Chapter 5 Lab

Cross-validation and Bootstrap Lab

Adrien Osakwe

The Validation Set Approach

```
library(ISLR2)
```

```
## Warning: package 'ISLR2' was built under R version 4.1.2
```

```
set.seed(1)
train <- sample(392,196)
lm.fit <- lm(mpg ~ horsepower, data  = Auto , subset = train)
attach(Auto)
lin1 <- mean((mpg - predict(lm.fit,Auto))[-train]^2)

lm.fit2 <- lm(mpg ~ poly(horsepower,2), data  = Auto , subset = train)
quad1 <- mean((mpg - predict(lm.fit2,Auto))[-train]^2)

lm.fit3 <- lm(mpg ~ poly(horsepower,3), data  = Auto , subset = train)
cube1 <- mean((mpg - predict(lm.fit3,Auto))[-train]^2)

first_set <- rbind(c("Linear","Quadratic","Cubic"),c(lin1,quad1,cube1))
first_set
```

```
##      [,1]               [,2]               [,3]
## [1,] "Linear"           "Quadratic"        "Cubic"
## [2,] "23.2660086465003" "18.7164594933828" "18.7940067973945"
```

```
## We find that the polynomial regressions improve the MSE
## Will now try with a different validation set
set.seed(2)
train <- sample(392,196)
lm.fit <- lm(mpg ~ horsepower, data  = Auto , subset = train)
attach(Auto)
```

```
## The following objects are masked from Auto (pos = 3):
##
##     acceleration, cylinders, displacement, horsepower, mpg, name,
##     origin, weight, year
```

```r
lin2 <-mean((mpg - predict(lm.fit,Auto))[-train]^2)

lm.fit2 <- lm(mpg ~ poly(horsepower,2), data  = Auto , subset = train)
quad2 <- mean((mpg - predict(lm.fit2,Auto))[-train]^2)

lm.fit3 <- lm(mpg ~ poly(horsepower,3), data  = Auto , subset = train)
cube2 <- mean((mpg - predict(lm.fit3,Auto))[-train]^2)

## This validation set has on average MSEs than the previous one
first_set <- rbind(first_set,c(lin2,quad2,cube2))
first_set
```

```
##      [,1]               [,2]               [,3]
## [1,] "Linear"           "Quadratic"        "Cubic"
## [2,] "23.2660086465003" "18.7164594933828" "18.7940067973945"
## [3,] "25.7265106448139" "20.4303642741463" "20.3853268638776"
```

```r
## Overall the quadratic regression seems to have the lowest MSE for modeling
## mpg based on horsepower
```

Leave-One-Out Cross-Validation

```r
library(boot)
#The cv.glm function can be used to run the LOOCV method
glm.fit <- glm(mpg ~ horsepower, data = Auto)
cv.err <- cv.glm(Auto, glm.fit)
#This contains the cross-validation results
cv.err$delta
```

```
## [1] 24.23151 24.23114
```

```r
cv.error <- rep(0, 10)
for (i in 1:10) {
  glm.fit <- glm( mpg ~ poly(horsepower, i), data = Auto )
  cv.error[i] <- cv.glm(Auto , glm.fit)$delta[1]
}
cv.error
```

```
##  [1] 24.23151 19.24821 19.33498 19.42443 19.03321 18.97864 18.83305 18.96115
##  [9] 19.06863 19.49093
```

k-Fold Cross-Validation

```r
set.seed(17)
cv.error.10 <- rep(0, 10)
for (i in 1:10) {
  glm.fit2 <- glm( mpg ~ poly(horsepower, i), data = Auto )
  cv.error.10[i] <- cv.glm(Auto , glm.fit2, K = 10)$delta[1]
}
cv.error.10
```

```
##  [1] 24.27207 19.26909 19.34805 19.29496 19.03198 18.89781 19.12061 19.14666
##  [9] 18.87013 20.95520
```

```
## Computation time is a lot shorter for this (and hence why it is an advantage
## in comparison to LOOCV)
```

Side note from ISLR:

"Notice that the computation time is shorter than that of LOOCV. (In principle, the computation time for LOOCV for a least squares linear model should be faster than for k-fold CV, due to the availability of the formula (5.2) for LOOCV; however, unfortunately the cv.glm() function does not make use of this formula.)"

The Bootstrap

```
library(ISLR2)
library(boot)

alpha.fn <- function(data,index){
  X <- data$X[index]
  Y <- data$Y[index]
  (var(Y) - cov(X,Y))/(var(Y) + var(X) - 2*cov(X,Y))
}

#Parameter estimate
alpha.fn(Portfolio,1:100)
```

```
## [1] 0.5758321
```

```
set.seed(7)
alpha.fn(Portfolio, sample(100,100, replace =T))
```

```
## [1] 0.5385326
```

```
##Bootstrapping the parameter estimate using boot()

boot(Portfolio,alpha.fn, R =1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Portfolio, statistic = alpha.fn, R = 1000)
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 0.5758321 0.0007959475  0.08969074
```

```r
##Estimate Linear regression model accuracy
boot.fn <- function(data , index ) +
  coef(lm( mpg ~ horsepower , data = data , subset = index ))
boot.fn(Auto , 1:392)
```

```
## (Intercept)  horsepower
##  39.9358610  -0.1578447
```

```r
set.seed(1)
boot.fn(Auto,sample(392,392, replace = T))
```

```
## (Intercept)  horsepower
##  40.3404517  -0.1634868
```

```r
boot.fn(Auto,sample(392,392, replace = T))
```

```
## (Intercept)  horsepower
##  40.1186906  -0.1577063
```

```r
boot(Auto,boot.fn,R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Auto, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##       original        bias    std. error
## t1* 39.9358610  0.0544513229 0.841289790
## t2* -0.1578447 -0.0006170901 0.007343073
```

```r
summary(lm( mpg ~ horsepower , data = Auto ))$coef
```

```
##                Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept) 39.9358610 0.717498656  55.65984 1.220362e-187
## horsepower  -0.1578447 0.006445501 -24.48914  7.031989e-81
```

```r
## bootstrap gives a more accurate estimation of the SE as it does not make
# same assumptions as the summary() method


## Adding quadratic variable to model
boot.fn2 <- function(data , index ) +
  coef(lm( mpg ~ horsepower + I(horsepower^2),
           data = data , subset = index ))
set.seed(1)
boot.fn2(Auto,sample(392,392, replace = T))
```

```
##     (Intercept)      horsepower I(horsepower^2) 
##    57.474669648    -0.479632716     0.001284905
```

```r
boot(Auto,boot.fn2,R = 1000)
```

```
## 
## ORDINARY NONPARAMETRIC BOOTSTRAP
## 
## 
## Call:
## boot(data = Auto, statistic = boot.fn2, R = 1000)
## 
## 
## Bootstrap Statistics :
##         original        bias     std. error
## t1* 56.900099702  3.538219e-02 2.0311700583
## t2* -0.466189630 -7.043881e-04 0.0324424001
## t3*  0.001230536  2.812430e-06 0.0001172544
```

```r
summary(lm( mpg ~ horsepower + I(horsepower^2) , data = Auto ))$coef
```

```
##                     Estimate   Std. Error   t value      Pr(>|t|)
## (Intercept)     56.900099702 1.8004268063  31.60367 1.740911e-109
## horsepower      -0.466189630 0.0311246171 -14.97816  2.289429e-40
## I(horsepower^2)  0.001230536 0.0001220759  10.08009  2.196340e-21
```

```r
#Using a design formula that better matches the trend seen in the data gives a
# Standard Error estimate that is much closer to the bootstrap estimate
```

Lab Complete