More Regression Bootstrapping

2018-01-10

Introduction and Research Question

In this set of notes, you will continue learning about bootstrapping in regression analysis. To do so, we will use the *riverside.csv* data to examine whether education level is related to income. The data come from C. Lewis-Beck & Lewis-Beck (2016) and contain five attributes collected from a random sample of $n = 32$ employees working for the city of Riverview, a hyopothetical midwestern city. The attributes include:

- `education`: Years of formal education
- `income`: Annual income (in U.S. dollars)
- `seniority`: Years of seniority
- `gender`: Employee's gender
- `male`: Dummy coded gender variable (0 = Female, 1 = Male)
- `party`: Political party affiliation

Preparation

```
# Load libraries
library(tidyverse)
library(broom)
library(sm)

# Read in data
city = read_csv(file = "~/Dropbox/epsy-8252/data/riverside.csv")
head(city)
```

education	income	seniority	gender	male	party
8	37449	7	male	1	Democrat
8	26430	9	female	0	Independent
10	47034	14	male	1	Democrat
10	34182	16	female	0	Independent
10	25479	1	female	0	Republican
12	46488	11	female	0	Democrat

Normal-Theory Regression Analysis

To begin, we will fit a regression model using education and seniority to predict inclome.

```
lm.1 = lm(income ~ 1 + education + seniority, data = city)

# Model-level estimates
glance(lm.1)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.742	0.724	7646	41.7	0	3	-330	668	674	1695313285	29

```
# Coefficient-level estimates
tidy(lm.1)
```

term	estimate	std.error	statistic	p.value
(Intercept)	6769	5373	1.26	0.218
education	2252	335	6.73	0.000
seniority	739	210	3.52	0.001

Let's examine the assumptions.

```
out.1 = augment(lm.1)
head(out.1)
```

income	education	seniority	.fitted	.se.fit	.resid	.hat	.sigma	.cooksd	.std.resid
37449	8	7	29956	2950	7493	0.149	7628	0.066	1.062
26430	8	9	31433	2875	-5003	0.141	7714	0.027	-0.706
47034	10	14	39631	2378	7403	0.097	7641	0.037	1.019
34182	10	16	41108	2502	-6926	0.107	7657	0.037	-0.959
25479	10	1	30026	3213	-4547	0.177	7723	0.031	-0.655
46488	12	11	41918	1879	4570	0.060	7730	0.008	0.617

```
sm.density(out.1$.std.resid, model = "normal")

ggplot(data = out.1, aes(x = .fitted, y = .std.resid)) +
  geom_point() +
  geom_hline(yintercept = 0) +
  theme_bw() +
  xlab("Fitted values") +
  ylab("Standardized residuals")
```

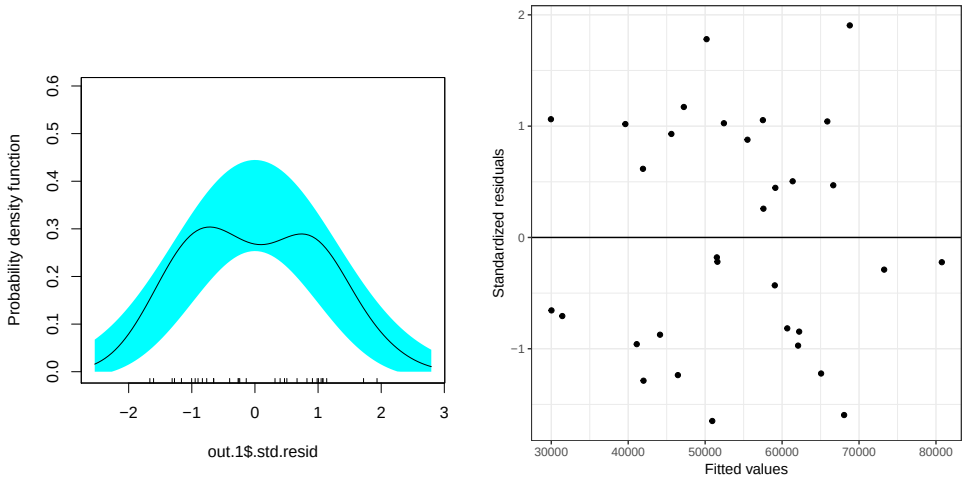

Figure 1. Density plot ofthe standardized residuals (left) and scatterplot of the standardized residuals versus the fitted values (right).

- Linearity—Seems reasonably satisfied.
- Independence—Seems ok since the observed data were randomly sampled; see C. Lewis-Beck & Lewis-Beck (2016).

- Normality—Potentially bimodal (?) although the model envelope suggests this may be due to sampling error.
- Homoskedasticity—Perhaps some heteroskedasticity; although again, this is hard to discern given the small sample size.
-

Bootstrapping Coefficient-Level Estimates

Let's imagine that we are suspicious of whether the normality or homogeneity of variance assumptions are met and thus do not believe that the SEs (and hence the t-statistics and p-values) for the coefficients are valid. To obtain better estimates of these values, we will carry out a bootstrap analysis to obtain bootstrap estimates of the SEs.

In the previous notes, we used the `do()` and `resample()` functions from the **mosaic** library to carry out the bootstrapping. The problem was that these functions were computationally expensive (they took a lot of time). Now, we will use functionality from the **broom** and **dplyr** functions to carry out the bootstrapping. This produces the `tidy()` summary of each bootstrap replication, combined into a single data.frame.

