Lecture 6: Model Selection and Regularization

STAT 1361/2360: Statistical Learning and Data Science

University of Pittsburgh
Prof. Lucas Mentch

Overview

- The next three sets of lectures will follow the book fairly closely (ISLR Chapters 6-8)
- This lecture coincides with Chapter 6 – BIG chapter
  - Part I: Linear Model Selection (AIC, BIC, Adjusted R^2, C_p)
  - Part II: Shrinkage and Regularization (Ridge, Lasso)
  - Part III: Dimension Reduction (PCA, PLS)
  - Part IV: High Dimensional Issues
  - Part V: Degrees of Freedom and Signal-to-Noise Ratios (not covered in book)
- As motivation, let's start by reviewing cross-validation:

k-fold Cross-validation

Big Idea: Suppose we want to estimate the "generalizability error" of a particular model (e.g. Linear Regression with 2 predictors, Linear Regression with 3 predictors, kNN for some fixed k)

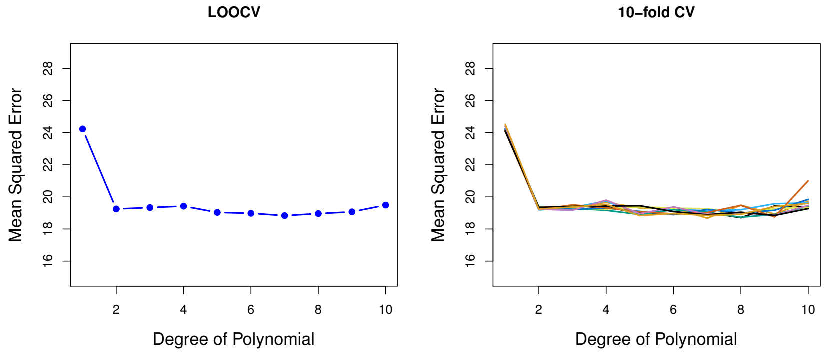
- Split data into k groups
- Construct the model k total times, each time holding out one group to use as the test set
- In doing so, obtain k estimates of the test error
- Average over these k estimates to obtain the final estimate of generalizability error

k-fold Cross-validation

- k-fold Cross-validation gives us a robust way of estimating generalizability error of a particular model
  - Much better than using only training error or even a single train/test split

- Very useful in terms of deciding which models are outperforming others
  - Can be used to compare across different kinds of models (e.g. Logistic Regression vs. LDA v.s. QDA) or choosing the “best" from a class of models (e.g. choosing the value of a tuning parameter like the k in kNN or the number of terms in linear models)

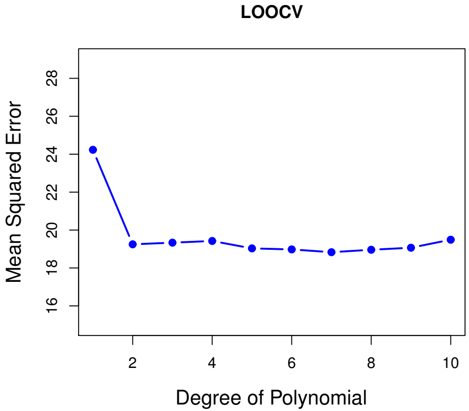
Cross-validation Example



ISLR Fig. 5.4: Cross-validation error (MSE) from a linear model when different degrees of polynomial for one covariate (feature) are included in the model. Left: LOOCV. Right: Several different runs of 10-fold CV.

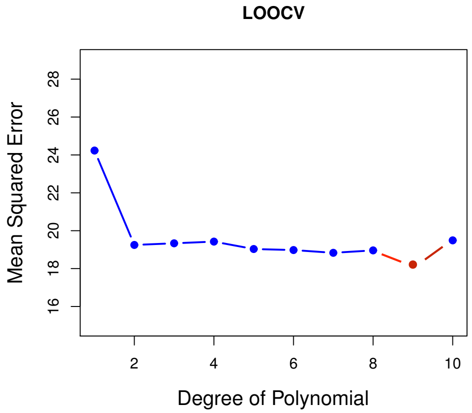
Cross-validation Example

Let's look closer at the left hand side – what would this tell us about which model we might want to use?



Cross-validation Example

What if we altered that cross-validation plot just slightly?



Model Selection

- Remember when we talked about model selection with cross-validation (CV), we said we were *only* interested in model fit
  - Ignored practical concerns such as (i) model complexity and (ii) understanding what kinds of scientific questions we might want/need to be able to answer
  - This is rarely the case – would we really want to fit a linear model with 9 polynomial terms for only a slight bump in accuracy over a model with only 2? Probably not.

Model Selection

- Another thing to keep in mind: recall that the whole purpose of doing CV in the first place was to get several estimates of test error and average them
- The reason for the train-test split in the first place was that the training error will be optimistic. In other words, $\text{Err}_{Train} < \text{Err}_{Test}$
- Equivalently, we could write $\text{Err}_{Train} + c = \text{Err}_{Test}$
- Note that c will be a function of model complexity $\implies$ the more complex the model, the better the training data / model will (over) fit
- In Part I of Chapter 6, we'll look at several measures of this form – these will be in the context of choosing an optimal linear regression model, though many of these measures have analogues or can be generalized for other types of models

Part I: Linear Model Selection

Linear Model Review

Recall that with a **linear model** we have n ordered pairs (X, Y), where each $X = (X_1, \ldots, X_p)$ where

- $Y \in \mathbb{R}$ is the continuous response
- $X_1, \ldots, X_p$ are the predictor variables (covariates/features)
- We will consider modeling the response Y as a linear function of $d \leq p$ of the predictors with parameters (coefficients) $\beta = (\beta_0, \ldots, \beta_d)$:

$$Y = f(X) + \epsilon = \beta_0 + \beta_1 X_1 + \cdots + \beta_d X_d + \epsilon$$

- Define $\sigma^2 = \text{var}(\epsilon)$ (the variance of the irreducible error) which we estimate with $\hat{\sigma}^2$

Linear Model Review

Recall a few facts about linear models:

- We don't necessarily believe that the true model is linear, but we think there could be a good linear approximation
- The d predictor variables $X_1, \ldots, X_d$ we use in the model could be d separate variables entirely, or they could be functions of other predictors.
  - E.g. we could have $X_2 = X_1^2$ (polynomial term) or $X_3 = X_1 X_2$ (interaction term)
  - Doesn't really matter what the covariates are – a linear model need only be linear in the parameters β

Linear Model Review

- We define the *total sum of squares* as

$$TSS = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

and the *residual sum of squares* as

$$RSS = \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 = \sum_{i=1}^{n} \big(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_d x_{di})\big)^2$$

- With ordinary least squares (OLS), we choose the coefficient estimates $\hat{\beta}_{\text{OLS}}$ as the values of β that minimize the RSS

Linear Model Review

Finally, recall again that the training error is going to *underestimate* the generalizability (test) error

- More flexible models are always going to look better – in the case of linear models, models with more terms (predictors/coefficients) are more flexible and thus will have lower RSS and R^2
- Again, we can think of the following 4 measures as “adjustments" to the training error that account for this fact and adjust for models with more predictors

Mallow's C_p

Measure 1 Mallow's C_p =

$$\frac{1}{n}\left(RSS + 2d\hat{\sigma}^2\right)$$

Mallow's C_p

Measure 1 Mallow's C_p =

$$\frac{1}{n}\left(RSS + \underbrace{2d\hat{\sigma}^2}_{\text{Penalty Term}}\right)$$

Mallow's C_p

Measure 1 Mallow's C_p =

$$\frac{1}{n}\left(RSS + \underbrace{2d\hat{\sigma}^2}_{\textit{Penalty Term}}\right)$$

- We add a *penalty* to the RSS that increases with the number of parameters d to account for the fact that RSS decreases with d

Mallow's C_p

Measure 1 Mallow's C_p =

$$\frac{1}{n}\left(RSS + \underbrace{2d\hat{\sigma}^2}_{\textit{Penalty Term}}\right)$$

- We add a *penalty* to the RSS that increases with the number of parameters d to account for the fact that RSS decreases with d
- Can think of this as an adjustment to the training error that is meant to estimate the test error (if $\hat{\sigma}^2$ is unbiased for σ^2, then Mallow's C_p is an unbiased estimate of the test error)

AIC

Measure 2 (Akaike Information Criterion) AIC =

$$\frac{1}{n\hat{\sigma}^2}\left(RSS + 2d\hat{\sigma}^2\right)$$

- Note that this is proportional to Mallow's C_p (true for models fit via least squares)
- **Note:** AIC is technically only defined for models fit via maximum likelihood; this definition though is proportional to the actual definition

BIC

Measure 3 (Bayesian Information Criterion) BIC =

$$\frac{1}{n}\left(RSS + \log(n) d \hat{\sigma}^2\right)$$

BIC

Measure 3 (Bayesian Information Criterion) BIC =

$$\frac{1}{n}\left(RSS + \log(n)d\hat{\sigma}^2\right)$$

- Here, the penalty term is $\log(n)d\hat{\sigma}^2$ instead of the $2d\hat{\sigma}^2$ with Mallow's C_p

Whenever $n > 7, \log(n) > 2$

$\implies \log(n)d\hat{\sigma}^2 > 2d\hat{\sigma}^2$

$\implies$ BIC assigns a harsher penalty than C_p and AIC

$\implies$ BIC will “prefer" models with fewer parameters

Measures 1-3 Recap

- Same ultimate goal for each measure: adjust the training error in order to account for model complexity
    - If we have two models M_1 and M_2 with similar (training) accuracy but M_1 uses fewer predictor variables (less complexity), then M_1 would be preferred
- Each measure can be seen as an estimate of the test error
    - Smaller values would be preferred
    - Which other method we recently discussed at length could fit with this list?

