Linear Regression Models
P8111

Lecture 11

Jeff Goldsmith

February 25, 2016

THE DEPARTMENT OF BIOSTATISTICS

Columbia University MAILMAN SCHOOL OF PUBLIC HEALTH

Today’s Lecture

- Review of tests
- The bootstrap
- Permutation testing
- Cross validation

Individual coefficients

For individual coefficients

- We can use the test statistic

$$T = \frac{\hat{\beta}_j - \beta_j}{\widehat{se}(\hat{\beta}_j)} = \frac{\hat{\beta}_j - \beta_j}{\sqrt{\hat{\sigma}^2 (\boldsymbol{X}^T \boldsymbol{X})^{-1}_{jj}}} \sim t_{n-p-1}$$

- For a two-sided test of size α, we reject if

$$|T| > t_{1-\alpha/2, n-p-1}$$

- The p-value gives $P(|t_{n-p-1}| > |T_{obs}||H_0)$

Note that t is a symmetric distribution that converges to a Normal as $n - p - 1$ increases.

Inference for linear combinations

Sometimes we are interested in making claims about $c^T\boldsymbol{\beta}$ for some c.

- Define $H_0 : c^T\boldsymbol{\beta} = c^T\boldsymbol{\beta}_0$ or $H_0 : c^T\boldsymbol{\beta} = 0$
- We can use the test statistic

$$T = \frac{c^T\hat{\boldsymbol{\beta}} - c^T\boldsymbol{\beta}}{\widehat{se}(c^T\hat{\boldsymbol{\beta}})} = \frac{c^T\hat{\boldsymbol{\beta}} - c^T\boldsymbol{\beta}}{\sqrt{\hat{\sigma}^2 c^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}c}}$$

- This test statistic is asymptotically Normally distributed
- For a two-sided test of size α, we reject if

$$|T| > z_{1-\alpha/2}$$

Global F tests

- Compute the test statistic

$$F_{obs} = \frac{(RSS_S - RSS_L)/(df_S - df_L)}{RSS_L/df_L}$$

- If H_0 (the null model) is true, then $F_{obs} \sim F_{df_S - df_L, df_L}$
- Note $df_s = n - p_S - 1$ and $df_L = n - p_L - 1$
- We reject the null hypothesis if the p-value is above α, where

$$\text{p-value} = P(F_{df_S - df_L, df_L} > F_{obs})$$

The Wald test

For a vector of coefficients, we can test $H_0 : \boldsymbol{\beta} = \boldsymbol{\beta}_0$:

- Use the test statistic

$$W = (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0)^T [Var(\hat{\boldsymbol{\beta}})]^{-1} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0)$$

- Under the null, this test statistic has an asymptotic χ^2_p distribution
- In practice, we replace $Var(\hat{\boldsymbol{\beta}})$ with $\widehat{Var}(\hat{\boldsymbol{\beta}})$ and use an F distribution

The LRT

If we are using maximum likelihood estimation (we'll cover this soon – turns out to be least squares in MLR), we can use a LRT:

- Use the test statistics

$$\Delta = -2\log\frac{L_0}{L_1} = -2(l_0 - l_1)$$

- This test statistic has an asymptotic χ^2_d distribution where d is the difference in the number of parameters between the two models.

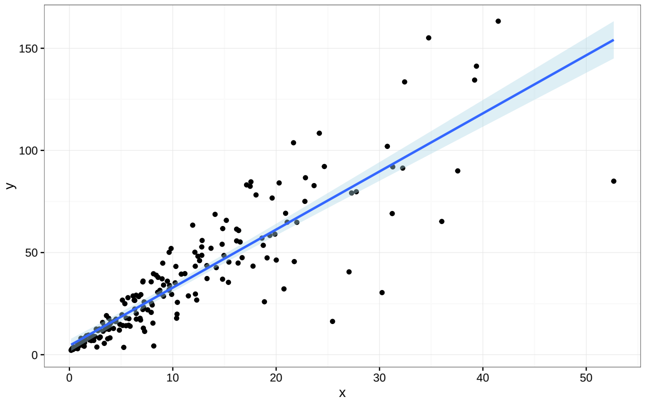
Inference: departure from assumptions

- In large samples, $\hat{\beta}$ is approximately Normal even if the errors are not
- In smaller samples, especially when our assumptions are not justified, the inferential methods we've developed are not valid
- Might also want variance estimates for quantities that are difficult to derive analytically

