# Resampling Methods: Bootstrap Method

## Section 5.2

Cathy Poliak, Ph.D.
cpoliak@central.uh.edu

Department of Mathematics
University of Houston

# Resampling Methods

- **Resampling methods** involve repeatedly drawing samples from a training set and refitting a model of interest on each sample in order to obtain additional information about the fitted model.
- Could potentially be computationally expensive, because they involve fitting the same statistical method multiple times using different subsets of the training data.
- Two most commonly used resampling methods:
  - Cross-validation - can be used to estimate the test error associated with a given statistical learning method in order to evaluate its performance.
  - Bootstrap - can be used to provide a measure of accuracy of a parameter estimate or of a given statistical learning method.

# The Bootstrap Methods

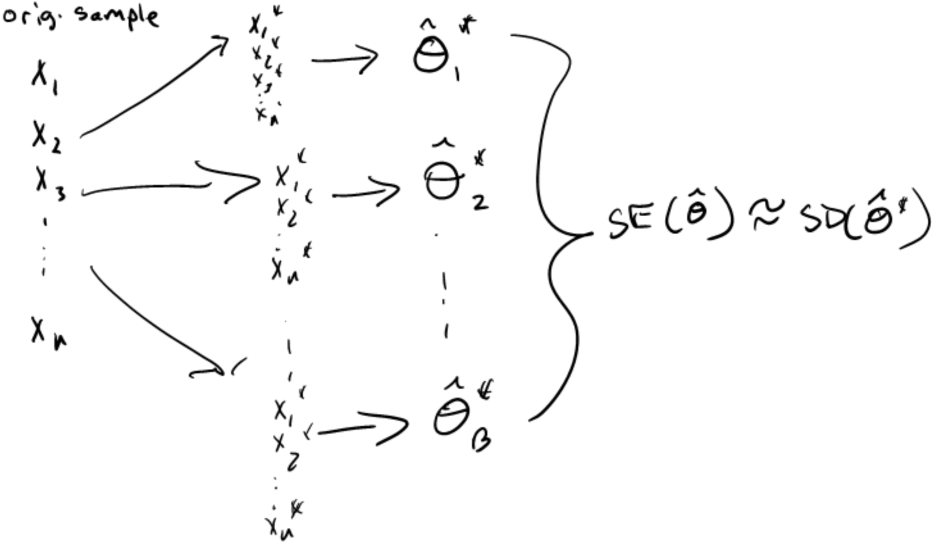
- The **bootstrap** is a widely applicable and extremely powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method.
- The machine learning techniques that use the bootstrap is the *tree*-based models: Bagging, Random Forest, ect.
- The power of the bootstrap lies in the fact that it can be easily applied to a wide range of statistical learning methods, including some for which a measure of variability is otherwise difficult to obtain and is not automatically output by statistical software.

# Idea of The Bootstrap

- Resample from the original data - either directly or via a fitted model - to create data sets, from which the variability of the quantities of interest can be assessed with out long-winded and error-prone analytical calculations.
- This approach involves repeating the original data analysis procedure with many replicate sets of data.
- The central goal is to obtain reliable standard errors, confidence intervals, and other measures of uncertainty for a wide range of problems.
- This approach can be applied in simple problems to check the adequacy of standard measures of uncertainty, to relax assumptions, and to give quick approximate solutions.
- The basic idea of bootstrap is to make inference about an estimate (such as the sample mean or sample coefficients $\hat{\beta}_j$) for a population parameter $\theta$ (such as the population mean or coefficients $\beta_j$) on sample data.

# Steps for Bootstrap

1. Get a sample from a population with sample size *n*.
2. Draw a sample from the original sample data **with replacement** with size *n*, and replicate *B* times, each re-sampled sample is a called a **Bootstrap Sample**, and there will be totally *B* Bootstrap Samples.
3. Evaluate the statistic of $\theta$ for each Bootstrap Sample, and there will be a total of *B* estimates of $\theta$.
4. Construct a **sampling distribution** with these *B* Bootstrap statistics and use it to make further statistical inference, such as:
   - Estimating the standard error of the statistic for $\theta$.
   - Obtaining a confidence interval for $\theta$.



B times

$\hat{\theta}_i^*$ is the statistic calculated from each resample

# The `boot` Function in `R`

Performing a bootstrap analysis in `R` entails only two steps:

1. Create a function that computes that statistic of interest.
2. Use the `boot()` function, which is part of the `boot` library, to perform the bootstrap by repeatedly sampling observations form the data set with replacement.