Measures 1-3 Recap

- Same ultimate goal for each measure: adjust the training error in order to account for model complexity
  - If we have two models M_1 and M_2 with similar (training) accuracy but M_1 uses fewer predictor variables (less complexity), then M_1 would be preferred
- Each measure can be seen as an estimate of the test error
  - Smaller values would be preferred
  - Which other method we recently discussed at length could fit with this list? **Cross-Validation!** (Remember, CV gives estimate of test error, but doesn't penalize more complex methods)

Adjusted R^2

Measure 4 Adjusted R^2 =

$$1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$$

Adjusted R^2

Measure 4 Adjusted R^2 =

$$1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$$

- Recall (non-adjusted) $R^2 = 1 - \frac{RSS}{TSS}$

Adjusted R^2

Measure 4 Adjusted R^2 =

$$1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$$

- Recall (non-adjusted) $R^2 = 1 - \frac{RSS}{TSS}$
- Since RSS decreases with more predictors (larger d), R^2 increases (gets closer to 1)
- Here we account for d, so *adjusted* R^2 only increases with d when those additional predictors make the model substantially more accurate

Measures 1-4 Recap

These measures give us a way to "objectively" choose a good model while balancing accuracy and complexity

- Given a set of k candidate models $M_1, ..., M_k$, choose the model with the lowest C_p, AIC, BIC, or CV error, or the model with the highest adjusted R^2
- Is each measure guaranteed to select the same model?

Measures 1-4 Recap

These measures give us a way to “objectively" choose a good model while balancing accuracy and complexity

- Given a set of k candidate models $M_1, \ldots, M_k$, choose the model with the lowest C_p, AIC, BIC, or CV error, or the model with the highest adjusted R^2
- Is each measure guaranteed to select the same model? **NO!** – but usually won’t choose drastically different models
- Ok, so now we have a way to choose between models – How do we go about finding those “candidate" linear models in the first place? A few different methods ...

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Given a total of p possible predictors, build every possible model using every possible subset of the p predictor variables. More specifically,

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Given a total of p possible predictors, build every possible model using every possible subset of the p predictor variables. More specifically,

1. Construct *null model* M_0 with no predictors

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Given a total of p possible predictors, build every possible model using every possible subset of the p predictor variables. More specifically,

1. Construct *null model* M_0 with no predictors
2. For $i = 1, \ldots, p$, construct all $\binom{p}{i}$ possible models containing i predictors. Denote the best among these by M_i

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Given a total of p possible predictors, build every possible model using every possible subset of the p predictor variables. More specifically,

1. Construct *null model* M_0 with no predictors
2. For $i = 1, \ldots, p$, construct all $\binom{p}{i}$ possible models containing i predictors. Denote the best among these by M_i

   How to choose the best?

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Given a total of p possible predictors, build every possible model using every possible subset of the p predictor variables. More specifically,

1. Construct *null model* M_0 with no predictors
2. For $i = 1, \ldots, p$, construct all $\binom{p}{i}$ possible models containing i predictors. Denote the best among these by M_i

   How to choose the best? Can use RSS or R^2 since models are the same size

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Given a total of p possible predictors, build every possible model using every possible subset of the p predictor variables. More specifically,

1. Construct *null model* M_0 with no predictors
2. For $i = 1, \ldots, p$, construct all $\binom{p}{i}$ possible models containing i predictors. Denote the best among these by M_i

   How to choose the best? Can use RSS or R^2 since models are the same size
3. Given the best of each size model $M_0, M_1, \ldots, M_p$, choose the best among these using C_p, AIC, BIC, CV, or Adjusted R^2

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Does this guarantee that we'll find the best model?

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Does this guarantee that we'll find the best model? Yes* – we're building every possible model!

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Does this guarantee that we'll find the best model? Yes* – we're building every possible model!

Disadvantage:

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Does this guarantee that we'll find the best model? Yes* – we're building every possible model!

Disadvantage: We're building every possible model!

$$1 + \sum_{i=1}^{p} \binom{p}{i} \text{ total models}$$

$$\implies \text{Very computationally expensive}$$

Best Subset Selection

Method 1 Best Subset Selection (Best SS):

Does this guarantee that we'll find the best model? Yes* – we're building every possible model!

Disadvantage: We're building every possible model!

$$1 + \sum_{i=1}^{p} \binom{p}{i} \text{ total models}$$

$$\Longrightarrow \text{Very computationally expensive}$$

*No - but this remains the popular narrative; Part V of this lecture will investigate why

Forward (Stepwise) Selection

Method 2 Forward (Stepwise) Selection (FSS):

1. Construct the *null model* M_0 with no predictors

Forward (Stepwise) Selection

Method 2 Forward (Stepwise) Selection (FSS):

1. Construct the *null model* M_0 with no predictors
2. Construct all possible models with only one predictor; keep the best and call this M_1. Suppose the best 1-variable model contains the predictor X_i

Forward (Stepwise) Selection

Method 2 Forward (Stepwise) Selection (FSS):

1. Construct the *null model* M_0 with no predictors
2. Construct all possible models with only one predictor; keep the best and call this M_1. Suppose the best 1-variable model contains the predictor X_i
3. Keeping X_i in the model, try adding every other predictor to the model to construct the best 2-variable model, M_2.

Forward (Stepwise) Selection

Method 2 Forward (Stepwise) Selection (FSS):

1. Construct the *null model* M_0 with no predictors
2. Construct all possible models with only one predictor; keep the best and call this M_1. Suppose the best 1-variable model contains the predictor X_i
3. Keeping X_i in the model, try adding every other predictor to the model to construct the best 2-variable model, M_2.
4. Continue this process to get the best of each size model $M_0, M_1, ..., M_p$. Choose the best among these using C_p, AIC, BIC, CV, or Adjusted R^2

Forward (Stepwise) Selection

Method 2 Forward (Stepwise) Selection (FSS):

Does this guarantee that we'll find the best model (or at least a very good one)?

Forward (Stepwise) Selection

Method 2 Forward (Stepwise) Selection (FSS):

Does this guarantee that we'll find the best model (or at least a very good one)? No! Suppose the best 1-variable model is

$$f(X) = \beta_0 + \beta_1 X_1$$

but the best 2-variable model is

$$f(X) = \beta_0 + \beta_1 X_2 + \beta_2 X_3$$

Advantage: *A lot* more computationally efficient and tends to work reasonably well in practice

Backward (Stepwise) Selection

Method 3 Backward (Stepwise) Selection (BSS):

1. Construct the *full model* M_p with all predictors

Backward (Stepwise) Selection

Method 3 Backward (Stepwise) Selection (BSS):

1. Construct the *full model* M_p with all predictors
2. Construct all possible models with $p - 1$ predictors by removing them one at a time. Keep the best and call this M_{p-1}.

Backward (Stepwise) Selection

Method 3 Backward (Stepwise) Selection (BSS):

1. Construct the *full model* M_p with all predictors
2. Construct all possible models with $p-1$ predictors by removing them one at a time. Keep the best and call this M_{p-1}.
3. Beginning with M_{p-1}, construct all possible models with $p-2$ predictors by removing them one at a time. Keep the best and call this M_{p-2}.

Backward (Stepwise) Selection

Method 3 Backward (Stepwise) Selection (BSS):

1. Construct the *full model* M_p with all predictors
2. Construct all possible models with $p-1$ predictors by removing them one at a time. Keep the best and call this M_{p-1}.
3. Beginning with M_{p-1}, construct all possible models with $p-2$ predictors by removing them one at a time. Keep the best and call this M_{p-2}.
4. Continue this process to get the best of each size model $M_0, M_1, \ldots, M_p$. Choose the best among these using C_p, AIC, BIC, CV, or Adjusted R^2

Backward (Stepwise) Selection

Method 3 Backward (Stepwise) Selection (BSS):

Does this guarantee that we'll find the best model (or at least a very good one)?

Backward (Stepwise) Selection

Method 3 Backward (Stepwise) Selection (BSS):

Does this guarantee that we'll find the best model (or at least a very good one)? No! Suppose the best 1-variable model is

$$f(X) = \beta_0 + \beta_1 X_1$$

but the best 2-variable model is

$$f(X) = \beta_0 + \beta_1 X_2 + \beta_2 X_3$$

Advantage: *A lot* more computationally efficient and tends to work reasonably well in practice

Model Selection Recap

Some Questions:

1. Would Best SS, FSS, and BSS produce the same models?

Model Selection Recap

Some Questions:

1. Would Best SS, FSS, and BSS produce the same models?

   No - not guaranteed to and usually don't

Model Selection Recap

Some Questions:

1. Would Best SS, FSS, and BSS produce the same models?

   No - not guaranteed to and usually don't

2. What about just FSS and BSS?

Model Selection Recap

Some Questions:

1. Would Best SS, FSS, and BSS produce the same models?

   No - not guaranteed to and usually don't

2. What about just FSS and BSS?

   No - see the previous example

Model Selection Recap

Some Questions:

1. Would Best SS, FSS, and BSS produce the same models?

   No - not guaranteed to and usually don't

2. What about just FSS and BSS?

   No - see the previous example

3. So which should we trust?

Model Selection Recap

Some Questions:

1. Would Best SS, FSS, and BSS produce the same models?

   No - not guaranteed to and usually don't

2. What about just FSS and BSS?

   No - see the previous example

3. So which should we trust?

   Can use the measures we defined previously (C_p, AIC, BIC, CV, Adjusted R^2) to compare results from different approaches

Part I Recap

- The measures we discussed – (C_p, AIC, BIC, Adjusted R^2) – provide a means for selecting models that trades off accuracy and model complexity
  - Cross-validation only accounts for accuracy, but still very useful as these other measures apply only to certain classes of models

- Best SS, FSS, and BSS provide a means for finding “candidate models" we can evaluate with the above measures
  - FSS and BSS don’t guarantee the best (or even a good) model, but are more computationally efficient.

