Cross-Validation
&
Regularized Regression

"Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani "

Overview

- Subset Selection of Predictors
- Cross-Validation
- K-fold Cross-Validation

- Bias-Variance Tradeoff
- Regularized Regression
  - Lasso
  - Ridge

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

I want to pare down my model!

Subset selection - choose subset of p predictors

Regularization – keep p predictors, shrink coefficient estimates towards 0 (some variable selection for Lasso)

Dimension Reduction – Project p predictors into M-dim space where $M < p$

Subset Selection

- Best subset: Try every model. Every possible combination of *p* predictors
  - Computationally intensive, especially for *p* large
  - Also, huge search space. Higher chance of finding models that look good on training data but have little predictive power on future data

- Stepwise
  - In practice, commonly done
  - Forward, Backward, Forward + Backward

Subset Selection - Forward Stepwise

$M_0 \rightarrow M_1 \rightarrow M_2 \rightarrow \ldots\ldots \rightarrow M_p$

+1 predictor with smallest RSS ⇔ largest R^2

+1 predictor with smallest RSS ⇔ largest R^2

Now we have *p* candidate models

Are RSS and R^2 good ways to decide amongst the *p* candidates?

Subset selection

Choosing among *p* candidate models…

- Cross-validation - always a great standby
- Mallow's C_p
- AIC
- BIC
- Adjusted R^2

OLS Regression Results

Dep. Variable:	y	R-squared:	0.933
Model:	OLS	Adj. R-squared:	0.928
Method:	Least Squares	F-statistic:	211.8
Date:	Mon, 03 Nov 2014	Prob (F-statistic):	6.30e-27
Time:	14:45:06	Log-Likelihood:	-34.438
No. Observations:	50	AIC:	76.88
Df Residuals:	46	BIC:	84.52
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf.	Int.]
x1	0.4687	0.026	17.751	0.000	0.416	0.522
x2	0.4836	0.104	4.659	0.000	0.275	0.693
x3	-0.0174	0.002	-7.507	0.000	-0.022	-0.013
const	5.2058	0.171	30.405	0.000	4.861	5.550

Omnibus:	0.655	Durbin-Watson:	2.896
Prob(Omnibus):	0.721	Jarque-Bera (JB):	0.360
Skew:	0.207	Prob(JB):	0.835
Kurtosis:	3.026	Cond. No.	221.

Subset selection

$$C_p = \frac{1}{n}(RSS + 2p\hat{\sigma}^2)$$

Mallow's Cp
p is the total # of parameters
$\hat{\sigma}^2$ is an estimate of the variance of the error, ε

$$AIC = -2logL + 2 \cdot p$$

L is the maximized value of the likelihood function for the model estimated

$$BIC = \frac{1}{n}(RSS + log(n)p\hat{\sigma}^2)$$

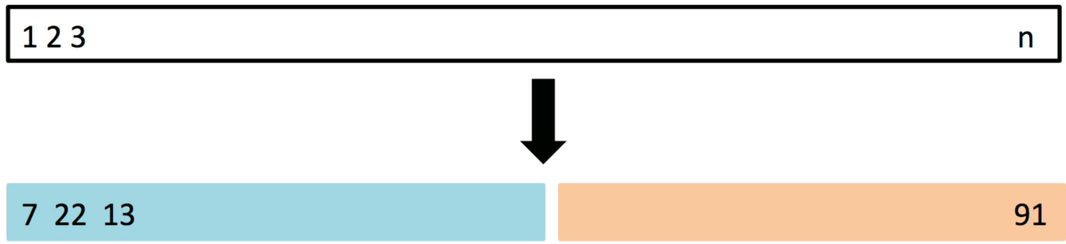
This is AIC, except 2 is replaced by log(n). log(n) > 2 for n>7, so BIC generally exacts a heavier penalty for more variables

