STAT 400 Data Analysis Oct. 31, 2005

Today: The Bootstrap

Introduction to the Bootstrap

The Bootstrap:

Suppose you have sample data and a population parameter for which the distribution theory of the desired parameter estimator is unknown. (It may be unknown because the population class of distributions is unknown, or because the theory of this estimator is intractable, or because you just don't know the theory!) The choice of estimator is assumed to be known – the issue is that you want to know the variability of your estimator, since that tells you about the precision of your estimate. The bootstrap gives a way to find this precision, based only on the sample data you have used to come up with the estimate. The amazing thing is that it does this without any assumptions about the population distribution. Moreover, the approach can be used in very complicated situations, as we shall see.

The operation of the basic bootstrap (the only version described here) is simple: You just compute the estimate for a large number of resamples of your original sample, and observe the variability (by computing the SD of the resample estimates). A "resample" here is a sample of the same size as the original sample, selected from the original sample at random and with replacement.

The justification for this procedure is this:

1. The ECDF is an estimate of the CDF.
2. Resampling is the same as using the inverse-CDF-transform method of simulating a distribution for a given CDF. The "given CDF" in this case is the ECDF.
3. The resample estimates thus reflect approx. the same variability as if the CDF itself had been sampled.

The inverse transform method of generating a random sample from a population whose CDF is F( ) is justified by the Probability Integral Transform Theorem. It says that if X is a RV having CDF F( ), then F(X) has a U(0,1) Distribution. The proof is done by computing P(F(X)<t) = P(X<inverse fcn of F at t) = F(inverse function of F at t) =t which is the CDF of the U(0,1).

A demo of this can be had using the following program, since it shows that the bootstrap can estimate that the sample mean of 20 values from a distribution has a SD that turns out to be very close to s/sqrt(n), and yet it does not use this formula.

```
my.boot
function (x,m=100,single=T)
{
```

```
        # This program uses bootstrap to estimate the
        # sd of the sample mean of data in vector x
        n=length(x)
        a=matrix(nrow=m,ncol=1)
        for (i in 1:m) {a[i]=mean(sample(x,n,replace=T))}
        s.form=sd(x)/(n^.5)  #the usual square root formula
        s.boot=sd(a) # the comparable bootstrap estimate
        # single = T if this program is used for a single
        # sample, otherwise just pass ests to other prog.
        if (single==T) {print("SEM from s/sqrt(n)")
                        print(s.form)
                        print("SEM from bootstrap")
                        print(s.boot)}
        invisible(list(s.boot,s.form))
}
```

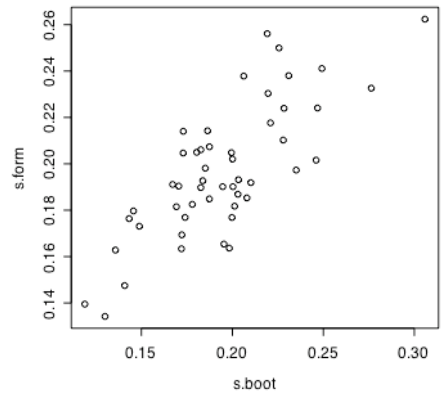
Using this program on a random sample of size 25 from N(0,1), so we know the sample mean has sd=1/sqrt(25) = .200

```
> my.boot(rnorm(25))
[1] "SEM from s/sqrt(n)"
[1] 0.2181372
[1] "SEM from bootstrap"
[1] 0.2040285
```

And if we were to run this program 50 times, and plot the two estimates each time in a scatterplot ....

```
> my.boot.test
function (n=25,m=50,k=50)
{       #n is sample size, m is boot reps, k redo for new sample
        s.boot=matrix(nrow=k,ncol=1)
        s.form=matrix(nrow=k,ncol=1)
        for (j in 1:k) {a=rnorm(n)
                s.boot[j]=my.boot(a,m,single=F)[[1]]
                s.form[j]=my.boot(a,m,single=F)[[2]]}
        plot(s.boot,s.form)
        mean.s.boot=mean(s.boot)
        mean.s.form=mean(s.form)
        print(mean.s.boot)
        print(mean.s.form)
}
```

