# Bootstrapping

Adam J Sullivan, PhD

04/30/2018

# Boostrapping

# Goal of Statistics

- Make inferences about a population based on data.
- How??
  - Traditional statistical inference is based on the assumption of drawing repeated samples from a population
  - A statistic (e.g. the mean or a regression coefficient) is assumed to be fixed in the population.

# What happens?

- If a researcher were to draw multiple samples, the statistic would be different each time.
- The term sampling distribution refers to the distribution that would result from taking multiple samples from a population and re-estimating a statistic on each new set of observations.
- Many times the sampling distribution is assumed to be normal, and a statistic is declared to be “significant” if the 95% confidence interval (the area under the curve excluding the smallest and largest 2.5% of values) does not include zero.

# What if Assumptions are violated?

- Many times we do not have normally distributed variables.
- For example, consider a linear regression, we assume the residuals are normally distributed.
- If we do not have normally distributed residuals then our standard errors are off and therefore our individual significance tests for the estimates in our regression are off.

# In Comes Bootstrapping!

- Bootstrapping is a method of resampling.
- Researchers use this to help us understand the variance of different estimates.

# How does it work?

- We just discussed using a sample to make inferences about a population.
- In Bootstrapping, we will treat our sample data as a population.
- Then we will draw many sample from this new "population."

# How does it work?

- For each sample, we calculate our test statistic and save the information.
- If we do this say 500 times, we have 500 estimates of our test statistic.
- We then can get an idea of how variable our test statistic is.

# Example

- Let's simulate a dataset so we know what the truth is.

```
truth <- function(x){
  2.34 + 2.68*x - 3.42*x^2
}

noise <- function(x){
  rnorm(length(x), sd=0.1)
}
```

# Example Data

```
set.seed(124)
data <- data_frame(
  x = runif(n=100, min=0, max = 1),
  y = truth(x) + noise(x)
)
```

# Example Data

Show 10 entries Search:

| | X | Y |
|---|---|---|
| | 0.0830154397990555 | 2.49270699323396 |
| | 0.408794660819694 | 2.84153304720815 |
| | 0.515284994151443 | 2.72824530612371 |
| | 0.39688234962523 | 2.87224597883287 |
| | 0.222722708247602 | 2.73974273183848 |
| | 0.292349642841145 | 2.79255277433931 |
| | 0.584065895760432 | 2.73400152707463 |
| | 0.490911490749568 | 2.74885362924033 |
| | 0.92299912404269 | 1.81464257885602 |
| | 0.279794845962897 | 2.83398863519112 |