$$Adjusted\ R^2 = 1 - \frac{RSS/(n-p-1)}{TSS/(n-1)}$$

Similar to R^2, but pays price for more variables

Can show AIC and Mallow's Cp are equivalent for linear case

Cross-Validation



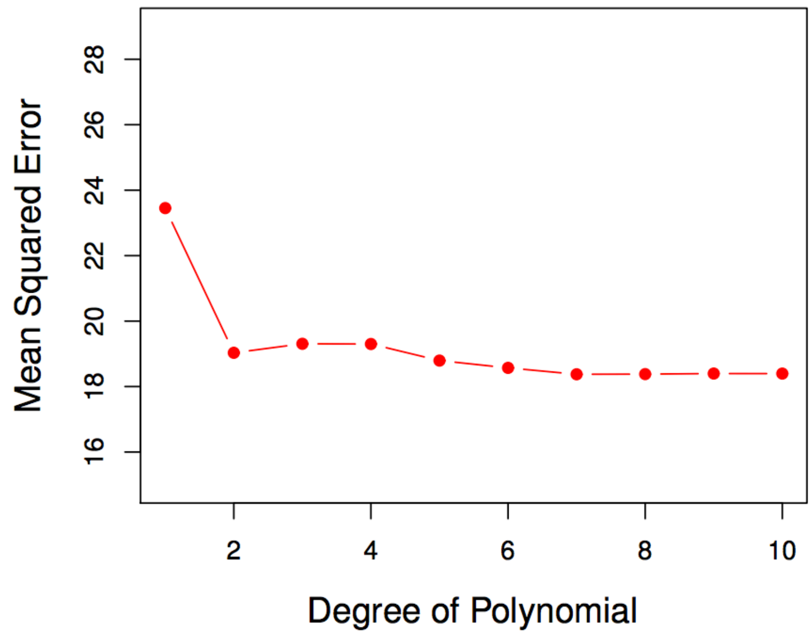
Randomly divide data into training set and validation set

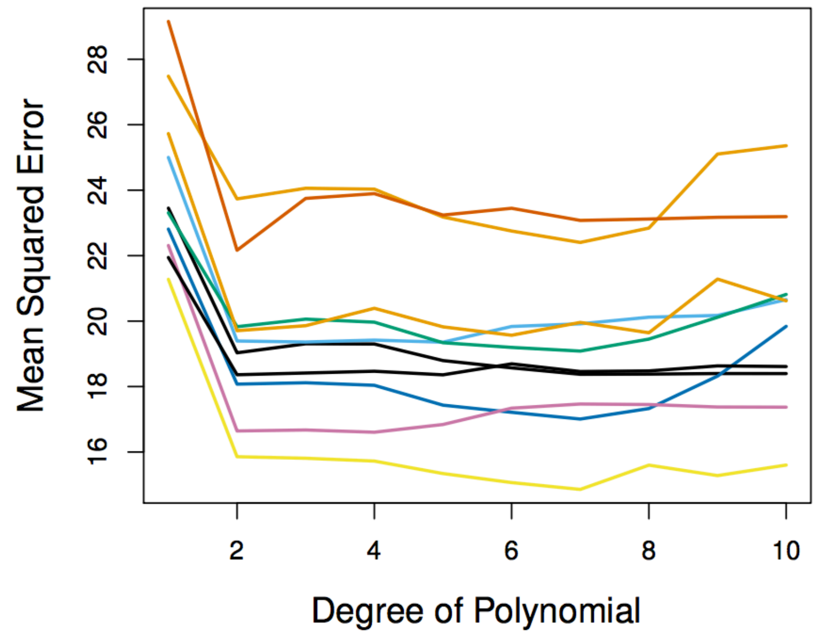
- 50/50, 60/40, 70/30, 80/20, no rule…

1. Fit model on training set
2. Use fitted model in 1. to predict responses for validation set
3. Compute validation-set error
   - Quantitative Response: Typically MSE
   - Qualitative Response: Typically Misclassification Rate

Why might validation-set error rate underestimate test-set error rate?

Cross-Validation

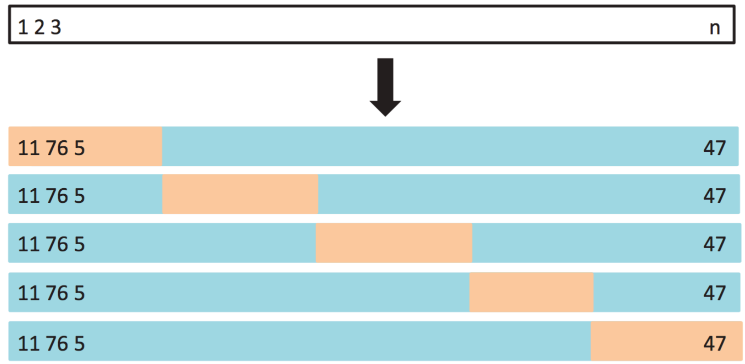




- Fitting MPG (Y) from Horsepower (X)
- Try different polynomial fits
  - Y~X+X^2
  - Y~X+X^2+X^3
  - Y~X+X^2+X^3+X^4

- Validation error can be highly variable depending on random split

K-Fold Cross-Validation



Randomly divide data into K=5 folds. Typically choose K=5 or 10.

