Stat 4620: Day 14

(3/14)

1 Sec. 4.9: Bootstrap

If the underlying distribution f is known, then $V(\overline{X})$ can be calculated using the formula $V(\overline{X}) = \sigma_f^2/n$ or estimated by generating Monte Carlo samples of size n from f. If f is unknown, we cannot generate Monte Carlo samples from f, but what if we generate Monte Carlo samples from $\hat{f}$? This is the idea behind the bootstrap procedure. Properly implemented, it can give good estimates of standard error, confidence interval, or p-values.

Example:

Let $x_1, \ldots, x_n$ be a random sample from an unknown distribution. Let $\overline{x}$ be the estimate of the population mean μ. Provide a standard error for $\overline{x}$.

Solution. Bootstrap plan:

1. Let $\hat{F}_n$ be the empirical cdf, i.e. the cdf whose point mass distribution is

x	x_1	x_2	$\cdots$	x_n
$p(x)$	$1/n$	$1/n$	$\cdots$	$1/n$

2. Draw $(x_1^*, x_2^*, \ldots, x_n^*)$ with replacement from $\hat{F}_n$. Calculate

$$\overline{x}^* = \frac{x_1^* + \cdots + x_n^*}{n} \tag{1}$$

3. Repeat B times, so we get: $\overline{x}^*_{(1)}, \overline{x}^*_{(2)}, \ldots, \overline{x}^*_{(B)}$

4. A bootstrap standard error for the sample mean:

$$SE(\overline{x}) = \sqrt{\frac{\sum_{i=1}^{B}\left(\overline{x}^*_{(i)} - \overline{\overline{x}}^*\right)}{B-1}}$$

A bootstrap 95% confidence interval for the sample mean is

$$\left(\overline{y}^*_{(.025B)}, \overline{y}^*_{(.975B)}\right)$$

where $\overline{y}^*_{(1)} \leq \overline{y}^*_{(2)} \leq \ldots \leq \overline{y}^*_{(B)}$ are the ordered values of $\overline{x}^*_{(1)}, \overline{x}^*_{(2)}, \ldots, \overline{x}^*_{(B)}$.

Example 4.9.1 90% confidence interval for population mean μ

```
> w<-c(131.7, 182.7, 73.3, 10.7, 150.4, 42.3, 22.2, 17.9, 264.0, 154.4,
       4.3, 265.6, 61.9, 10.8, 48.8, 22.5, 8.8, 150.6, 103.0, 85.9)
> index<-sample(1:20, size=20, replace=T)
> index
 [1]  3  1 13 14 11 19 16 12 19 18 11  9 13  6 18 11 11  7  3  5
> wb<-w[index]
> wb
 [1]  73.3 131.7  61.9  10.8   4.3 103.0  22.5 265.6 103.0 150.6   4.3 264.0
[13]  61.9  42.3 150.6   4.3   4.3  22.2  73.3 150.4
> cbind(w, index, wb)
          w index    wb
 [1,] 131.7     3  73.3
 [2,] 182.7     1 131.7
 [3,]  73.3    13  61.9
 [4,]  10.7    14  10.8
 [5,] 150.4    11   4.3
 [6,]  42.3    19 103.0
 [7,]  22.2    16  22.5
 [8,]  17.9    12 265.6
 [9,] 264.0    19 103.0
[10,] 154.4    18 150.6
[11,]   4.3    11   4.3
[12,] 265.6     9 264.0
[13,]  61.9    13  61.9
[14,]  10.8     6  42.3
[15,]  48.8    18 150.6
[16,]  22.5    11   4.3
[17,]   8.8    11   4.3
[18,] 150.6     7  22.2
[19,] 103.0     3  73.3
[20,]  85.9     5 150.4

> # Bootstrap CI: Repeat B=3000 times
> B=3000
> meanstore<-rep(0,B)   # Initialize storage
> for(b in 1:B){
+   index<-sample(1:20, size=20, replace=T)
+   wb<-w[index]
+   meanstore[b]<-mean(wb)
+ }
> length(meanstore)
[1] 3000
> head(meanstore, 20)         # Print first 20
```

```
 [1]  69.760  79.725 103.170  53.265 100.715  64.410  90.170  56.235  84.150
[10]  66.730 103.090 103.315 105.810  98.795 102.420 111.720  65.770  49.380
[19]  92.620 119.655
> meanstore<-sort(meanstore)  # Sort
> head(meanstore, 20)         # Print first 20
 [1] 33.770 41.440 44.210 45.115 45.265 45.305 45.605 45.975 46.535 46.670
[11] 47.440 47.680 48.080 48.250 48.735 48.785 49.340 49.380 49.515 49.750
> tail(meanstore, 20)         # Print last 20
 [1] 138.460 138.460 138.850 138.915 139.170 139.540 139.605 139.930 141.230
[10] 141.805 141.895 143.025 143.825 145.395 146.900 148.655 148.980 150.120
[19] 150.120 150.350
> lower90<-meanstore[.05*B]
> lower90
[1] 62.775
> upper90<-meanstore[.95*B]
> upper90
[1] 121.6
>
> # Compare with classical normal-based 90% cI
> mean(w)-1.645*sd(w)/sqrt(20)
[1] 60.28329
> mean(w)+1.645*sd(w)/sqrt(20)
[1] 120.8967
```

