MSA220 - Statistical Learning for Big Data

Lecture 14

Rebecka Jörnsten

Mathematical Sciences
University of Gothenburg and Chalmers University of Technology

Big n!

Our final theme!

- When the sample size is large, there's a couple of things we need to be concerned about
- p-values become "meaningless" - simply reflecting that all models are approximations of the real world
- Computations can become impossible or slow, even for simple statistical tasks

Big n statistics

- Most methods bear a strong resemblance to stuff you're already familiar with
- Cross-validation, subsampling (bagging, RF) and bootstrap
- We will review bootstrap first because of this

The Bootstrap

An excellent book (can be found online): *An Introduction to the Bootstrap* by R. Tibshirani and B. Efron

- What and why?
- We often just trust confidence intervals that package methods spit out without thinking about underlying assumptions
- For linear models this is often OK as long as the error distribution isn't too messy or skewed and as long as the sample size is fairly big
- In other modeling scenarios, or for particular statistics of interest, we should be careful!

The Bootstrap

- In nonlinear modeling and generalized linear models, CIs provided are approximations!
- Involves linear approximations to obtain standard errors and large-sample asymptotic arguments to motivate the format of the CI
- Why not let the data do the work for us instead!

Recap

- Your observed data is a sample from the underlying population
- If you were able to repeatedly sample from this population, each time estimating your statistic of interest, $\hat{\theta}$...
- then the distribution of the $\hat{\theta}$ across the samples reflects the *sampling distribution* of this statistics
- and can be used to construct CI and for testing

Recap

- In the real life case you only observe one sample
- You therefore work through the properties of the statistics so that you can compute the sampling distribution statistics without having access to multiple samples
- Example: Draw $x_i, i = 1, \cdots, n$ independently from population. Goal: estimate the mean μ of the population.
- $\hat{\mu} = \bar{x}$ and from iid assumption $SE(\hat{\mu}) = \frac{1}{\sqrt{n}}\hat{\sigma}$ where $\hat{\sigma} = \sqrt{\sum_i (x_i - \bar{x})^2 / n - 1}$
- If we assume $X\tilde{N}(\mu, \sigma^2)$ then $\frac{\hat{\mu}-\mu}{SE} \sim t_{n-1}$
- or even without this assumption if n is large we have $\frac{\hat{\mu}-\mu}{SE} \sim N(0,1)$

Recap

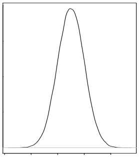
- What if your statistic is more complicated so the format of the SE is unknown?
- Work it out? Use linear approximation methods (Taylor expansion, delta-method)
- Use the bootstrap!

The bootstrap

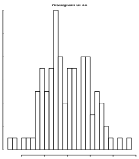
- The idea is that you mimic the sampling from the population with a repeated sampling from the observed data
- The sampling distribution estimate you obtain by repeated sampling from the observed data can be a very good estimate of the true sampling distribution
- It doesn't always work: for extremes or "weird" statistics that are non-continuous on the true distribution or for small sample sizes.

The bootstrap

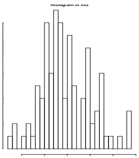
Population



Histogram of observed sample



Histogram of bootstrap data



The bootstrap

- For $b = 1, \cdots, B$ (B large, 1000-10000), draw a bootstrap sample from your observed data
- Non-parametric, classic: draw n samples with replacement.
- Alternatives: draw $m < n$ without replacement (m-out-of-n), draw from a smooth density estimate of the data, draw from a parametric distribution
- Estimate your statistics θ_b from each of the bootstrap data
- The distribution of θ_b across b is an estimate of the sampling distribution of $\hat{\theta}$, the estimate from the original data

The bootstrap

- $\bar{\theta} = \frac{1}{B}\sum_b \theta_b$ is NOT a better estimate than $\hat{\theta}$ - the purpose of bootstrap is not to improve on the estimate this way
- Bootstrap SE: $\sqrt{\frac{\sum_b(\theta_b - \bar{\theta})^2}{B-1}}$ can be used to construct CI
- Bias estimate: $\hat{\theta} - \bar{\theta}$ can be used to construct a bias-corrected estimate BUT it only reflects bias with respect to estimation NOT bias induced by the wrong model assumption (that would be magic).

Bootstrap CI

- Bootstrap SE: $\sqrt{\frac{\sum_b (\theta_b - \bar{\theta})^2}{B-1}}$ can be used to construct CI
- Basic CI: $\hat{\theta} \pm z_{1-\alpha/2} SE$
- Note: here we are using a normal assumption for the sampling distribution BUT we could go further using the bootstrap distribution instead

Bootstrap CI

- How get around the normal assumption
- Double-bootstrap
- For each bootstrap estimate θ_b, run a second bootstrap on this bootstrap sample to obtain SE_b and compute the *pivotal element*
$$z_b = \frac{\theta_b - \hat{\theta}}{SE_b}$$
- Use the quantiles of the z_b instead of the normal quantiles
- This is called the bootstrap-t

Percentile method

- A conceptually simple approach is the *percentile method*
- Simple construct your confidence interval from the quantiles of the θ_b (e.g. the 2.5% and 97.5%)!!!
- Supersimple.... BUT behaves poorly in many real-life situations.