# Example

A thermostat used in an electrical devise is to be checked for the accuracy of its design setting of 200°F. Ten thermostats were tested to determine their actual settings, resulting in the following data:

202.2   203.4   200.5   202.5   206.3   198.0   203.7   200.8   201.3   199.0

We wish to estimate the true median value of this thermostat.

function of stat

```
> median.fun = function(dat, idx) median(dat[idx],na.rm = TRUE)
> boot.out.median = boot(data = temp, statistic = median.fun, R = 100)
> boot.out.median
```

B

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = temp, statistic = median.fun, R = 10000)


Bootstrap Statistics :
    original   bias    std. error
t1*   201.75 0.002145   0.8643586
```

SD(of 100 medians)

median of temp

201.75 + .002145 = mean of the medians

# The Ideal and Reality in Statistics World

Ideal World

- A standard error of our sample mean can be easily estimated and can find the estimated standard error.
- We assume we know or can estimate about the estimator's population.

Real World

- Hard to know the information about the population or it's distribution.
- The standard error of an estimate is hard to evaluate in general.

When the assumptions are violated, or when no formula exists for estimating standard errors, bootstrap is the powerful choice.

# Why Does the Simulation of the Bootstrap Work?

Let $X_1, X_2, \ldots, x_n$ be a random sample from a population $P$ with cumulative distribution function $F$. And let $M = g(X_1, X_2, \ldots, X_n)$ be our statistic for the parameter of interest. What we desire to is to know $Var(M)$. We resample $B$ times.

By the **Law of Large Numbers**:

$$\bar{m} = \frac{1}{B}\sum_{j=1}^{B} M_j \xrightarrow{P} E(M), \text{ as } B \to \infty$$

Where $E(M)$ is the true mean of the statistic $M$.

In addition, the sample variance of these $B$ statistics converges to the true variance of statistic $M$ as $B \to \infty$.

$$s^2 = \frac{\sum_{j=1}^{B}(M_j - \bar{m})^2}{B-1} \xrightarrow{P} Var(M), \text{ as } B \to \infty$$

Where $Var(M)$ is the true variance of the statistic $M$.

# Empirical Cumulative Distribution Function

- The ECDF (empirical cumulative distribution function) $F_n$ is a step function with jumps $i/n$ at observation values, where $i$ is the number of tied observations at that value.
- Missing values are ignored.
- For observations $x = (x_1, x_2, \ldots, x_n)$, $F_n$ is the fraction of observations less or equal to t, i.e.,

$$F_n(t) = \frac{\text{number of elements in the sample} \leq t}{n}$$

# Small Example

| | | | | | Samples | | | | | | Medians |
|---|---|---|---|---|---|---|---|---|---|---|---|
| x1* | 206.3 | 200.5 | 200.8 | 202.2 | 200.8 | 201.3 | 202.5 | 200.5 | 199 | 200.5 | 200.8 |
| x2* | 203.4 | 202.5 | 203.4 | 203.7 | 202.5 | 202.5 | 198 | 198 | 203.7 | 198 | 202.5 |
| x3* | 198 | 200.5 | 200.8 | 201.3 | 203.4 | 202.5 | 199 | 206.3 | 198 | 202.2 | 201.05 |
| x4* | 206.3 | 200.5 | 206.3 | 203.7 | 202.5 | 206.3 | 201.3 | 198 | 203.7 | 203.7 | 203.7 |
| x5* | 203.4 | 200.8 | 201.3 | 201.3 | 200.5 | 203.4 | 206.3 | 200.5 | 202.5 | 200.5 | 201.3 |
| x6* | 199 | 202.5 | 203.7 | 198 | 200.8 | 203.4 | 201.3 | 202.2 | 200.8 | 198 | 201.05 |
| x7* | 198 | 206.3 | 203.7 | 206.3 | 202.2 | 200.8 | 203.4 | 200.5 | 203.7 | 198 | 202.8 |
| x8* | 201.3 | 203.4 | 206.3 | 202.2 | 203.4 | 202.5 | 203.4 | 203.7 | 200.8 | 203.4 | 203.4 |
| x9* | 203.7 | 200.5 | 203.7 | 206.3 | 203.4 | 199 | 202.2 | 202.5 | 200.8 | 203.4 | 202.95 |
| x10* | 202.5 | 199 | 202.5 | 201.3 | 206.3 | 200.8 | 203.4 | 200.8 | 199 | 201.3 | 201.3 |

$B = 10$

$\hat{\Theta} = \text{median} = 201.75$

$\text{mean}(200.8, 202.5, 201.5, 203.7, 201.3, 201.05, 202.8, 203.4, 202.95, 201.3) = 202.085 = \hat{\Theta}^*$

$\text{bias} = \hat{\Theta}^* - \hat{\Theta} = 202.085 - 201.75 = 0.335$

# Using the boot Function