```
boot_reg = city %>%
  bootstrap(1000) %>%
  do( tidy( lm(income ~ 1 + education + seniority, data = .) ) )

head(boot_reg)
```

```
## # A tibble: 6 x 6
## # Groups:   replicate [2]
##   replicate        term estimate std.error statistic      p.value
##       <int>       <chr>    <dbl>     <dbl>     <dbl>        <dbl>
## 1         1 (Intercept)    10139      6054      1.67 0.1047273265
## 2         1   education     2221       319      6.96 0.0000001195
## 3         1   seniority      713       203      3.50 0.0015133286
## 4         2 (Intercept)     9736      5453      1.79 0.0846374941
## 5         2   education     2167       294      7.36 0.0000000415
## 6         2   seniority      618       210      2.95 0.0062412899
```

This is still computationally expensive, although not as expensive as using the functions from the **mosaic** package. Now we can use `group_by()` and `summarize()` to obtain the SEs.

```
boot_reg %>%
  group_by(term) %>%
  summarize(SE = sd(estimate))
```

term	SE
(Intercept)	5102
education	290
seniority	204

Bootstrap-Based Confidence Intervals

Examining the bootstrap distribution for each of the coefficients:

```
library(gridExtra)

p1 = boot_reg %>%
  filter(term == "(Intercept)") %>%
  ggplot(data = ., aes(x = estimate)) +
    geom_density() +
    theme_bw() +
    ggtitle("Intercept")

p2 = boot_reg %>%
  filter(term == "education") %>%
  ggplot(data = ., aes(x = estimate)) +
    geom_density() +
    theme_bw() +
    ggtitle("Education")

p3 = boot_reg %>%
  filter(term == "seniority") %>%
  ggplot(data = ., aes(x = estimate)) +
    geom_density() +
    theme_bw() +
    ggtitle("Seniority")

grid.arrange(p1, p2, p3, nrow = 1)
```

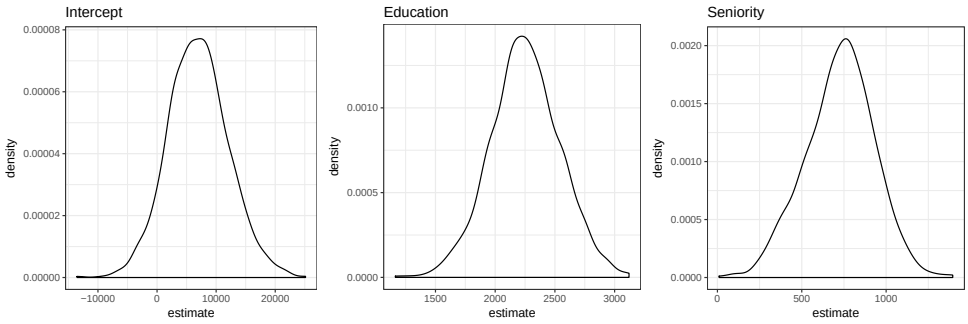

Figure 2. Density plots of the bootstrap distributions for each of the three regression coefficients.

The first two bootstrap distributions (intercept and education) seem relatively symmetric. The bootstrap distribution for the seniority predictor, however, seems a bit negatively skewed. To be safe, we will compute the CIs using the percentile method.

