MA685 HW2

Jiayuan Shi

Feb 3, 2016

Exercise 1 (Conceptional)

Exercise 2(p. 197): Derive the probabilty that a given sample is part of a bootstrap sample.

(a)

There are n observations. Because bootstrap sampling draws items with replacement, we draw from the same sample and with the same probability each time. There are n-1 items excluding the jth item. So the probability that the first bootstrap observation is not the jth observation from the original sample is $1 - \frac{1}{n}$.

(b)

The probability is also $1 - \frac{1}{n}$, since the set of observation we draw is the same, and we are sampling with replacement.

(c)

The probability that jth observation is not the first sample in bootstrap is $1 - \frac{1}{n}$. The total bootstrap sample size is n, so we need to pick n different observations, and none of them should be the jth one. Since bootstrapping does sampling with replacement, the probability of each observation is independent of the other. Applying the product rule, we just have to multiply $1 - \frac{1}{n}$, n times, and we get $(1 - \frac{1}{n})^n$

(d)

When n = 5: Pr(in) = 1 - Pr(out) = $(1 - \frac{1}{n})^n = (1 - \frac{1}{5})^5 = 0.672$

(e)

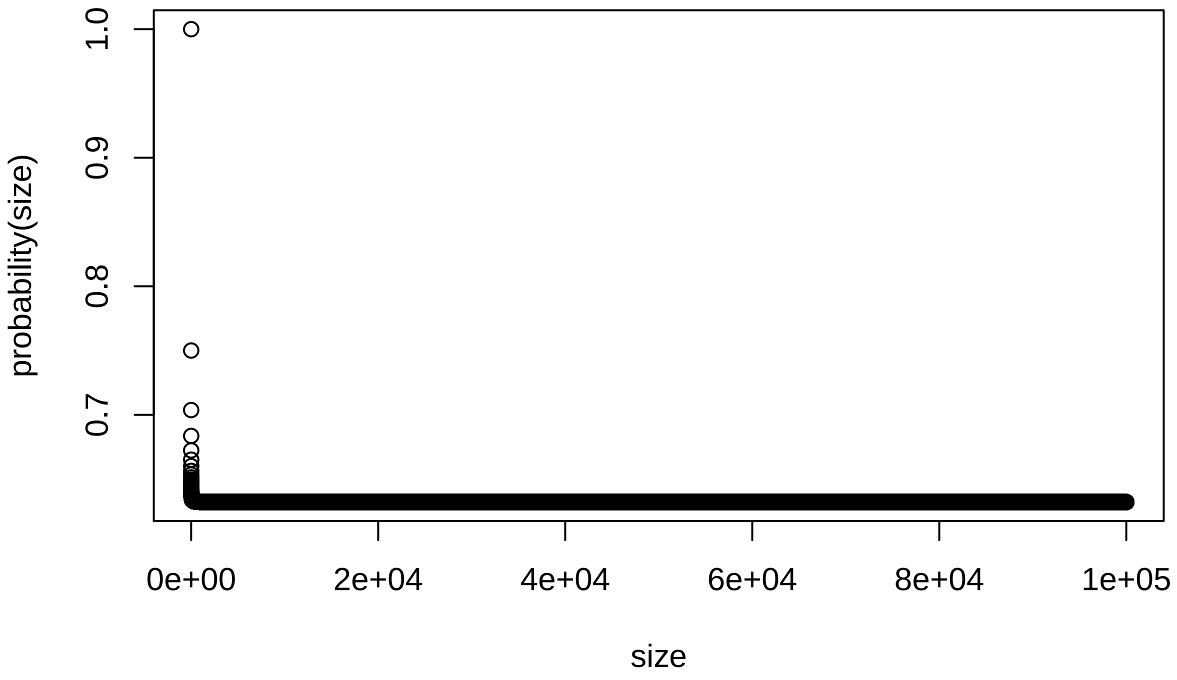
When n = 100: Pr(in) = 1 - Pr(out) = $(1 - \frac{1}{n})^n = (1 - \frac{1}{100})^{100} = 0.634$

(f)

When n = 10000: Pr(in) = 1 - Pr(out) = $(1 - \frac{1}{n})^n = (1 - \frac{1}{10000})^{10000} = 0.632$

(g)

```r
probability <- function(n) {
  1 - (1 - 1/n)^n
  }
size <- c(1:100000)
plot(size, probability(size))
```



With the increase of the sample size, the probability that the jth observation is in the bootstrap sample seems to converge on about 0.632 quickly and then stay there.

(h)

```r
store <- rep(NA, 10000)
for(i in 1:10000){
  store[i] <- sum(sample(1:100, rep=TRUE)==4)>0
}
mean(store)
```

```
## [1] 0.6316
```

This made a list of length 10000. Each time we sampled 0 to 100 with replacement, and checked if j = 4 is in the list. The numerical results show with an approximate mean probability of 0.6357, the list contains the number 4. This result is close to the result 0.632 we get in (g).

