Math 455 Class 32 April 16

The "Bootstrap"

Idea: you have a data set and you want to get more information out of it.

Solution: Sample your data set (with replacement) to get a new data set. In fact, many new data sets.

Problem: We have n rows of data, and sample n times (with replacement). What is the average number of rows that get left out of our sample?

Solution: every time we pick a row in our sampling row 1 has a $\left(1-\frac{1}{n}\right)$ chance of being left out.

Sampling is independent, so chance row 1 is left out after n samples is $\left(1-\frac{1}{n}\right)^n$. If n is large a good approximation is: $e^{-1} \approx 0.37$

Calculus: $\lim_{n\to\infty} \left(1+\frac{x}{n}\right)^n = e^x$

The same analysis applies to any row, not just row 1.

So the average number of rows left out is

test sets on average over many iterations.

This idea can be automated and made much fancier. Example Faraway p. 127 – 128.

Third idea for Bootstrap:

When we produce estimates $\hat{\beta}_i$, they come with a standard error. These standard errors were derived from the assumptions of linear regression and the theory of least-squares.

Suppose that we have another method of estimation. which doesn't come with a theory of its standard error. (Examples are found in 8.4, Robust Regression. Also many ML methods satisfy this.)

How can we get an estimate of std. err for such an estimator? The bootstrap answers this question empirically: get a lot of data sets, compute the estimator on each one, and obtain the std. error from these numbers.

Q: Why do we need a std. error?

$\left(1 - \frac{1}{n}\right)^n \cdot n \approx e^{-1} n \approx (0.37)n$

Note: ML texts sometimes simplify further, to $\frac{1}{3}n$.

Second idea: the "left-out" data is still useful.

If we fit our model to the data, this doesn't answer the important question: how good is our model on new data that it <u>hasn't</u> seen?

These "left-out" rows provide a "test set" of data that the model hasn't seen.

We can check how good the model is for this "test set".

Of course, the model could turn out to be good or bad on this test set by coincidence.

We can handle this with the earlier idea: we can make many samples: all will produce different "training sets" of size $\approx 0.63\,n$ and different "test sets" of size $\approx 0.37\,n$.

We can look at how good our model is on

A: We want to know how reliable our results are.

Two principles: ① We should try to predict something continuous.

② Our predictions should have an error estimate.

Q: What if the thing we want to predict isn't continuous?

A: Transfer your attention to something that is continuous.

Example: Instead of predicting the winner of the basketball game, try to predict the number of points scored by each team.

Or we could predict a probability of winning.

(Logistic Regression, math 532)

Example from "Robust Regression"

Things we could do other than least squares:

1. Least Absolute deviation (LAD)

Minimize $\sum \left| (Y_i - \beta_0 - \beta_1 X_i) \right|$

2. "Huber Method"

least squares up to some point, and then least absolute deviation afterward.

Define: $$\rho(x) = \begin{cases} x^2/2 & \text{if } |x| \le c \\ c|x| - c^2/2 & \text{if } |x| > c. \end{cases}$$

Minimize $$\sum \rho\left(Y_i - \beta_0 - \beta_1 X_i\right)$$

3. Least Trimmed Squares.

~~Less~~ errors $e_i = Y_i - \beta_0 - \beta_1 X_i$

Ordered errors (by absolute value)

$$|e_{(1)}| \le |e_{(2)}| \le \cdots \le |e_{(n)}|$$

Drop the k highest errors and minimize

$$\sum_{i=1}^{n-k} e_{(i)}^2$$

The point of this: these methods are all LESS sensitive to outliers.

LTS does NOT come with std. error estimates.

But we can get them using the bootstrap.

We look at R: Faraway p. 127.