```r
boot_reg %>%
  group_by(term) %>%
  summarize(
    LL = quantile(estimate, prob = .025),
    UL = quantile(estimate, prob = .975)
    )
```

term	LL	UL
(Intercept)	-2980	17774
education	1672	2819
seniority	315	1100

Coefficient-Level Hypothesis Testing

In the normal-theory output, we are given the results for testing whether each of the individual parameters are differnt from zero. This is tested using a t-test with an appropriate df. In our example each t-test uses 29 df.

To compute the t-value, we take the coefficient estimate and divide by the SE. In the normal-theory output, the t-value for the intercept was,

$$t = \frac{6769}{5373} = 1.26$$

Then, to compute the p-value we find the area in the t-distribution with 29 df that is at least as extreme as 1.26. This is two-tailed.

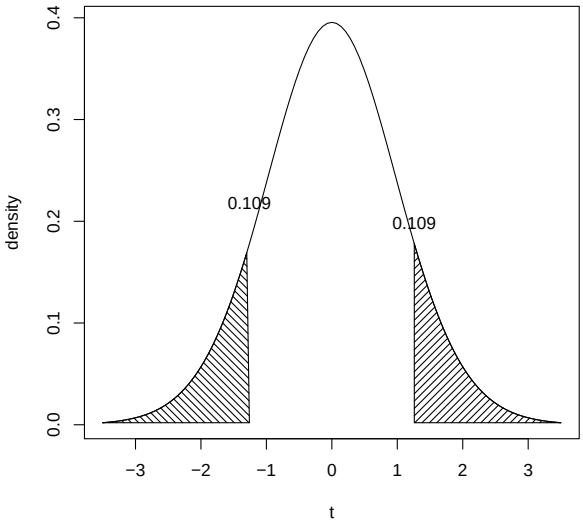

Figure 3. The figure shows a t-distribution with 29 df. The shaded area represents the proportion of the distribution that is at least as extreme as a t-value of 1.26.

The combined shaded area in this t-distribution is 0.218. This is the normal-theory p-value. To determine this directly, we can use the `pt()` function. This function computes the area from $-\infty$ to some value `q=` in a t-distribution with a specified df. Below we compute the area from $-\infty$ to -1.26 (the lower shaded tail).

```r
pt(q = -1.26, df = 29)
```

```
## [1] 0.109
```

To compute the two-tailed p-value, we multiply this by two.

```
2 * pt(q = -1.26, df = 29)
```

```
## [1] 0.218
```

Bootstrap-Based t- and p-Values

Once we have computed the bootstrapped SE, we can use that along with the observed value of the coefficient to obtain a modified t-value. For the intercept,

$$t = \frac{6769}{4820} = 1.40$$

Then, we can use this modified t-value to compute the p-value.

```
2 * pt(q = -1.4, df = 29)
```

```
## [1] 0.172
```

Although the p-value is still non-significant (it is likely that $\beta_0 = 0$), using the bootstrapped SE gives slgihtly more statistical power (produces a smaller p-value) when the assumptions are violated.

We can similarly compute t-values and p-values using the bootstrapped SEs for the other coefficients as well.

Table 1.

Results of Hypothesis Tests for Each of the Regression Coefficients Using the Bootstrapped SEs.

Predictor	Estimate	SE	t	p
Intercept	6769	4820	1.40	0.17083280
Education	2252	283	7.96	0.00000001
Seniority	739	199	3.71	0.00086555

Better p-Values

In the previous example, we used the bootstrapped SEs, but still input those values into a t-distribution. So we are still making an assumption that the distributions are t-distributed. This is a *semi-parametric* analysis in that we did not make an assumption about the distribution to compute the SEs but we did make an assumption about the distribution of to compute the p-value from.

In this case, the use of the semi-parametric method produces MORE reasonable p-values than the normal-based theory, but these still may not be correct. It depends on whether the bootstrap distributions are t-distributed with 29 *df*. In many cases, it is better to make NO ASSUMPTIONS about the shape of the distributions, at any point. This is referred to as *nonparametric* analysis.

The p-value gives us the probability of observing a regression coefficient at least as extreme as the one we observed. For example for the intercept, we are interested in the proportion (probability) of bootstrapped coefficients that are at least as extreme as 6769, UNDER the null hypothesis.

Under the null hypothesis, the sampling distribution is centered at zero. The bootstrap distribution is centered at 6769. To center this at zero, we subtract 6769 from each value. Then we compute the proportion of values that are more extreme than 6769.