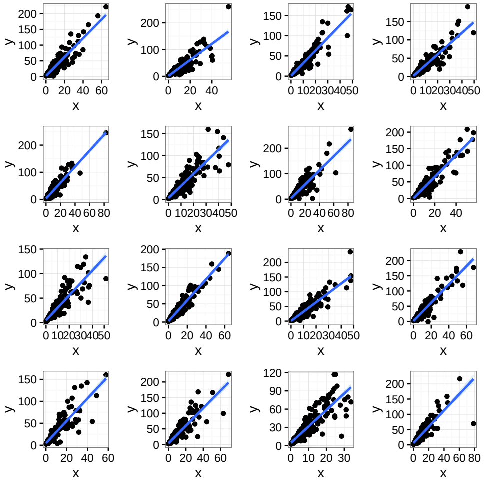
Resampling methods

- Fundamental tool in modern statistics
- Build on the intuition that, if we could, we'd keep gathering new datasets – but we can't
- Use repeated samples of a training set to understand variability
- Computationally intensive ... but we have computers

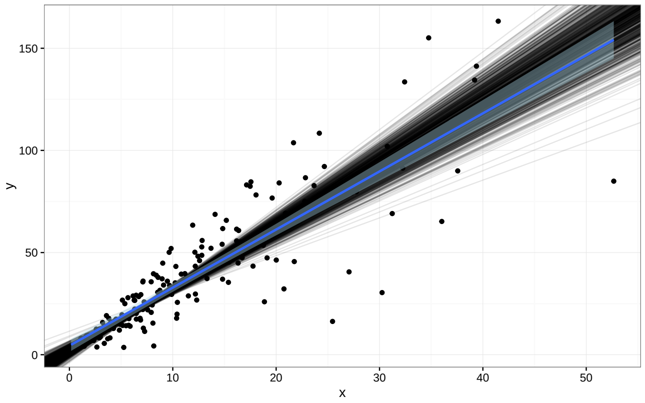
Motivating the Bootstrap



Motivating the Bootstrap



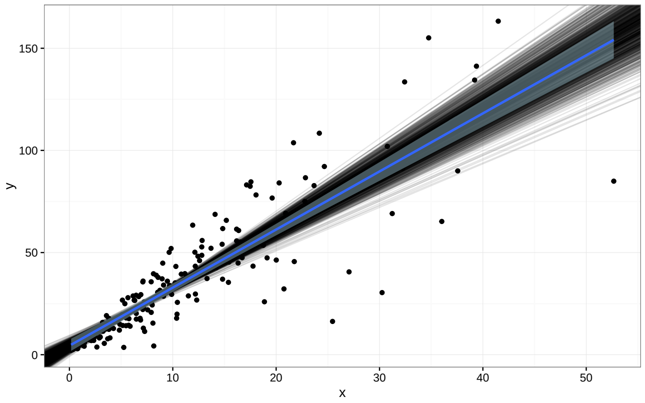
Motivating the Bootstrap



The Bootstrap

- The basic idea is that the observed data mimics the underlying distribution, whatever that may be
- Drawing samples (with replacement) from the observed data mimics drawing samples from the underlying distribution
- Recalculating regression parameters for the “new” samples gives an idea of the distribution of regression coefficients

Implementing the Bootstrap

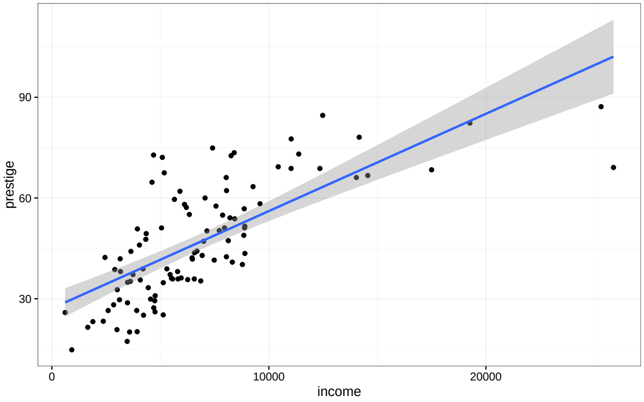


Bootstrap example

Prestige dataset

- Information on 102 occupations
- Variables include education, income, proportion women, job type, and prestige
- Source: 1971 Canadian census