    E.g. With 20 predictors ($p = 20$), Best SS requires building 1,048,576 models; FSS and BSS only 211

Part II: Shrinkage and Regularization

Part II Overview

- In Part I, we talked about linear model selection
  - Measures for evaluating model fit and complexity (C_p, AIC, BIC, Adj. R^2)
  - Methods for finding candidate models these measures can be tried on (Best Subset, Forward/Backward Selection)
- Most measures of model fit included an explicit *penalty term* to down-weight models with more variables (higher complexity / more flexibility)
- Here we'll again see this idea of penalty terms, but here we use them in fitting the models rather than to evaluate models already fit

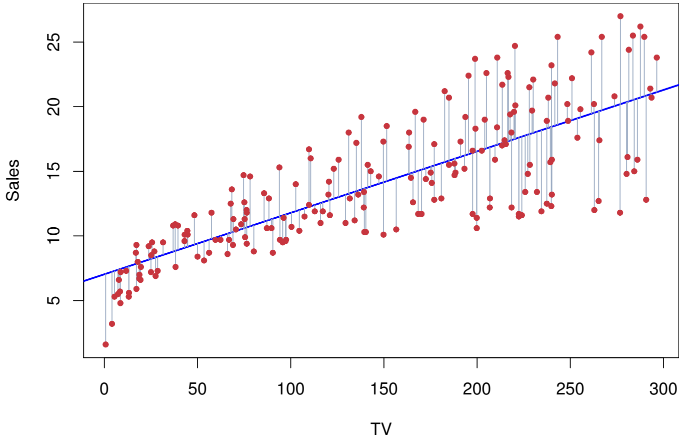
Fitting Linear Models

- Recall from Part I (and every other time you've ever seen linear models) that we begin by defining the *residual sum of squares*

$$RSS = \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 = \sum_{i=1}^{n} \big(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\big)^2$$

- We can think of the RSS as a *loss function* (squared-error loss)
- To fit the model, we choose the coefficient estimates $\hat{\beta}$ as the values of β that minimize this loss
- We call this approach ordinary least squares (OLS) and denote the estimates by $\hat{\beta}_{\text{OLS}} = \big(\hat{\beta}_{0,\text{OLS}}, \hat{\beta}_{1,\text{OLS}}, \ldots, \hat{\beta}_{p,\text{OLS}}\big)$

Ordinary Least Squares (OLS)



ISL Fig. 3.1: Recall this Figure from Chapter 3 depicting OLS Linear regression. We said that this is not the *only* way to choose the parameter estimates, but this method provides nice, useful properties.

Ridge Regression

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\sum_{i=1}^{n} \big(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\big)^2 \quad + \quad \lambda \sum_{i=1}^{p} \beta_i^2$$

Ridge Regression

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\underbrace{\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2}_{\text{RSS (Least Squares Term)}} \quad + \quad \underbrace{\lambda \sum_{i=1}^{p} \beta_i^2}_{\text{Penalty Term}}$$

Ridge Regression

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\underbrace{\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2}_{\text{RSS (Least Squares Term)}} \quad + \quad \underbrace{\lambda \sum_{i=1}^{p} \beta_i^2}_{\text{Penalty Term}}$$

- What's being penalized here?

Ridge Regression

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\underbrace{\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2}_{\text{RSS (Least Squares Term)}} \quad + \quad \underbrace{\lambda \sum_{i=1}^{p} \beta_i^2}_{\text{Penalty Term}}$$

- What's being penalized here? **The magnitude of the coefficients**

Ridge Regression

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\underbrace{\sum_{i=1}^{n}\left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2}_{\text{RSS (Least Squares Term)}} \quad + \quad \underbrace{\lambda \sum_{i=1}^{p} \beta_i^2}_{\text{Penalty Term}}$$

- What's being penalized here? **The magnitude of the coefficients**
- What effect would this have? How would the coefficient values chosen here compare to $\hat{\beta}_{\text{OLS}}$?

Ridge Regression

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\underbrace{\sum_{i=1}^{n}\left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2}_{\text{RSS (Least Squares Term)}} \quad + \quad \underbrace{\lambda \sum_{i=1}^{p} \beta_i^2}_{\text{Penalty Term}}$$

- What's being penalized here? **The magnitude of the coefficients**
- What effect would this have? How would the coefficient values chosen here compare to $\hat{\beta}_{\text{OLS}}$? **These coefficients would often be smaller. The penalty term has the effect of *shrinking* the magnitude of the coefficients**

Ridge Regression

- This idea is referred to broadly as *regularization*
  - In the specific context of linear regression, this typically means adding some kind of penalty to the least squares approach

- Adding this *specific* penalty – what we call an L_2 penalty – is referred to as *ridge regression* and we write the estimates as $\hat{\beta}_{\text{Ridge}}$
  - The “2" in L_2 refers to fact that we’re squaring the coefficient values in the penalty term

Ridge Regression

- Notice the λ (≥ 0) in front of the sum in the penalty term – this is a tuning parameter (analogous to the k in k-nearest neighbors) that controls how much weight we give the penalty:
  - When $\lambda = 0$, the penalty goes away so that $\hat{\beta}_{\text{Ridge}} = \hat{\beta}_{\text{OLS}}$
  - For large values of λ, the penalty term becomes larger $\Longrightarrow$ the coefficient estimates shrink by a large amount (i.e. $\hat{\beta}_{\text{Ridge}} \longrightarrow 0$)
- How can/should we choose the value of λ?

Ridge Regression

- Notice the λ (≥ 0) in front of the sum in the penalty term – this is a tuning parameter (analogous to the k in k-nearest neighbors) that controls how much weight we give the penalty:
  - When $\lambda = 0$, the penalty goes away so that $\hat{\beta}_{\text{Ridge}} = \hat{\beta}_{\text{OLS}}$
  - For large values of λ, the penalty term becomes larger $\implies$ the coefficient estimates shrink by a large amount (i.e. $\hat{\beta}_{\text{Ridge}} \longrightarrow 0$)
- How can/should we choose the value of λ? **The same way we talked about choosing any tuning parameter – Cross-Validation**

Ridge Regression

One important note:

- When we fit a model via OLS, if we scale-up one predictor variable (i.e. X_i becomes cX_i for some constant c), the corresponding coefficient estimate $\hat{\beta}_{c,\text{OLS}}$ simply gets weighted down by a factor of $1/c$
- However, with ridge regression, if we change the scale of one predictor X_i, the resulting change in $\hat{\beta}_{c,\text{Ridge}}$ will depend on c, λ, and the scaling of the other predictors
  - For this reason, we typically want to first standardize the values of the predictors before fitting: The i^{th} observation of the j^{th} predictor variable x_{ij} becomes

$$\tilde{x}_{ij} = \frac{x_{ij}}{\widehat{sd}(X_j)} = \frac{x_{ij}}{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}$$

Ridge Regression

Big, obvious, lingering question ...

Ridge Regression

Big, obvious, lingering question ... **Why do this? We spent more than a chapter praising the virtues of OLS!**

Ridge Regression

Big, obvious, lingering question ... **Why do this? We spent more than a chapter praising the virtues of OLS!**

- We said that fitting via OLS produced estimates with good properties, namely ...

Ridge Regression

Big, obvious, lingering question ... **Why do this? We spent more than a chapter praising the virtues of OLS!**

- We said that fitting via OLS produced estimates with good properties, namely ... **Unbiasedness: the OLS estimates won't systematically under/over-estimate the "true" value of the β's**
- So why might we want to give that up?

Ridge Regression

Big, obvious, lingering question ... **Why do this? We spent more than a chapter praising the virtues of OLS!**

- We said that fitting via OLS produced estimates with good properties, namely ... **Unbiasedness: the OLS estimates won't systematically under/over-estimate the "true" value of the β's**
- So why might we want to give that up? **Bias-variance tradeoff! Maybe we can give up a little bias for a big reduction in variance.**
  - Once again, tuning parameter controls this: when $\lambda = 0$, we have little (zero) bias, but possibly high variance
  - For large λ, we get a more rigid (less flexible) model; $\hat{\beta}_{\text{Ridge}}$ close to 0; $\implies$ higher bias, lower variance

Ridge Regression

- So when might we expect ridge regression (with $\lambda > 0$) to do well?

Ridge Regression

- So when might we expect ridge regression (with $\lambda > 0$) to do well? **When the OLS estimates have high variance!**

Ridge Regression

- So when might we expect ridge regression (with $\lambda > 0$) to do well? **When the OLS estimates have high variance!**
- What if we have more predictor variables than we do observations ($p > n$)?

Ridge Regression

- So when might we expect ridge regression (with $\lambda > 0$) to do well? **When the OLS estimates have high variance!**
- What if we have more predictor variables than we do observations ($p > n$)? **OLS solution is not well-defined (no unique solution)**

  $\Longrightarrow$ **Very high (infinite) variance**

  $\Longrightarrow$ **OLS doesn't work at all**

But ridge regression isn't the only option here ...

Lasso

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2 \quad + \quad \lambda \sum_{i=1}^{p} \left|\beta_i\right|$$

Lasso

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2 \quad + \quad \lambda \sum_{i=1}^{p} \left|\beta_i\right|$$

- This regularization method (L_1 penalty) is called the *Lasso* (**L**east **A**bsolute **S**hrinkage and **S**election **O**perator) – take that as a lesson in branding ☺

Lasso

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2 \quad + \quad \lambda \sum_{i=1}^{p} |\beta_i|$$

- This regularization method (L_1 penalty) is called the *Lasso* (**L**east **A**bsolute **S**hrinkage and **S**election **O**perator) – take that as a lesson in branding ☺
- No big deal, right? Just a slightly different penalty ...

Lasso

- Suppose instead we chose $\hat{\beta}$ to minimize

$$\sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})\right)^2 \quad + \quad \lambda \sum_{i=1}^{p} \left|\beta_i\right|$$

- This regularization method (L_1 penalty) is called the *Lasso* (**L**east **A**bsolute **S**hrinkage and **S**election **O**perator) – take that as a lesson in branding ☺

- No big deal, right? Just a slightly different penalty ...

WRONG!

Lasso

- Lasso tries to do the same thing as ridge regression – shrink the magnitude of the coefficients – but
  - Ridge regression does this in more of a uniform way – predictors are still left in the model
  - **Lasso does this by setting many of the coefficients equal to 0** (i.e. removing predictor variables from the model)

    $\Longrightarrow$ Lasso fits a *sparse* model – only involves a subset of the predictors

    $\Longrightarrow$ Shrinkage plus a sort of automated model selection
- How sparse? What controls this?