> my.boot.test(n=25)



[1] 0.1948124 # this is the mean of 50 ests of sd(s.mean) from formula
[1] 0.1968160 #this is the mean of 50 ests of sd(s.mean) from bootstrap
The estimates are always close to each other and to the true value 0.2.
In a situation where the theory is unknown, such as the variability of the sample 90th percentile (from an unknown population, perhaps), the following bootstrap procedure will provide an estimate of the SD of the sample 90th percentile.

```
> my.boot.90.test
function (n=25,m=100,l=10,k=250)
{
        a=matrix(ncol=1,nrow=k)
        s.boot.90=matrix(ncol=1,nrow=l)
        for (i in 1:k) {a[i]=sort(rnorm(n))[round(.9*n)]}
        s.90=sd(a)
        print("True SD of 90th sample percentile")
        print(s.90)
        print("Bootstrap estimates of SD of 90th sample percentile")
        for (j in 1:l){smpl=rnorm(n)
        s.boot.90[i]=my.boot.90(smpl,m,single=F)
        print(s.boot.90[i])}

my.boot.90
function (x,m=100,single=T)
{
        # This program uses bootstrap to estimate the
        # sd of the 90th percentile of data in vector x
```

```
        a=matrix(ncol=1,nrow=m)
        n=length(x)
        for (i in 1:m) {a[i]=sort(sample(x,n,replace=T))[round(.9*n)]}
        s.boot.90=sd(a) # the comparable bootstrap estimate
        # single = T if this program is used for a single
        # sample, otherwise just pass ests to other prog.
        if (single==T) {print("est 90th pctl from bootstrap")
                                                print(s.boot.90)}
        invisible(as.numeric(s.boot.90))
}
```

The program output is shown below:

```
> my.boot.90.test()
[1] "True SD of 90th sample percentile"
[1] 0.3124393
[1] "Bootstrap estimates of SD of 90th sample percentile"
[1] 0.1303692
[1] 0.2285510
[1] 0.1430806
[1] 0.3684711
[1] 0.3781018
[1] 0.2413628
[1] 0.2353158
[1] 0.4517218
[1] 0.2928408
[1] 0.2079172
```

Next Day we will examine bootstrap of a stepwise regression procedure.

The following is from 2003 – might be useful ...

1. Does the Bootstrap produce correct results for the SD of the sample mean?

We computed the SD of the sample mean, based only on one particular random sample of size n=20. We showed that this method gave a value similar to the theoretical value and the usual estimator of it. ($\sigma/\sqrt{n}$ and $s/\sqrt{n}$). These latter formulas did not need to be known to do this.

For example, for a particular sample of N(0,1) data, we found:

Standard deviation of sample, s = 1.14

$s/\sqrt{20} =$ 0.26

bootSD of sample mean = 0.24

Since in this case we know Population SD = 1,
Population SD of mean = $1/\sqrt{20} = 0.22$

So the bootstrap has performed well here.

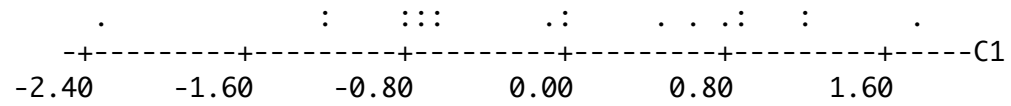
2. Does the Bootstrap produce correct results of the SD of the 90th percentile?

We compute the SD of the sample 90th percentile, based only on one random sample.

The theoretical formula is not widely known and so this was an example of how the bootstrap could estimate this SD in such a situation. Today we will check this bootstrap approach by simulating the correct SD since we will start with a known distribution.

For example:

Here is a sample n=20 from N(0,1):



The 90th percentile could be estimated by the 18th order statistic, in this case 1.22. But how precise is this estimate? Bootstrap it!