Exercise 2 (Applied: CV)

Exercise 8 (p. 200): Perform cross-validation on a simulated data set.

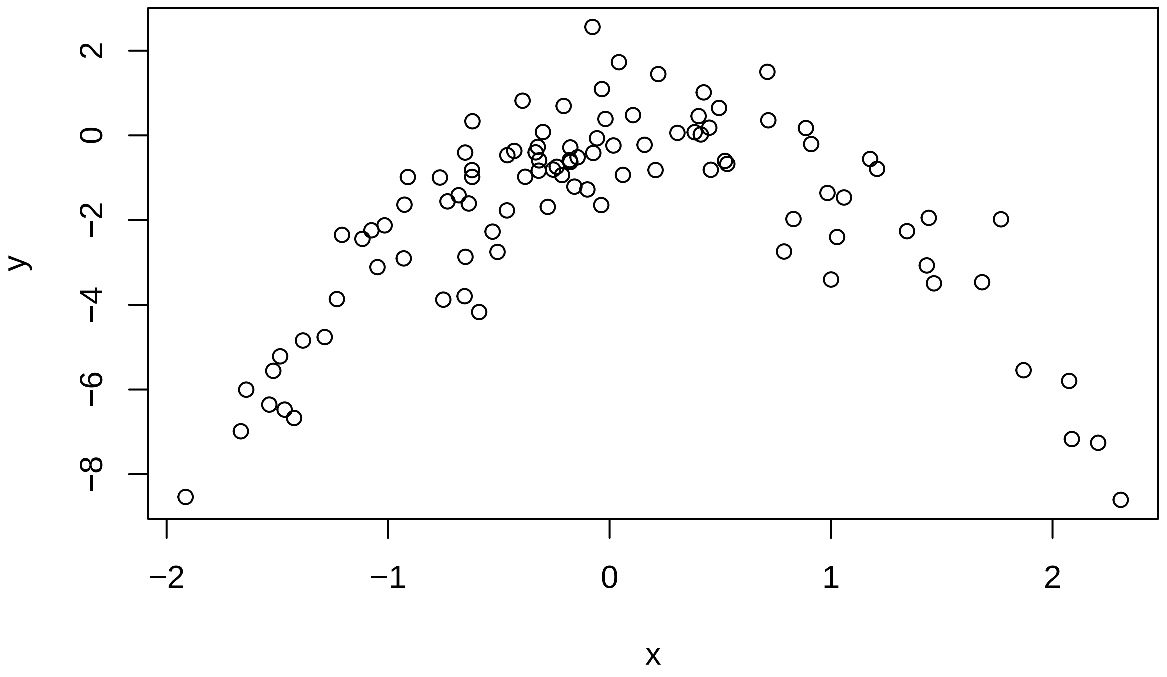
(a)

```r
set.seed(1)
y  <-  rnorm(100)
x  <-  rnorm(100)
y  <-  x - 2*x^2 + rnorm(100)
```

In this dataset, n is 100 and p is 2. $Y = X - 2X^2 + \epsilon$.

(b)

```r
plot(x, y)
```



X is quadratic in terms of Y. X is from about -2 to 2, and Y from about -8 to 2.

(c)

```r
library(boot)
data  <-  data.frame(x,y)
set.seed(1)
glm.fit1  <-  glm(y~x)
cv.glm(data, glm.fit1)$delta
```

```
## [1] 5.890979 5.888812
```

```r
glm.fit2  <-  glm(y~poly(x,2))
cv.glm(data, glm.fit2)$delta
```

```
## [1] 1.086596 1.086326
```

```r
glm.fit3  <-  glm(y~poly(x,3))
cv.glm(data, glm.fit3)$delta
```

```
## [1] 1.102585 1.102227
```

```
glm.fit4  <-  glm(y~poly(x,4))
cv.glm(data, glm.fit4)$delta
```

```
## [1] 1.114772 1.114334
```

(d)

```
set.seed(10)
glm.fit1  <-  glm(y~x)
cv.glm(data, glm.fit1)$delta
```

```
## [1] 5.890979 5.888812
```

```
glm.fit2  <-  glm(y~poly(x,2))
cv.glm(data, glm.fit2)$delta
```

```
## [1] 1.086596 1.086326
```

```
glm.fit3 <- glm(y~poly(x,3))
cv.glm(data, glm.fit3)$delta
```

```
## [1] 1.102585 1.102227
```

```
glm.fit4 <- glm(y~poly(x,4))
cv.glm(data, glm.fit4)$delta
```

```
## [1] 1.114772 1.114334
```

The results are the same, because LOOCV will not change, and it is the same with any iteration given the same undelrying data and model.

(e)

This was expected because it matches the true form of Y. The poly(x,2) model had the smallest LOOCV error. This was expected because it matches the model used to generate the data in (a).