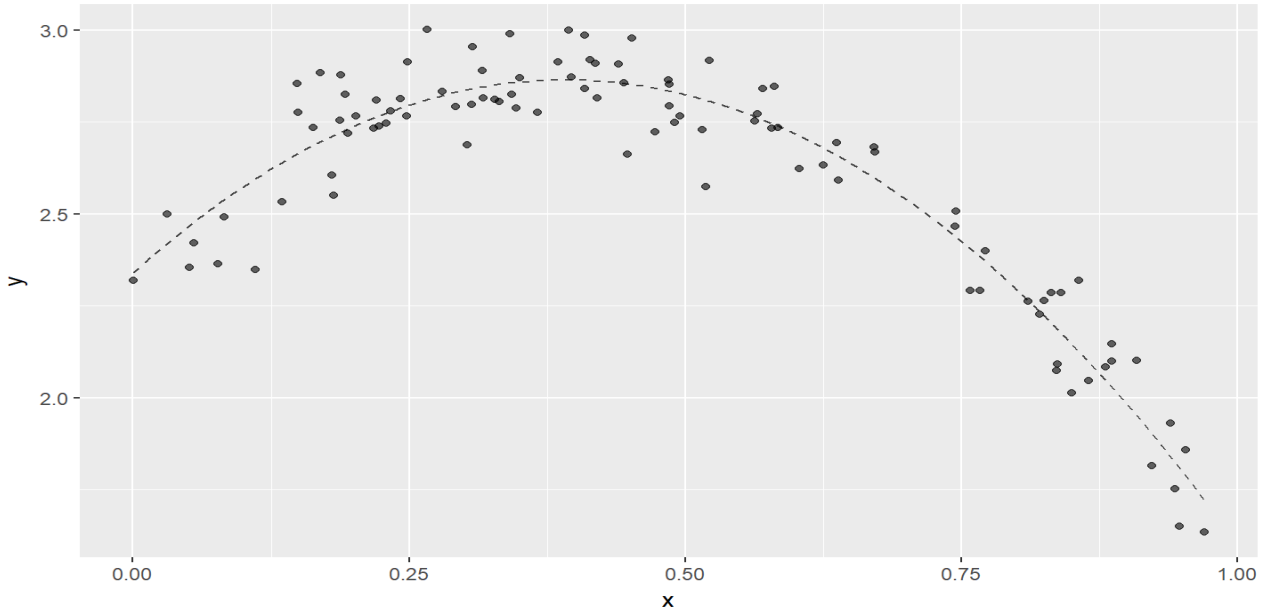
Showing 1 to 10 of 100 entries

Previous 1 2 3 4 5 … 10 Next

# Example Data: Graph

```
ggplot(data, aes(x = x, y = y)) +
  stat_function(fun = truth, color = "black", alpha = 0.7, linetype = "dashed") +
  geom_point(alpha = 0.6)
```

# Example Data: Graph



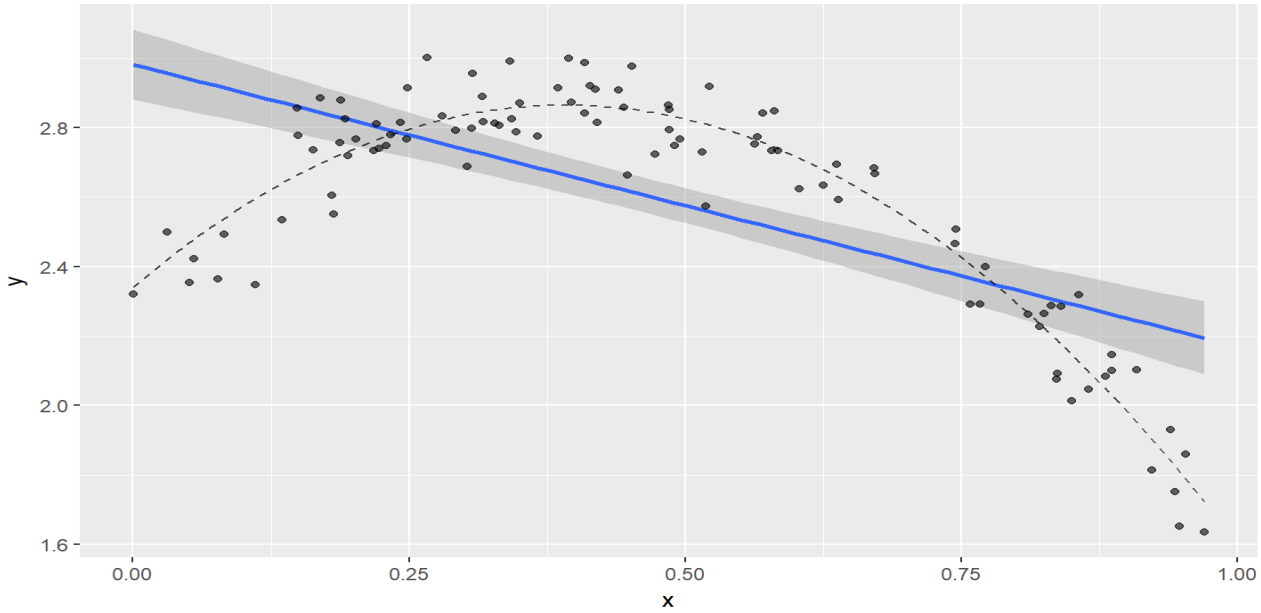
# Our Linear Model

```
library(broom)
mod <- lm(y~poly(x, 2, raw = TRUE), data)
tidy(mod, conf.int=T)[,-c(3:4)]
```

