Assignment 4

Tina Roha

3/26/2021

Resampling Methods

Chapter 5 (page 197): Questions 3, 5, 6 and 9

Question-3

3. We now review k-fold cross-validation.

a. Explain how k-fold cross-validation is implemented.

The k-fold cross-validation is implemented by taking a group of observations and dividing them into k-folds at random and of about the same size. The first fold becomes a validation set and the method then focuses on the fit of the other k-folds. Then the computation of the mean squared error occurs for the held-out fold. The process continues to repeat itself k amount of times where a different observation group is utilized as the validation set. Overall, the end result of the process approximates the test error, MSE1,MSE2,…,MSEk. Then by averaging these values we get the estimation of the k-fold cross validation.

b. What are the advantages and disadvantages of k-fold cross-validation relative to:

i. The validation set approach?

The advantages of the k-fold cross-validation relative to the validation set approach includes that the validation estimate of the test error rate may be overly variable. Furthermore, validation set error rate usually causes the occurrence of the overestimation of the test error rate for the overall fit of the model inclusive of the whole data set. On the other hand, the disadvantages of this approach is the simplicity of concepts and that it is implemented with ease.

ii. LOOCV?

By definition, the LOOCV cross-validation approach is a case of k-fold cross-validation in which k=n. An advantage of k-fold cross-validation relative to LOOCV includes the fact that one must fit the statistical learning method n times which may involve an excessive amount of computations. On the other hand, a disadvantage is that LOOCV has less bias when compared to k-fold cross-validation.

Question-5

5. In Chapter 4, we used logistic regression to predict the probability of default using income and balance on the Default data set. We will now estimate the test error of this logistic regression model using the validation set approach. Do not forget to set a random seed before beginning your analysis.

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

```r
library(ISLR)
attach(Default)
set.seed(1)
fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial")
summary(fit.glm)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4725  -0.1444  -0.0574  -0.0211   3.7245
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
## income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
## balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1579.0  on 9997  degrees of freedom
## AIC: 1585
##
## Number of Fisher Scoring iterations: 8
```

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.

```r
train <- sample(dim(Default)[1], dim(Default)[1] / 2)
```

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

```
fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
summary(fit.glm)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3583  -0.1268  -0.0475  -0.0165   3.8116
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.208e+01  6.658e-01 -18.148   <2e-16 ***
## income       1.858e-05  7.573e-06   2.454   0.0141 *
## balance      6.053e-03  3.467e-04  17.457   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1457.0  on 4999  degrees of freedom
## Residual deviance:  734.4  on 4997  degrees of freedom
## AIC: 740.4
##
## Number of Fisher Scoring iterations: 8
```

iii. Obtain a prediction of default status for each individual in the validation set by computing 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.

```
probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
pred.glm <- rep("No", length(probs))
pred.glm[probs > 0.5] <- "Yes"
```

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

```
mean(pred.glm != Default[-train, ]$default)
```

```
## [1] 0.0286
```

The result of this output indicates that there is a 2.86% test error rate with the validation set approach.

c. Repeat the process in (b) three times, using three different splits of the observations into a training set and a validation set. Comment on the results obtained.

```
train <- sample(dim(Default)[1], dim(Default)[1] / 2)
fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
pred.glm <- rep("No", length(probs))
pred.glm[probs > 0.5] <- "Yes"
mean(pred.glm != Default[-train, ]$default)
```

```
## [1] 0.0236
```

```
train <- sample(dim(Default)[1], dim(Default)[1] / 2)
fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
pred.glm <- rep("No", length(probs))
pred.glm[probs > 0.5] <- "Yes"
mean(pred.glm != Default[-train, ]$default)
```

```
## [1] 0.028
```

```
train <- sample(dim(Default)[1], dim(Default)[1] / 2)
fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
pred.glm <- rep("No", length(probs))
pred.glm[probs > 0.5] <- "Yes"
mean(pred.glm != Default[-train, ]$default)
```

```
## [1] 0.0268
```

The results obtained emphasize that the validation estimate of the test error can be variable. Furthermore, this depends on which set the observations are included with which is either the training set or the validation set.

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. Comment on whether or not including a dummy variable for student leads to a reduction in the test error rate.

```
train <- sample(dim(Default)[1], dim(Default)[1] / 2)
fit.glm <- glm(default ~ income + balance + student, data = Default, family = "binomial", subset = train)
pred.glm <- rep("No", length(probs))
probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
pred.glm[probs > 0.5] <- "Yes"
mean(pred.glm != Default[-train, ]$default)
```

```
## [1] 0.0264
```

The produced results show that including a dummy variable for “student” does not lead to a reduction in the test error rate.

