36-617: Applied Linear Models

Variable Selection

Brian Junker
132E Baker Hall
brian@stat.cmu.edu

Announcements

- Reading
  - This week: Sheather 6.4, 6.5, 6.6, 7.1, 7.2
    (supplemental: ISLR 3.3.3; G&H Ch 4)
  - Next week: Sheather, 7.3, 7.4, 8.1, 8.2
    (supplemental: ISLR 3.3.3 & Ch 6; G&H Ch 4)
- Quiz 03
  - Quiz 3 Results.pdf and Quiz 3 Data Generation.r are in the files area on Canvas
  - (I also put quiz results for the earlier quizzes there, for the curious!)
- HW04 is out – Due Mon, 1159pm as usual
- Next week: Take-home midterm instead of regular HW.
  - More on that next week!

Outline

- Variable selection – An Overview
- Traditional variable selection
  - MSE-based indices
  - Likelihood-based indices
  - Stepwise and all-subsets
- Modern variable selection
  - Variable selection by penalized likelihood
  - Are All Subsets and Stepwise really so different from Ridge and Lasso?
- Inference after variable selection
- What do I really do?

Variable Selection – The ***Dark Underbelly*** of Statistical Modeling

- Large search space (*p* predictors: 2^p or 2^{2^p} models)
  - Heuristics to select "good paths" through model space
- Multiple-inference problems and non-nested model comparisons as we sift through models
  - (trad) Indices instead of statistical tests
  - (mod) Jointly estimate model & select variables
- Inference after model selection is vulnerable to capitalization on chance
  - Training/test samples; cross-validation methods

Traditional Variable Selection

RSS & MSE-based indices

- $R^2 = \frac{SS_{reg}}{SST} = 1 - \frac{RSS}{SST}$.

$$RSS_{(y=X\beta+x'\beta'+\epsilon)} \leq RSS_{(y=X\beta+\epsilon)}$$

  hence RSS always decreases (R^2 increases) when you add predictors.

- $R^2_{adj} = 1 - \frac{RSS/(n-p-1)}{SST/(n-1)} = 1 - \frac{MSE}{MST}$ is an attempt to fix this.

- While not as sensitive to overfit as RSS, MSE (and hence R^2_{adj}) still tends to choose overly complex models.

  - In-sample measure: same data used to estimate coefficients & measure error/fit

- Another common measure[1], Mallows' $C_p = p+1+\frac{(MSE-\hat{\sigma}^2)(n-p-1)}{\hat{\sigma}^2}$, shares these flaws.

[1]See https://onlinecourses.science.psu.edu/stat501/node/330/ for a discussion of C_p, for example.

Digression: ML estimation

We know

$$\begin{aligned} L(\beta, \sigma^2) &= \prod_{i=1}^{n} f(y_i|X_i) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(y_i - X_i\beta)^2\right\} \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - X_i\beta)^2\right) \end{aligned}$$

So

$$\log L(\beta, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\sigma^2 - \frac{1}{2\sigma^2}\sum_{i-1}^{n}(y_i - X_i\beta)^2$$

If we are not interested in estimating σ^2, the maximized likelihood is

$$\begin{aligned} \log L(\hat{\beta}, \sigma^2) &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\sigma^2 - \frac{1}{2\sigma^2}RSS \\ &= c_1(n, \sigma^2) - c_2(\sigma^2) \cdot RSS \end{aligned}$$

and if we want to estimate σ^2 at the same time as β, the maximized log-likelihood will be

$$\begin{aligned} \log L(\hat{\beta}, \hat{\sigma}^2) &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(RSS/n) - \frac{1}{2RSS/n}RSS \\ &= c_1(n) - c_2(n)\log(RSS) \end{aligned}$$

Digression: Likelihood Ratio (LR) Test

- Let the full model be $y = X\beta + x'\beta' + \varepsilon$ and the reduced model be $y = X\beta + \varepsilon$
- Under H_0: $\beta'=0$, the LR test statistic

$$
\begin{aligned}
&-2[\log L(\hat{\beta}_{red}, \hat{\sigma}^2_{red}) - \log L(\hat{\beta}_{full}, \hat{\sigma}^2_{full})] \\
&\qquad = \quad -2c_2(n)[-\log(RSS_{red}) + \log(RSS_{full})] \\
&\qquad = \quad n[\log(RSS_{red}) - \log(RSS_{full})]
\end{aligned}
$$

  will be[1] asymptotically χ^2 with df = the number of constraints (parameters set to zero) under H_0
  - Like the partial F test
  - Works only for nested models
  - Tends to become significant as n alone increases