Run K times

1. Fit model on training set, using (K-1) folds
2. Use fitted model in 1. to predict responses for validation set, 1 of the folds
3. Compute validation-set error
   - Quantitative Response: Typically MSE
   - Qualitative Response: Typically Misclassification Rate

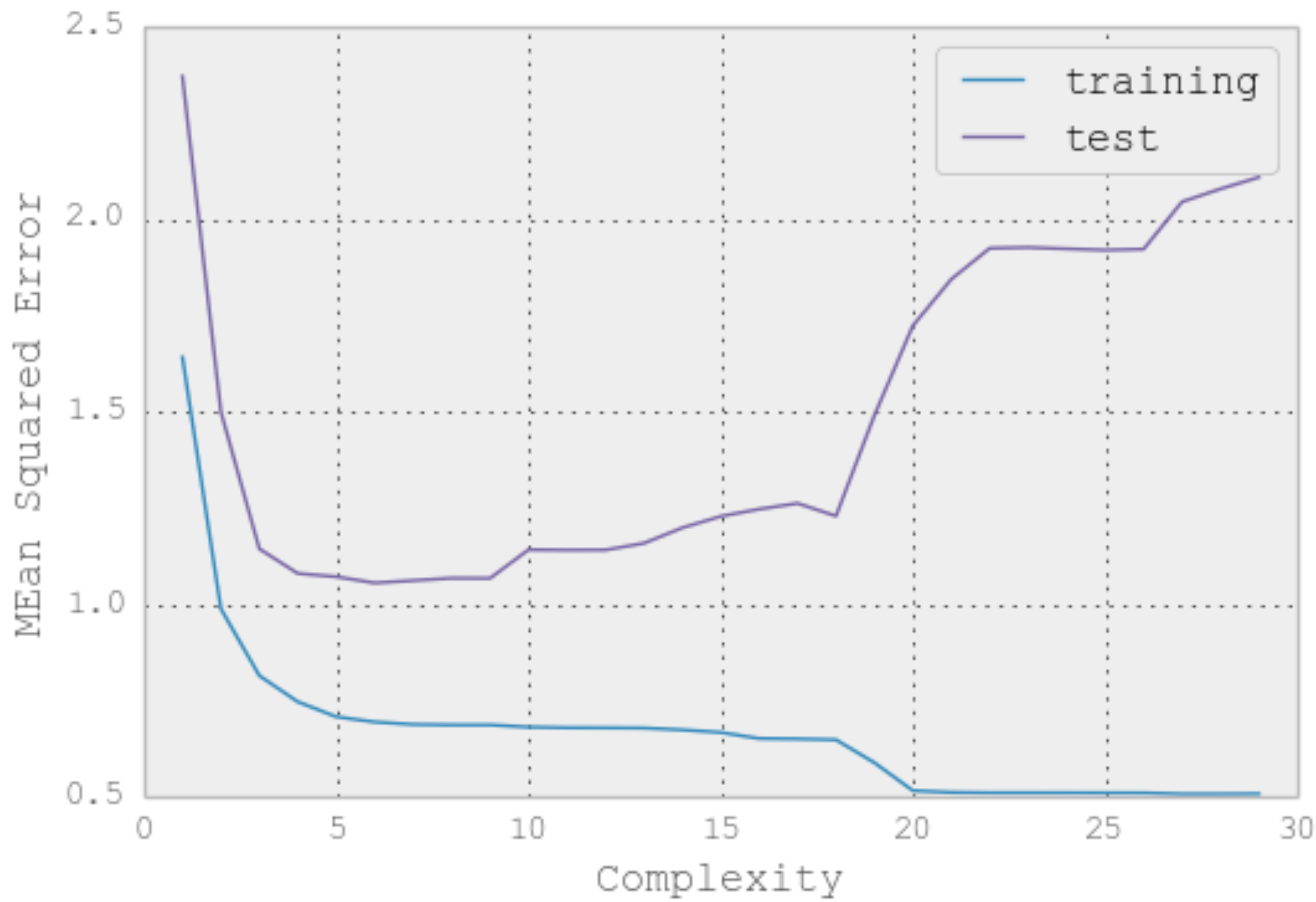
$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i$$