Non-normal inference

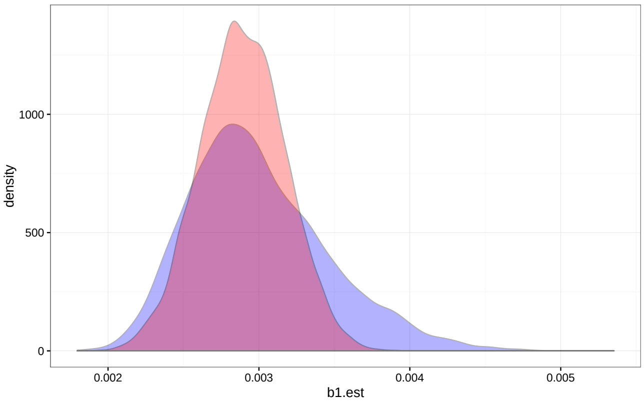


Bootstrap code

```
## define a vector for the bootstrapped estimates
betaHatBS = data.frame(b1.est = rep(NA, 10000))

## use a loop to do the bootstrap
for(i in 1:10000){
  data.cur = sample_frac(Prestige, size = 1, replace = TRUE)
  betaHatBS$b1.est[i] = lm(prestige ~ income, data = data.cur)$coef[2]
}
```

Bootstrap results



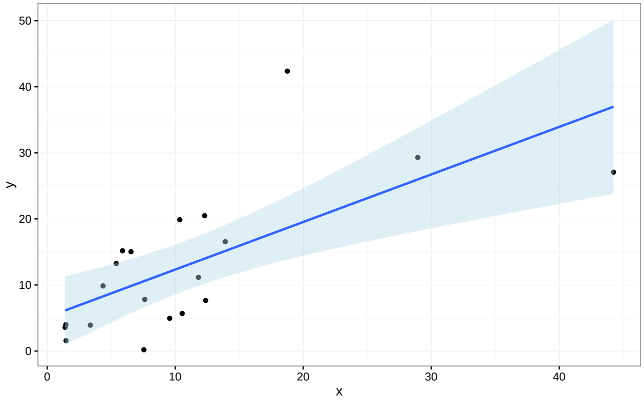
Permutation testing

- Bootstrapping helps understand variability
- What about testing?
- One option – invert the CI from a bootstrap
- Another option – understand distribution of “test statistic” under the null

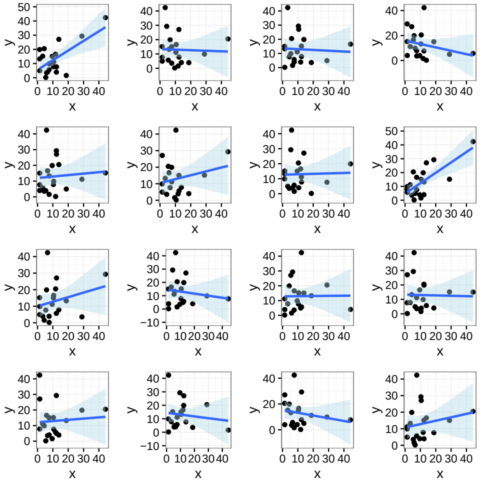
Permute the data

- If we permute the data, there should be no association
- Easy for comparing two groups or SLR; harder for MLRs

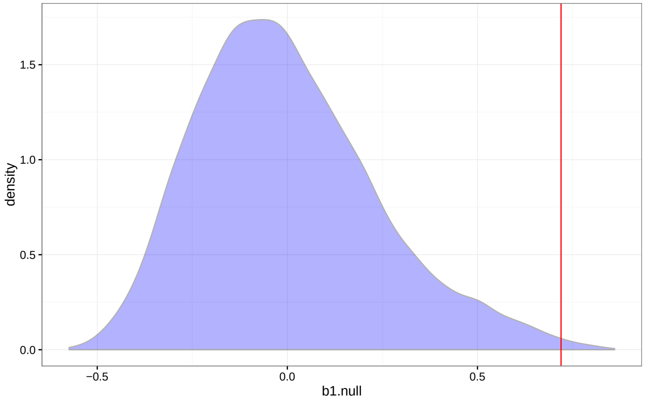
Permutation test example



Permutation test example



Permutation test example



Implementing permutation tests

```
## do enough permutations to test
obs.coef = coef(lm(y ~ x, data = data.noncst))[2]

b1 = data.frame(b1.null = rep(NA, 10000))
for(i in 1:10000){
  data.noncst.cur = mutate(data.noncst, x = sample(x, length(x), replace = FALSE))
  b1$b1.null[i] = coef(lm(y ~ x, data = data.noncst.cur))[2]
}
```