Lasso

- Lasso tries to do the same thing as ridge regression – shrink the magnitude of the coefficients – but
  - Ridge regression does this in more of a uniform way – predictors are still left in the model
  - **Lasso does this by setting many of the coefficients equal to 0** (i.e. removing predictor variables from the model)

    $\Longrightarrow$ Lasso fits a *sparse* model – only involves a subset of the predictors

    $\Longrightarrow$ Shrinkage plus a sort of automated model selection
- How sparse? What controls this? **The tuning parameter (can be selected via cross-validation). Larger $\lambda \Longrightarrow$ more shrinkage $\Longrightarrow$ sparser model (i.e. more coefficients set to 0; fewer predictors in the model)**

Lasso

- We discussed when we might expect to see ridge regression outperform OLS. We might expect Lasso to outperform OLS under the same kinds of circumstances.

  But ...

- When might we expect Lasso to outperform ridge regression?

Lasso

- We discussed when we might expect to see ridge regression outperform OLS. We might expect Lasso to outperform OLS under the same kinds of circumstances.

  But ...

- When might we expect Lasso to outperform ridge regression? **When there are a lot of predictors, *not* all of which are relevant (i.e. important for predicting the response)**

Lasso

Question: Why does Lasso do variable selection and ridge regression does not?

Lasso

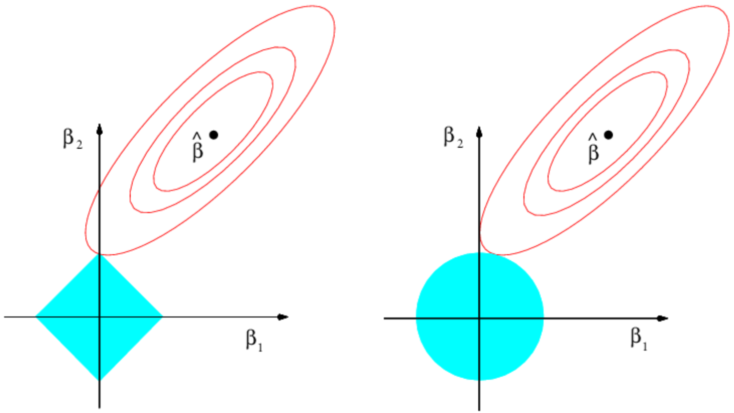
Question: Why does Lasso do variable selection and ridge regression does not?

- When we talked about fitting these models, we did so in the context of minimizing a loss function. However, we could equivalently write these as optimization (minimization problems):

$$\textbf{Lasso:} \quad \text{minimize } RSS \text{ subject to } \sum_{i=1}^{p} \left|\beta_i\right| \leq s$$

$$\textbf{Ridge Regression:} \quad \text{minimize } RSS \text{ subject to } \sum_{i=1}^{p} \beta_i^2 \leq s$$

The Lasso Picture



ISLR Fig. 6.7: The classic lasso picture.

Turquoise = regions of constraint.

Red Ellipses = regions of constant RSS (points on same ellipse have same RSS).

Part II Recap

- We mentioned in ISLR Chapter 3 when first discussing linear models that OLS was just one way we could think of choosing our parameter estimates $\hat{\beta}$
- Here we saw two new alternative ways, both of which involve minimizing the least squares loss function (RSS or squared error loss) but with an additional added penalty term tacked on
- Estimating a linear model in this fashion means our $\hat{\beta}$'s are no longer unbiased, but the hope is that what we add in bias we will more than make up for in a drop in variance

Part III: Dimension Reduction

Part III Overview

- In Part I, we talked about linear model selection – how do we generate a reasonable set of candidate models and how do we measure the differences between them
- In Part II, we talked about alternative approaches for fitting linear models (besides OLS) that involved shrinking the values of the coefficients
- The common theme was the notion that sometimes we might prefer models that are a bit smaller, simpler, and more interpretable
- In Part III, we'll talk about more direct approaches to this that involve using a different set of predictor variables

Dimension Reduction

Big Idea: Instead of building a linear model of the form

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

let's build a linear model of the form

$$Y = \theta_0 + \theta_1 Z_1 + \cdots + \theta_m Z_m + \epsilon$$

where the Z's are linear combinations of the original predictors:

$$Z_i = \phi_{1,i} X_1 + \phi_{2,i} X_2 + \cdots + \phi_{p,i} X_p$$

and generally we want m to be small.

Dimension Reduction

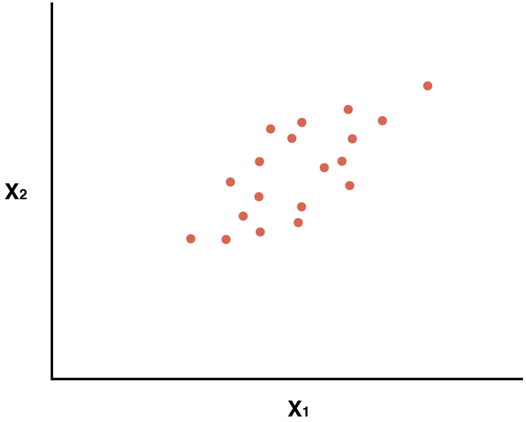
- Very importantly, note that **we're no longer using the original predictors directly**
- Instead, we're using a new, smaller set of *transformed* predictors
- When these new predictors are chosen well, this kind of model can do substantially better than OLS

  $\Longrightarrow$ Ideally, better fit with less predictors

Why would we expect something like this to work?

Dimension Reduction

Often times, different predictor variables contain similar information about the response:



Dimension Reduction

- If we know X_1, adding X_2 to the model isn't likely to help substantially (and vice versa)
- Instead, maybe we could find a single variable that summarizes the information in both X_1 and X_2 – these are the kinds of transformations we're looking for
- **Big Question:** How do we decide which transformations to use? We'll talk about two possible options:
  1. Principal Component Analysis (sometimes called Principal Component Regression)
  2. Partial Least Squares

Principal Component Analysis

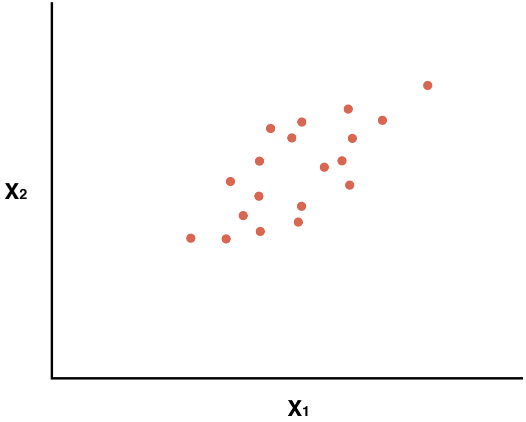
Option 1: Principal Component Analysis (PCA)

- Each transformation (i.e. each new predictor, Z_i) is called a *principal component*
- The first principal component is *the direction in which the data varies the most*
  - This means that if we were to project our data onto a line, we would want the line that would result in the largest variance in the projections

Let's look at an example where we only have two variables, X_1 and X_2

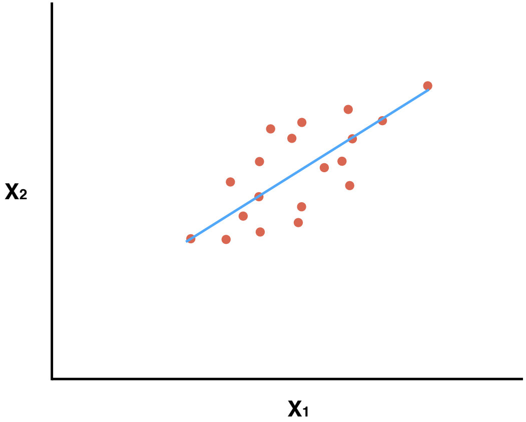
Principal Component Analysis

Here's the same data as before:



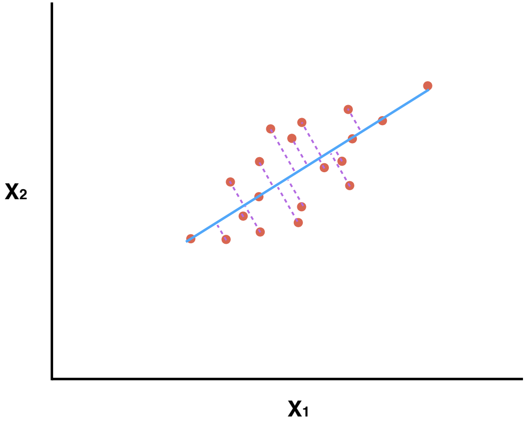
Principal Component Analysis

This is the “best" line to project the data onto:



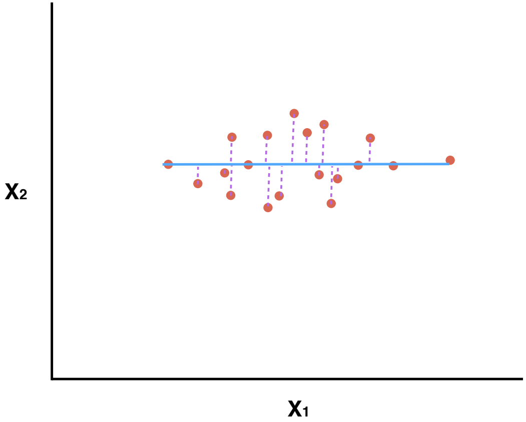
Principal Component Analysis

Projecting the data ...