K-Fold Cross-Validation

TRY STUFF
- tune parameter 1
- tune parameter 2
- feature engineering
- feature selection
- scale factors 1 way
- scale factors another way
- etc. etc.

Test Set

Don't touch until end for final evaluation.
Gives best estimate of future error.



training
test
MEan Squared Error
Complexity
2.5
2.0
1.5
1.0
0.5
0
5
10
15
20
25
30

Bias-Variance Tradeoff

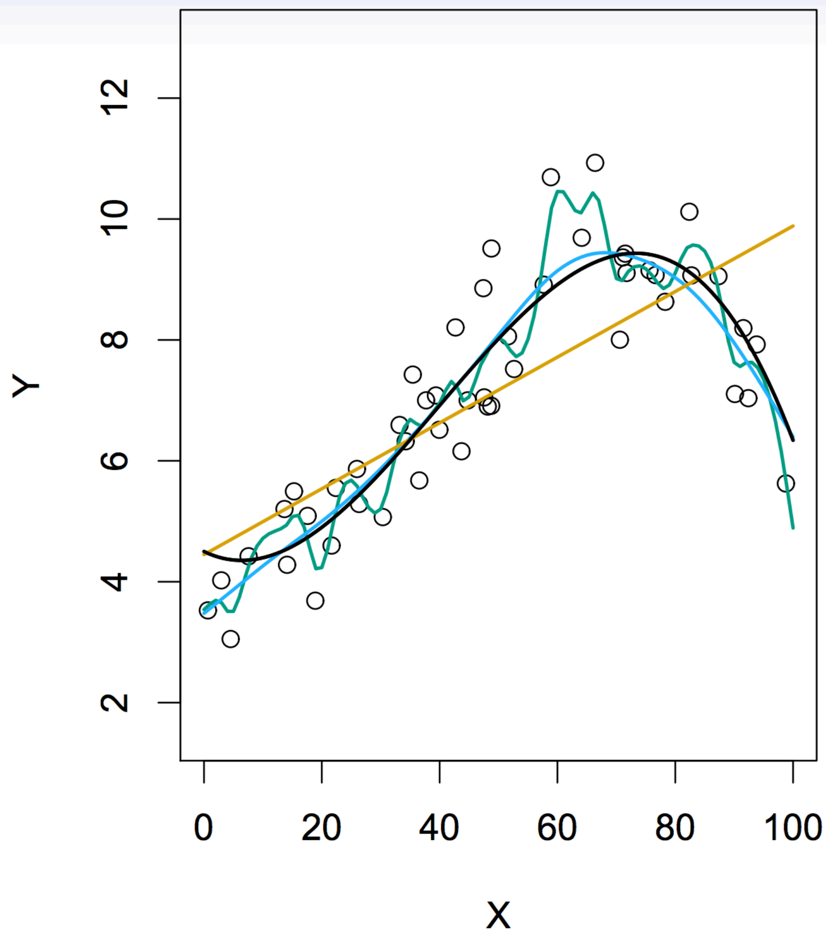
Suppose we fit a model $\hat{f}(x)$ to some training data
Let (x_0, y_0) be a test observation from the population.
If the true model is $Y = f(X) + \epsilon$, where $f(x) = E(Y|X = x)$ then…

$$\boxed{E(y_0 - \hat{f}(x_0))^2 = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon)}$$

Ok….what is going on here?

