CrossValidation and Bootstrap

Abhirup Sen

11/05/2021

```
# tinytex::install_tinytex()

library(ISLR)
library(tinytex)
```

Validation Set Approach

random selection of 196 samples out of 392 observations.

```
set.seed(1)
train = sample(392, 196)
```

```
lm.fit = lm(mpg~ horsepower, data=Auto, subset = train)
```

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

```
## [1] 23.26601
```

poly() is used to estimate the test error for the quadratic and cubic regressions.

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

```
## [1] 18.71646
```

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

```
## [1] 18.79401
```

choosing a different training set

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

```
## [1] 25.72651
```

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

```
## [1] 20.43036
```

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

```
## [1] 20.38533
```

Leave-One-Out-Cross-Validation

The **LOOCV** model can be automatically computed for any generalized linear model using $glm()$ and $cv.glm()$ functions.Since we are not specifying any type - it would behave as linear regression == lm()

```
glm.fit = glm(mpg~horsepower, data=Auto)
coef(glm.fit)
```

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

```
lm.fit = lm(mpg~horsepower, data=Auto)
coef(lm.fit)
```

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

As expected!!!

```
library(boot)
glm.fit = glm(mpg~horsepower, data = Auto)
cv.err = cv.glm(Auto, glm.fit)
cv.err$delta
```

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

```
start.time <- Sys.time()


cv.error = rep(0,5)

for ( i in 1:5){
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
```

```
end.time <- Sys.time()
time.taken <- end.time - start.time
time.taken
```

```
## Time difference of 22.81967 secs
```

K fold Cross Validation

```
start.time <- Sys.time()

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

```
##  [1] 24.27207 24.08261 24.29863 24.25307 24.22995 24.24427 24.28884 24.17441
##  [9] 24.16991 24.21398
```

```
end.time <- Sys.time()
time.taken <- end.time - start.time
time.taken
```

```
## Time difference of 0.8991339 secs
```

** note the time taken in K-fold CV is much much lesser**

The Bootstrap

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

```
alpha.fn(Portfolio , 1:100)
```

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

use the **sample()** function to randomly select 100 values from a set of observations

```
set.seed(1)
alpha.fn(Portfolio, sample(100,100, replace = T))
```

```
## [1] 0.7368375
```

```
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 -0.001695873  0.09366347
```

```
boot.fn=function(data , index)
  return(coef(lm(mpg~horsepower, data = data, subset = index)))
boot.fn(Auto, 1:392)
```

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

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

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

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

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

```
boot.fn(Auto, sample(100,392,replace =T))
```

```
## (Intercept)  horsepower
## 31.30831939 -0.09989368
```

```
boot(Auto, boot.fn, 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Auto, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##       original        bias    std. error
## t1* 39.9358610  0.0525074429 0.840128665
## t2* -0.1578447 -0.0006042412 0.007333203
```

```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
boot.fn = function(data, index)
  coefficients(lm(mpg~horsepower+I(horsepower^2), data = data, subset=index))
set.seed(1)
boot(Auto,boot.fn, 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Auto, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##         original        bias     std. error
## t1* 56.900099702  3.511640e-02 2.0300222526
## t2* -0.466189630 -7.080834e-04 0.0324241984
## t3*  0.001230536  2.840324e-06 0.0001172164
```

```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
```