Principal Component Analysis

… and shifting to be on a horizontal axis:



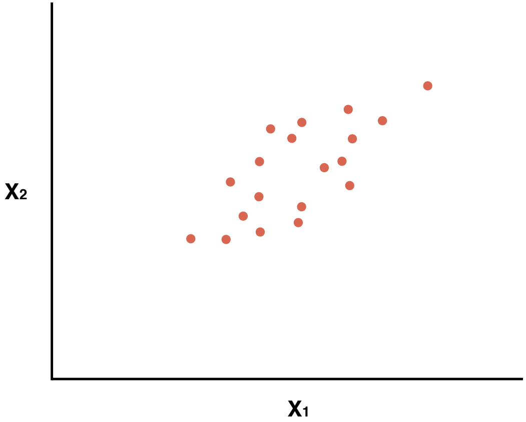
Principal Component Analysis

- Why is this the best projection? It's the one that results in the maximum variance in the data after projected onto it
- This is a bit difficult to visualize:
  - After we project the data, think of shifting it horizontally – the resulting picture is no different than if you took any typical kind of sample from a population – we're looking at spread

Let's look at another example where use a slightly different projection:

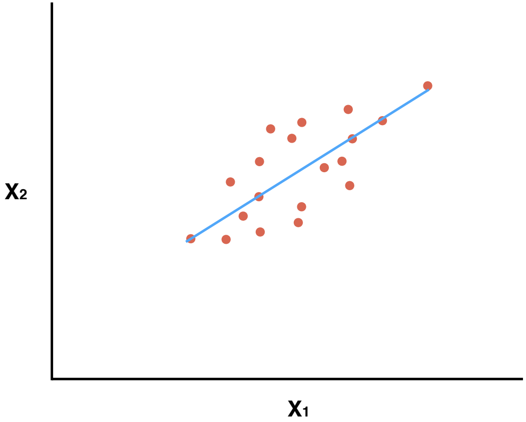
Principal Component Analysis

Again, here's the same original data:



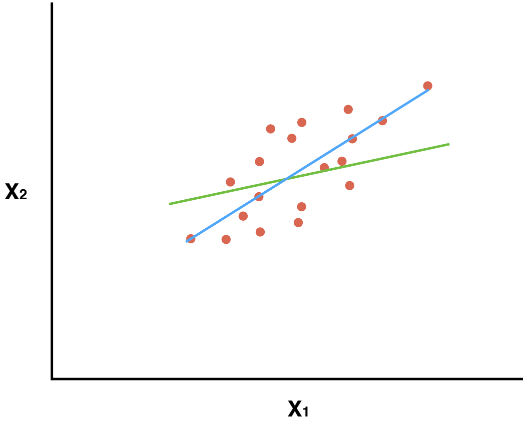
Principal Component Analysis

And the “best" line to project the data onto from before:



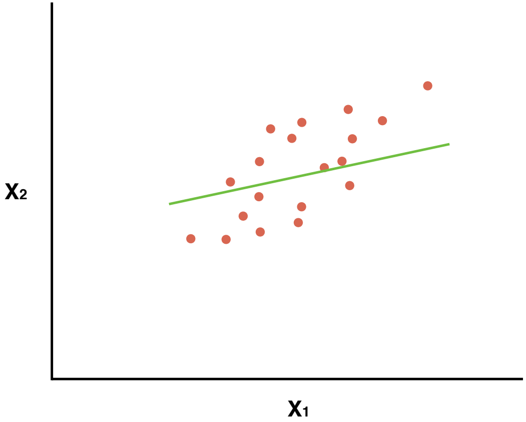
Principal Component Analysis

And a new projection we could consider:



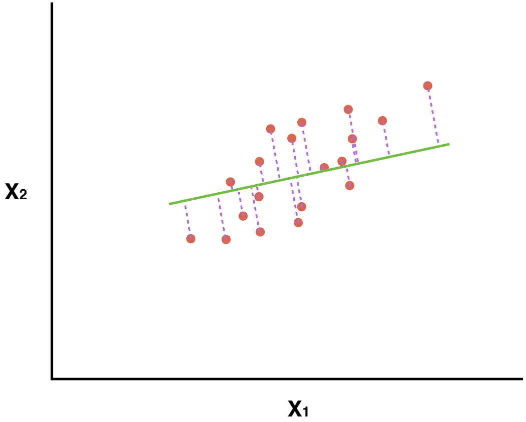
Principal Component Analysis

And a new projection we could consider:



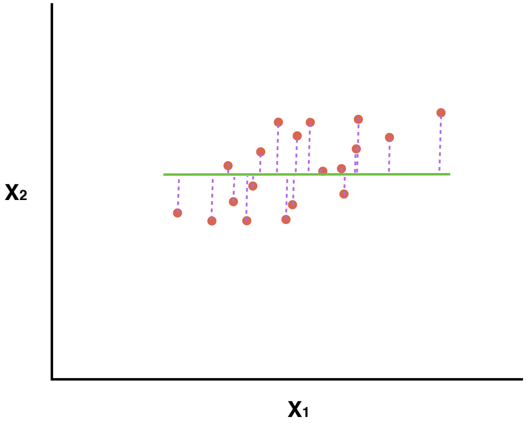
Principal Component Analysis

I do the projection ...



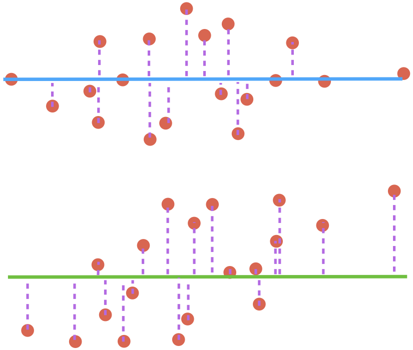
Principal Component Analysis

... and shift



Principal Component Analysis

Now let's compare: it's a little hard to see, but the variance of the data on the blue line (top) is greater (it helps to know the blue and green lines are the same length; notice that the data extend to the end of the blue line)



Principal Component Analysis

- This first principal component can be written as a linear combination of X_1 and X_2 of the form

$$Z_1 = c_1(X_1 - \bar{X}_1) + c_2(X_2 - \bar{X}_2)$$

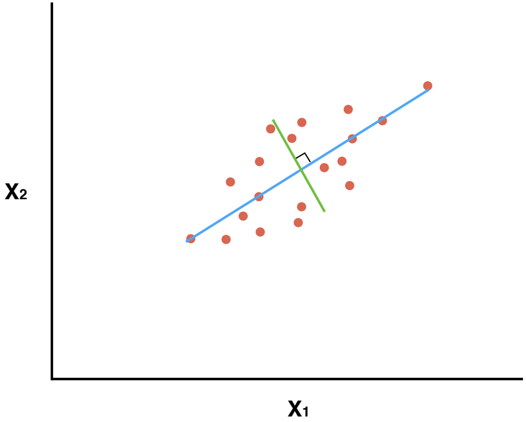
$\Longrightarrow$ See book for details

- The next principal component Z_2 is the next direction in which the data varies the most, **but** we insist that Z_2 not be correlated with Z_1

$\Longrightarrow$ Means that Z_2 must be orthogonal to Z_1

Principal Component Analysis

Second principal component shown in green:



Principal Component Analysis

- We can continue in this fashion to construct up to p principal components $Z_1, ..., Z_p$
  - ▶ Idea is that these should be ordered in terms of information: Z_1 explains the most about Y, Z_2 explains the next most about Y given the information already contained in Z_1 etc.
  - ▶ Keep in mind that while we *can* create p principal components, we generally only want to use the first few (in practice, think 20-30 predictor variables down to 2 or 3 principal components)
  - ▶ Number of principal components we use in the model can be decided via cross-validation

Principal Component Analysis

- **Downside to PCA:** Regression is on Z_i, not the original X_i, so depending on the model you fit, you can lose some interpretability
- Because the Z_i still depend on the original predictors, we consider this kind of process *dimension reduction* instead of *variable selection*
- There's one aspect of PCA that is a bit strange that we still haven't addressed:

Principal Component Analysis

- **Downside to PCA:** Regression is on Z_i, not the original X_i, so depending on the model you fit, you can lose some interpretability
- Because the Z_i still depend on the original predictors, we consider this kind of process *dimension reduction* instead of *variable selection*

- There's one aspect of PCA that is a bit strange that we still haven't addressed: **We're inherently assuming that the direction of max variation in the data is also the most informative about Y**

$\Longrightarrow$ This leads us to Partial Least Squares

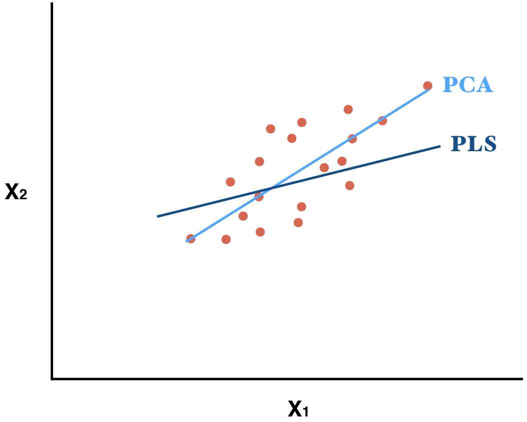
Partial Least Squares

Option 2: Partial Least Squares (PLS)

- Very similar idea to PCA, but instead of the direction (projection) based *only* on what maximizes the variance, we choose the new predictors Z_i based on a projection that both summarizes the original predictors **and** relates to the response
- In the two-predictor case, the new predictors take a similar form
$$Z_1 = c_1(X_1 - \bar{X}_1) + c_2(X_2 - \bar{X}_2)$$
but PLS is going to give more weight to the variable that marginally (i.e. by itself) is more strongly correlated with Y (e.g. if X_1 more correlated with Y, magnitude of c_1 will be larger)

Partial Least Squares

For example, we may see something like this ...



Partial Least Squares

- Suppose we did see something like this. If we think of the two projections as just lines, what's the difference between the PCA line and the PLS line?

Partial Least Squares

- Suppose we did see something like this. If we think of the two projections as just lines, what's the difference between the PCA line and the PLS line? **The PLS line has a smaller slope (with X_1 on the x-axis and X_2 on the y-axis)**

Partial Least Squares

- Suppose we did see something like this. If we think of the two projections as just lines, what's the difference between the PCA line and the PLS line? **The PLS line has a smaller slope (with X_1 on the x-axis and X_2 on the y-axis)**

$\implies$ Less change in X_2 per unit change in X_1

Partial Least Squares

- Suppose we did see something like this. If we think of the two projections as just lines, what's the difference between the PCA line and the PLS line? **The PLS line has a smaller slope (with X_1 on the x-axis and X_2 on the y-axis)**

$\implies$ Less change in X_2 per unit change in X_1

$\implies$ We're capturing "more" of X_1

Partial Least Squares

- Suppose we did see something like this. If we think of the two projections as just lines, what's the difference between the PCA line and the PLS line? **The PLS line has a smaller slope (with X_1 on the x-axis and X_2 on the y-axis)**

$\implies$ Less change in X_2 per unit change in X_1

$\implies$ We're capturing "more" of X_1

$\implies$ X_1 must have been more correlated with the response

Part III Recap

- PCA and PLS have the same ultimate goal: Instead of fitting a bigger linear model on a lot of predictors, transform the predictors and fit a small model on the transformations
- In both cases, cross-validation can be used to determine the number of components to include in the new models
- Potentially lose some interpretability (though with small PCA and PLS models this is sometimes manageable)
- Can think of PLS has a “supervised alternative" to PCA (PLS uses information about Y to make the transformations)
    - PLS is not always better though – it can often reduce bias (more flexible), but comes with added variance

Part IV: High Dimensional Issues

Part IV: Overview

- “Big data" has become a ubiquitous buzz word – what does it really mean?
- Could means different things but usually,
  1. Large n (many observations), or
  2. Large p (many predictors), or
  3. Both large n and large p, or
  4. Large p relative to n
- “High-dimensional data" usually means something more specific (at least in statistics): p large relative to n

$$p \approx n, \quad p > n, \text{ or } \quad p \gg n$$

Big Data: $n > p$

- The situation we're familiar with is $n > p$ (and often $n \gg p$)
- For a fixed number of predictors p, is having a larger sample size n always better?