BootSD = 0.27 so we might say our estimate of the 90th percentile is 1.22±.27 (mean ± SD)

But is this right? Lets simulate some samples from N(0,1), estimate the 90th percentile from each sample (18th order statistic) and compute the SD when we have enough of them.

simSD90 0.35

So our estimate of .27 was a bit low. But of course, the .27 was based on a sample of n=20, and the .35 is a population-based SD, so we don't expect perfection. The question is, is this bootstrap method seriously biased in this case?

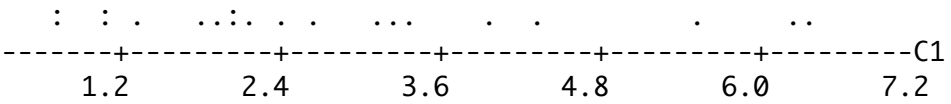
We can investigate this with simulation. I did the whole experiment (generate 20 N(0,1), bootstrap the 90th percentile estimate and compute its SD) 15 more times. Here are the 15 values of the bootstrap estimates of SD of the 90th percentile:
.36, .35, .34, .27. .24, .14, .40, .36, .49, .46, .16, .40, .36, .38, .34

and the average of these is .34 – pretty close to .35. If the method is biased, the bias would be modest in this case. But there is no reason to expect unbiasedness – it would depend on the situation.

Does the good performance of the bootstrap depend on the N(0,1) sample we used?

These demos used "data" that was generated from a N(0,1) distribution, but this information was not used. Someone asked if the performance would be as good for some less symmetrical distribution, so I will show how it works with Gamma (3,1)

Here is a Gamma (3,1), n=20, which is strongly right skewed:



In this case the bootstrap SD of the 90th percentile estimate is

bootSD 1.07

The true SD (using the n=known population Gamma(3,1) of the 90th percentile is

simSD90 0.82

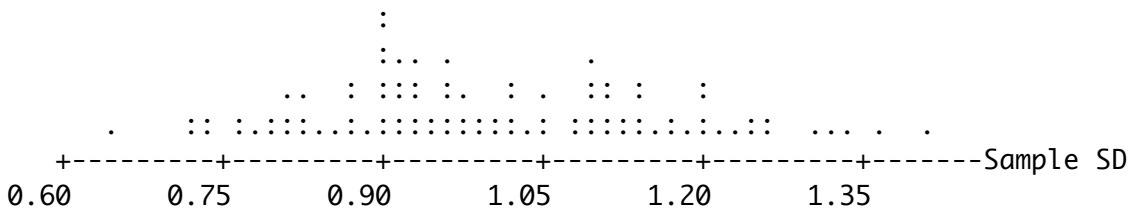
Our one sample produced a pretty good estimate of the SD of the 90th percentile. To see if it is usually this good, you can do the simulation using the "boot2" program 15 times, starting with a new Gamma sample each time in column 1.

Is the variability of the bootstrap estimate of SD of a statistic too variable to be useful?

We have shown that the bootstrap does provide variability estimates which are almost unbiased, but they do vary quite a bit in samples of size 20. However, any estimate of variability will vary quite a bit if based on such a small sample. For example, the sample SD as an estimator of the population SD has quite high variability, even in the case of N(0,1) data.

Here is the result of an experiment on this:

.



mean of sample SD = 1.00 SD of sample SD = 0.16 range in n=100 trials is (0.63, 1.41)
So the estimate of the SD which we know is 1 is only within ± .16
In our 15 repetitions of the bootstrap estimate of the 90th percentile, the
SD of the SD of the estimate was only 0.10. This is less than the 0.16 which is the variability of the sample SD that we habitually use as our estimate of the population SD.

Of course, larger samples provide much better estimates. The small sample used in these experiments (n=20) was used to demonstrate the surprising efficiency of the bootstrap – in a case where it might be expected to fail, it worked pretty well.