Lab #13 - Resampling Methods

Econ 224

October 23rd, 2018

Introduction

In this lab you will work through Section 5.3 of ISL and record your code and results in an RMarkdown document. I have added section headings below to help you organize your results. You do not have to submit this lab, so you don't have to type up a detailed description of what you've done. However, I'd suggest that you write down some notes for your own future reference. These will be helpful on the problem set.

You do not need to follow the code in ISL exactly: feel free to use your preferred coding style. In particular, rather than using the `attach` command suggested by ISL, feel free to instead use `dplyr` or related commands to produce more readable and error-resistant code. If you choose to do this, you will need to select the rows of a tibble *by position*, i.e. by row index. You can do this using the `slice` function in `dplyr`.

You will need the `ISLR` package for the lab, so please install it if you have not done so already. Note that this lab uses two different datasets: `Auto` and `Portfolio`. Both of these are included with `ISLR`. Make sure to read the documentation for each dataset in the R help files before proceeding. There are also a few new R functions that you will encounter in this tutorial `poly`, `cv.glm`, and `boot`. Make sure to read the help files for these functions to make sure that you understand how they work.

The Validation Set Approach

Work through section 5.3.1 of ISL and add your code and results below.

```
library(ISLR)
library(tidyverse)
#--------------------- create training and test samples
set.seed(1)
train_indices <- sample(nrow(Auto), 196)
train <- Auto %>% slice(train_indices)
test <- Auto %>% slice(-train_indices)

#--------------------- fit regressions using training sample
fit1 <- lm(mpg ~ horsepower, train)
fit2 <- lm(mpg ~ poly(horsepower, 2), train)
fit3 <- lm(mpg ~ poly(horsepower, 3), train)

#--------------------- predict test data
predicted1 <- predict(fit1, test)
predicted2 <- predict(fit2, test)
predicted3 <- predict(fit3, test)

#--------------------- compare MSE of predictions
test %>% summarize(MSE1 = mean((mpg - predicted1)^2),
                   MSE2 = mean((mpg - predicted2)^2),
                   MSE3 = mean((mpg - predicted3)^2))
```

```
      MSE1     MSE2     MSE3
1 26.14142 19.82259 19.78252
```

```
#-------------------- try re-running with a different seed
```

Leave-One-Out Cross-Validation

Work through section 5.3.2 of ISL and add your code and results below.

```
library(boot)
#--------------------------- Function to automate LOO-CV for polynomial fits
loo_cv_poly <- function(degree) {
  glm_fit <- glm(mpg ~ poly(horsepower, degree), data = Auto)
  cv_error <- cv.glm(Auto, glm_fit)
  cv_error$delta
}
sapply(1:5, loo_cv_poly)
```

```
         [,1]     [,2]     [,3]     [,4]     [,5]
[1,] 24.23151 19.24821 19.33498 19.42443 19.03321
[2,] 24.23114 19.24787 19.33448 19.42371 19.03242
```

k-Fold Cross-Validation

Work through section 5.3.3 of ISL and add your code and results below.

```
#--------------------------- Function to automate LOO-CV for polynomial fits
tenfold_cv_poly <- function(degree) {
  glm_fit <- glm(mpg ~ poly(horsepower, degree), data = Auto)
  cv_error <- cv.glm(Auto, glm_fit, K = 10)
  cv_error$delta[1]
}
set.seed(17)
sapply(1:10, tenfold_cv_poly)
```

```
 [1] 24.20520 19.18924 19.30662 19.33799 18.87911 19.02103 18.89609
 [8] 19.71201 18.95140 19.50196
```

The Bootstrap

Work through section 5.3.4 of ISL and add your code and results below.

Estimating the Accuracy of a Statistic of Interest

```
#--------------------------- Function for use with boot() in portfolio example
get_alpha <- function(dat, indices) {
  X <- dat$X[indices]
  Y <- dat$Y[indices]
  (var(Y) - cov(X,Y)) / (var(X) + var(Y) - 2 * cov(X,Y))
}
```

```r
set.seed(1)

#--------------------------- Bootstrap sample "by hand"
get_alpha(Portfolio, sample(100, 100, replace = TRUE))
```

```
[1] 0.5963833
```

```r
#--------------------------- Bootstrap using boot()
boot(Portfolio, get_alpha, R = 1000)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Portfolio, statistic = get_alpha, R = 1000)


Bootstrap Statistics :
     original        bias    std. error
t1* 0.5758321 -7.315422e-05  0.08861826
```

Estimating the Accuracy of a Linear Regression Model

```r
#--------------------------- Function to calculate regression coefs for use with boot()
get_coefs <- function(dat, indices) {
  reg <- lm(mpg ~ horsepower, dat, subset = indices)
  coef(reg)
}
#--------------------------- Bootstrap samples "by hand"
set.seed(1)
get_coefs(Auto, sample(392, 392, replace = TRUE))
```

```
(Intercept)  horsepower
 38.7387134  -0.1481952
```

```r
get_coefs(Auto, sample(392, 392, replace = TRUE))
```

```
(Intercept)  horsepower
 40.0383086  -0.1596104
```

```r
#--------------------------- Bootstrap using boot()
boot(Auto, get_coefs, R = 1000)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
Call:
boot(data = Auto, statistic = get_coefs, R = 1000)


Bootstrap Statistics :
      original      bias    std. error
t1* 39.9358610  0.02972191 0.860007896
t2* -0.1578447 -0.00030823 0.007404467
```

```
#---------------------------- Compare to output from lm
summary(lm(mpg ~ horsepower, Auto))
```

```
Call:
lm(formula = mpg ~ horsepower, data = Auto)

Residuals:
     Min       1Q   Median       3Q      Max
-13.5710  -3.2592  -0.3435   2.7630  16.9240

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 39.935861   0.717499   55.66   <2e-16 ***
horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.906 on 390 degrees of freedom
Multiple R-squared:  0.6059,    Adjusted R-squared:  0.6049
F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

```
#---------------------------- Repeat the above for a quadratic regression
get_coefs2 <- function(dat, indices) {
  reg <- lm(mpg ~ poly(horsepower, 2), dat, subset = indices)
  coef(reg)
}
set.seed(1)
boot(Auto, get_coefs2, R = 1000)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Auto, statistic = get_coefs2, R = 1000)


Bootstrap Statistics :
      original      bias    std. error
t1*   23.44592 0.003943212   0.2255528
t2* -120.13774 0.117312678   3.7008952
t3*   44.08953 0.047449584   4.3294215
```

```r
summary(lm(mpg ~ poly(horsepower, 2), Auto))
```

```
Call:
lm(formula = mpg ~ poly(horsepower, 2), data = Auto)

Residuals:
     Min       1Q   Median       3Q      Max
-14.7135  -2.5943  -0.0859   2.2868  15.8961

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)           23.4459     0.2209  106.13   <2e-16 ***
poly(horsepower, 2)1 -120.1377     4.3739  -27.47   <2e-16 ***
poly(horsepower, 2)2   44.0895     4.3739   10.08   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.374 on 389 degrees of freedom
Multiple R-squared:  0.6876,    Adjusted R-squared:  0.686
F-statistic:   428 on 2 and 389 DF,  p-value: < 2.2e-16
```