Big Data: $n > p$

- The situation we're familiar with is $n > p$ (and often $n \gg p$)
- For a fixed number of predictors p, is having a larger sample size n always better?
  - Yes, of course, but **assuming** the quality of the data is the same
  - Smaller sample sizes may actually be better if the data *quality* is better than that of larger datasets that may be available
  - See Xiao-Li Meng's "Statistical Paradises and Paradoxes in Big Data" video

High Dimensional Data: $p > n$

- With modern data, it's quite common that $p > n$

  That's good because more data is always better, right?

High Dimensional Data: $p > n$

- With modern data, it's quite common that $p > n$

  That's good because more data is always better, right?

WRONG!

High Dimensional Data: $p > n$

- With modern data, it's quite common that $p > n$

  That's good because more data is always better, right?

WRONG!

- We used to have a formal system in place which prevented this kind of situation. What was it called?

High Dimensional Data: $p > n$

- With modern data, it's quite common that $p > n$

  That's good because more data is always better, right?

WRONG!

- We used to have a formal system in place which prevented this kind of situation. What was it called?

SCIENCE
(i.e. variable selection by humans)

$\implies$ Get's back Lecture 1 – shift not just in the amount of data available, but in the fundamental ways we think about learning information about the world

High Dimensional Data: $p > n$

- It used to be generally understood that in order to understand the effects of each predictor, you need sufficient data and replicates at each level (i.e. $n > p$), so why is so much data now $p > n$?
- Well, one could draw an analogy with missing data. That too used to be considered a bad situation, yet it's now extremely common, especially in clinical trial and biomedical data. Why?

High Dimensional Data: $p > n$

- It used to be generally understood that in order to understand the effects of each predictor, you need sufficient data and replicates at each level (i.e. $n > p$), so why is so much data now $p > n$?
- Well, one could draw an analogy with missing data. That too used to be considered a bad situation, yet it's now extremely common, especially in clinical trial and biomedical data. Why?

In some cases, it really is unavoidable. However, the danger is that it's become all too commonplace because we have tools (books on missing data imputation) to "fix" the issue.

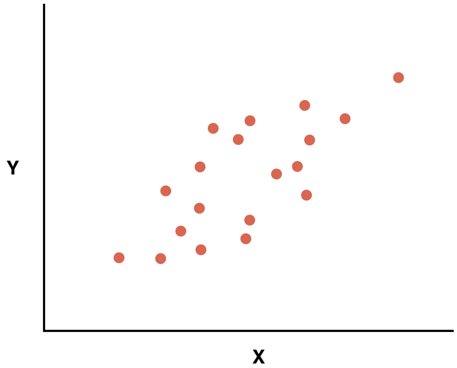
High Dimensional Data: $p > n$

- But therein lies the danger: having missing data is never good, but because there are “solutions" (variety of imputation methods under various assumptions), it’s an issue that gets overlooked far too often

- Much the same claim could be made about high dimensional data issues: having $p > n$ is not a good thing, but because we have regularization tools like the lasso, it’s not an issue that gets properly considered in much of scientific practice (i.e. because there are now tools to “solve" these kinds of problems, it’s often no longer even considered a negative situation)

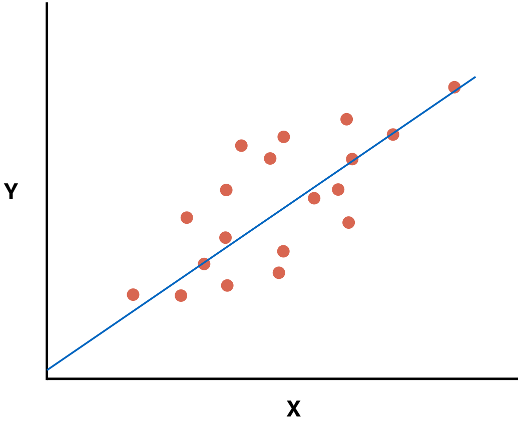
High Dimensional Data: $p > n$

So what's the big deal? Well let's think about what happens even when $p \approx n$ but with p, n both small so we can visualize it. Let's start with $n > p$, same data as before:



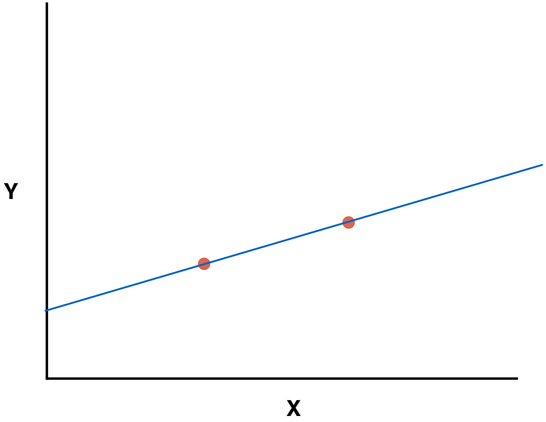
High Dimensional Data: $p > n$

We can fit a linear model as usual:



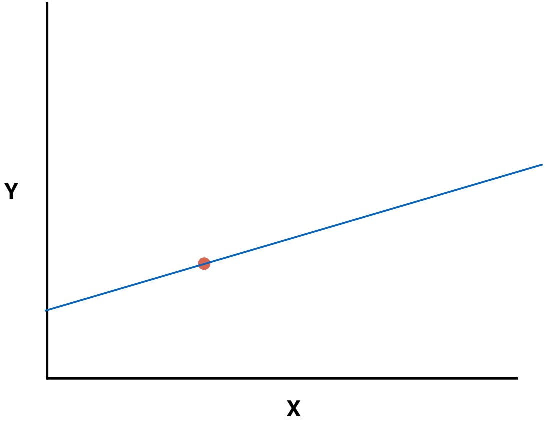
High Dimensional Data: $p > n$

Now still have $n > p$ but, unique perfect fit.



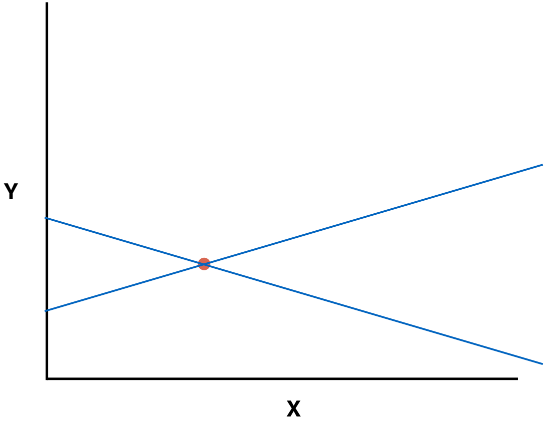
High Dimensional Data: $p > n$

Now $n = p$, model still fits perfectly but ...



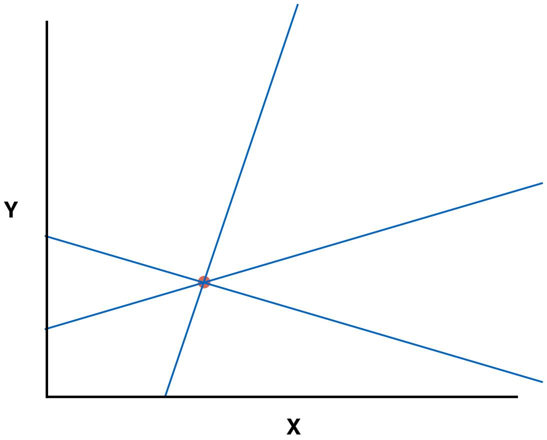
High Dimensional Data: $p > n$

So does this one



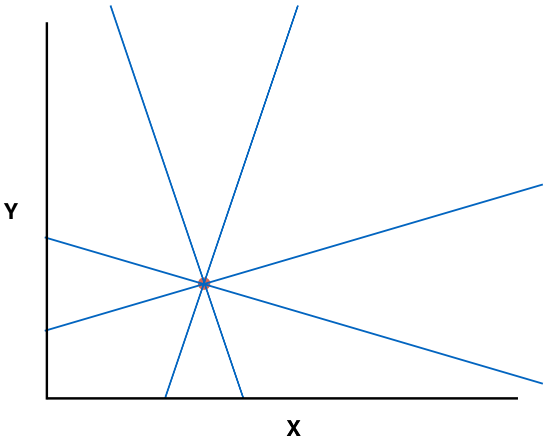
High Dimensional Data: $p > n$

And this one



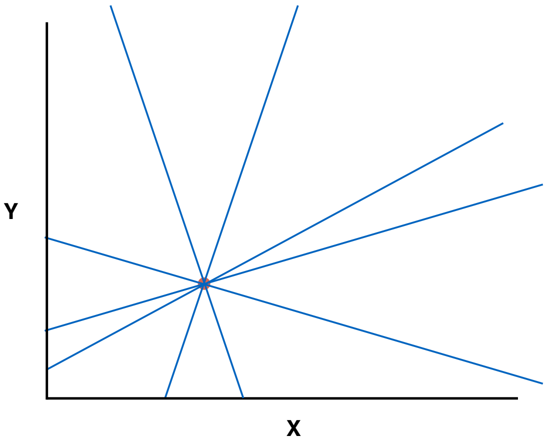
High Dimensional Data: $p > n$

And this one



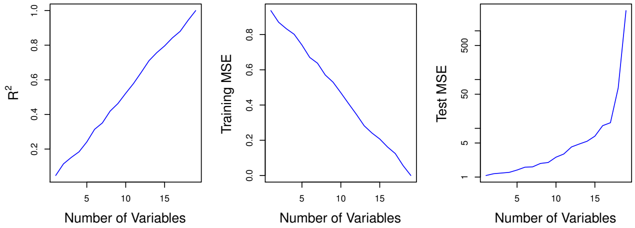
High Dimensional Data: $p > n$

And this one



High Dimensional Data: $p > n$

- For $p \geq n$, we can think of this as no bias (perfect fit) but infinite variance – same holds for larger values of n and p
- As p increases and becomes larger than n, $RSS \longrightarrow 0$ and $R^2 \longrightarrow 1$



ISLR Fig. 6.23: Measures of fit for simulation with $n = 20$ and increasing p

High Dimensional Data: $p > n$

- And oh yeah, in the previous figure,

High Dimensional Data: $p > n$

- And oh yeah, in the previous figure,

  NONE OF THE PREDICTORS ARE IN ANY WAY RELATED TO THE RESPONSE

High Dimensional Data: $p > n$

- And oh yeah, in the previous figure,

  NONE OF THE PREDICTORS ARE IN ANY WAY RELATED TO THE RESPONSE

- But ok, that's the training error. Why don't we use something like C_p, AIC, BIC, or Adjusted R^2 like we talked about?