```
boot_intercepts = boot_reg %>%
  filter(term == "(Intercept)") %>%
  mutate(centered_estimate = estimate - 6769) %>%
  select(replicate, estimate, centered_estimate)

head(boot_intercepts)
```

```
## # A tibble: 6 x 3
## # Groups:   replicate [6]
##   replicate estimate centered_estimate
##       <int>    <dbl>             <dbl>
## 1         1    10139              3370
## 2         2     9736              2967
## 3         3    12971              6202
## 4         4     7376               607
## 5         5     5887              -882
## 6         6     6188              -581
```

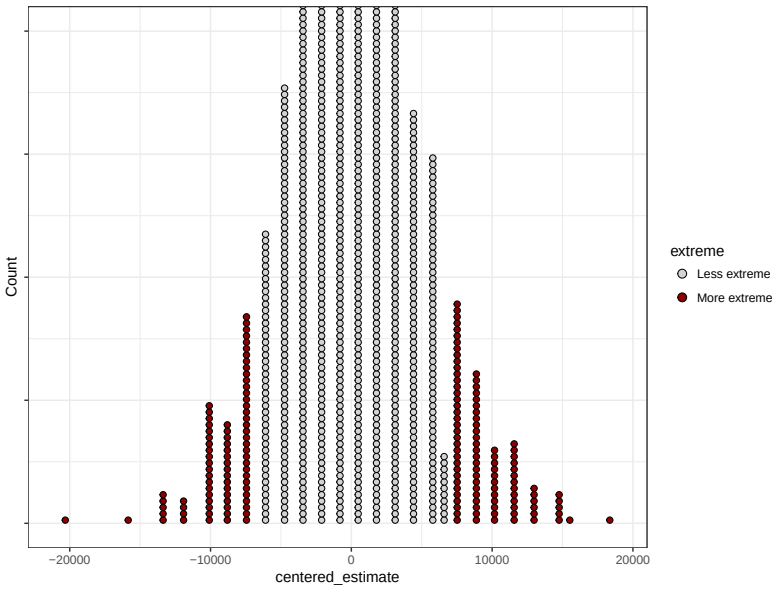

Figure 5. Dotplot of the 1,000 centered bootstrapped intercept values. Centered values greater than 6769 or less than −6769 are colored red.

To actually compute the proportion of the centered replicates that are more extreme than 6769 we use the `sum()` function on a logical statement.

```
sum(boot_intercepts$centered_estimate >= 6769)
```

```
## [1] 97
```

```
sum(boot_intercepts$centered_estimate <= -6769)
```

```
## [1] 79
```

```
# Combine these into a single statement
sum(abs(boot_intercepts$centered_estimate) >= 6769)
```

```
## [1] 176
```

To compute the proportion, we divide this by the total number of replicates.

```r
sum(abs(boot_intercepts$centered_estimate) >= 6769) / 1000
```

```
## [1] 0.176
```

This is the nonparmaetric bootstrapped p-value.

We can compute this in a similar manner for the other coefficients. I do it for the education coefficient below, and leave it as an exercise to compute the seniority p-value.

```r
# Center the distribution
boot_educ = boot_reg %>%
  filter(term == "education") %>%
  mutate(centered_estimate = estimate - 2252) %>%
  select(replicate, estimate, centered_estimate)

head(boot_intercepts)
```

```
## # A tibble: 6 x 3
## # Groups:   replicate [6]
##   replicate estimate centered_estimate
##       <int>    <dbl>             <dbl>
## 1         1    10139              3370
## 2         2     9736              2967
## 3         3    12971              6202
## 4         4     7376               607
## 5         5     5887              -882
## 6         6     6188              -581
```

```r
# Compute p-value
sum(abs(boot_educ$centered_estimate) >= 2252) / 1000
```

```
## [1] 0
```

None of the 1,000 centered bootstrap replicates had a value that was equal to or more extreme than 2252. The proportion is 0. We would report this as less than 1 out of 1000 ($p < .001$).

Adjustment for Simulation Error

Remember that by bootstrapping we introduced simulation error into the estimates. There has been some suggestion that to compensate for this, when computing a p-value, we should add one to the numerator and denominator of our proportion (e.g., Davison & Hinkley, 1997).