- Applies to modeling in general, beyond Linear Regression
- Want your model to minimize the expected test MSE on LHS. But how?
  - Var(ε), or "Irreducible Error". Can't do anything about that!
  - Can reduce Variance
  - Can reduce Bias

Bias-Variance Tradeoff



$$\mathrm{Var}(\hat{f}(x_0))$$

Amount by which $\hat{f}$ would change if estimated it using a different training dataset

$$\mathrm{Bias}(\hat{f}(x_0))] = E[\hat{f}(x_0)] - f(x_0)$$

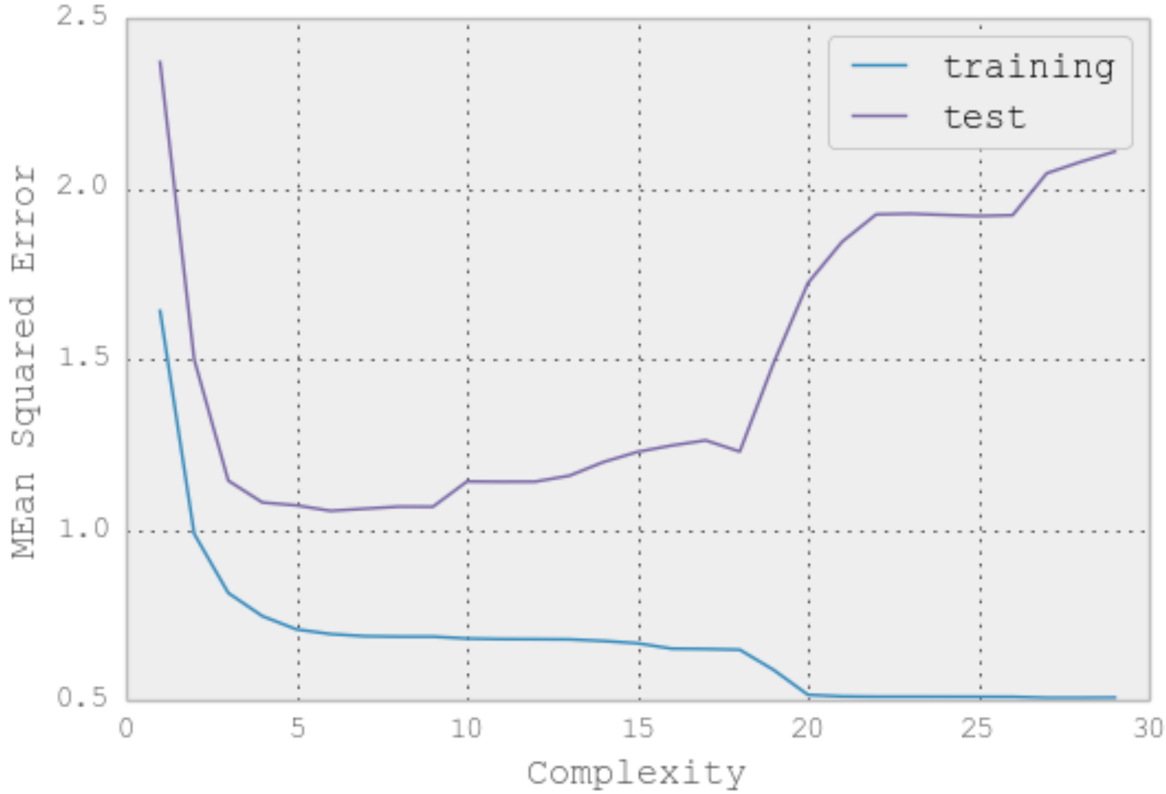
Difference between expected prediction of our model and correct value we are trying to predict