```
(boot2 = boot(data = temp,statistic = median.fun, R = 10))
Bootstrap Statistics :
    original  bias    std. error
t1*   201.75    0.12   0.9905666
sort(boot2$t)
[1] 200.50 200.90 201.05 201.05 201.50 202.35 202.50 202.50 202.95 203.40
```

CI for $\mu$: $\bar{X} \pm t_{\alpha/2, n-1} SE(\bar{x})$ $\frac{S}{\sqrt{n}} = SE(\bar{x})$

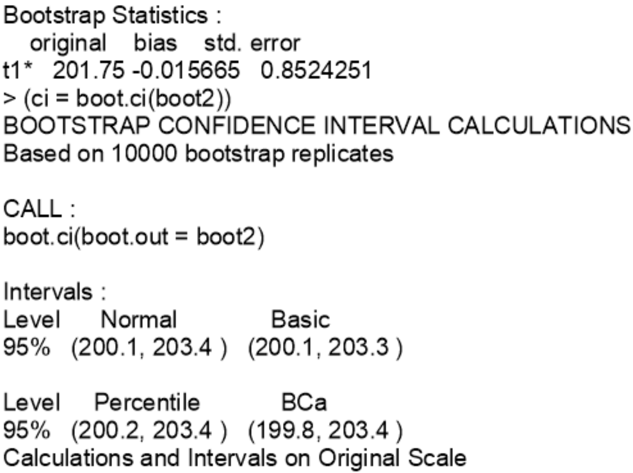
$\bar{X} \pm z_{\alpha/2} SE(\bar{x})$

CI for $\hat{\theta}$: $\hat{\theta} - \text{bias} \pm z_{\alpha/2} SE(\bar{x})$ Normal Approx.

Percentile CI: LL = quantile at 2.5%

UL = quantile at 97.5%

Basic Bootstrap: $(2\hat{\theta} - \hat{\theta}^*_{0.975}, 2\hat{\theta} - \hat{\theta}^*_{0.025})$



```
Bootstrap Statistics :
    original    bias    std. error
t1*   201.75 -0.015665   0.8524251
> (ci = boot.ci(boot2))
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 10000 bootstrap replicates

CALL :
boot.ci(boot.out = boot2)

Intervals :
Level      Normal              Basic
95%   (200.1, 203.4 )   (200.1, 203.3 )

Level     Percentile            BCa
95%   (200.2, 203.4 )   (199.8, 203.4 )
Calculations and Intervals on Original Scale
```

# Confidence Intervals

```
(ci.median = boot.ci(boot.out.median))
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 10000 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out.median)

Intervals :
Level      Normal              Basic
95%   (200.1, 203.4 )   (200.1, 203.6 )

