Bootstrap - Interactions and Seeds

SDS 291

March 4, 2020

Two things about the bootstrap for homeworks:

1. The bootstrap will take a different sample every time - and you'll get different answers. The solution is to *set a seed* so that every time you knit the file, you will start in the same place, get the same distribution, and thus get the same answer.
2. Also, it's better practice to *knit your whole file over and over again* rather than running code chunks iteratively so that you minimize discrepancies between the answers from your knitted verson with one seed and the answers you got when running the code in a code chunk that used a different seed.
3. Naming conventions for interaction terms differ between how the coefficient is saved from a regression model and how it's saved in the bootstrap distribution

Set the Seed

I'm going to set the seed at the beginning, in the code chunk below.

```
knitr::opts_chunk$set(echo = TRUE)
require(Stat2Data)
require(mosaic)
require(magrittr)
require(tidyverse)
data("FirstYearGPA")

set.seed(8675309)
```

The seed can be any number – today's date, your birthday, your telephone number, your zipcode – and R will be consistent when creating the bootstrap sample to pick the same starting point for resampling. It's an abstract concept, which is why the seed doesn't actually need to be, say, a number in the dataset.

Fit the regression equation

Fit the simple linear regression equation $GPA = \beta_0 + \beta_1 SATM + \beta_2 HSGPA + \beta_3(SATM \times HSGPA) + \epsilon$ and construct a 95% CI for the estimate for β_3. [Hint: Remember that you can use `confint()` to calculate the 95% CI for all terms in the model (put the name of the model in the parentheses)].

```
SATM_orig<-lm(GPA~SATM*HSGPA, data=FirstYearGPA)
summary(SATM_orig)
```

```
##
## Call:
## lm(formula = GPA ~ SATM * HSGPA, data = FirstYearGPA)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00571 -0.31168  0.05113  0.30683  0.82901
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.4644149  2.3415700   0.625    0.532
## SATM        -0.0003161  0.0036773  -0.086    0.932
## HSGPA        0.3235961  0.6833125   0.474    0.636
## SATM:HSGPA   0.0003258  0.0010693   0.305    0.761
##
## Residual standard error: 0.4149 on 215 degrees of freedom
## Multiple R-squared:  0.2163, Adjusted R-squared:  0.2054
## F-statistic: 19.78 on 3 and 215 DF,  p-value: 2.315e-11
```

```
confint(SATM_orig)
```

```
##                    2.5 %      97.5 %
## (Intercept) -3.150958000 6.079787785
## SATM        -0.007564155 0.006932016
## HSGPA       -1.023253163 1.670445334
## SATM:HSGPA  -0.001781796 0.002433416
```

You'll note that here the coefficient from the model SATM_orig is saved as `SATM:HSGPA` (with a colon). This becomes important later, because it gets saved with a different name in the bootstrap distribution that you will calculate below.