$$\mathrm{Var}(\epsilon)$$

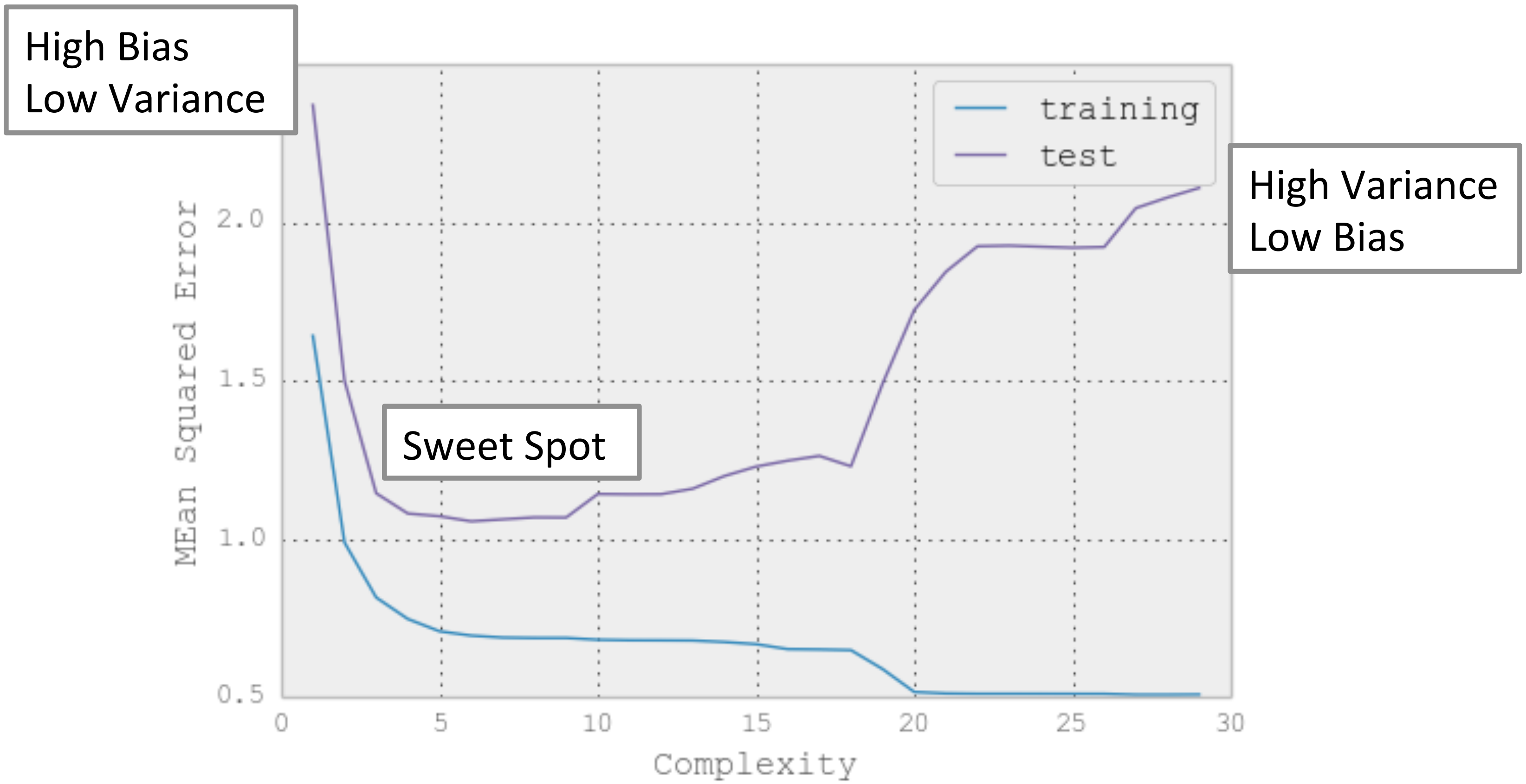
Simply because $Y = f(X) + \epsilon$

Generally speaking, the *more flexible* the model, the *greater the variance*.

Model Framework - Evaluation



Model Framework - Evaluation



- Can break this complexity tradeoff into what we call “bias” and “variance”

Managing the Bias-Variance Tradeoff with Linear Regression

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

I want to pare down my model, reducing the variance!

Subset selection - choose subset of p predictors

Regularization – keep p predictors, shrink coefficient estimates towards 0 (some variable selection for Lasso)

Dimension Reduction – Project p predictors into M-dim space where M < p

Regularization – Ridge regression

In Linear Regression, we find the estimates for $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ that minimize….

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

In Ridge Regression, we find the estimates for $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ that minimize….

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = \mathrm{RSS} + \boxed{\lambda \sum_{j=1}^{p} \beta_j^2}$$

Regularization – Ridge regression

In Linear Regression, we find the estimates for $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ that minimize....

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

In Ridge Regression, we find the estimates for $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ that minimize....

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = \mathrm{RSS} + \boxed{\lambda \sum_{j=1}^{p} \beta_j^2}$$

λ is tuning parameter to be determined!

j is not zero!