# Our Linear Model

```
##                     term estimate  p.value conf.low conf.high
## 1             (Intercept)     2.32 3.04e-84     2.26      2.39
## 2 poly(x, 2, raw = TRUE)1     2.76 1.10e-32     2.45      3.06
## 3 poly(x, 2, raw = TRUE)2    -3.49 1.58e-42    -3.77     -3.20
```

# Our Linear Model



# Bootstrapping

- We first create 10,000 new datasets

```
data_bootstrap <-
  data %>%
  modelr::bootstrap(10000)
```

# Boostrapping

Show 10 ▼ entries Search:

| STRAP | .ID |
|---|---|
| [object Object] | 00001 |
| [object Object] | 00002 |
| [object Object] | 00003 |
| [object Object] | 00004 |
| [object Object] | 00005 |
| [object Object] | 00006 |
| [object Object] | 00007 |
| [object Object] | 00008 |
| [object Object] | 00009 |
| [object Object] | 00010 |

Showing 1 to 10 of 10,000 entries

Previous 1 2 3 4 5 … 1000 Next

# Running the `lm()` on Our Bootstrap

- We first need to create a function that we wish to run

```
fn_mod <- function(data){
  lm(y~poly(x, 2, raw = TRUE), data=data)
}
```

# Running the `lm()` on Our Bootstrap

- Then we run the `fn_mod` over the bootstraps

```
data_bootstrap_model <-
  data_bootstrap %>%
  mutate(model = map(strap, fn_mod))
```

# Running the `lm()` on Our Bootstrap

Show 10 ▼ entries Search:

| STRAP | .ID | MODEL |
|---|---|---|
| [object Object] | 00001 | [object Object] |
| [object Object] | 00002 | [object Object] |
| [object Object] | 00003 | [object Object] |
| [object Object] | 00004 | [object Object] |
| [object Object] | 00005 | [object Object] |
| [object Object] | 00006 | [object Object] |
| [object Object] | 00007 | [object Object] |
| [object Object] | 00008 | [object Object] |
| [object Object] | 00009 | [object Object] |
| [object Object] | 00010 | [object Object] |

Showing 1 to 10 of 10,000 entries

Previous 1 2 3 4 5 … 1000 Next

# Getting the Parameters

- we use `tidy()` from the `broom` package.

```
data_bootstrap_param <-
  data_bootstrap_model %>%
  mutate(param = map(model, tidy)) %>%
  select(.id, param) %>%
  unnest()
```

# Getting the Parameters

Show 10 ▼ entries Search:

| .ID | TERM | ESTIMATE | STD.ERROR | STATISTIC | P.VALUE |
|---|---|---|---|---|---|
| 00001 | (Intercept) | 2.39012178930062 | 0.0371826299820924 | 64.2806006582035 | 2.50203776389026e-81 |
| 00001 | poly(x, 2, raw = TRUE)1 | 2.47668131375546 | 0.177500020427133 | 13.9531325562421 | 6.35046471404864e-25 |
| 00001 | poly(x, 2, raw = TRUE)2 | -3.22156084596636 | 0.168541598052562 | -19.1143366574801 | 1.15982273342521e-34 |
| 00002 | (Intercept) | 2.30386838623072 | 0.0336765578170282 | 68.4116351424071 | 6.76907087959256e-84 |
| 00002 | poly(x, 2, raw = TRUE)1 | 2.94477405501404 | 0.150398396856775 | 19.5798234326817 | 1.80855762724377e-35 |
| 00002 | poly(x, 2, raw = TRUE)2 | -3.69795537014388 | 0.137739046103648 | -26.8475459555687 | 1.07040065499199e-46 |
| 00003 | (Intercept) | 2.33640081941268 | 0.0281337136842963 | 83.0462997395477 | 6.31803692678483e-92 |
| 00003 | poly(x, 2, raw = TRUE)1 | 2.67719590581824 | 0.132351855559363 | 20.2278683173993 | 1.4216423473885e-36 |
| 00003 | poly(x, 2, raw = TRUE)2 | -3.4365747870883 | 0.138901442905594 | -24.7411021455264 | 1.08928740649482e-43 |
| 00004 | (Intercept) | 2.31968616054047 | 0.0336456689342167 | 68.9445695098489 | 3.23706157774023e-84 |