(f)

```
summary(glm.fit4)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -2.8914  -0.5244   0.0749   0.5932   2.7796
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.8277     0.1041 -17.549   <2e-16 ***
## poly(x, 4)1   2.3164     1.0415   2.224   0.0285 *
## poly(x, 4)2 -21.0586     1.0415 -20.220   <2e-16 ***
## poly(x, 4)3  -0.3048     1.0415  -0.293   0.7704
## poly(x, 4)4  -0.4926     1.0415  -0.473   0.6373
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.084654)
##
##     Null deviance: 552.21  on 99  degrees of freedom
## Residual deviance: 103.04  on 95  degrees of freedom
## AIC: 298.78
##
## Number of Fisher Scoring iterations: 2
```

Yes, when we do a poly(x,4) model, we see that the x and x^2 terms are the two that end up statistically significant. These results agree with the conclusions drawn based on the cross-validation results.

Exercise 3 (Applied: Bootstrap)

Exercise 9 (p. 201): Perform bootstrap on Boston housing data.

(a)

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

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

This estimate mu is 22.5328.

(b)

```
se <- sd(medv) / sqrt(length(medv))
se
```

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

The estimate of standard error of sample mean is 0.4089.

(c)

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

```
##
## 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
```

The estimate of standard error of the sample mean 0.4119 is similar to answer 0.4089 from (b).

(d)

```r
t.test(medv)
```

```
##
##  One Sample t-test
##
## data:  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
```

```r
# 21.72953 23.33608
c(bstrap$t0 - 2*0.4119, bstrap$t0 + 2*0.4119)
```

```
## [1] 21.70901 23.35661
```

```r
# 21.70901 23.35661
```

Bootstrap estimate only 0.02 away for t.test estimate. They are very similar. The 95% confidence interval of the bootstrap estimate from (c) is only 0.02 away for the results obtained using t.test.

(e)

```
mu_med <- median(medv)
mu_med
```

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

The estimate mu of med is 21.2.

(f)

```
set.seed(1)
boot.fn <- function(data,index){
  median(data[index])
}
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 -0.01615   0.3801002
```

The estimate of standard error of the median 0.3801 is lower than that of the mean 0.4119.

(g)

```
mu_tenth <- quantile(medv, c(0.1))
mu_tenth
```

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

The estimate for the tenth percentile of medv is 12.75.

(h)

```
set.seed(1)
boot.fn <- function(data,index){
  quantile(data[index],p=0.1)
}
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 0.01005    0.505056
```

The estimate of standard error of the tenth-percentile 0.5051 is higher than that of the mean 0.4119 and of the the median 0.3801. Apparently these outliers must be more sensitive to which subset is chosen than the mean and median.

Exercise 4 (To be graded in detail)

(a)

```r
bootsample <- matrix(c(4.85, 2.61, 1.95, 4.95, 2.61, 1.95, 4.63,
                      3.31, 4.95, 2.61, 3.31, 4.63, 4.95, 4.63,
                      1.95, 2.61, 3.31, 1.95, 2.61, 4.85, 1.95,
                      1.95, 3.31, 2.61, 4.85, 1.95, 1.95, 4.95,
                      1.95, 4.95, 4.57, 4.95, 4.57, 4.63, 3.31,
                      4.63, 4.85, 4.85, 3.31, 4.95, 3.31, 1.95,
                      4.95, 4.95, 4.57, 1.95, 4.57, 4.85, 1.95,
                      4.95, 1.95, 1.95, 2.61, 4.95, 2.61, 1.95,
                      4.57, 4.85, 1.95, 3.31, 4.57, 4.85, 2.61,
                      1.95, 2.61, 1.95, 4.85, 4.95, 4.63, 4.57), nrow = 10, byrow = T)
B <- nrow(bootsample)
Tboot <- rep(0,B)
for(i in 1:B){
  Tboot[i] <- mean(bootsample[i,])
}
theta. <- mean(Tboot)
theta.
```

```
## [1] 3.578571
```

```r
sample.mean <- mean(c(4.63, 2.61, 4.95, 1.95, 3.31, 4.85, 4.57))
sample.mean
```

```
## [1] 3.838571
```

The bootstrap mean estimate $\widehat{\theta}^*(\cdot)$ is 3.579, and the sample mean is 3.839 after we get rid of the last three numbers (because in the question the sample size is seven). The bootstrap mean estimate is a little bit smaller than the sample mean.

(b)

```
theta_b <- c(3.36, 4.06, 2.75, 3.08, 4.13, 3.98, 3.97, 3.00, 3.82, 3.64)
dif <- rep(c(0),B)
for(i in 1:B){
   dif[i] <- theta_b[i]-theta.
}
dif
```

```
##  [1] -0.21857143  0.48142857 -0.82857143 -0.49857143  0.55142857
##  [6]  0.40142857  0.39142857 -0.57857143  0.24142857  0.06142857
```

```
se.theta <- sqrt((1/(B-1))*sum((dif)^2))
se.theta
```

```
## [1] 0.4977607
```

The bootstrap estimate for the standard error $\hat{se}(\hat{\theta}(x))$ of the mean of X is 0.498.