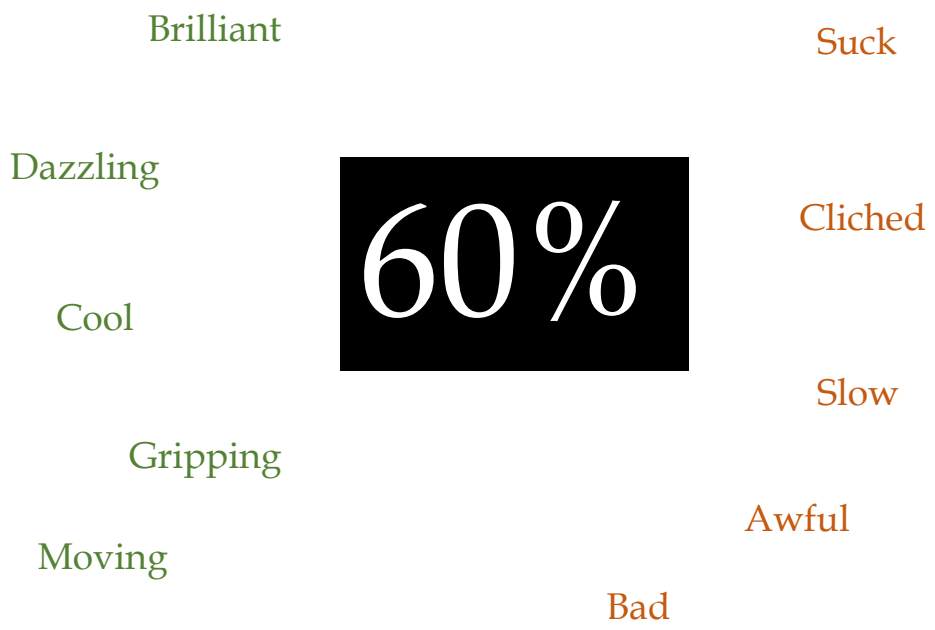
Supervised learning: how does it work?

- Movie reviews (Pan, Lee, Vaithyanathan)
  - Intelligent methods, trying to copy humans, perform poorly.

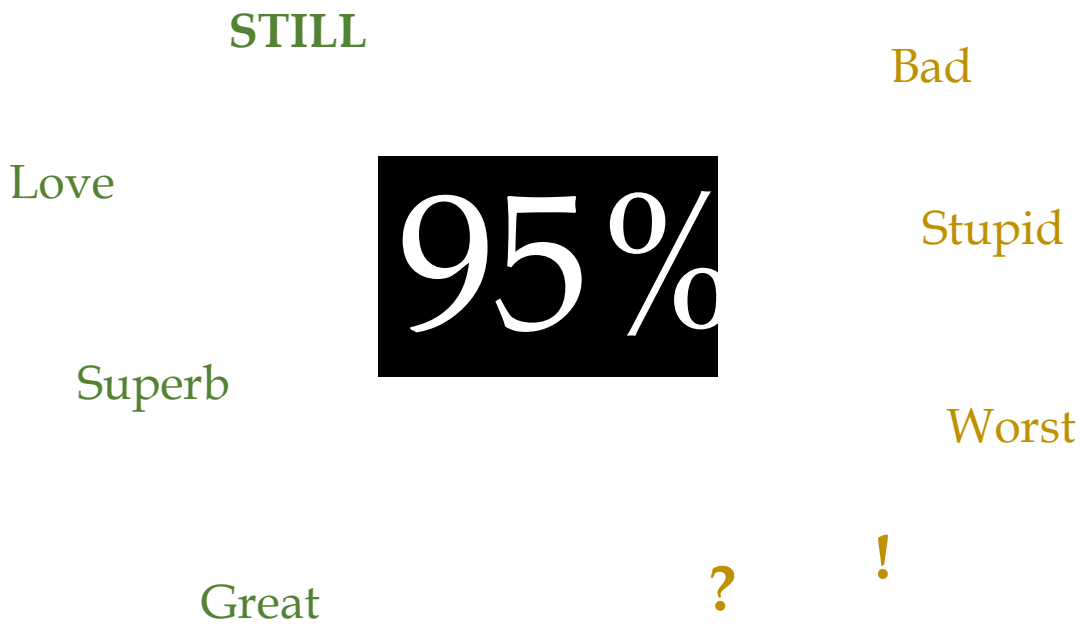


What is so hard?

