Stat 151 Fall 2015

Homework 4 Solutions

November 9, 2015

1. (a) We have iid observations $X_1, \ldots, X_n$ from a dsitribution with unknown variance σ^2. We shall use bootstrap to calculate a confidence interval for σ

Algorithm 1 $1-\alpha$ bootstrap confidence interval of σ

```
Input X = (X_1, ..., X_n), B (number of bootstraps), α
for k ∈ 1 : B do
  Construct X* = SRSWOR(X, n)
  Calculate σ^(i) = SD(X*)
end for
Output confidence interval of σ − {α/2 quantile ofσ^(), (1 − α/2) quantile ofσ^()}
```

(b) A 95% bootstrap confidence interval of σ should cover the true value of σ 95% of the time. We shall now validate this by repeating this entire experiment $M = 1000$ on datasets of size $n = 100$. We shall further set $B = 1000$ meaning that from each dataset we draw B bootstraps to compute confidence intervals for σ. After M replications we count the number of confidence intervals that covered the true value σ. For the sake of experimentation we set $X_i \sim N(5, 1)$.

```
n = 100
M = 1000
B = 1000
count = 0
#pb = txtProgressBar(min = 0, max = B, style = 3)

for (i in 1:M){
  x = rnorm(n,5,1)
  sigvec = array(0,dim = B)
  for (j in 1:B){
    xstar = sample(x,replace = TRUE)
    sigvec[j] = sd(xstar)
  }
  ci = quantile(sigvec,c(0.025,0.975))
  if ((ci[1]<=1)&&(ci[2]>=1))
    count = count + 1
  #setTxtProgressBar(pb, i)
}
print(count)

## [1] 932
```

We see that 932 out of 1000 replications contain the true σ which is fairly close to the expected coverage of 950 out of 1000.

2. Let us begin with creating a new dataframe with only the relevant variables

```
load('twoyear.RData')
tyd = data[,c('lwage','jc','univ','exper')]
```

(a) To perform t-test to check whether $\beta_1 = \beta_2$, we notice that the model

$$lwage = \beta_0 + \beta_1 jc + \beta_2 univ + \beta_3 exper$$

can be rewritten as

$$lwage = \beta_0 + (\beta_1 - \beta_2)jc + \beta_2(jc + univ) + \beta_3 exper$$

in which we need to perform a t-test to check whether the coefficient of `jc` is 0, this test can be read off from the output of lm

```
lm1 = lm(lwage ~ jc + I(jc + univ) + exper, data = tyd)
summary(lm1)$coefficients[2,4]

## [1] 0.142244
```

which shows a p-value of 14.2% which fails to reject the test at 95% level.

(b) The null model is

$$lwage = \beta_0 + \beta_1(jc + univ) + \beta_3 exper$$

comparing the null model with the full model using anova gives us the desired F-test, alternatively we also know that this F-statistic can be obtained simply by squaring the relevant t-statistic from previous subquestion and the corresponding p-value will be exactly the same

```
M = lm(lwage ~ jc + univ + exper,data =tyd)
m = lm(lwage ~ I(jc + univ) + exper, data = tyd)
anova_result = anova(m,M)
print(anova_result)

## Analysis of Variance Table
##
## Model 1: lwage ~ I(jc + univ) + exper
## Model 2: lwage ~ jc + univ + exper
##   Res.Df    RSS Df Sum of Sq     F Pr(>F)
## 1   6760 1250.9
## 2   6759 1250.5  1   0.39853 2.154 0.1422

anova_result$F[2]

## [1] 2.154016
```

giving a F-statistic of 2.154 with the same p-value of 14.2% which fails to reject at 95%