[1]See any good book on stat theory, e.g. Casella & Berger (2008), Lehmann & Casella (2003), Lehmann & Romano (2006) or Bickel & Doksum (2015)

"Fixing" the LR test

- Two problems:
  - Larger sample size tends to ring the significance bell
  - Works only for nested models
- Fixes: penalized likelihood indices (small is good):

$$-2\log L(\hat{\beta},\hat{\sigma}^2) + (\text{complexity penality})$$

  - AIC: $-2\log L(\hat{\beta},\hat{\sigma}^2) + 2(p+2)$
  - CAIC: $-2\log L(\hat{\beta},\hat{\sigma}^2) + 2\frac{n+2}{n-p-1}(p+2)$
  - BIC: $-2\log L(\hat{\beta},\hat{\sigma}^2) + (\log n)(p+2)$
- Focus on differences, eg $AIC_{M1} - AIC_{M2}$
  - *Puzzle*: R replaces *p+2* with *p* when it calculates AIC or BIC… why doesn't this matter?

Comments[1] on AIC, CAIC, BIC

- AIC is motivated as an approximation to the K-L information distance between the linear model and the "true" distribution of the data
  - As sample size grows, AIC (and CAIC) tends to pick *the model that minimizes prediction error.*
  - In "small" samples, AIC picks models that are too complex. CAIC picks less complex models.
- BIC is motivated as an approximation to the log-posterior probability of the linear model when several models are considered
  - As sample size grows, BIC tends to pick *the true model.*
  - In "small" samples, BIC picks models that are too simple.

[1]Sheather pp. 230ff. sketches some details. For a more complete story, see Burnham & Anderson (2004).

Model comparison indices xIC (x = A, CA, B):

- Cannot tell you that a model fits well (or poorly)
- Can only tell you that one model fits better (or worse) than another *(smaller is better!)*
- Models to compare
  - Must be based on same data
  - Must be in the same "family" so any ignored "normalizing constants" $c_1(n)$, $c_2(n)$, etc., are the same
  - For variable selection in regression these are automatic[1]
- Rules of thumb for $\Delta(\text{xIC}) = \text{xIC}_{M1} - \text{xIC}_{M2}$:
  - $\Delta \sim 3$ might be interesting; $\Delta \sim 10$ might be compelling

[1]If the data has embedded NA's for different X's, R will choose different subsets of the data for different models – be careful!

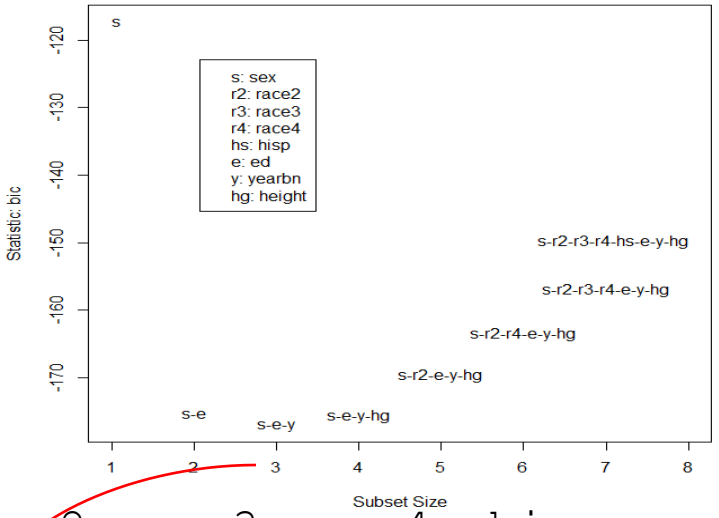
"Automatic" heuristics for searching model space: *All Subsets Regression*

- For each fixed number of predictors p, choose the model that minimizes RSS
  - This also optimizes R^2_{adj}, C_p, AIC, CAIC & BIC – why??
- Choose among the "winners" at each predictor set size, to get an overall winner
  - Typically want AIC or CAIC, and BIC, to be (near-) minimum at the same model – not always possible!
- In R[1]:
  - library(leaps): regsubsets(), summary(), coef()
  - library(car): subsets(); library(MASS): AIC(), BIC()