The bootstrap

- The rationale behind the percentile method is that for a normally distributed $\frac{\hat{\theta}-\theta}{\sigma} \sim N(0,1)$ we have that

  $$(\hat{\theta}_{\alpha/2}, \hat{\theta}_{1-\alpha/2})$$

  is a $1-\alpha$ CI
- With the percentile method, we assume that there is some monotone transformation $g(\hat{\theta})$ such that its sampling distribution is approximately normal $N(0,1)$
- Why does percentile method then fail sometimes?
- There may not exist one transform that has this normalizing and variance-stabilizing effect

BCa

- Brad Efron (The Bootstrap Guy!) proposed an improved percentile method as follows:
- Perhaps we need to correct the simple monotone transform with some bias constant and acceleration constant to make the approximate normal assumption hold

$$\phi = g(\theta), \quad \frac{\hat{\phi} - \phi}{\sigma} \sim N(-z_0, 1), \ \sigma = 1 + a\phi$$

- The bias correction is obtained by the normal quantile of $P_B(\theta_b < \hat{\theta})$
- The acceleration constant is obtained from an estimate of the skewness of the θ_b distribution
- We adjust which quantiles in θ_b to actually use to construct the $1 - \alpha$ CI.
- R package `boot()`!!!

Back to big n

- New methods for dealing with large sample size
- Parallelization or
- online updates

Bag of Little Bootstraps

- This method is based on Bickel et al.'s m-out-of-n bootstrap
- m-out-of-n was shown to have better properties than regular sample-with-replacement bootstrap
- When you use m-out-of-n, you need to correct the SEs by a factor $\sqrt{m/n}$ - but otherwise it works pretty much the same way as regular bootstrap
- Here, is it used to reduce sample size!!!

Bag of Little Bootstraps

- We draw s subsets of data of size $m < n$
- For each of the s subsets, draw r samples of size n
- Obtain point estimate and e.g. CIs from the r bootstraps
- Finally, combine the results across the s subsets

Bag of Little Bootstraps

- Wait a minute! Didn't this just make the computations explode?
- Actually, no - drawing a sample of size n from the subset s of size m is equivalent to assigning *weights* to the m observations in s
- So the computation is actually performer only on the smaller sample size m

Bag of Little Bootstraps

The algorithm

- For $j = 1, \cdots, s$, draw a sample of size m (or disjoint partition of the original data)
    - For $k = 1, \cdots, r$,
    - Draw weights from Multinomial(n,m)
    - Estimate your statistics of interest
- Combine by averaging quantities of interest across s (e.g. estimates, lower and upper CI limits, etc)

Bag of Little Bootstraps

- Recommended size of $m = n^\gamma$, $\gamma \in [.5, 1]$
- In the original BLB paper (Kleiner et al, 2014) they use $\gamma = 0.6$ (reducing a data set of 10^6 to about 4000 for computation).
- Kleiner et al found that BLB is fairly robust to choices of m, consistency of estimates and good convergence rates
- Completely parallelizable for each set of size m so allows for fast and scalable computing
- Implemented in the `datadr` R package

Leveraging

- Another variant for subsampling was proposed by Ma and Sun (2013)
- Like the BLB, they suggest that we estimate model parameters from a much smaller data set and then combine the results
- However, they differ in how the subsampling is done

Leveraging

- Recap from regression
- $y = X\beta + \epsilon$
- LS: $\min_\beta ||y - X\beta||^2$
- $\hat{\beta} = (X'X)^{-1}X'y$
- $\hat{y} = X\hat{\beta} = X(X'X)^{-1}X'y = Hy$

Leveraging

- Specifically, $\hat{y}_i = \sum_j h_{ij} y_j$
- Element h_{ii} is called the *leverage* of observation i, i.e. how much it influences its own fitted values
- Leverage basically captures if observation i is close or far from the center of the data. Observations near the center (in X-space) have limited contribution to the fit.

Leveraging

- Sample r observations from the original n where $r << n$
- The sampling probability π_i for observations i is $\pi = \frac{h_{ii}}{\sum_j h_{jj}}$
- Estimate to regression parameters
  - Alt 1: use standard OLS
  - Alt 2: use weighted LS, where the weights are the inverse sampling probabilities

Leveraging

- Ma and Sun found that regular OLS works better than the weighted version
- Seems simple enough!
- BUT we do need the leverage h_{ii}.
- Hm..
- The matrix H is $n \times n$ so we don't want to have to compute that - we only care about the diagonal anyway.

Leveraging

- SVD to the rescue (again!)
- $X = UDV'$
- $H = X(X'X)^{-1}X' = UU'$
- and so $h_{ii} = ||u_i||^2$ for u_i i-th row in U
- Moreover, fast randomized SVD methods exist

Leveraging

- Fast and simple
- A bit careful about outliers....
- A big pro: can use the subsample to visualize the data
- Model diagnostics in a big-n world - and we could remove outliers at this point...