- Decide what words makes for a positive review
  - What combination of words do you look for?
    - “Some people say this talk was great”
- This problem was endemic to every problem
  - Driving a car: What is a tree?
  - Language: Which noun does this pronoun modify?
- This approach stalled
  - “Trivial” problems proved impossible
  - Forget about the more complicated problems like language

ML Approach

- Collect example dataset:
  - 2000 movie reviews
  - 1000 good and 1000 bad reviews
- Now just ask what combination of words predicts being a good review.



What is the magic trick?

- Stop trying to “figure it out”
- Stop looking for an insight
- Just look at the data
- Treat the known like we treat the unknown

Widespread use

- Post office uses machines to read addresses
- Voice recognition (Siri)
- Spam filters
- Movie and other recommendations

- String together many smaller tasks
  - Driverless cars

- *Empirical* intelligence

The problem

- Find function f(x)= $\hat{y}$ to minimize $L(y, \hat{y})$ in new data.
  - Loss function: $L(y, \hat{y})$, typically MSE of prediction error.
- Two components
  - Choose function f class: regression tree, linear, etc.
  - Select regularizer: bias-variance trade-off.

Examples

Table 2

Some Machine Learning Algorithms

Function class $\mathcal{F}$ (and its parametrization)	*Regularizer $R(f)$*
Global/parametric predictors	
Linear $\beta' x$ (and generalizations)	Subset selection $\|\|\beta\|\|_0 = \sum_{j=1}^{k} 1_{\beta_j \neq 0}$
	LASSO $\|\|\beta\|\|_1 = \sum_{j=1}^{k} \|\beta_j\|$
	Ridge $\|\|\beta\|\|_2^2 = \sum_{j=1}^{k} \beta_j^2$
	Elastic net $\alpha\|\|\beta\|\|_1 + (1-\alpha)\|\|\beta\|\|_2^2$
Local/nonparametric predictors	
Decision/regression trees	Depth, number of nodes/leaves, minimal leaf size, information gain at splits
Random forest (linear combination of trees)	Number of trees, number of variables used in each tree, size of bootstrap sample, complexity of trees (see above)
Nearest neighbors	Number of neighbors
Kernel regression	Kernel bandwidth
Mixed predictors	
Deep learning, neural nets, convolutional neural networks	Number of levels, number of neurons per level, connectivity between neurons
Splines	Number of knots, order
Combined predictors	
Bagging: unweighted average of predictors from bootstrap draws	Number of draws, size of bootstrap samples (and individual regularization parameters)
Boosting: linear combination of predictions of residual	Learning rate, number of iterations (and individual regularization parameters)
Ensemble: weighted combination of different predictors	Ensemble weights (and individual regularization parameters)

Linear Machine Learning Models

Readings:
An Introduction to Statistical Learning with Applications in R,

- Chapter 6

The Elements of Statistical Learning

- Chapter 3.

In praise of linear models!

- Despite its simplicity, the linear model has distinct advantages in terms of its *interpretability* and often shows good *predictive performance*.
- Hence we discuss in this lecture some ways in which the simple linear model can be improved, by replacing ordinary least squares fitting with some alternative fitting procedures.

Why consider alternatives to least squares?

- *Prediction Accuracy:* especially when $p > n$, to control the variance.
- *Model Interpretability:* By removing irrelevant features — that is, by setting the corresponding coefficient estimates to zero — we can obtain a model that is more easily interpreted. We will present some approaches for automatically performing *feature selection*.

Three classes of methods

- *Subset Selection*. We identify a subset of the p predictors that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.
- *Shrinkage*. We fit a model involving all p predictors, but the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage (also known as *regularization*) has the effect of reducing variance and can also perform variable selection.
- *Dimension Reduction*. We project the p predictors into a M-dimensional subspace, where $M < p$. This is achieved by computing M different *linear combinations*, or *projections*, of the variables. Then these M projections are used as predictors to fit a linear regression model by least squares.