[1]https://www.statmethods.net/stats/regression.html

Example: heights.dta data

```
> heights.complete <-
+ heights[apply(heights,1,
+ function(x){!any(is.na(x))}),]
> all.subsets <-
+ regsubsets(log(earn+1) ~ .,
+ data=heights.complete)
> subsets(all.subsets)
> coef(all.subsets,1:8)
(Int)   sex    ed yearbn height race2 race3 race4  hisp
11.87 -2.14
 7.84 -2.05  0.29                                      AIC       BIC
 8.53 -2.08  0.30  -0.02                          7104.329  7130.463
 2.86 -1.66  0.30  -0.02   0.08                   7100.063  7131.424
 2.81 -1.66  0.30  -0.02   0.08  0.33
 2.81 -1.67  0.30  -0.02   0.08  0.33          0.99
 3.04 -1.69  0.30  -0.02   0.07  0.33 -0.68    0.99
 3.03 -1.69  0.30  -0.02   0.07  0.32 -0.68    0.99  0.00
```



"Automatic" heuristics for searching model space: *Stepwise Regression*

- Forward selection
  - Start with a "smallest" model
  - Add 1 term at a time that causes largest drop in xIC
    - Until no added term causes a drop in xIC
- Backwards elimination
  - Start with a "largest" model
  - Drop 1 term at a time that causes largest drop xIC
    - Until no dropped term causes a drop in xIC
- Both
  - Drop or add term that causes largest drop in xIC
    - Until no add or drop causes drop in xIC

Example: heights.dta data

```
> stepAIC(lm(log(earn+1) ~ .,data=heights),direction="both",k=2)
Error in stepAIC: number of rows in use has changed: remove
missing values?
> heights.complete <- heights[apply(heights,1,function(x)
+ {!any(is.na(x))}),]
> stepAIC(lm(log(earn+1) ~ .,
+ data=heights.complete),direction="both",k=2)
log(earn + 1) ~ sex + race + hisp + ed + yearbn + height
log(earn + 1) ~ sex +        hisp + ed + yearbn + height
log(earn + 1) ~ sex +               ed + yearbn + height
> n <- dim(heights.complete)[1]
> stepAIC(lm(log(earn+1) ~ .,
+ data=heights.complete),direction="both", k=log(n))
log(earn + 1) ~ sex + race + hisp + ed + yearbn + height
log(earn + 1) ~ sex +        hisp + ed + yearbn + height
log(earn + 1) ~ sex +               ed + yearbn + height
log(earn + 1) ~ sex +               ed + yearbn
```

Output greatly abbreviated!

Modern Variable Selection

Penalized Estimation[1] (a.k.a. "Regularization", "Shrinkage")

- We have seen that, e.g. under collinearity, $\hat{\beta}$'s become unstable and $SE(\hat{\beta})$ can explode
  - This leads to poor prediction error[1]
- If we can control the size of the $\hat{\beta}$'s , they will become more stable and $SE(\hat{\beta})$ will be controlled
  - Prediction error will actually be improved[1]
- Basic idea: Instead of maximizing $\log L(\beta, \sigma^2)$ we will maximize

$$\log L(\beta, \sigma^2) - \text{penalty}(\beta)$$

[1]Much useful detail at http://statweb.stanford.edu/~tibs/sta305files/Rudyregularization.pdf

Ridge Regression (a.k.a. L^2 penalty)

- Maximize
  $\log L(\beta, \sigma^2) - \lambda \sum_{j=0}^{p} \beta_j^2 = \log L(\beta, \sigma^2) - \lambda ||\beta||_2^2$
- We usually (and henceforth) replace $\log L(\beta, \sigma^2)$ with just RSS, so we want to minimize