High Dimensional Data: $p > n$

- And oh yeah, in the previous figure,

  NONE OF THE PREDICTORS ARE IN ANY WAY RELATED TO THE RESPONSE

- But ok, that's the training error. Why don't we use something like C_p, AIC, BIC, or Adjusted R^2 like we talked about? **Because these are perfect fits – we'd estimate $\hat{\sigma} = 0$. Similar issues with adjusted R^2.**

High Dimensional Data: $p > n$

- In general, as the dimension (p) grows, so does the test error
  - This is the idea of the **curse of dimensionality**
- But what if those features we're adding in really are relevant?

High Dimensional Data: $p > n$

- In general, as the dimension (p) grows, so does the test error
  - This is the idea of the **curse of dimensionality**
- But what if those features we're adding in really are relevant?

  Great! But the reduction in bias (improved fit) you may see will have to outweigh the additional variance you're adding in by including more terms.

High Dimensional Data: $p > n$

- How about multicollinearity? Remember this from Chapter 3 ...
  - The idea that some combination of variables (almost entirely) explained some other variable
- In high dimensional settings, this **MUST BE** (i.e. is always) the case
  - In fact, we must have “perfect" multicollinearity: any/every predictor can be written as a linear combination of the others
  - In linear algebra terms, this means we *can't* have linearly independent columns

High Dimensional Data: $p > n$

What are the practical implications of this?

- Even if you can find a good “small" set of predictors, there are almost certainly many such good, small sets
- Thus, while finding such a set is useful for predicting, you can’t really ever claim to have found the “best" set
- In practice, the best option is often to find a good set that are easy to measure (in the real world) and/or that make the most sense in a particular context

Part IV Recap

- For a fixed p, having a larger n is always preferable as long as the bigger dataset is of the same quality
- For a fixed n, having a larger p is problematic
- Regularization methods provide a solution when $p > n$ where OLS does not, but those solutions will **never** be as good as if we had started with the “right" (smaller) set of predictors to begin with
  - Unfortunately, because “solutions" like the Lasso exist, don’t look for this to change anytime soon
  - Brings up an interesting ethical dilemma regarding whether it’s really a good thing to try and “solve" problems like this in, for example, high dimensional and missing data settings

Part V: Degrees of Freedom and Signal-to-Noise Ratios

DoF and SNR

- The typical understanding of best subset selection is something like the following:

  Best subset selection is obviously the best thing to do, but for even reasonably small datasets, there's just no way we can do that. Thus, FSS, BSS, lasso are computationally efficient alternatives that tend to work well in many practical settings.

- This belief is near universal outside of statistics and even held by a surprising amount of statisticians

- In Part I of this lecture, we said this wasn't true – now in Part V we'll take an in-depth look at why

DoF and SNR

Let's start by looking at a really interesting and important paper in *Annals of Statistics* by Bertsimas et al. (2016):

- Everyone obviously wants to always do best subset selection, but can't for computational reasons, but ...
- Best subset selection can be reformulated as a mixed integer optimization (MIO) problem
- MIO problems are fast (200,000,000,000 x faster than in 1990)
- Even with n, p in the 1000s, can find near optimal solutions in minutes

DoF and SNR

Let's start by looking at a really interesting and important paper in *Annals of Statistics* by Bertsimas et al. (2016):

- Everyone obviously wants to always do best subset selection, but can't for computational reasons, but ...
- Best subset selection can be reformulated as a mixed integer optimization (MIO) problem
- MIO problems are fast (200,000,000,000 x faster than in 1990)
- Even with n, p in the 1000s, can find near optimal solutions in minutes

$\implies$ Thanks for playing, lasso. Enjoy your retirement

DoF and SNR

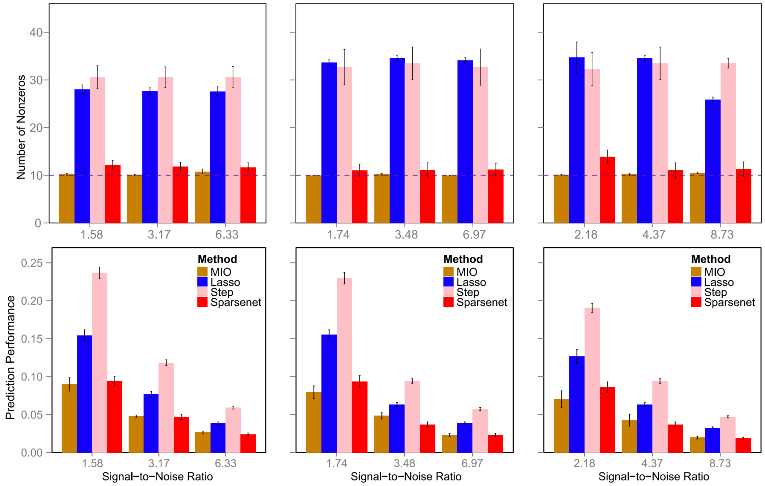


Figure 4 from Bertsimas et al. (2016).

DoF and SNR

Before going further, we need to pause and discuss what we mean by a “signal-to-noise" ratio (SNR):

For a particular pair (x, y) with $y = f(x) + \epsilon$, SNR is defined as:

$$SNR = \frac{\text{var}(f(x))}{\text{var}(\epsilon)}$$

We can also define the percent of variance explained (PVE) for a particular estimate $\hat{f}$:

$$PVE = 1 - \frac{\mathbb{E}(y - \hat{f}(x))^2}{\text{var}(y)}$$

When $\hat{f} = f$, we have that

$$PVE = \frac{SNR}{1 + SNR}$$

DoF and SNR

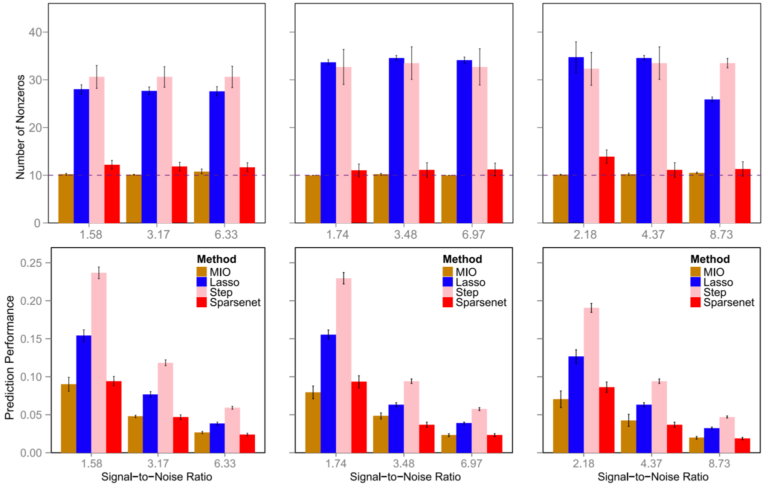


Figure 4 from Bertsimas et al. (2016).

DoF and SNR

Take a close look at the bottom row of plots in that figure. What do you notice?

DoF and SNR

Take a close look at the bottom row of plots in that figure. What do you notice?

For larger signal to noise ratios (SNRs), each method seems to be doing reasonably well in terms of prediction error. Should make some sense:

$$\begin{aligned} \text{High SNR} &\implies \text{best model(s) more obvious} \\ &\implies \text{Even "bad" methods can still do well.} \end{aligned}$$

DoF and SNR

Take a close look at the bottom row of plots in that figure. What do you notice?

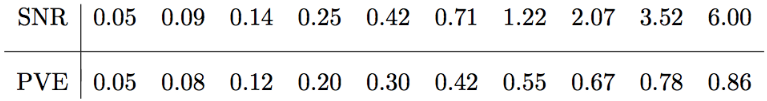
For larger signal to noise ratios (SNRs), each method seems to be doing reasonably well in terms of prediction error. Should make some sense:

$$\begin{aligned} \text{High SNR} &\implies \text{best model(s) more obvious} \\ &\implies \text{Even "bad" methods can still do well.} \end{aligned}$$

But does the opposite hold (as implied)? Looks like MIO is doing much better than lasso/stepwise selection at low SNRs ...

DoF and SNR

Hastie, Tibshirani, and Tibshirani (2017) wrote a response to this paper and provided an extended set of comparisons. Bertsimas et al. consider SNR values from about 1.5 to 8. Is that reasonable? HTT don't think so ...



SNR	0.05	0.09	0.14	0.25	0.42	0.71	1.22	2.07	3.52	6.00
PVE	0.05	0.08	0.12	0.20	0.30	0.42	0.55	0.67	0.78	0.86

[Table from HTT (2017) page 10.]

HTT argue that in realistic, real-data settings, it's not be reasonable to think that the data explain upwards of 60% of the total variance. So how does MIO/BSS do at low SNRs?