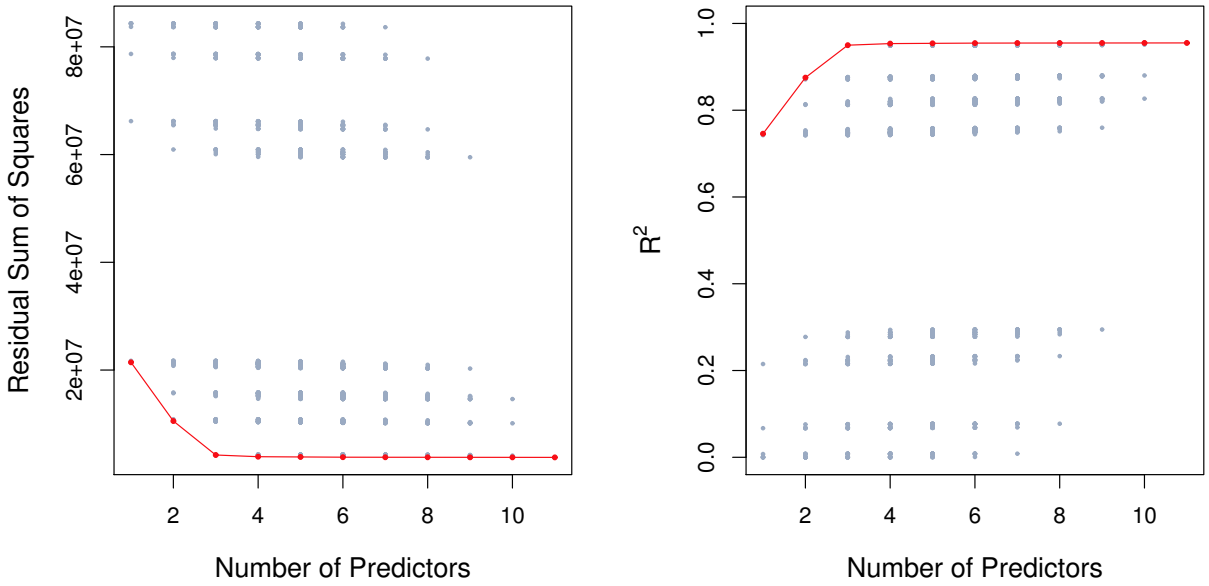
Subset Selection

Best subset and stepwise model selection procedures

Best Subset Selection

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = 1, 2, \ldots p$:
   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.
   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Example- Credit data set



For each possible model containing a subset of the ten predictors in the `Credit` *data set, the RSS and* R^2 *are displayed. The red frontier tracks the best model for a given number of predictors, according to RSS and* R^2*. Though the data set contains only ten predictors, the* x*-axis ranges from* 1 *to* 11*, since one of the variables is categorical and takes on three values, leading to the creation of two dummy variables*

Extensions to other models

- Although we have presented best subset selection here for least squares regression, the same ideas apply to other types of models, such as logistic regression.
- The *deviance*— negative two times the maximized log-likelihood— plays the role of RSS for a broader class of models.

Stepwise Selection

- For computational reasons, best subset selection cannot be applied with very large p. *Why not?*
- Best subset selection may also suffer from statistical problems when p is large: larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data.
- Thus an enormous search space can lead to *overfitting* and high variance of the coefficient estimates.
- For both of these reasons, *stepwise* methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection.

Forward Stepwise Selection

- Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all of the predictors are in the model.
- In particular, at each step the variable that gives the greatest *additional* improvement to the fit is added to the model.

In Detail

Forward Stepwise Selection

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   - 2.1 Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   - 2.2 Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

More on Forward Stepwise Selection

- Computational advantage over best subset selection is clear.
- It is not guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors. *Why not? Give an example.*

Credit data example

# Variables	Best subset	Forward stepwise
One	`rating`	`rating`
Two	`rating`, `income`	`rating`, `income`
Three	`rating`, `income`, `student`	`rating`, `income`, `student`
Four	`cards`, `income`	`rating`, `income`,
	`student`, `limit`	`student`, `limit`

The first four selected models for best subset selection and forward stepwise selection on the `Credit` *data set. The first three models are identical but the fourth models differ.*

Backward Stepwise Selection

- Like forward stepwise selection, *backward stepwise selection* provides an efficient alternative to best subset selection.
- However, unlike forward stepwise selection, it begins with the full least squares model containing all p predictors, and then iteratively removes the least useful predictor, one-at-a-time.

Backward Stepwise Selection: details

Backward Stepwise Selection

1. Let $\mathcal{M}_p$ denote the *full* model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$:
   - 2.1 Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   - 2.2 Choose the *best* among these k models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

