9 = '7-9';
                              10:24 = 'More than 9';  else=NA ",
                              levels = c("Less than 7", "7-9", "More than 9"))
table(dat.full$sleep, useNA = "always")

# High blood pressure
```

```r
dat.full$high.blood.pressure <- dat.full$BPQ020
table(dat.full$high.blood.pressure, useNA = "always")
dat.full$high.blood.pressure <- car::recode(dat.full$high.blood.pressure,
                                            " 'Yes'='Yes'; 'No'='No'; else=NA ",
                                            levels = c("No", "Yes"))
table(dat.full$high.blood.pressure, useNA = "always")

# General health condition
dat.full$general.health <- dat.full$HUQ010
table(dat.full$general.health, useNA = "always")
dat.full$general.health <- car::recode(dat.full$general.health,
                                       "c('Excellent,', 'Very good,')=
                                       'Very good or Excellent';
                                       'Good,'='Good';
                                       c('Fair, or', 'Poor?') ='Poor or Fair';
                                       else=NA  ",
                                       levels = c("Poor or Fair", "Good",
                                                  "Very good or Excellent"))
table(dat.full$general.health, useNA = "always")
```

1(c) Keep relevant variables

Let's keep only the relevant variables for this exercise:

```r
# Keep relevant variables
vars <- c(
  # Unique identifier
  "SEQN",

  # Survey features
  "survey.weight", "psu", "strata",

  # Eligibility
  "RIDAGEYR", "BMXBMI", "RIDEXPRG",

  # Outcome
  "obesity",

  # Predictors
  "age.cat", "gender", "race", "education", "smoking", "physical.activity",
  "sleep", "high.blood.pressure", "general.health")

dat.full2 <- dat.full[,vars]
```

1(d) Weight normalization

Large weights can cause issues when evaluating model performance. Let's normalize the survey weights to address this problem:

```r
dat.full2$wgt <- dat.full2$survey.weight * nrow(dat.full2)/sum(dat.full2$survey.weight)
summary(dat.full2$wgt)
```

1(e) Analytic dataset

The authors restricted their study to - adults aged 20 years and more, - non-missing body mass index, and - non-pregnant

```
# Aged 20 years or more
dat.analytic <- subset(dat.full2, RIDAGEYR>=20) # N = 5,769

# Non-missing outcome
dat.analytic <- subset(dat.analytic, !is.na(BMXBMI)) # N = 5,520

# Non-pregnant
table(dat.analytic$RIDEXPRG)
dat.analytic <- subset(dat.analytic, is.na(RIDEXPRG) | RIDEXPRG !=
                         "Yes, positive lab pregnancy test") # N = 5,455
nrow(dat.analytic)

# Drop irrelevant variables
dat.analytic$RIDAGEYR <- dat.analytic$BMXBMI <- dat.analytic$RIDEXPRG <- NULL
```

1(f) Complete case data

Below is the code for creating the complete case dataset (no missing for the outcome or predictors):

```
# Drop missing values
dat.complete <- dat.analytic[complete.cases(dat.analytic),] # N = 5,433
```

1(g) Save daatsets

```
save(dat.full, dat.full2, dat.analytic, dat.complete, file = "Data/machinelearning/Flegal2016_v2.RData")
```

Question 2: Importing data and creating Table 1

2(a) Importing dataset

Let's load the dataset:

```
load("Data/machinelearning/Flegal2016_v2.RData")
ls()
```

```
## [1] "dat.analytic" "dat.complete" "dat.full"     "dat.full2"
```

Here,

- dat.full: the full dataset with all variables
- dat.full2: the full dataset with only relevant variables for this exercise
- dat.analytic: the analytic dataset with only adults aged 20 years and more, non-missing BMI, and non-pregnant

- dat.complete: the complete case dataset without missing values in the outcome and predictors

```
names(dat.full2)
```

```
##  [1] "SEQN"                "survey.weight"       "psu"
##  [4] "strata"              "RIDAGEYR"            "BMXBMI"
##  [7] "RIDEXPRG"            "obesity"             "age.cat"
## [10] "gender"              "race"                "education"
## [13] "smoking"             "physical.activity"   "sleep"
## [16] "high.blood.pressure" "general.health"      "wgt"
```

2(b) Creating Table 1

Let's create Table 1 for the complete case dataset with unweighted frequencies:

```
library(tableone)
predictors <- c("age.cat", "gender", "race", "education", "smoking",
                "physical.activity", "sleep", "high.blood.pressure",
                "general.health")
tab1 <- CreateTableOne(vars = predictors, strata = "obesity",
                       data = dat.complete, test = F, addOverall = T)
print(tab1, format="f") # Showing only frequencies
```

```
##                            Stratified by obesity
##                             Overall 0    1
##   n                         5433    5023 410
##   age.cat
##      [20,40)                1806    1659 147
##      [40,60)                1892    1728 164
##      [60,Inf)               1735    1636  99
##   gender = Female           2807    2531 276
##   race
##      White                  2333    2157 176
##      Black                  1109     976 133
##      Hispanic               1208    1125  83
##      Other                   783     765  18
##   education
##      <High school           1171    1092  79
##      High school            1216    1116 100
##      >High school           3046    2815 231
##   smoking
##      Never smoker           3054    2828 226
##      Former smoker          1261    1159 102
##      Current smoker         1118    1036  82
##   physical.activity = Yes    984     917  67
##   sleep
##      7-9                    3174    2971 203
##      Less than 7            2084    1896 188
##      More than 9             175     156  19
##   high.blood.pressure = Yes 2037    1810 227
##   general.health
##      Poor or Fair           1260    1084 176
```

```
##      Good                   2065    1911 154
##      Very good or Excellent 2108    2028  80
```

Question 3: Prediction using split sample approach

In this exercise, we will use the split-sample approach to predict obesity. We will create our training and test data using a 60-40 split for the training and test data. We will use the following two methods to predict obesity:

- Design-adjusted logistic with all survey features (psu, strata, and survey weights)
- LASSO with survey weights

3(a) Split the data into training and test

Let us create our training and test data using the split-sample approach:

```
set.seed(900)
dat.complete$datasplit <- rbinom(nrow(dat.complete), size = 1, prob = 0.6)
table(dat.complete$datasplit)
```

```
##
##    0    1
## 2130 3303
```

```
# Training data
dat.train <- dat.complete[dat.complete$datasplit == 1,]
dim(dat.train)
```

```
## [1] 3303   16
```

```
# Test data
dat.test <- dat.complete[dat.complete$datasplit == 0,]
dim(dat.test)
```

```
## [1] 2130   16
```

3(b) Prediction with design-adjusted logistic

We will use the design-adjusted logistic regression to predict obesity with the following predictors:

- age.cat, gender, race, high.blood.pressure, general.health

Instructions:

- 1: Create the survey design on the full data and subset the design for those individuals in the training data.
- 2: Use the **training data design** created in step 1 to fit the model
- 3: Use the test data to predict the probability of obesity.

- 4: Calculate AUC on the test data.
- 5: Calculate calibration slope with 95% confidence interval on the test data.

Hints:

- `WeightedAUC` and `WeightedROC` are helpful functions in calculating AUC.
- The `Logit` function from the `DescTools` package is helpful in calculating the logit of predicted probabilities for calculating calibration slope.
- Use the **normalized weight** variable to calculate the AUC and calibration slope.

Simpler Model

```
library(survey)
library(DescTools)
library(WeightedROC)
library(Publish)
library(boot)
library(scoring)

# Design
dat.full2$miss <- 1
dat.full2$miss[dat.full2$SEQN %in% dat.train$SEQN] <- 0
svy.design0 <- svydesign(strata = ~strata, id = ~psu, weights = ~survey.weight,
                         data = dat.full2, nest = TRUE)
svy.design <- subset(svy.design0, miss == 0)

# Formula
predictors <- c("age.cat", "gender", "race", "education",
                "high.blood.pressure", "general.health")
Formula <- formula(paste("obesity ~ ", paste(predictors, collapse=" + ")))

# Model
fit.glm <- svyglm(Formula, design = svy.design, family = binomial)
publish(fit.glm)
```

```
##              Variable                   Units OddsRatio       CI.95    p-value
##               age.cat                [20,40)       Ref
##                                      [40,60)      0.63 [0.42;0.95]   0.091799
##                                     [60,Inf)      0.31 [0.17;0.57]   0.018946
##                gender                    Male       Ref
##                                        Female      1.90 [1.32;2.73]   0.026347
##                  race                   White       Ref
##                                         Black      1.56 [1.12;2.16]   0.056561
##                                      Hispanic      0.91 [0.53;1.57]   0.758727
##                                         Other      0.37 [0.18;0.75]   0.050646
##             education           <High school       Ref
##                                  High school      1.34 [0.87;2.07]   0.252853
##                                 >High school      1.97 [1.13;3.46]   0.076212
##  high.blood.pressure                      No       Ref
##                                          Yes      2.20 [1.61;3.02]   0.007835
##       general.health           Poor or Fair       Ref
```