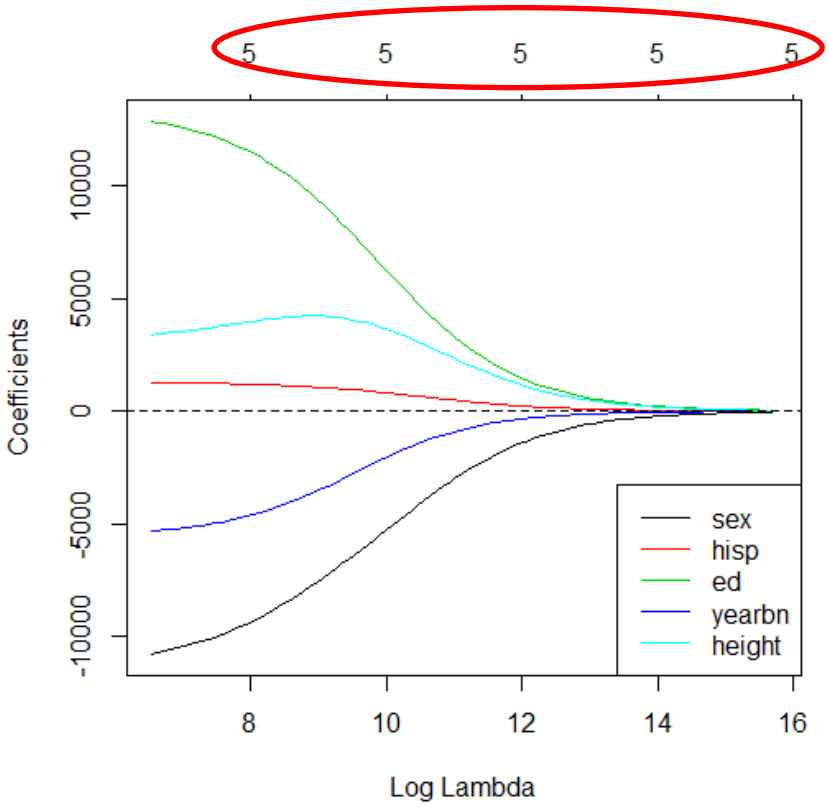
$$RSS + \lambda ||\beta||_2^2 = ||y - X\beta||_2^2 + \lambda ||\beta||_2^2$$

  - λ controls how much "regularization":
    - $\lambda = 0$: just ordinary LS
    - $\lambda \to \infty$: all β's equal 0
  - To treat all β's equally, should standardize columns of X
  - $|\beta| < 1$ not much affected; $|\beta| > 1$ reduced in magnitude
- In R: library(glmnet), function glmnet(…,alpha=0)

Example: heights.dta data

```
> Z <- heights.complete[,-c(1,3)]
> # omit "earn" and "race"
> Z <- apply(Z,2,
+ function(x) rescale(x,"full"))
> znames <- dimnames(Z)[2]
> z.ridge.fits <-
+ glmnet(Z,heights.complete[,1],
+ alpha=0)
> # alpha=0 for ridge regression
> plot(z.ridge.fits,
+ xvar="lambda")
> abline(h=0,lty=2)
> legend("bottomright",
+ legend=znames,col=1:5,lty=1)
```

How many nonzero β's



We omit **race** because it is a group of dummy variables that should be considered together. `glmnet` can't deal with that.

LASSO (a.k.a. L^1 penalty)

- Minimize

$$RSS + \lambda \sum_{j=0}^{p} |\beta_j| = ||y - X\beta||_2^2 + \lambda ||\beta||_1$$

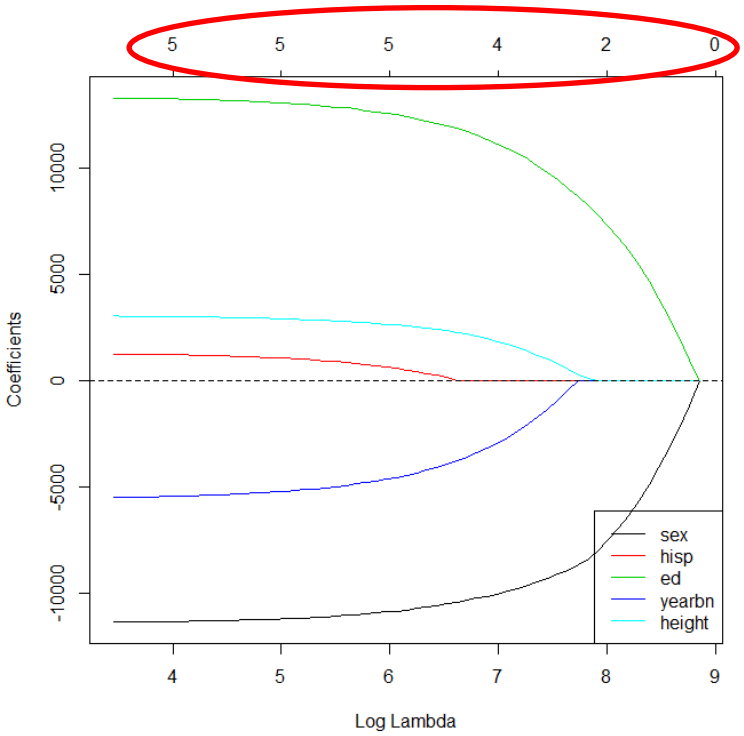
  - λ controls how much "regularization":
    - $\lambda = 0$: just ordinary LS
    - $\lambda \to \infty$: all β's equal 0
  - To treat all β's equally, should standardize columns of X
- All β's reduced in magnitude
- Geometry of L^1 distance means that smaller β's are forced to zero – *provides a variable selection tool*
- In R: library(glmnet), function glmnet(…,alpha=1)

