Chapter 5 Resampling Methods

Jishen Yin

2020/5/6

```
knitr::opts_chunk$set(echo = TRUE)
library(ISLR)
library(MASS)
library(class)
library(tidyverse)
library(GGally)
library(gridExtra)
library(grid)
```

Problem 5

`Default` data set

```
data(Default)
```

(a) Fit a logistic regression model that uses `income` and `balance` to predict `default`.

(b) Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps:

i. Split the sample set into a training set and a validation set.

ii. Fit a multiple logistic regression model using only the training observations.

iii. Obtain a prediction of default status for each individual in the validation set by compuing the posterior probability of default for that individual, and classifying the individual to the `default` category if the posterior probability is greater than 0.5.

iv. Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

```
set.seed(7)
train <- sample(1:nrow(Default), 8000)
Default_tr <- Default[train, ]
Default_val <- Default[-train, ]
```

```
lr <- glm(default ~ balance + income, data = Default_tr, family = binomial)
pred <- ifelse(predict(lr, Default_val, type = "response") > 0.5, "Yes", "No")
mean(pred != Default_val$default)
```

```
## [1] 0.0235
```

(c) Repeat the process in (b) three times, using three different splits.

```
err <- sapply(1:100, function(x){
  train <- sample(1:nrow(Default), 8000)
  Default_tr <- Default[train, ]
  Default_val <- Default[-train, ]
  lr <- glm(default ~ balance + income, data = Default_tr, family = binomial)
  pred <- ifelse(predict(lr, Default_val, type = "response") > 0.5, "Yes", "No")
  return(mean(pred != Default_val$default))
})
summary(err)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.01800 0.02400 0.02550 0.02565 0.02800 0.03300
```

(d) Now consider a logistic regression model that predicts the probability of `default` using `income`, `balance`, and a dummy variable for `student`. Estimate the test error for this model using the validation set approach.

```
err2 <- sapply(1:100, function(x){
  train <- sample(1:nrow(Default), 8000)
  Default_tr <- Default[train, ]
  Default_val <- Default[-train, ]
  lr <- glm(default ~ ., data = Default_tr, family = binomial)
  pred <- ifelse(predict(lr, Default_val, type = "response") > 0.5, "Yes", "No")
  return(mean(pred != Default_val$default))
})
summary(err2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.01900 0.02500 0.02700 0.02707 0.02950 0.03400
```

Problem 6

In the previous problem, use a logistic regression model to predict the probability of `default` using `income` and `balance` on the `Default` data set.

(a) Compute estimates for the standard errors of coefficients using the `summary()` and `glm()` functions.

```
lr <- glm(default ~ income + balance, data = Default, family = binomial)
summary(lr)$coefficients
```

```
##                   Estimate   Std. Error    z value      Pr(>|z|)
## (Intercept) -1.154047e+01 4.347564e-01 -26.544680 2.958355e-155
## income       2.080898e-05 4.985167e-06   4.174178  2.990638e-05
## balance      5.647103e-03 2.273731e-04  24.836280 3.638120e-136
```

(b) Using bootstrap to contain the standard errors

```
boot.fn <- function(data, idx){
  lr <- glm(default ~ income + balance, data = data[idx, ], family = binomial)
  return(lr$coefficients[2:3])
}

bootstrap <- lapply(1:1000, function(x){
  idx <- sample(1:10000, 10000, replace = TRUE)
  return(boot.fn(Default, idx))
})
```

```
coefs <- do.call(rbind, bootstrap)
sd(coefs[,1])
```

```
## [1] 4.729736e-06
```

```
sd(coefs[,2])
```

```
## [1] 0.0002287495
```

Problem 7

Use the `WWeekly` data set. Fit a logistic regression mdel that predicts `Direction` using `Lag1` and `Lag2`. Compute the LOOCV error.

```
data(Weekly)
```

```
loocv_pred <- sapply(1:nrow(Weekly), function(i){
  lr <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-i,], family = binomial)
  return(predict(lr, Weekly[i,], type = "response"))
})
```

```
loocv <- ifelse(loocv_pred > 0.5, "Up", "Down")
mean(loocv == Weekly$Direction)
```

```
## [1] 0.5500459
```

Problem 9

Use the `Boston` housing data set, from the `MASS` library.

(a) Based on this data set, provide an estimate for the population mean of `medv`. Call this estimate $\hat{\mu}$.

```
data(Boston)
mean(Boston$medv)
```

```
## [1] 22.53281
```

(b) Provide an estimate of standard error of $\hat{\mu}$ by definition.

```
sqrt(var(Boston$medv)/nrow(Boston))
```

```
## [1] 0.4088611
```

(c) Now estimate the standard error of $\hat{\mu}$ using bootstrap.

```
mu <- sapply(1:10000, function(x){
  idx <- sample(1:nrow(Boston), nrow(Boston), replace = TRUE)
  tmp <- Boston[idx,]
  return(mean(tmp$medv))
})

sqrt(var(mu))
```

```
## [1] 0.4058924
```

(d) Provide a 95% confidence interval for the mean of `medv` based on the bootstrap estimate from (c). Compare it to the result obtained using `t.test(Boston$medv)`.

```
c(mean(Boston$medv)-qnorm(0.975)*sqrt(var(mu)), mean(Boston$medv)+qnorm(0.975)*sqrt(var(mu)))
```

```
## [1] 21.73727 23.32834
```

```
t.test(Boston$medv)
```

```
##
##  One Sample t-test
##
## data:  Boston$medv
## t = 55.111, df = 505, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  21.72953 23.33608
## sample estimates:
## mean of x
##  22.53281
```

(e) Provide an estimate, $\hat{\mu_{med}}$ for the population median.

```
median(Boston$medv)
```

```
## [1] 21.2
```

(f) Estimate the standard error of the median using the bootstrap.

```
med <- sapply(1:10000, function(x){
  idx <- sample(1:nrow(Boston), nrow(Boston), replace = TRUE)
  tmp <- Boston[idx,]
  return(median(tmp$medv))
})

sqrt(var(med))
```

```
## [1] 0.3751242
```

(g) Provide an estimate for the tenth percentile of `medv` in Boston suburbs. Call this quantity $\hat{\mu}_{0,1}$.

```
quantile(Boston$medv, 0.1)
```

```
##   10%
## 12.75
```

(h) Provide an estimate for the standard error of $\hat{\mu}_{0,1}$.

```
quat10 <- sapply(1:10000, function(x){
  idx <- sample(1:nrow(Boston), nrow(Boston), replace = TRUE)
  tmp <- Boston[idx,]
  return(quantile(tmp$medv, 0.1))
})

sqrt(var(quat10))
```

```
## [1] 0.5016506
```