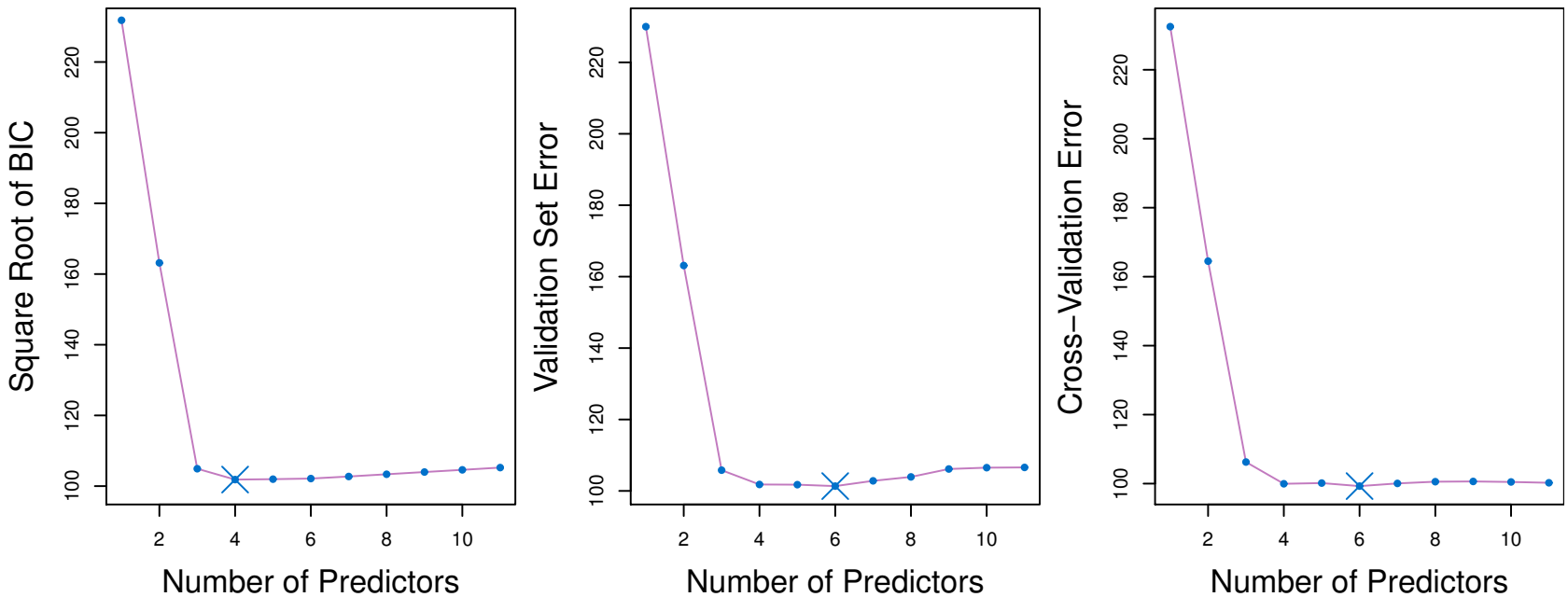
More on Backward Stepwise Selection

- Like forward stepwise selection, the backward selection approach searches through only $1 + p(p+1)/2$ models, and so can be applied in settings where p is too large to apply best subset selection
- Like forward stepwise selection, backward stepwise selection is not guaranteed to yield the *best* model containing a subset of the p predictors.
- Backward selection requires that the *number of samples n is larger than the number of variables p* (so that the full model can be fit). In contrast, forward stepwise can be used even when $n < p$, and so is the only viable subset method when p is very large.

Validation and Cross-Validation

- Each of the procedures returns a sequence of models $\mathcal{M}_k$ indexed by model size $k = 0, 1, 2, \ldots$. Our job here is to select $\hat{k}$. Once selected, we will return model $\mathcal{M}_{\hat{k}}$
- We compute the validation set error or the cross-validation error for each model $\mathcal{M}_k$ under consideration, and then select the k for which the resulting estimated test error is smallest.
- This procedure has an advantage relative to AIC, BIC, C_p, and adjusted R^2, in that it provides a direct estimate of the test error, and *doesn't require an estimate of the error variance* σ^2.
- It can also be used in a wider range of model selection tasks, even in cases where it is hard to pinpoint the model degrees of freedom (e.g. the number of predictors in the model) or hard to estimate the error variance σ^2.

Credit data example



Details of Previous Figure

- The validation errors were calculated by randomly selecting three-quarters of the observations as the training set, and the remainder as the validation set.
- The cross-validation errors were computed using $k = 10$ folds. In this case, the validation and cross-validation methods both result in a six-variable model.
- However, all three approaches suggest that the four-, five-, and six-variable models are roughly equivalent in terms of their test errors.
- In this setting, we can select a model using the *one-standard-error rule*. We first calculate the standard error of the estimated test MSE for each model size, and then select the smallest model for which the estimated test error is within one standard error of the lowest point on the curve. *What is the rationale for this?*

Shrinkage Methods

Ridge regression and *Lasso*

- The subset selection methods use least squares to fit a linear model that contains a subset of the predictors.
- As an alternative, we can fit a model containing all p predictors using a technique that *constrains* or *regularizes* the coefficient estimates, or equivalently, that *shrinks* the coefficient estimates towards zero.
- It may not be immediately obvious why such a constraint should improve the fit, but it turns out that shrinking the coefficient estimates can significantly reduce their variance.