https://newonlinecourses.science.psu.edu/stat508/lesson/5/5.4
https://stats.stackexchange.com/questions/74542/why-does-the-lasso-provide-variable-selection

Example: heights.dta data

```
> z.lasso.fits <-
+ glmnet(Z,heights.complete[,1],
+ alpha=1)
> # alpha=1 for lasso
> plot(z.lasso.fits,
+ xvar="lambda")
> abline(h=0,lty=2)
> legend("bottomright",
+ legend=znames,col=1:5,lty=1)
> coef(z.lasso.fits,s=exp(6:9))
```

How many nonzero β's



log λ = 6: sex, hisp, ed, yearbn, height
log λ = 7: sex, , ed, yearbn, height
log λ = 8: sex, , ed, ,
log λ = 9: Intercept only

Not found by xIC methods

Also found by xIC methods

A newer package, `gglasso`, can deal with groups of variables, like **race** (there are over 70 lasso packages on `https://cran.r-project.org`/!!)

Elasticnet: $\alpha L^1 + (1-\alpha)L^2$ penalty

- Minimize

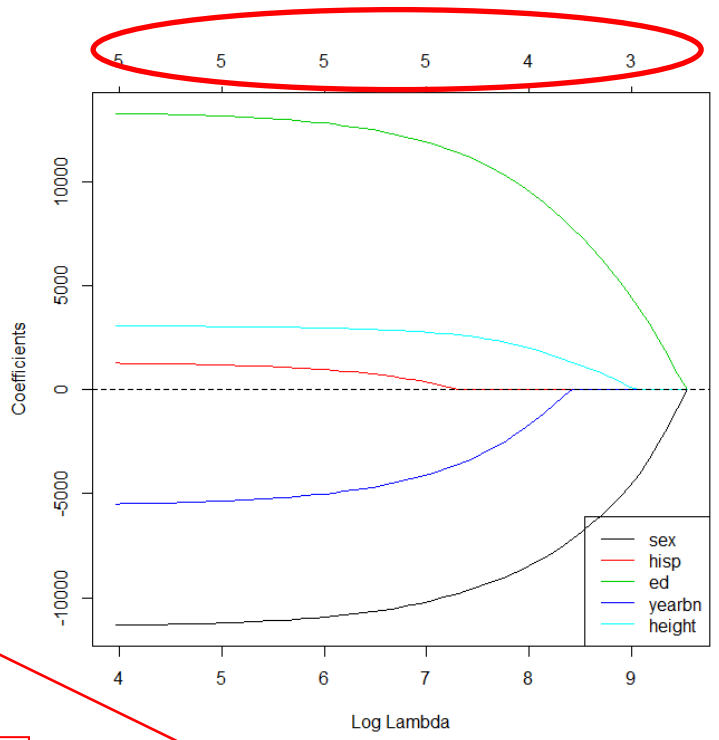
$$||y - X\beta||_2^2 + \lambda(\alpha||\beta||_1 + (1-\alpha)||\beta||_2^2)$$

  - λ controls how much "regularization":
    - $\lambda = 0$: just ordinary LS
    - $\lambda \to \infty$: all β's equal 0
  - To treat all β's equally, should standardize columns of X
- α controls tradeoff between lasso & ridge
- Smaller β's are forced to zero (lasso); $|\beta|>1$ tend to be more quickly reduced in magnitude (ridge)
- In R: library(glmnet), function glmnet(…,alpha=??)