```
##                                       Good      0.52 [0.36;0.74]   0.022344
##                       Very good or Excellent      0.22 [0.14;0.36]   0.003885
```

```r
# Prediction on the test set
dat.test$pred.glm <- predict(fit.glm, newdata = dat.test, type = "response")
summary(dat.test$pred.glm)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.003889 0.027737 0.052118 0.073342 0.094836 0.413143
```

```r
# AUC on the test set with sampling weights
# unfortunately strata and cluster omitted
auc.glm <- WeightedAUC(WeightedROC(dat.test$pred.glm, dat.test$obesity,
                                   weight = dat.test$wgt))
auc.glm
```

```
## [1] 0.6903502
```

```r
# Function to calculate AUC for bootstrap samples
# unfortunately strata and cluster omitted
calc_auc <- function(data, indices) {
  d <- data[indices, ]
  roc_obj <- WeightedROC(d$pred.glm, d$obesity, weight = d$wgt)
  return(WeightedAUC(roc_obj))
}

# Perform bootstrapping
set.seed(123)
boot_obj <- boot(data = dat.test, statistic = calc_auc, R = 150)

# Get 95% confidence intervals
ci_auc <- boot.ci(boot_obj, type = "perc")
ci_auc
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 150 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_obj, type = "perc")
##
## Intervals :
## Level     Percentile
## 95%   ( 0.6357,  0.7582 )
## Calculations and Intervals on Original Scale
## Some percentile intervals may be unstable
```

```r
# Weighted calibration slope
# unfortunately strata and cluster omitted
dat.test$pred.glm.logit <- DescTools::Logit(dat.test$pred.glm)
slope.glm <- glm(obesity ~ pred.glm.logit, data = dat.test, family = quasibinomial,
                 weights = wgt)
publish(slope.glm)
```

```
##        Variable Units Coefficient       CI.95 p-value
##  pred.glm.logit              0.79 [0.61;0.98] < 1e-04
```

```r
# Calculate the weighted Brier score
brier_score <- mean(brierscore(dat.test$obesity ~ dat.test$pred.glm,
                               data = dat.test,
                               wt = dat.test$wgt))
brier_score
```

```
## [1] 0.06878706
```

3(c) Prediction with design-adjusted logistic with added covariates [90% grade]

Add the following variables in the existing model, and assess the model performance in terms of the same 3 performance measures:

- education, smoking, physical.activity, sleep

Model with more variables

```r
# Formula
# predictors2 <- ...
# Formula2 <- ...

# Model
# fit.glm2 <- ...

# Prediction on the test set

# AUC on the test set with sampling weights
# unfortunately strata and cluster omitted
# auc.glm2 <-...

# Function to calculate AUC for bootstrap samples
# unfortunately strata and cluster omitted

# Get 95% confidence intervals from bootstrapping
# ci_auc2 <- ...

# Weighted calibration slope
# unfortunately strata and cluster omitted
# slope.glm2 ...

# Calculate the weighted Brier score
# brier_score2 <- ...
```

3(d) Interpretation [10%]

Interpret the AUC, calibration slope and Brier Score based on the following criteria, and suggest which model you would choose:

AUC	Interpretation
0.50	No better than a random chance
0.51-0.70	Poor discrimination ability
0.71-0.80	Acceptable discrimination ability
0.81-0.90	Excellent discrimination ability
0.90-1.00	Outstanding discrimination ability

Calibration slope	Interpretation
1 and 95% CI includes 1	Well-calibration
Significantly less than 1	Overfitting
Significantly greater than 1	Underfitting

Brier Score	Interpretation
0	Perfect prediction
0.01-0.1	Very good model performance
0.11-0.2	Good model performance
0.21-0.3	Fair model performance
0.31-0.5	Poor model performance
> 0.5	Very poor model performance (no skill)

- AUC: Higher is better (closer to 1).
- Calibration Slope: Closer to 1 is better.
- Brier Score: Lower is better (closer to 0).

Response:

3(e) Prediction with LASSO [Optional]

Now we will use the LASSO method to predict obesity. We will incorporate sampling weights in the model to account for survey data (no psu or strata). Note that we are not interested in the statistical significance of the beta coefficients. Hence, not utilizing psu and strata should not be an issue in this prediction problem.

Instructions:

- 1: Use the training data with normalized weight to fit the model.
- 2: Find the optimum lambda using 5-fold cross-validation. Consider the lambda value that gives the minimum prediction error.
- 3: Predict the probability of obesity on the test set
- 3: Calculate AUC on the test data.
- 4: Calculate calibration slope with 95% confidence interval on the test data.

Model

