Lecture 6: Bootstrapping

36-402, Advanced Data Analysis

31 January 2013
Knowing the sampling distribution of a statistic tells us about statistical uncertainty (standard errors, biases, confidence sets)

The bootstrap principle: *approximate* the sampling distribution by *simulating* from a good model of the data, and treating the simulated data just like the real data

Sometimes we simulate from the model we’re estimating (parametric bootstrap)

Sometimes we simulate by re-sampling the original data (nonparametric bootstrap)

As always, stronger assumptions mean less uncertainty *if we’re right*
Re-run the experiment (survey, census, ...) and we get more or less different data
∴ everything we calculate from data (estimates, test statistics, policies, ...) will change from trial to trial as well
This variability is (the source of) **statistical uncertainty**
Quantifying this is a way of being honest about what we do and do not know
Measures of Statistical Uncertainty

Standard error = standard deviation of an estimator
could equally well use median absolute deviation, etc.

\( p \)-value = Probability we’d see a signal this big if there was just noise

Confidence region = All the parameter values we can’t reject at low error rates:

1. *Either* the true parameter is in the confidence region
2. *or* we are very unlucky
3. *or* our model is wrong

etc.
The Sampling Distribution Is the Source of All Knowledge

Data $X \sim P_{X, \theta_0}$, for some true $\theta_0$
We calculate a statistic $T = \tau(X)$ so it has distribution $P_{T, \theta_0}$
If we knew $P_{T, \theta_0}$, we could calculate $\text{Var}[T]$ (and so standard error),
$E[T]$ (and so bias), quantiles (and so confidence intervals or $p$-values), etc.
Problem 1: Most of the time, $P_{X, \theta_0}$ is very complicated
Problem 2: Most of the time, $\tau$ is a very complicated function
Problem 3: We certainly don’t know $\theta_0$
Upshot: We don’t know $P_{T, \theta_0}$ and can’t use it to calculate anything
Classically (≈ 1900–≈ 1975): Restrict the model and the statistic until you can calculate the sampling distribution, at least for very large $n$
Modern (≈ 1975–): Use complex models and statistics, but simulate calculating the statistic on the model
some use of this idea back to the 1940s at least
The Bootstrap Principle

1. Find a good estimate $\hat{P}$ for $P_{X,\theta_0}$
2. Generate a simulation $\tilde{X}$ from $\hat{P}$, set $\tilde{T} = \tau(\tilde{X})$
3. Use the simulated distribution of the $\tilde{T}$ to approximate $P_{T,\theta_0}$

Refinements: improving the initial estimate $\hat{P}$
reducing the number of simulations or speeding them up
transforming $\tau$ so the final approximation is more stable
First step: find a good estimate $\hat{P}$ for $P_{X,\theta_0}$
If we are using a model, our best guess at $P_{X,\theta_0}$ is $P_{X,\hat{\theta}}$, with our best estimate $\hat{\theta}$ of the parameters

**THE PARAMETRIC BOOTSTRAP**

1. Get data $X$, estimate $\hat{\theta}$ from $X$
2. Repeat $b$ times:
   1. Simulate $\tilde{X}$ from $P_{X,\hat{\theta}}$ (simulate data of same size/“shape” as real data)
   2. Calculate $\tilde{T} = \tau(\tilde{X})$ (treat simulated data the same as real data)
3. Use empirical distribution of $\tilde{T}$ as $P_{T,\theta_0}$
Concrete Example

Is Moonshine over-weight?
Data on weights of 144 cats; fit Gaussian, find 95th percentile

library(MASS); data(cats); summary(cats)
(q95.gaussian <- qnorm(0.95, mean = mean(cats$Bwt), sd = sd(cats$Bwt)))
Simulate from fitted Gaussian; bundle up estimating 95th percentile into a function

```r
cats.gaussian <- function() {
    rnorm(n=nrow(cats),mean=mean(cats$Bwt),sd=sd(cats$Bwt))
}

est.q95.gaussian <- function(x) {
    m <- mean(x)
    s <- sd(x)
    return(qnorm(0.95,mean=m,sd=s))
}
```
Simulate, plot the sampling distribution from the simulations

```r
sampling.dist.gaussian <- replicate(1000, est.q95.gaussian(rcats.gaussian()))
plot(hist(sampling.dist.gaussian, breaks=50), freq=FALSE)
plot(density(sampling.dist.gaussian))
abline(v=q95.gaussian, lty=2)
```
Find standard error and a crude confidence interval

\[sd(sampling\_dist\_gaussian)\]
\[quantile(sampling\_dist\_gaussian,c(0.025,0.975))\]
The crude confidence interval uses the distribution of $\tilde{\theta}$ under $\hat{\theta}$
But really we want the distribution of $\hat{\theta}$ under $\theta$
Observation: Generally speaking,

$$\Pr_{\hat{\theta}} (\tilde{\theta} - \hat{\theta} \leq a) \rightarrow \Pr_{\theta_0} (\hat{\theta} - \theta_0 \leq a)$$

faster than

$$\Pr_{\hat{\theta}} (\tilde{\theta} \leq a) \rightarrow \Pr_{\theta_0} (\hat{\theta} \leq a)$$