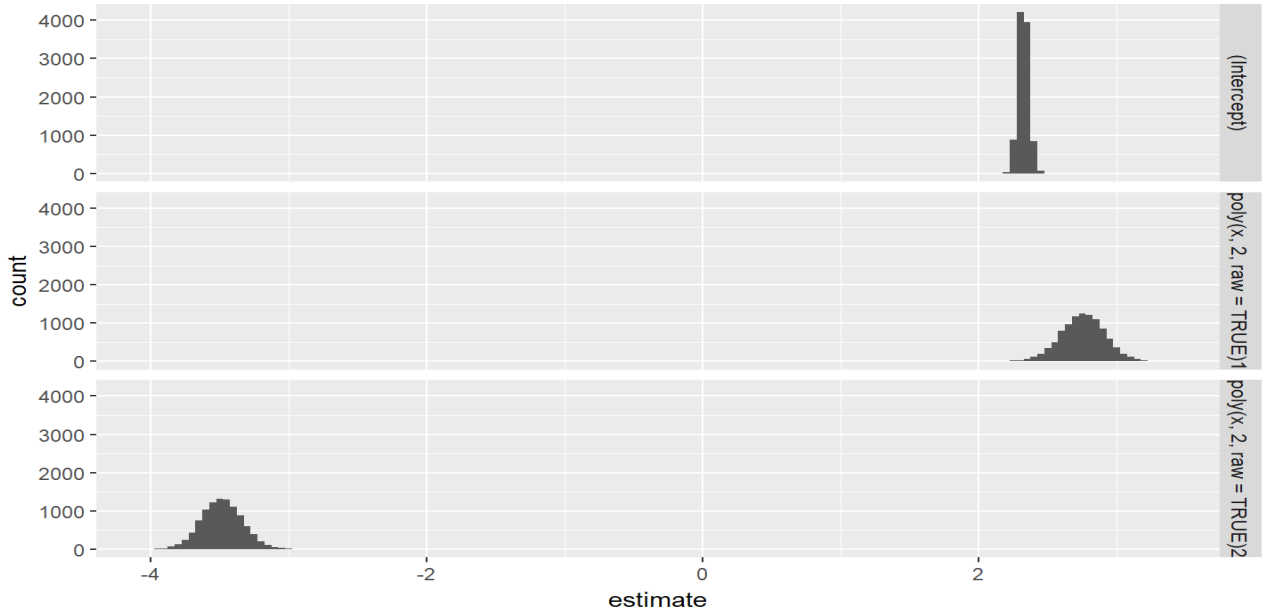
Showing 1 to 10 of 30,000 entries

Previous 1 2 3 4 5 … 3000 Next

# Distribution Plots

```
data_bootstrap_param %>%
  ggplot(aes(x = estimate)) +
  geom_histogram(binwidth = 0.05) +
  facet_grid(term ~ ., scales = "free_x")
```

# Distribution Plots



# Summarizing the Results

```
data_bootstrap_param_mean <-
  data_bootstrap_param %>%
  group_by(term) %>%
  summarize(estimate_mean = mean(estimate))
```