Regularization – Lasso regression

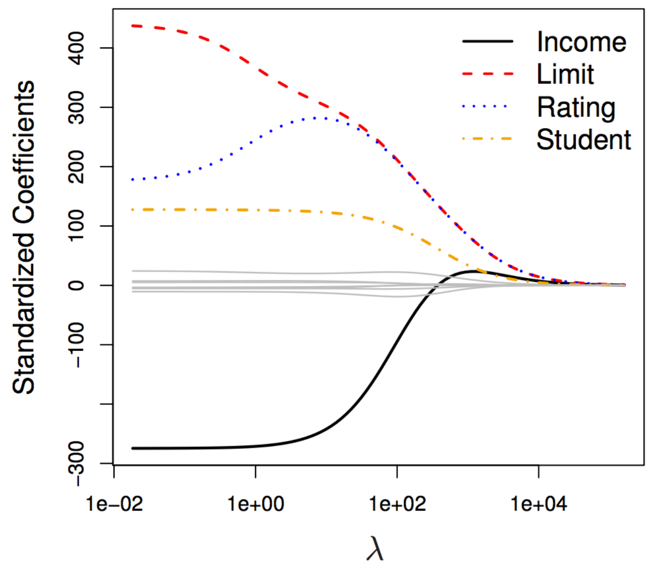
In Lasso Regression, we find the estimates for $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ that minimize….

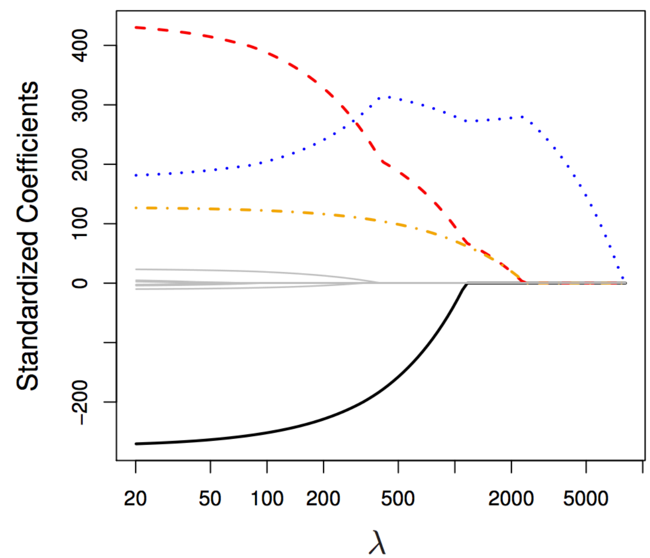
$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \mathrm{RSS} + \boxed{\lambda\sum_{j=1}^{p}|\beta_j|}$$