(errors converge faster, as in CLT)
$\hat{\theta} - \theta_0$ is (nearly) “pivotal”
The Basic, Pivotal CI

\[ q_{\alpha/2}, q_{1-\alpha/2} = \text{quantiles of } \tilde{\theta} \]

\[ 1 - \alpha = \Pr_{\hat{\theta}} \left( q_{\alpha/2} \leq \tilde{\theta} \leq q_{1-\alpha/2} \right) \]

\[ = \Pr_{\hat{\theta}} \left( q_{\alpha/2} - \hat{\theta} \leq \tilde{\theta} - \hat{\theta} \leq q_{1-\alpha/2} - \hat{\theta} \right) \]

\[ \approx \Pr_{\theta_0} \left( q_{\alpha/2} - \hat{\theta} \leq \tilde{\theta} - \theta_0 \leq q_{1-\alpha/2} - \hat{\theta} \right) \]

\[ = \Pr_{\theta_0} \left( q_{\alpha/2} - 2\hat{\theta} \leq -\theta_0 \leq q_{1-\alpha/2} - 2\hat{\theta} \right) \]

\[ = \Pr_{\theta_0} \left( 2\hat{\theta} - q_{1-\alpha/2} \leq \theta_0 \leq 2\hat{\theta} - q_{\alpha/2} \right) \]

Basically: re-center the simulations around the empirical data
Switch to R

Find the basic CI

\[ 2 \times \text{q95.gaussian} - \text{quantile(sampling.dist.gaussian, c(0.975, 0.025))} \]
As always, if the model isn’t right, relying on the model is asking for trouble
How good is the Gaussian as a model for the distribution of cats’ weights?
Compare histogram to fitted Gaussian density and to a smooth density estimate

```r
plot(hist(cats$Bwt), freq=FALSE)
curve(dnorm(x, mean=mean(cats$Bwt), sd=sd(cats$Bwt)), add=TRUE, col="purple")
lines(density(cats$Bwt), lty=2)
```
Problem: Suppose we don’t have a trust-worthy parametric model
Resource; We do have data, which tells us a lot about the
distribution
Solution: **Resampling**, treat the sample like a whole population

**THE NONPARAMETRIC BOOTSTRAP**

1. Get data $X$, containing $n$ samples
2. Repeat $b$ times:
   1. Generate $\tilde{X}$ by drawing $n$ samples from $X$ *with replacement*
      (resample the data)
   2. Calculate $\tilde{T} = \tau\tilde{X}$ (treat simulated data the same as real data)
3. Use empirical distribution of $\tilde{T}$ as $P_{T,\theta}$
Model-free estimate of the 95th percentile is the 95th percentile of the data
How precise is that?
Resampling, re-estimating, and finding sampling distribution, standard error, bias, CIs

(q95.np <- quantile(cats$Bwt, 0.95))
resample <- function(x) {
  sample(x, size = length(x), replace = TRUE)
}
est.q95.np <- function(x) {
  quantile(x, 0.95)
}
sampling.dist.np <- replicate(1000, est.q95.np(resample(cats$Bwt)))
plot(density(sampling.dist.np))
abline(v = q95.np, lty = 2)
sd(sampling.dist.np)
mean(sampling.dist.np - q95.np)
quantile(sampling.dist.np, c(0.025, 0.975))
2*q95.np - quantile(sampling.dist.np, c(0.975, 0.025))
A regression is a model for $Y$ conditional on $X$

$$Y = m(X) + \text{noise}$$

Silent about distribution of $X$, so how do we simulate?
Options, putting less and less trust in the model:

1. Hold $x_i$ fixed, set $\hat{y}_i = \hat{m}(x_i) + \text{noise from model’s estimated noise distribution (e.g., Gaussian)}$
2. Hold $x_i$ fixed, set $\hat{y}_i = \hat{m}(x_i) + \text{a resampled residual}$
3. Resample $(x_i, y_i)$ pairs (resample data-points or resample cases)
The cats data set has weights for cats’ hearts, as well as bodies much cuter than an actual photo of cats’ hearts.

Source: http://yaleheartstudy.org/site/wp-content/uploads/2012/03/cat-heart1.jpg

How does heart weight relate to body weight?
(Useful if Moonshine’s vet wants to know how much heart medicine to prescribe)
Plot the data with the regression line

```r
plot(Hwt ~ Bwt, data=cats, xlab="Body weight (kg)", ylab="Heart weight (g)"")
cats.lm <- lm(Hwt ~ Bwt, data=cats)
abline(cats.lm)
```
Coefficients and “official” confidence intervals:

```r
coefficients(cats.lm)
confint(cats.lm)
```
The residuals don’t look very Gaussian:

```r
plot(cats$Bwt, residuals(cats.lm))
plot(density(residuals(cats.lm)))
```
Find CIs for coefficients by resampling cases:

```r
coefs.cats.lm <- function(subset) {
  fit <- lm(Hwt~Bwt, data=cats, subset=subset)
  return(coefficients(fit))
}
cats.lm.sampling.dist <- replicate(1000, coefs.cats.lm(resample(1:nrow(cats))))
(limits <- apply(cats.lm.sampling.dist,1,quantile,c(0.025,0.975)))
```
Sources of Error in Bootstrapping

**Simulation**  Using only $b$ bootstrap replicates
- Make this small by letting $b \to \infty$
- Costs computing time

**Approximation** Using $\hat{P}$ instead of $P_{X, \theta_0}$
- Make this small by careful statistical modeling

**Estimation** Only a finite number of samples
- Make this small by being careful about what we simulate (e.g., basic interval vs. crude interval)

Generally: for fixed $n$, nonparametric bootstrap shows more uncertainty than parametric bootstraps, but is less at risk to modeling mistakes
- Yet another bias-variance tradeoff
Standard errors, biases, confidence regions, $p$-values, etc., could all be calculated from the sampling distribution of our statistic.

The bootstrap principle: simulate from a good estimate of the real process, use that to approximate the sampling distribution.
- Parametric bootstrapping simulates an ordinary model.
- Nonparametric bootstrapping resamples the original data.

Simulations get processed *just like* real data.

Bootstrapping works for regressions and for complicated models as well as distributions and simple models.