```
library(glmnet)
library(DescTools)
library(WeightedROC)
```

```r
# Training data - X: predictor, y: outcome
X.train <- model.matrix(Formula2, dat.train)[,-1]
y.train <- as.matrix(dat.train$obesity)

# Test data - X: predictor, y: outcome
X.test <- model.matrix(Formula2, dat.test)[,-1]
y.test <- as.matrix(dat.test$obesity)

# Find the best lambda using 5-fold CV
fit.cv.lasso <- cv.glmnet(x = X.train, y = y.train, nfolds = 5, alpha = 1,
                          family = "binomial", weights = dat.train$wgt)

# Prediction on the test set
dat.test$pred.lasso <- predict(fit.cv.lasso, newx = X.test, type = "response",
                               s = fit.cv.lasso$lambda.min)

# AUC on the test set with sampling weights
auc.lasso <- WeightedAUC(WeightedROC(dat.test$pred.lasso, dat.test$obesity,
                                     weight = dat.test$wgt))
auc.lasso

# Weighted calibration slope
dat.test$pred.lasso.logit <- DescTools::Logit(dat.test$pred.lasso)
slope.lasso <- glm(obesity ~ pred.lasso.logit, data = dat.test,
                   family = quasibinomial, weights = wgt)
publish(slope.lasso)
```

Interpretation [optional]

Interpret the AUC and calibration slope.

Question 4: Prediction using croos-validation approach [optional]

Use LASSO with 5-fold cross-validation to predict obesity with the same set of predictors (from larger model) used in Question 2. Report the average AUC and average calibration slope with 95% confidence interval over 5 folds.

```r
library(glmnet)
library(DescTools)
library(WeightedROC)

k <- 5
set.seed(604)
nfolds <- sample(1:k, size = nrow(dat.complete), replace = T)
table(nfolds)

auc.lasso <- cal.slope.lasso <- cal.slope.se.lasso <- NULL
for (fold in 1:k) {
  # Training data
  dat.train <- dat.complete[nfolds != fold, ]
```

```
  X.train <- model.matrix(Formula2, dat.train)[,-1]
  y.train <- as.matrix(dat.train$obesity)

  # Test data
  dat.test <- dat.complete[nfolds == fold, ]
  X.test <- model.matrix(Formula2, dat.test)[,-1]
  y.test <- as.matrix(dat.test$obesity)

  # Find the optimum lambda using 5-fold CV
  fit.cv.lasso <- cv.glmnet(x = X.train, y = y.train, nfolds = 5, alpha = 1,
                            family = "binomial", weights = dat.train$wgt)

  # Prediction on the test set
  dat.test$pred.lasso <- predict(fit.cv.lasso, newx = X.test, type = "response",
                                 s = fit.cv.lasso$lambda.min)

  # AUC on the test set with sampling weights
  auc.lasso[fold] <- WeightedAUC(WeightedROC(dat.test$pred.lasso,dat.test$obesity,
                                             weight = dat.test$wgt))

  # Weighted calibration slope
  dat.test$pred.lasso.logit <- DescTools::Logit(dat.test$pred.lasso)
  mod.cal <- glm(obesity ~ pred.lasso.logit, data = dat.test, family = binomial,
                 weights = wgt)
  cal.slope.lasso[fold] <- summary(mod.cal)$coef[2,1]
  cal.slope.se.lasso[fold] <- summary(mod.cal)$coef[2,2]
}

# Average AUC
mean(auc.lasso)

# Average calibration slope
mean(cal.slope.lasso)

# 95% CI for calibration slope
cbind(mean(cal.slope.lasso) - 1.96 * mean(cal.slope.se.lasso),
      mean(cal.slope.lasso) + 1.96 * mean(cal.slope.se.lasso))
```

Knit your file

Please knit your file once you finished and submit the knitted PDF or doc file. Please also fill-up the following table:

Group name: **Put the group name here**

Student initial	% contribution
Put Student 1 initial here	Put % contribution for Student 1 here
Put Student 2 initial here	Put % contribution for Student 2 here
Put Student 3 initial here	Put % contribution for Student 3 here