DoF and SNR

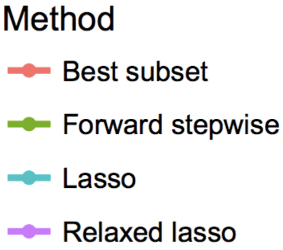
In their response, HTT provide a large set of comparisons between (1) best subset selection (MIO), (2) forward stepwise selection, (3) the lasso, and (4) the relaxed lasso. Relaxed Lasso is defined as

$$\hat{\beta}_{RL}(\lambda, \alpha) = \alpha \cdot \hat{\beta}_{lasso}(\lambda) + (1 - \alpha) \cdot \hat{\beta}_{OLS\,|\,Lasso}(\lambda)$$

- $\hat{\beta}_{OLS\,|\,Lasso}$ is “least-squares-after-lasso" – OLS on the coefficients selected by lasso
- Relaxed lasso is just a weighted average of lasso and OLS-after-lasso coefficients. Weighting can be chosen by cross or external validation.

DoF and SNR

In each of the following plots:



DoF and SNR

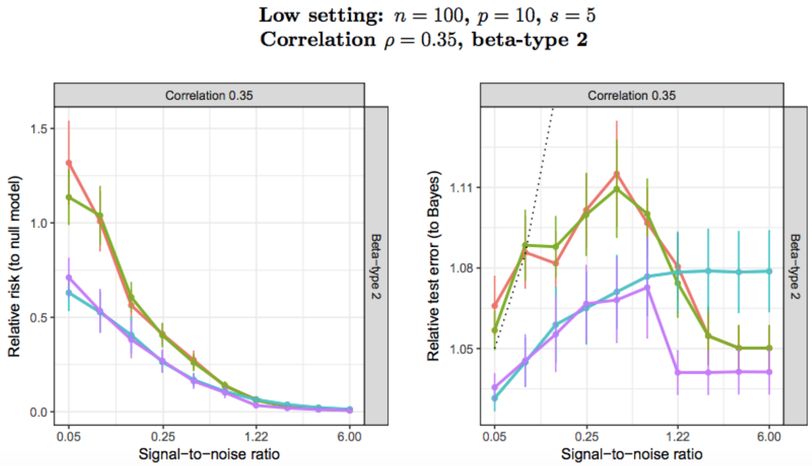


Figure 5 from HTT (2017).

DoF and SNR

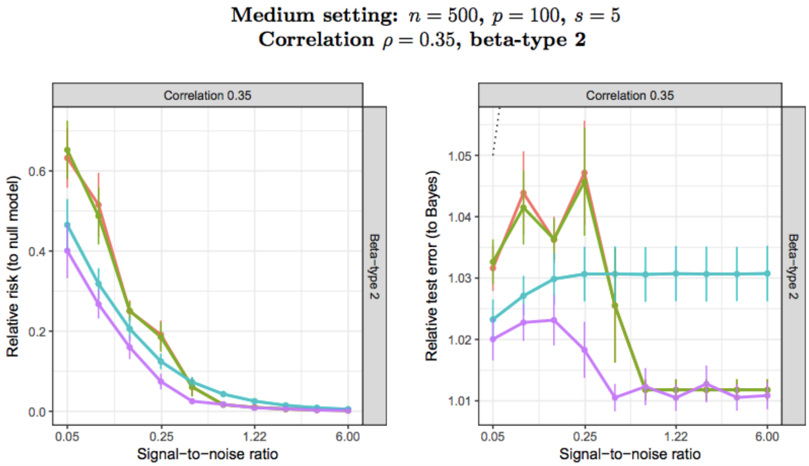


Figure 6 from HTT (2017).

DoF and SNR

High-5 setting: $n = 50$, $p = 1000$, $s = 5$
Correlation $\rho = 0.35$, **beta-type 2**

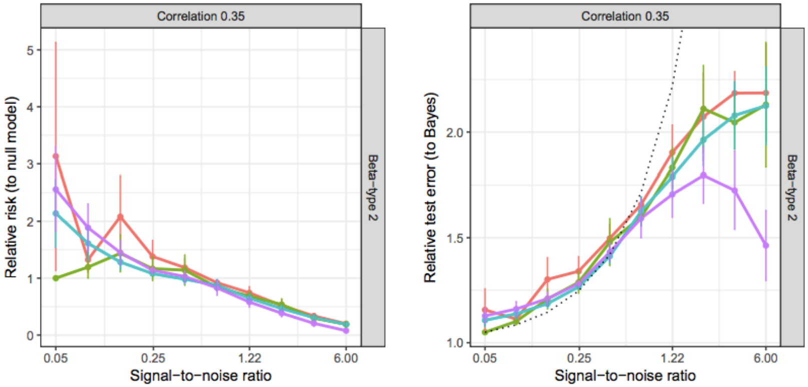


Figure 7 from HTT (2017).

DoF and SNR

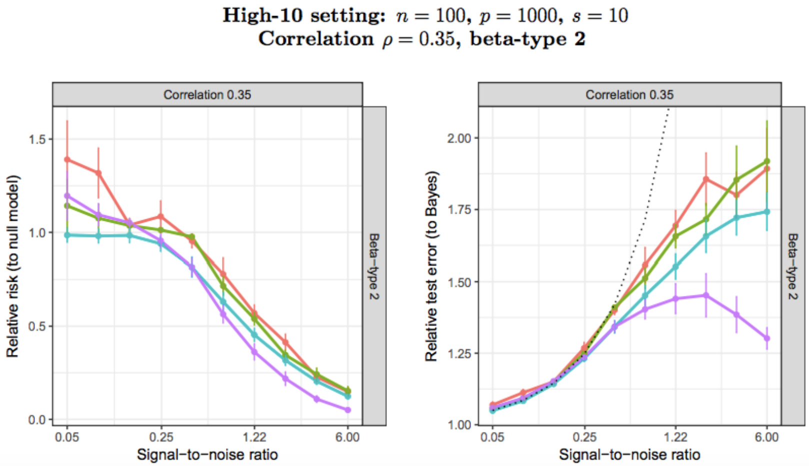


Figure 8 from HTT (2017).

DoF and SNR

So what’s going on here? HTT explain this behavior in terms of the “degrees of freedom" of the different procedures.

DoF and SNR

So what's going on here? HTT explain this behavior in terms of the "degrees of freedom" of the different procedures.

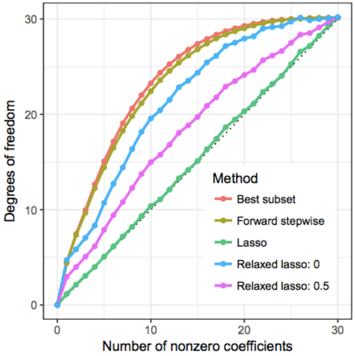


Figure 4: *Degrees of freedom for the lasso, forward stepwise, best subset selection, and the relaxed lasso with $\gamma = 0.5$ and $\gamma = 0$. The problem setup is the same as that in the left panel of Figure 1. Note that the relaxed lasso has an inflated degrees of freedom compared to the lasso and generally has a larger degrees of freedom than the expected number of nonzero coefficients. But, even when $\gamma = 0$, its degrees of freedom is smaller than that of forward stepwise and best subset selection throughout their model paths.*

Figure 4 from HTT (2017).

DoF and SNR

What are degrees of freedom (dof)?

DoF and SNR

What are degrees of freedom (dof)?

You may have heard this in a linear modeling course as the number of terms in the model ... outside of linear models it's often thought of as the "effective" number of parameters.

More generally, we can define the degrees of freedom of a particular model $\hat{f}$ as

$$df(\hat{f}) = \frac{1}{\sigma^2} \sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i)$$

where σ^2 is the variance of the error term ϵ. Note that the sum is over the entire training set.

DoF and SNR

Let's dig into the definition a little more ...

- Degrees of freedom is a measure of how much each prediction $\hat{y}_i$ depends on it's true value y_i
  - More dependence = more dof
- In other words, how much would my prediction change if I instead got a different value of y_i
- i.e. This is really a measure of how our model is behaving in terms of bias and variance ... so this is a measure of model *flexibility/complexity*
  - More dependence = more variance / less bias = more flexible/complex = higher dof

DoF and SNR

Let's have another look at the plots provided by HTT:

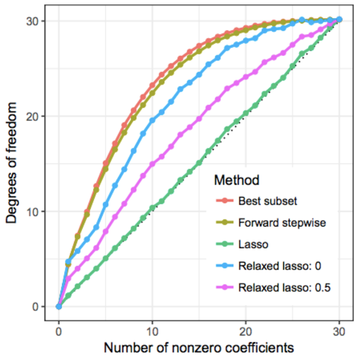


Figure 4: *Degrees of freedom for the lasso, forward stepwise, best subset selection, and the relaxed lasso with* $\gamma = 0.5$ *and* $\gamma = 0$. *The problem setup is the same as that in the left panel of Figure 1. Note that the relaxed lasso has an inflated degrees of freedom compared to the lasso and generally has a larger degrees of freedom than the expected number of nonzero coefficients. But, even when* $\gamma = 0$, *its degrees of freedom is smaller than that of forward stepwise and best subset selection throughout their model paths.*

Figure 4 from HTT (2017).

DoF and SNR

So in terms of how this relates to the earlier plots of performance vs SNR ...

- Best subset / FSS have high dof – lasso has few dof (same as number of terms in the model) – relaxed lasso is in the middle
- At low SNRs, procedures with high dof tend to do poorly. At high SNRs, all methods begin to do well with higher dof procedures doing best.
- Why? At low SNRs, it's very difficult to see the "real" signal. Thus, high dof procedures are almost guaranteed to overfit, regardless of how you try to validate. Regularized procedures like the lasso are designed to increase bias / stability, so they excel in those settings.

DoF and SNR Takeaways

- Best subset is not the “gold standard" that nearly everyone believes
- Relative to lasso, best subset and FSS are low bias but high variance (spending many more dof to fit "same size" models)
- Thus, when SNR is low, most of what you "see" in the data is noise so these methods overfit to that noise
- When SNR is high, the right model is easy to distinguish from others so this "over-eagerness" to fit actually serves it well compared with lasso where the shrinkage adds too much bias
- **Something to consider:** Why do we often see a diverging performance between lasso and relaxed lasso as the SNR gets larger?