# Summarizing the Results

Show 10 ▼ entries Search:

| TERM | ESTIMATE_MEAN |
|---|---|
| (Intercept) | 2.32470949341213 |
| poly(x, 2, raw = TRUE)1 | 2.75245268430527 |
| poly(x, 2, raw = TRUE)2 | -3.48191932215691 |

Showing 1 to 3 of 3 entries Previous 1 Next

# Interquartile Range of Estimates

```
data_bootstrap_param %>%
  group_by(term) %>%
  summarize(
    q_25 = quantile(estimate, 0.25),
    median = median(estimate, 0.5),
    q_75 = quantile(estimate, 0.75)
  )
```

# Interquartile Range of Estimates

```
## # A tibble: 3 x 4
##   term                     q_25 median  q_75
##   <chr>                   <dbl>  <dbl> <dbl>
## 1 (Intercept)              2.30   2.32  2.35
## 2 poly(x, 2, raw = TRUE)1  2.65   2.76  2.86
## 3 poly(x, 2, raw = TRUE)2 -3.58  -3.48 -3.38
```

# Confidence Intervals

```
data_bootstrap_param %>%
  group_by(term) %>%
  summarize(
    lower_95 = quantile(estimate, 0.025),
    upper_95 = quantile(estimate, 0.975)
  )
```

# Confidence Intervals

```
## # A tibble: 3 x 3
##   term                    lower_95 upper_95
##   <chr>                      <dbl>    <dbl>
## 1 (Intercept)                 2.25     2.40
## 2 poly(x, 2, raw = TRUE)1     2.42     3.06
## 3 poly(x, 2, raw = TRUE)2    -3.77    -3.18
```

# Add Model Predictions to Plot of Original Data

```
library(tidyr)
library(modelr)
grid <-
  data %>%
  expand(x = seq_range(x, 20))

boot_pred <-
  data_bootstrap_model %>%
  transmute(
    .id,
    data = map2(list(grid), model, add_predictions, var = "y")
  ) %>%
  unnest()
```

# Add Model Predictions to Plot of Original Data

Show 10 ▼ entries Search:

| X |
|---|
| 0.000450939871370792 |
| 0.0514784060712708 |
| 0.102505872271171 |
| 0.153533338471071 |
| 0.204560804670971 |
| 0.255588270870871 |
| 0.306615737070771 |
| 0.357643203270671 |
| 0.408670669470571 |
| 0.459698135670471 |

Showing 1 to 10 of 20 entries Previous 1 2 Next

# Add Model Predictions to Plot of Original Data

Show 10 ▼ entries Search:

| .ID | X | Y |
|---|---|---|
| 00001 | 0.000450939871370792 | 2.39123796855969 |
| 00001 | 0.0514784060712708 | 2.50908017473733 |
| 00001 | 0.102505872271171 | 2.61014576579064 |
| 00001 | 0.153533338471071 | 2.69443474171962 |
| 00001 | 0.204560804670971 | 2.76194710252427 |
| 00001 | 0.255588270870871 | 2.81268284820458 |
| 00001 | 0.306615737070771 | 2.84664197876057 |
| 00001 | 0.357643203270671 | 2.86382449419222 |
| 00001 | 0.408670669470571 | 2.86423039449955 |
| .ID | X | Y |
| 00001 | 0.459698135670471 | 2.84785967968254 |

Showing 1 to 10 of 200,000 entries

Previous 1 2 3 4 5 … 20000 Next

# Plot Data and Model Estimates

```
ggplot(data = data, mapping = aes(x = x, y = y)) +
  geom_line(
    data = boot_pred %>% filter(as.numeric(.id) < 3000),
    aes(group = .id),
    color = "blue",
    alpha = 0.002
  ) +
  stat_function(fun = truth, color = "black", linetype = "dashed", size = 1) +
  geom_point(data = data, alpha = 0.5)
```

# Plot Data and Model Estimates