Question-6

We continue to consider the use of a logistic regression model to predict the probability of default using income and balance on the Default data set. In particular, we will now compute estimates for the standard errors of the income and balance logistic regression coefficients in two different ways: (1) using the bootstrap, and (2) using the standard formula for computing the standard errors in the glm() function. Do not forget to set a random seed before beginning your analysis.

a. Using the summary() and glm() functions, determine the estimated standard errors for the coefficients associated with income and balance in a multiple logistic regression model that uses both predictors.

```
set.seed(1)
attach(Default)
```

```
## The following objects are masked from Default (pos = 3):
##
##     balance, default, income, student
```

```
fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial")
summary(fit.glm)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4725  -0.1444  -0.0574  -0.0211   3.7245
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
## income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
## balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1579.0  on 9997  degrees of freedom
## AIC: 1585
##
## Number of Fisher Scoring iterations: 8
```

Utilizing the summary() and glm() functions it can be concluded that the the estimated standard errors for the coefficients $Beta_0$, $Beta_1$ and $Beta_2$ are 4.348e-01, 4.985e-06, and 2.274e-04 respectively.

b. Write a function, boot.fn(), that takes as input the Default data set as well as an index of the observations, and that outputs the coefficient estimates for income and balance in the multiple logistic regression model.

```
boot.fn <- function(data, index) {
    fit <- glm(default ~ income + balance, data = data, family = "binomial", subset = index)
    return (coef(fit))
}
```

c. Use the boot() function together with your boot.fn() function to estimate the standard errors of the logistic regression coefficients for income and balance.

```
library(boot)
boot(Default, boot.fn, 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Default, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##          original        bias     std. error
## t1* -1.154047e+01 -8.008379e-03 4.239273e-01
## t2*  2.080898e-05  5.870933e-08 4.582525e-06
## t3*  5.647103e-03  2.299970e-06 2.267955e-04
```

Utilizing the boot() function together with boot.fn() function it can be concluded that the bootstrap estimates of the standard error for the coefficients $Beta_0$, $Beta_1$ and $Beta_2$ are 4.239273e-01, 4.582525e-06 and 2.267955e-04 respectively.

d. Comment on the estimated standard errors obtained using the glm() function and using your bootstrap function.

Utilizing the glm() function and the bootstrap function, it can be concluded that the estimated standard errors obtained are close in value.

Question-9

We will now consider 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 ˆμ.

```
library(MASS)
attach(Boston)
mu.hat <- mean(medv)
mu.hat
```

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

b. Provide an estimate of the standard error of ˆμ. Interpret this result. Hint: We can compute the standard error of the sample mean by dividing the sample standard deviation by the square root of the number of observations.

```
se.hat <- sd(medv) / sqrt(dim(Boston)[1])
se.hat
```

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

c. Now estimate the standard error of ˆμ using the bootstrap. How does this compare to your answer from (b)?

```
set.seed(1)
boot.fn <- function(data, index) {
    mu <- mean(data[index])
    return (mu)
}
boot(medv, boot.fn, 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = medv, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##     original      bias    std. error
## t1* 22.53281 0.008517589   0.4119374
```

Estimating the standard error of ˆμ using bootstrap the value 0.4119 was obtained. When compared to the value of the estimate found in part (b), which had a value of 0.4089, it can be determined that they are both close in value.

d. Based on your bootstrap estimate from (c), provide a 95 % confidence interval for the mean of medv. Compare it to the results obtained using t.test(Boston$medv$). $Hint : You can approximate a 95$medv) are extremely close in value.

e. Based on this data set, provide an estimate, ˆμmed, for the median value of medv in the population.

```
med.hat <- median(medv)
med.hat
```

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

f. We now would like to estimate the standard error of ˆμmed. Unfortunately, there is no simple formula for computing the standard error of the median. Instead, estimate the standard error of the median using the bootstrap. Comment on your findings.

```
boot.fn <- function(data, index) {
    mu <- median(data[index])
    return (mu)
}
boot(medv, boot.fn, 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = medv, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##     original  bias    std. error
## t1*     21.2 -0.0098   0.3874004
```

The findings from the produced results shows an estimated value of 21.2. Similarly, in part (e) an estimated median value of 21.2 was also produced in the results. Additionally, the standard error in this case is 0.3874004 which is small in comparison to median value.

g. Based on this data set, provide an estimate for the tenth percentile of medv in Boston suburbs. Call this quantity ˆμ0.1. (You can use the quantile() function.)

```
percent.hat <- quantile(medv, c(0.1))
percent.hat
```

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

h. Use the bootstrap to estimate the standard error of ˆμ0.1. Comment on your findings.

```
boot.fn <- function(data, index) {
    mu <- quantile(data[index], c(0.1))
    return (mu)
}
boot(medv, boot.fn, 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = medv, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##     original  bias    std. error
## t1*    12.75 0.00515   0.5113487
```

The findings from the produced results show an estimated tenth percentile value of 12.75 which is also equivalent to the results produced in part (g). Additionally, in this case a standard error of 0.5113 was obtained which is small in comparison to percentile value.