■

1.0.1 Bootstrap test of hypothesis

Example 4.9.2

Let $\mathbf{x} = (x_1, \dots, x_{15})$ and $\mathbf{x} = (x_1, \dots, x_{15})$ be samples from a distribution with cdf $F(\cdot)$ with means μ_x and μ_y, respectively. From textbook,

$X : 94.2, 111.3, 90.0, 99.7, 116.8, 92.2, 166.0, 95.7, 109.3, 106.0, 111.7, 111.9, 111.6, 146.4, 103.9$

$Y : 125.5, 107.1, 67.9, 98.2, 128.6, 123.5, 116.5, 143.2, 120.3, 118.6, 105.0, 111.8, 129.3, 130.8, 139.8$

We want to test

$$H_0 : \mu_x = \mu_y \text{ vs } H_1 : \mu_x < \mu_y$$

1. Approach 1: Welch two-sample t-test

```
> x<-c(94.2, 111.3, 90.0, 99.7, 116.8, 92.2, 166.0, 95.7, 109.3, 106.0, 111.7,
       111.9, 111.6, 146.4, 103.9)
> y<-c(125.5, 107.1, 67.9, 98.2, 128.6, 123.5, 116.5, 143.2, 120.3, 118.6, 105.0,
       111.8, 129.3, 130.8, 139.8)
> cbind(x,y)
          x     y
 [1,]  94.2 125.5
```

```
 [2,] 111.3 107.1
 [3,]  90.0  67.9
 [4,]  99.7  98.2
 [5,] 116.8 128.6
 [6,]  92.2 123.5
 [7,] 166.0 116.5
 [8,]  95.7 143.2
 [9,] 109.3 120.3
[10,] 106.0 118.6
[11,] 111.7 105.0
[12,] 111.9 111.8
[13,] 111.6 129.3
[14,] 146.4 130.8
[15,] 103.9 139.8

> mean(x)
[1] 111.1133
> mean(y)
[1] 117.74

> t.test(x,y, alternative='less')

Welch Two Sample t-test

data:  x and y
t = -0.92983, df = 27.759, p-value = 0.1802
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
     -Inf 5.500446
sample estimates:
mean of x mean of y
 111.1133  117.7400
```

2. Approach 2: Bootstrap

   (a) Under H_0, the combined data $\mathbf{z} = (\mathbf{x}, \mathbf{y})$ is a random sample of size 30 from $F(\cdot)$ with mean μ.

   (b) Draw two samples of size 15 from $\mathbf{z}$: $(x_1^*, \ldots, x_{15}^*)$ and $(y_1^*, \ldots, y_{15}^*)$. Calculate

   $$v^* = \overline{y}^* - \overline{x}^*$$

   (c) Repeat $B = 3000$ times, get

   $$v_1^*, v_2^*, \ldots, v_{3000}^*$$

   (d) What percentage of times are the v^* greater than $v = 117.74 - 111.11 = 6.63$? This is the bootstrap p-value.

```
> # Bootstrap in R
>
> z<-c(x,y)          # Combine x and y samples
> z
 [1]  94.2 111.3  90.0  99.7 116.8  92.2 166.0  95.7 109.3 106.0 111.7 111.9
[13] 111.6 146.4 103.9 125.5 107.1  67.9  98.2 128.6 123.5 116.5 143.2 120.3
[25] 118.6 105.0 111.8 129.3 130.8 139.8
> xindex<-sample(1:30, size=15, replace=T)
> xindex
 [1]  8  6 13 17 26 18 24  9  7 24 28 27 24 28 20
> xb<-z[xindex]
> xb
 [1]  95.7  92.2 111.6 107.1 105.0  67.9 120.3 109.3 166.0 120.3 129.3 111.8
[13] 120.3 129.3 128.6
> yindex<-sample(1:30, size=15, replace=T)
> yindex
 [1] 23 17 10 14 22  9  5 25 27 14 12 17 13  9 28
> yb<-z[yindex]
> vb<-mean(yb)-mean(xb)
>
> # Repeat B=3000 times and store in vstore
> B<-3000
> vstore<-rep(0,B)   # Initialize storage for v
> for(b in 1:B){
+ xindex<-sample(1:30, size=15, replace=T)
+ xb<-z[xindex]
+ yindex<-sample(1:30, size=15, replace=T)
+ yb<-z[yindex]
+ vstore[b]<-mean(yb)-mean(xb)
+ }
> length(vstore)
[1] 3000
> head(vstore, 20)
 [1]   1.9666667  -6.8600000  -2.0466667   0.5666667 -12.7800000   2.7400000
 [7]  -0.2666667  -5.8933333   5.4800000   7.2600000   0.0400000   1.4666667
[13]  10.3866667   5.1266667  -4.4133333  10.2533333  -1.4000000 -15.6466667
[19]   4.0200000  -6.6933333
> mean(vstore> 117.7400-111.1133)
[1] 0.1713333
```