```r
# Simulation adjusted p-value for intercept
(sum(abs(boot_intercepts$centered_estimate) >= 6769) + 1) / (1000 + 1)
```

```
## [1] 0.177
```

```r
# Simulation adjusted p-value for education
(sum(abs(boot_educ$centered_estimate) >= 2252) + 1) / (1000 + 1)
```

```
## [1] 0.000999
```

These should be used rather than the non-adjusted p-values. This computation also alleviates ever getting a p-value of zero.

We can write-up the results of our bootstrap analysis as:

> A regression model was fitted to the data to examine predictors of income. Because of potential violation of the distributional assumptions of the model, nonparametric bootstrap tests were used to evaluate the predictors. Monte Carlo p-values were computed by drawing 1,000 bootstrap replicates from the data. Using a correction suggested by Davison and Hinkley (1997), the education predictor (B = 2252, p = .001) and seniority predictor (B = 739, p = .004) were statistically significant. The intercept (B = 6,769, p = .141) was not statistically significant.

Bootstrapping Model-Level Estimates

We can also bootstrap estimates at the model-level. For example, we might be interested in estimating the uncertainty for the RMSE or R^2. We can use the same bootstrap functionality we did at the coefficient-level, but instead of using the `tidy()` function, we will use the `glance()` function.

```
boot_reg2 = city %>%
  bootstrap(1000) %>%
  do( glance( lm(income ~ 1 + education + seniority, data = .) ) )

head(boot_reg2)
```

```
## # A tibble: 6 x 12
## # Groups:   replicate [6]
##   replicate r.squared adj.r.squared sigma statistic        p.value    df
##       <int>     <dbl>         <dbl> <dbl>     <dbl>          <dbl> <int>
## 1         1     0.703         0.682  7569      34.3 0.000000022753     3
## 2         2     0.792         0.778  7208      55.2 0.000000000130     3
## 3         3     0.814         0.801  7529      63.6 0.000000000025     3
## 4         4     0.734         0.716  7720      40.0 0.000000004575     3
## 5         5     0.792         0.778  6585      55.3 0.000000000127     3
## 6         6     0.777         0.762  7312      50.5 0.000000000356     3
## # ... with 5 more variables: logLik <dbl>, AIC <dbl>, BIC <dbl>,
## #   deviance <dbl>, df.residual <int>
```

Let's examine the bootstrap distribution for R^2.

```
ggplot(data = boot_reg2, aes(x = r.squared)) +
  geom_density() +
  theme_bw() +
  geom_vline(xintercept = 0.742, linetype = "dashed")
```

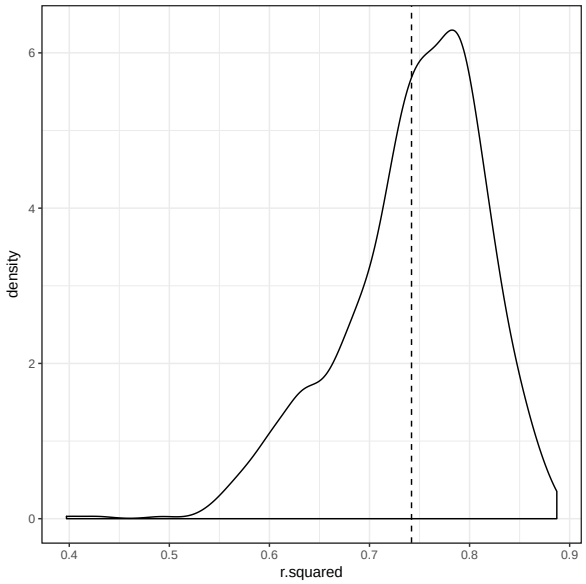

Figure 4. Bootstrap distribution for R^2. The vertical dashed line is drawn at the observed R^2 value of 0.742.

The bootstrap distribution for R^2 is clearly left-skewed. This suggests that the statistic of R^2 is positively biased (it tends to overestimate the population R^2 value). This asymmetry makes it hard to produce a confidence interval using normal-theory. With bootstrapping, however, we can simply compute a CI using the percentile method.

```
quantile(boot_reg2$r.squared, prob = c(0.025, 0.975))
```

```
##  2.5% 97.5%
## 0.587 0.855
```

Notice the confidence interval (like the bootstrap distribution) is asymmetric. In otherwords there is no one value we can choose to add/subtract to the observed R^2 of 0.742 that gives us the limits on this interval. (The upper limit is closer to the observed R^2 than the lower limit.) The asymmetry accounts for some of the positive bias.

Lastly, notice that there is a great deal of uncertainty in R^2. We initially thought that the two predictors accounted for 74.2% of the variation in incomes. Now we are less sure about that. It may be that they account for as little as 60% or as much as 85%. If you are concerned about having precise estimates of variation accounted for, you need extremely large sample sizes.

References

Davison, A., & Hinkley, D. V. (1997). *Bootstrap methods and their application*. New York: Cambridge University Press.

Lewis-Beck, C., & Lewis-Beck, M. (2016). *Applied regression: An introduction* (2nd ed.). Thousand Oaks, CA: Sage.