Example: heights.dta data

```
> z.elasticnet.fits <-
+ glmnet(Z,heights.complete[,1],
+ alpha=0.5)
> # any 0<alpha<1 gives tradeoff
> plot(z.elasticnet.fits,
+ xvar="lambda")
> abline(h=0,lty=2)
> legend("bottomright",
+ legend=znames,col=1:5,lty=1)
> coef(z.elasticnet.fits,s=exp(7:9))
```

How many nonzero β's



log λ = 7: sex, hisp, ed, yearbn, height
log λ = 8: sex, , ed, yearbn, height
log λ ~ 9: sex, , ed, , height

xIC methods preferred yearbn to height

Also found by xIC methods

Are All Subsets and Stepwise really so different from Ridge and Lasso?

- Our penalized likelihood methods all try to

$$\text{minimize} \quad \left[-2\log L(\beta, \sigma^2) + \lambda||\beta||\right]$$

- Choices for $||\beta||$:

$$\text{Ridge:} \quad ||\beta|| = ||\beta||_2^2 = \sum_1^p \beta_j^2$$

$$\text{Lasso:} \quad ||\beta|| = ||\beta||_1 = \sum_1^p |\beta_j|$$

$$\text{Trad:} \quad ||\beta|| = ||\beta||_0 = \sum_1^p 1_{\{\beta_j \neq 0\}}$$

- Choices for λ:
  - λ = fixed value like $2p$, $p\log n$, or shrinkage plot choice
  - λ chosen by cross-validation

Are All Subsets and Stepwise really so different from Ridge and Lasso?

- *Ridge*: Control $\hat{\beta}$ and $SE(\hat{\beta})$ by

$$\text{minimizing } \left[-2\log L(\beta,\sigma^2) + \lambda||\beta||\right]$$

  - $||\beta|| = ||\beta||_2^2 = \sum_1^p \beta_j^2$ (squared L^2 distance)
  - λ chosen from shrinkage plot or CV

- *Lasso*: Control $\hat{\beta}$, $SE(\hat{\beta})$ and select variables by

$$\text{minimizing } \left[-2\log L(\beta,\sigma^2) + \lambda||\beta||\right]$$

  - $||\beta|| = ||\beta||_1 = \sum_1^p |\beta_j|$ (L^1 distance)
  - λ chosen from shrinkage plot or CV

Are All Subsets and Stepwise really so different from Ridge and Lasso?

- *AIC & BIC*: select variables by

$$\text{minimizing } \left[-2\log L(\beta,\sigma^2)+\lambda||\beta||\right]$$

  - $||\beta|| = ||\beta||_0 = \sum_1^p 1_{\{\beta_j \neq 0\}}$ (L^0 distance)
  - $\lambda = 2p$ (AIC) or $\lambda = p\log n$ (BIC)
- Minimizing with a discrete component $||\beta||_0$ is hard
  - *All Subsets* provides exact minimum
  - *Stepwise* provides a heuristic approx. to *All Subsets* when *p* is large

Inference After Variable Selection

Assessing/comparing models after variable selection

- After “all subsets”, “stepwise” or “lasso”, we cannot expect test statistics to have “textbook” distributions.
- It is difficult (and in some cases a matter of current research) to get valid CI’s for β’s, prediction intervals, etc., after variable selection.
- A better approach can be to measure how well the final model, or a set of candidate models, can predict new data.

Measuring predictive ability of a model

- RSS measures "predictive ability" on the same data set as the model was fitted on
  - We know RSS decreases whenever we add more variables
  - More "degrees of freedom" to minimize RSS
- Instead, we can split the data into
  - A training data set
  - A test data set

  Do whatever variable selection and model building we like on the training data set, and then assess/compare models on the (independent) test data set.

Choosing a Training Set & Test Set

- The split into training and test sets should be "uninformative" about the model
  - Conceptually simplest to do a random split
  - If that is impossible or undesirable, a systematic split that is uninformative is fine
- Rules of thumb[1] for sizes (see next slide also):

Sample Size	Speed of Model Selection	Ratio of Training Set to Test Set
Small	Slow	50/50
Moderate to Large	Moderate	60/40, 80/20, 90/10
Enormous	Fast	99/1, 99.5/0.5

[1]https://stackoverflow.com/questions/13610074/is-there-a-rule-of-thumb-for-how-to-divide-a-dataset-into-training-and-validatio

Some remarks on test set sample size

- You are always making a tradeoff between
  - Minimizing $SE(\hat{\beta})$, etc. (large training set)
  - Minimizing $SE(prediction\ error)$, etc. (large test set)
- Power calculation for $SE(pred.error)$ → n_{test}
  - Exact calculations depend on model complexity (df), etc.
  - $1000 \leq n_{test} \leq 10000$ is often safe (e.g. 80/20 split of n = 6000 → n_{test} = 1200; but even smaller n_{test} may suffice
- If nothing else, we can assess $SE(pred.error)$ by resampling / bootstrapping / etc. in the test set.