(c) Going back to modified model in part (a)

$$lwage = \beta_0 + (\beta_1 - \beta_2)jc + \beta_2(jc + univ) + \beta_3 exper$$

we know that we need to test if the coefficient of `jc` is 0 or not, which we can do using permutation test as discussed in lecture/section

```
N = 1000 #number of permutations
null.stats = array(0,dim = N)
#pb = txtProgressBar(min = 0, max = N, style = 3)
for(i in 1:N){
  null.stats[i] = summary(lm(lwage ~ sample(jc) + I(jc + univ) + exper, data = tyd))$coefficients[2,3]
  #setTxtProgressBar(pb, i)
}
true.tstat = summary(lm(lwage ~ jc + I(jc + univ) + exper, data = tyd))$coefficients[2,3]
pval = mean(abs(true.tstat)<=abs(null.stats))
cat('p-value based on permutation test is ',pval)

## p-value based on permutation test is  0.131
```

giving a p-value of 13.1%

(d) Using the modified model

$$lwage = \beta_0 + (\beta_1 - \beta_2)jc + \beta_2(jc + univ) + \beta_3 exper$$

and denoting by $\gamma = \beta_1 - \beta_2$ we wish to construct a confidence interval of γ. In the algorithm, y denotes a column populated by `lwage`, X denotes a matrix with 3 columns, respectively populated by `jc, jc+univ, exper`. We shall use residual bootstrap

Algorithm 2 $1-\alpha$ residual bootstrap confidence interval of γ

Input y, X, B (number of bootstraps), α
Construct $\hat{y}$ and $\hat{e}$, the vectors of fitted values and residuals upon regressing y on X
for $k \in 1 : B$ **do**
  Construct $y^* = \hat{y} + SRSWOR(\hat{e}, n)$
  Calculate $\gamma^{(i)}$ the estimated coefficient of `jc` upon regressing y^* on X
end for
Output confidence interval of γ – $\{\alpha/2$ quantile $of \gamma^{()}, (1-\alpha/2)$ quantile $of \gamma^{()}\}$

```
B = 2000
boot.gamma = array(0,dim = B)
#pb = txtProgressBar(min = 0, max = B, style = 3)
lm1 = lm(lwage ~ jc + univ + exper,data = tyd)
yhat = lm1$fitted.values
ehat = lm1$residuals
for(i in 1:B){
  lwage.star = yhat + sample(ehat)
  boot.gamma[i] = summary(lm(lwage.star ~ tyd$jc + I(tyd$jc + tyd$univ) + tyd$exper))$coefficients[2,1]
  #setTxtProgressBar(pb, i)
}
ci = quantile(boot.gamma,c(0.025,0.975))
cat('The bootstrap confidence interval is ',ci)

## The bootstrap confidence interval is  -0.02331413 0.003787554
```

As we see above the bootstrap confidence interval contains 0.

3. (a)

```
library(faraway)
data(savings)
lm3 = lm(sr ~ pop15 + pop75 + dpi + ddpi, data = savings)
confint(lm3,c('pop15','pop75','dpi','ddpi'))

##              2.5 %       97.5 %
## pop15 -0.752517542 -0.169868752
## pop75 -3.873977955  0.490982602
## dpi   -0.002212248  0.001538444
## ddpi   0.014533628  0.804856227
```

(b)

```
B = 2000
boot.betas = matrix(0,B,4)
#pb = txtProgressBar(min = 0, max = B, style = 3)
lm3 = lm(sr ~ pop15 + pop75 + dpi + ddpi, data = savings)
yhat = lm3$fitted.values
ehat = lm3$residuals
for(i in 1:B){
  sr.star = yhat + sample(ehat)
  boot.betas[i,] = summary(lm(sr.star ~ savings$pop15 + savings$pop75 + savings$dpi + savings$ddpi))$coe
  #setTxtProgressBar(pb, i)
}
cis = matrix(0,4,2)
```

```r
cis[,1] = apply(boot.betas,2,function(z) quantile(z,0.025))
cis[,2] = apply(boot.betas,2,function(z) quantile(z,0.975))
rownames(cis) = c('pop15','pop75','dpi','ddpi')
print(cis)
```

```
##               [,1]         [,2]
## pop15 -0.725674974 -0.182269900
## pop75 -3.723841189  0.383302687
## dpi   -0.002012062  0.001343584
## ddpi   0.043705786  0.757476141
```

The bootstrap and theoretical confidence intervals approximately match in this instance, with the possible exception of β_2 corresponding to `pop75` which seems to have shrinked a little.