Construct the Bootstrap Distribution for Regression Coefficients

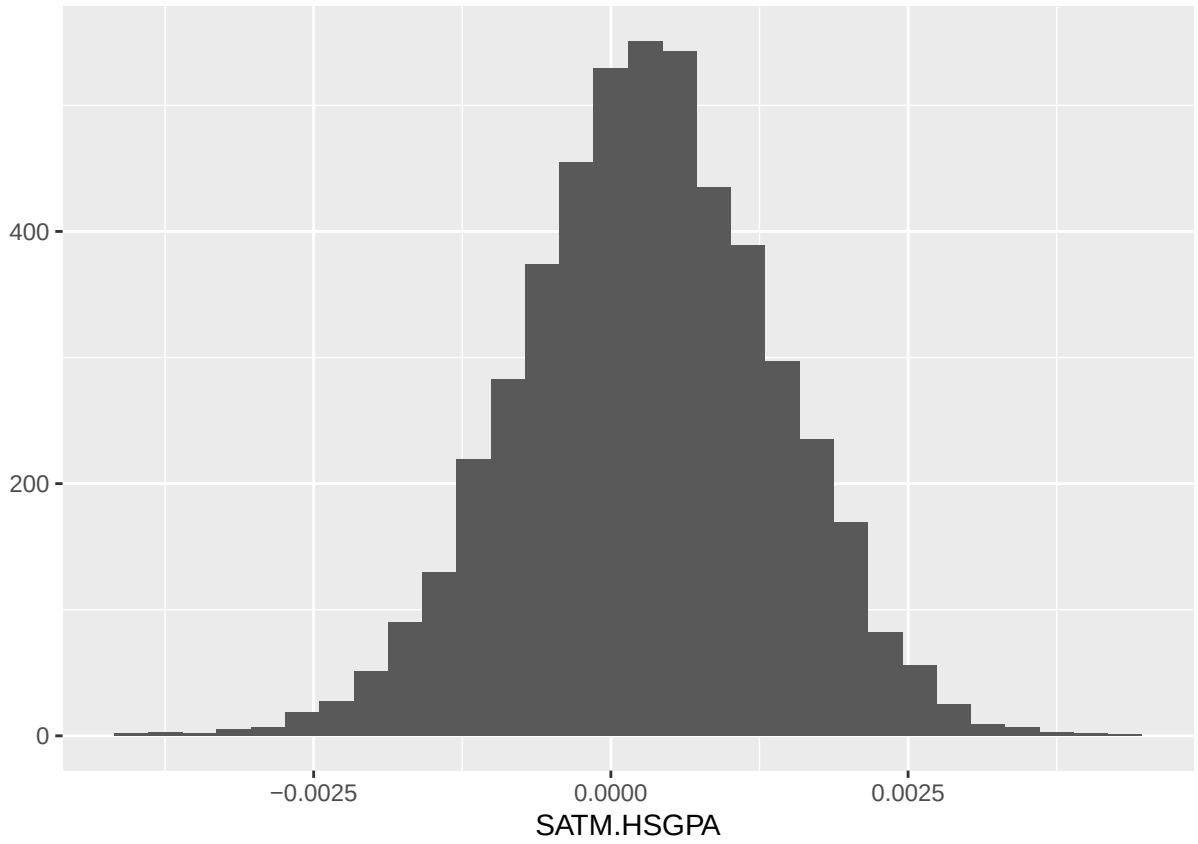
Construct the bootstrap distribution by estimating the coefficients from the regression model (i.e., the bootstrap statistic) in 5000 bootstrap samples. Make a histogram of the bootstrap distribution of the slope coefficient for SATM.

```
SATM_bootstrap<- do(5000) * coef(lm(GPA~SATM*HSGPA, data=resample(FirstYearGPA)))
glimpse(SATM_bootstrap)
```

```
## Observations: 5,000
## Variables: 4
## $ Intercept  <dbl> 7.31351325, 1.15982443, 2.73081168, -2.43560764, 3.30632...
## $ SATM       <dbl> -0.0092813434, 0.0005795890, -0.0027062226, 0.0055896367...
## $ HSGPA      <dbl> -1.33312407, 0.49189724, -0.10490374, 1.40107312, -0.232...
## $ SATM.HSGPA <dbl> 2.847030e-03, -5.476726e-05, 1.100933e-03, -1.332998e-03...
```

```
qplot(x=SATM.HSGPA, data=SATM_bootstrap)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Here, you'll notice that the interaction term is called `SATM.HSGPA` in the bootstrap distribution. Now we have different values about the interaction term saved with different names in two different places.

So, when we wan't to use the coefficient from the original regression (like in Methods 1 and 3), we'll need to use `coef(SATM_orig)["SATM:HSGPA"]` and when we want values from the bootstrap distribution to calculate the standard error or quantiles around of the bootstrap distribution, the value is called `SATM.HSGPA` from the `SATM_bootstrap` dataset we just built above.

Confidence Intervals

Estimate the CI with each of the three approaches and compare them at the end.

Method 1: Standard Deviation

```
zs <- qnorm(c(0.025, 0.975))
coef(SATM_orig)["SATM:HSGPA"] + zs * sd(~SATM.HSGPA, data=SATM_bootstrap)
```

```
## [1] -0.001782178  0.002433798
```

Method #2 - Quantiles from the Bootstrap Distribution

```
qdata(~SATM.HSGPA, p=c(0.025, 0.975), data=SATM_bootstrap)
```

```
##           quantile     p
## 2.5%  -0.001846888 0.025
```

```
## 97.5%  0.002353318 0.975
```

Method #3 - Reverse the quantiles from the bootstrap distribution

```
qs <- qdata(~SATM.HSGPA, p = c(0.025, 0.975), data=SATM_bootstrap)$quantile
coef(SATM_orig)["SATM:HSGPA"] - (qs - coef(SATM_orig)["SATM:HSGPA"])
```

```
## [1]  0.002498509 -0.001701698
```