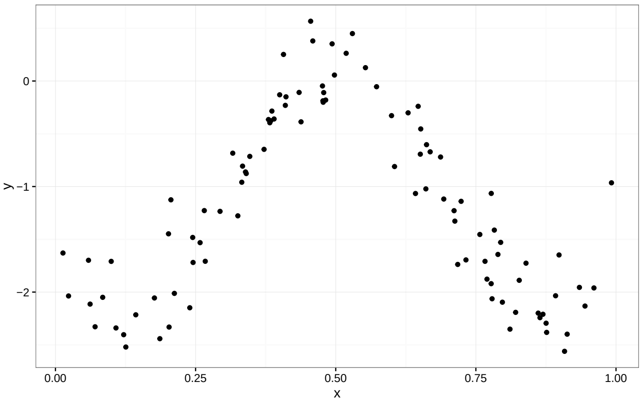
Cross Validation

- Focus is on model performance, quantified by prediction error
- We get in-sample performance ...
- But we want generalization to new data
- Most of the time, we don’t have an external testing dataset

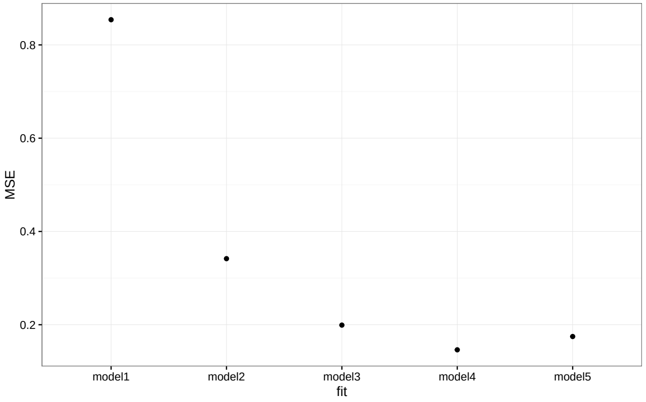
Cross Validation: one validation set

- Simplest case: create a validation set by randomly splitting the full dataset
- Fit model to training data; compute mean squared prediction error on test set
- Provides way of comparing models (any models ...)

Example data

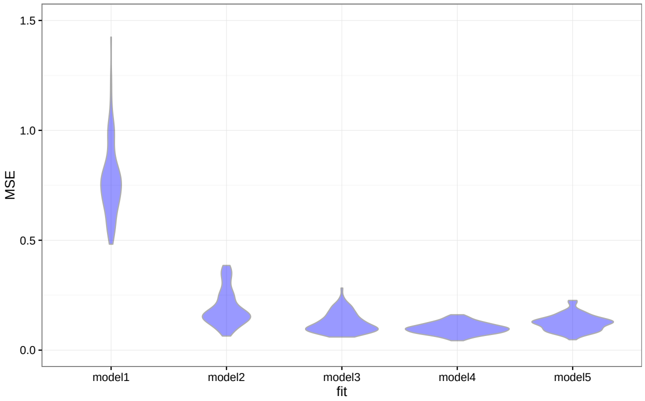


Cross Validation: one validation set



Cross Validation: many validation set

- How you split the data is random; can repeat to understand this source of uncertainty



Implementing CV

```
MSEs = data.frame(
  model1 = rep(NA, 100),
  ...
 )

for(i in 1:100){

  set.seed(i)
  data.nonlin = mutate(data.nonlin,
                       cv_group = sample(1:100, 100, replace = FALSE) <= 80,
                       cv_group = factor(cv_group, levels = c(TRUE, FALSE),
                                         labels = c("train", "test")))

  data.train = filter(data.nonlin, cv_group == "train")
  data.test = filter(data.nonlin, cv_group == "test")

  fit.1 = lm(y ~ x, data = data.train)
  MSEs[i,1] = mean((data.test$y - predict(fit.1, newdata = data.test))^2)

  fit.2 = lm(y ~ x + spline_5, data = data.train)
  MSEs[i,2] = mean((data.test$y - predict(fit.2, newdata = data.test))^2)

  ...

}
```

Cross Validation: folds

- Could use k-fold cross validation:
  - Divide data into k equal-sized folds
  - Use each one in turn as the validation set; average MSE across sets
  - k of 5 or 10 is pretty common

Today’s big ideas

- Resampling methods

- Suggested reading: ISLR chapter 5