Very similar to Ridge!
Except we use an "L1" penalty instead of an "L2" penalty

L1 is also known as least absolute deviations
L2 is also known as least squares

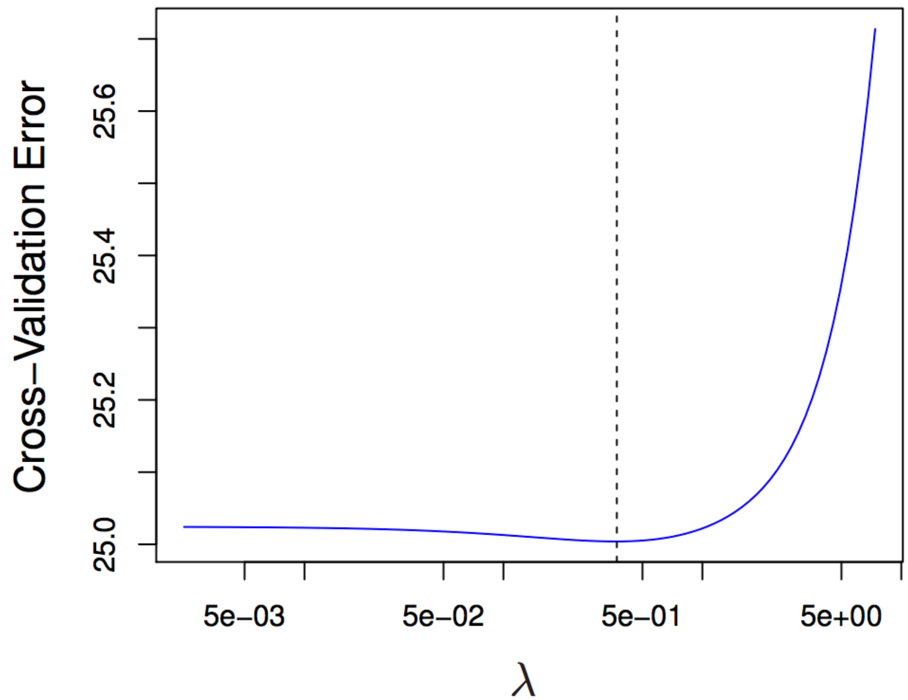
Ridge vs. Lasso

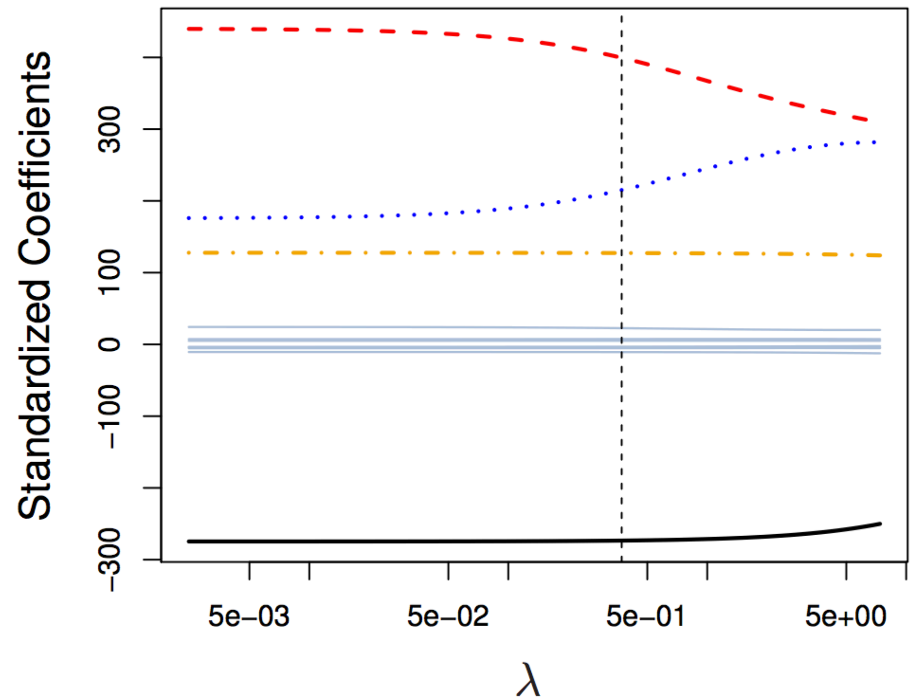




- When λ = 0, we simply have linear models.
- As λ increases, both models become less flexible, reducing variance, but increasing bias.
- Lasso has the advantage of variable selection as well (especially nice when *p* is large)
- Neither universally dominate, but in general one might expect Lasso to do better when response is function of relatively few predictors.
  - Of course you never actually know this, so use your friend, cross-validation!

Choosing λ





- Just increment λ along, fit a large number of models (1 per increment), and choose λ which minimizes cross-validated error, and voila! You have your corresponding optimized model for Ridge Regression.

Don't forget...

- In standard least squares *Linear Regression*, the beta coefficient estimates are scale equivariant.

  In other words, multiplying X_j by constant c leads to scaling of least squares coefficient estimates by 1/c, so that $X_j\hat{\beta}_j$ remains the same

- In *Ridge Regression*, the beta coefficient estimates can change substantially due to the penalty part of the ridge cost function.

  Therefore, it's best to first standardize the predictors using:

$$\boxed{\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij}-\overline{x}_j)^2}}}$$

Questions

- What is training error? validation error? test error?
- What are the steps to cross-validation?
  - How would you use it to compare say ***p*** different models?
- Same question as above, except with K-fold cross-validation
- What is the Bias-Variance tradeoff?
  - What happens with Bias and Variance at **low** levels of complexity?
  - What happens with Bias and Variance at **high** levels of complexity?
- How do Ridge and Lasso attempt to win at the Bias-Variance tradeoff?
  - What's being penalized exactly?