Example: heights.dta data

```
> c(.8,.2)*1371
[1] 1096.8  274.2
> indices <- sample(1:1371,size=275,replace=F)
> test.sel <- ifelse(1:1371 %in% indices,T,F); train.sel <- !test.sel
> Z.train <- Z[train.sel,]
> Z.test <- Z[test.sel,]
> X.train <- heights.complete[train.sel,]
> X.test <- heights.complete[test.sel,]
> log.earn.train <- log(X.train[,1]+1)
> log.earn.test <- log(X.test[,1]+1)
> train.all.subsets <-
+ regsubsets(log.earn.train ~ .,data=X.train[,-1])
> subsets(train.all.subsets)
> train.lasso <- glmnet(Z.train,log.earn.train,alpha=1)
> plot(train.lasso,xvar="lambda")
> legend("bottomright",
+ legend=names(heights.complete[-c(1,3)]),col=1:5,lty=1)
```

Set up training and test data sets

Model selection using all-subsets and lasso…

Models are listed on the next slide…

Example: heights.dta data

```
> ## skipping all of the diagnostics that I would normally do...
> ##   - nonlinearity
> ##   - nonnormality
> ##   - non-constant variance
> ##   - leverage
> lm.1 <- lm(log.earn.train ~ sex + ed,data=X.train)
> lm.2 <- lm(log.earn.train ~ sex + ed + yearbn,data=X.train)
> lm.3 <- lm(log.earn.train ~ sex + ed + yearbn +
+ height,data=X.train)
>
> BIC(lm.1) # [1] 5672.953
> BIC(lm.2) # [1] 5675.221
> BIC(lm.3) # [1] 5677.121
>
> AIC(lm.1) # [1] 5652.955
> AIC(lm.2) # [1] 5650.224
> AIC(lm.3) # [1] 5647.124
> ## AIC and BIC are telling opposite stories, on the training data...
```

Which model seems to give best xIC?

Example: heights.dta data

```
> yhat.1 <- predict(lm.1,newdata=X.test)
> yhat.2 <- predict(lm.2,newdata=X.test)
> yhat.3 <- predict(lm.3,newdata=X.test)
```

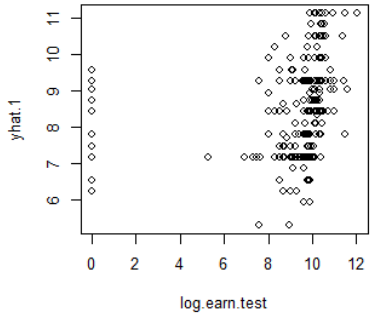
Check prediction error

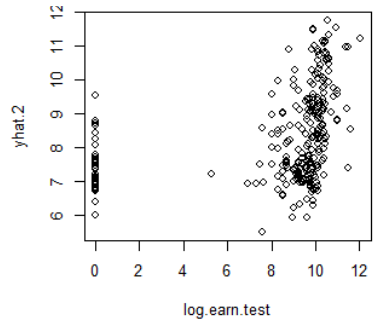
```
> (PSE.1 <- sum((log.earn.test-yhat.1)^2))
[1] 2880.098
> (PSE.2 <- sum((log.earn.test-yhat.2)^2))
[1] 2841.281
> (PSE.3 <- sum((log.earn.test-yhat.3)^2))
[1] 2835.232
```

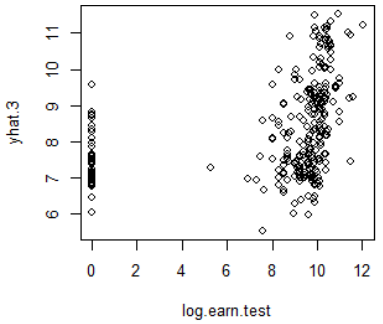
Agrees with AIC…

```
> plot(log.earn.test,yhat.1)
> plot(log.earn.test,yhat.2)
> plot(log.earn.test,yhat.3)
```

How good do the yhats look?