Level     Percentile            BCa
95%   (199.9, 203.4 )   (199.8, 203.4 )
Calculations and Intervals on Original Scale
```

For more information see: `https://www.r-bloggers.com/2019/09/understanding-bootstrap-confidence-interval-output-from-the-r-boot-packag`

# Examples

This lecture will illustrate the use of the bootstrap in two examples

- We illustrate the bootstrap on a toy example in which we wish to determine the best investment allocation under a simple model.

- An example involving estimating the accuracy of the linear regression model.

Lab questions are in red. These are not multiple choice questions. Just type in your answers in blackboard.

# Example 1

Suppose that we wish to invest a fixed sum of money in two financial assets that yield returns of $X$ and $Y$ , respectively, where $X$ and $Y$ are random quantities. We will invest a fraction $\alpha$ of our money in $X$, and will invest the remaining $1 - \alpha$ in $Y$ . Since there is variability associated with the returns on these two assets, we wish to choose $\alpha$ to minimize the total risk, or variance, of our investment. In other words, we want to minimize $Var(\alpha X + (1 - \alpha)Y)$. One can show that the value that minimizes the risk is given by

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

Where, $\sigma_X^2 = Var(X)$, $\sigma_Y^2 = Var(Y)$, and $\sigma_{XY} = Cov(X, Y)$.

# The Estimate

In reality the population variances and covariance is unknown so we have to use estimates, using a data set that contains past measurements for $X$ and $Y$. We can then estimate the value of $\alpha$ that minimizes the variance of our investment using

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

# Using R

1. Install the ISLR, we will be using the Portfolio data set.

   ```
   install.packages("ISLR")
   library(ISLR)
   ```

2. Create the function which takes as input $(X, Y)$ data as well as a vector indicating which observations should be used to estimate $\alpha$. The function then outputs the estimate for $\alpha$ based on the selected observations. The function is as follows:

   ```
   alpha.fn = function(data,index){
   X = data$X[index]
   Y = data$Y[index]
   return((var(Y) - cov(X,Y))/(var(X) + var(Y) - 2*cov(X,Y)))
   }
   ```

   Be careful about capitalization and lower case in these variables.

3. This function *returns* or outputs an estimate for $\alpha$ based on a applying the formula to the observations indexed by the argument index. For instance, the following command tells R to estimate $\alpha$ using all 100 observations.

# Lab Questions

Type and run in R

```
alpha.fn(Portfolio,1:100)
```

Question 1: From this command, give an estimate of $\alpha$.

The following command uses the sample function to randomly select 100 observations from the range 1 to 100, with replacement. This is equivalent to constructing a new bootstrap data set and recomputing $\alpha$ based on the new data set.

```
set.seed(10)
alpha.fn(Portfolio,sample(100,100,replace = TRUE))
```

Question 2: From this command, give an estimate of $\alpha$.

# Using the boot Function

We can implement a bootstrap analysis by performing this command many times, recording all of the corresponding estimate for $\alpha$, and computing the resulting standard deviation. However, the boot() function automates this approach. Below, is the function to produce $R = 1000$ bootstrap estimates for $\alpha$.

```
install.packages("boot")
library(boot)
alpha.boot = boot(Portfolio,alpha.fn,R = 1000)
alpha.boot
mean(alpha.boot$t)
```

Question 3: What is the original estimate of $\alpha$? What is $SE(\hat{\alpha})$? What is the mean of the bootstrap estimates of $\alpha$?

Question 4: Do the command sd(alpha.boot$t) compare that to the standard error from the boot() function. What does the standard error mean?

# Example 2

The bootstrap approach can be used to assess the variability of the coefficient estimates and predictions from a statistical learning method. We we use the bootstrap approach in order to assess the variability of the estimates for $\beta_0$ and $\beta_1$, the intercept and slope terms for the linear regression model that uses `horsepower` to predict `mpg` in the `Auto` data set.

$$mpg = \underset{\uparrow}{\beta_0} + \beta_1 \times hp + \varepsilon$$

find $\hat{\beta}_0$ and $\hat{\beta}_1$

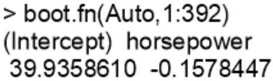
# Input in R

Create a function called boot.fun() which takes into the Auto data set as well as a set of indices for the observations, and returns the intercept and slope estimates for the linear regression model.

```
boot.fn = function(data,index)
return(coef(lm(mpg~horsepower,data = data,subset = index)))
boot.fn(Auto,1:392)
```

*Note*: We do not need { and } at the beginning and end of the defined function because it is only one line.

Question 5: Give the estimates of the intercept and slope.



```
> boot.fn(Auto,1:392)
(Intercept)  horsepower
 39.9358610  -0.1578447
```

# Lab Questions

- The boot.fn() function can also be used in order to create bootstrap estimates for the intercept on slope terms by randomly sampling from among the observations.
  Question 6: Run the following command twice give two of the bootstrap estimates for the intercept and the slope.

  ```
  boot.fn(Auto,sample(392,392,replace = TRUE))
  ```

- We can use the boot() function to compute the standard errors of 1000 bootstrap estimates for the intercept and slope terms.

  ```
  boot.out = boot(Auto,boot.fn,1000)
  boot.out
  ```

  Question 7: What is the standard errors of the bootstrap estimates for the intercept and the slope?

- The command below gives the estimates of the original data using the lm() function.

  ```
  summary(lm(mpg~horsepower,data = Auto))$coef
  ```

  Question 8: Compare the standard errors given in the summary from the command above and the bootstrap function. Describe why there is a difference in the standard errors.