Looking at -- and thinking about -- the data tells us there's much more to do!

Extension: K-fold cross-validation

- In the heights example we made an 80/20 split of the data: 80% for training, 20% for testing.
- We can make this 80/20 split 5 times with 5 disjoint tests sets.
  - Each time, compute the prediction squared error (PSE) or MSE = PSE/(size of test set) = PSE/n_{test}
  - Average these to produce a more stable estimate of prediction error (instead of making n_{test} larger)
- This is 5-fold cross-validation.

More on K-fold cross validation…

- For any K, we can split the data into K parts. Each part can be a test set, with the remaining data the training set.
  - We average PSE, MSE, or some other measure over the K cross-validation trials
- There is theory that says what the best K is, but in practice K = 5 or 10 is usually sufficient.
- Often K-fold cross-validation is implemented by randomly splitting the data, so different runs give slightly different answers.

Example: heights.dta again

```
# recall that heights.complete omits all observations with NA's
> library(boot) # for the cv.glm() function…
> clm.1 <- glm(log(earn+1) ~ sex + ed,data=heights.complete)
> clm.2 <- glm(log(earn+1) ~ sex + ed + yearbn,data=heights.complete)
> clm.3 <- glm(log(earn+1) ~ sex + ed + yearbn + height,
+   data=heights.complete)

> cv.glm(heights.complete,clm.1,K=5)$delta[1]
[1] 10.21179
> .Last.value * dim(heights.complete)[1]/5
[1] 2800.073
> cv.glm(heights.complete,clm.2,K=5)$delta[1]
[1] 10.16831
> .Last.value * dim(heights.complete)[1]/5
[1] 2788.151
> cv.glm(heights.complete,clm.3,K=5)$delta[1]
[1] 10.14349
> .Last.value * dim(heights.complete)[1]/5
[1] 2781.435
```

Refit models on "full" data set

Cross-val MSE for each model…

Rescale to PSE if we wish…

Conclusions similar to AIC…

What Do I Really Do?

Challenges of Consulting/Collaboration

- The people you work with will think
  - You are a "**high priest**" of variable selection and you know the "**right**" way to do it!
  - You can provide them with statistical cover for whatever model they *really really* want
- These are contradictory, and your collaborators will want both… Neither is true!
- Your job is to come up with the model that
  - Best reflects the substance (science, engineering, policy…)
  - Best satisfies modeling assumptions
  - Is most clearly indicated by the data

What do I do in practice[1]?

- Have a good conversation with my collaborator / client: Which variables are
  - Scientifically or policy-wise important
    - I will try to keep these in, or discuss eliminating with client
  - Related to design of the experiment/data collection
    - These must stay in the model
- Use t, F, xIC, best subsets, stepwise, lasso, etc. to see what the data will support
  - Use this together with knowledge of subject area to come up with 2-5 models
  - Conversation with client decides final model

[1]The whole process may iterate several times before settling on a "final" model!

Heuristic Principles[1] from Gelman & Hill (2009, p. 69)

1. Include all input variables that, for substantive reasons, might be expected to be important in predicting the outcome.
2. Sometimes inputs can be combined—for example, several inputs can be averaged or summed to create a "total score" that replaces them.
3. Inputs with large main effects, often have large interactions as well.
4. Looking at the t-statistics and signs of individual $\hat{\beta}$'s:
    a) If a predictor is not statistically significant and has the expected sign, it is generally fine to keep it in. It may not help predictions much but is also probably not hurting them.
    b) If a predictor is not statistically significant and does not have the expected sign (for example, incumbency having a negative effect on vote share), consider removing it.
    c) If a predictor is statistically significant and does not have the expected sign, then think hard if it makes sense. (E.g. you were expecting more tutoring to improve test scores, but students who sought more tutoring got lower scores). Try to gather data on potential lurking variables and include them in the analysis.
    d) If a predictor is statistically significant and has the expected sign, then by all means keep it in the model.

[1]These do not solve all problems!

Summary

- Variable selection – An Overview
- Traditional variable selection
  - MSE-based indices
  - Likelihood-based indices
  - Stepwise and all-subsets
- Modern variable selection
  - Variable selection by penalized likelihood
  - Are All Subsets and Stepwise really so different from Ridge and Lasso?
- Inference after variable selection
- What do I really do?