Ridge regression

- Recall that the least squares fitting procedure estimates $\beta_0, \beta_1, \ldots, \beta_p$ using the values that minimize

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

- In contrast, the ridge regression coefficient estimates $\hat{\beta}^R$ are the values that minimize

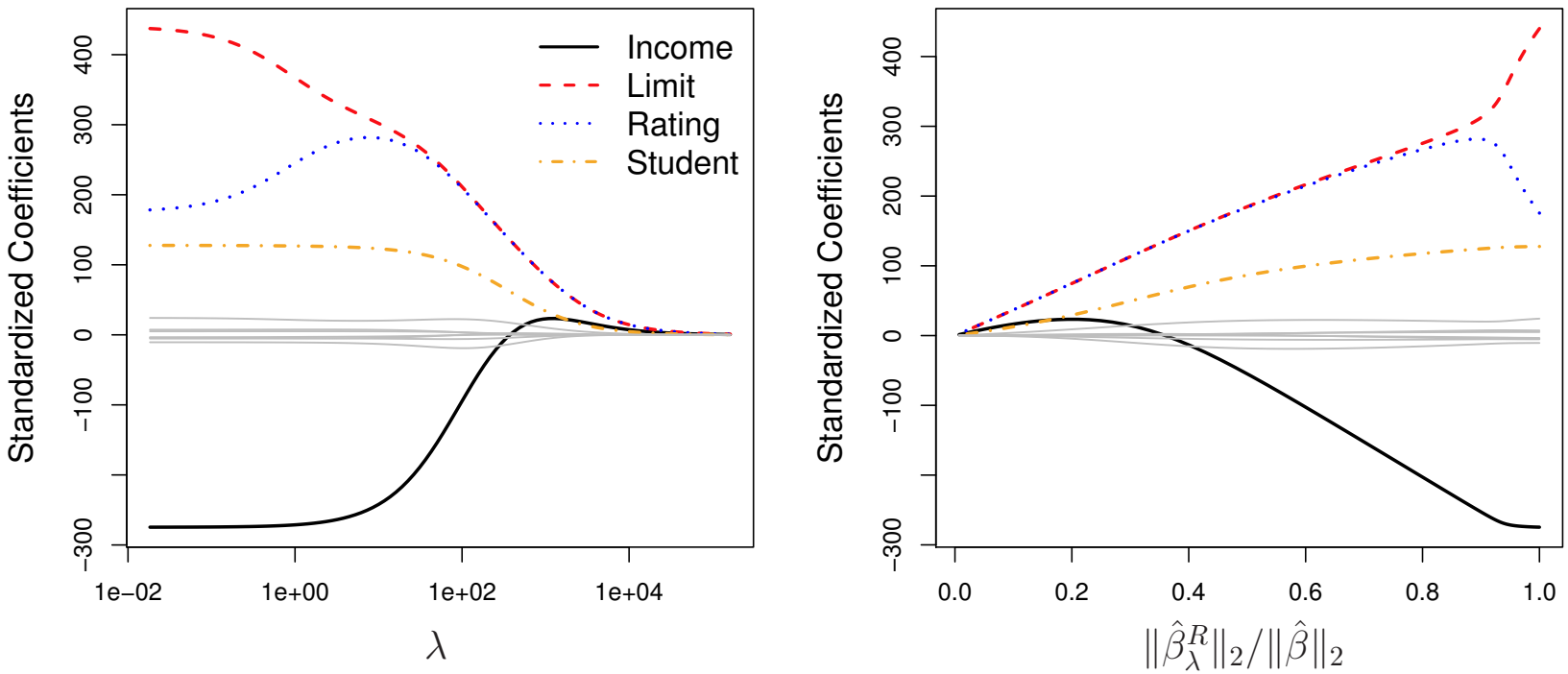
$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = \text{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2,$$

where $\lambda \geq 0$ is a *tuning parameter*, to be determined separately.

Ridge regression: continued

- As with least squares, ridge regression seeks coefficient estimates that fit the data well, by making the RSS small.
- However, the second term, $\lambda \sum_j \beta_j^2$, called a *shrinkage penalty*, is small when $\beta_1, \ldots, \beta_p$ are close to zero, and so it has the effect of *shrinking* the estimates of β_j towards zero.
- The tuning parameter λ serves to control the relative impact of these two terms on the regression coefficient estimates.
- Selecting a good value for λ is critical; cross-validation is used for this.

Credit data example



Details of Previous Figure

- In the left-hand panel, each curve corresponds to the ridge regression coefficient estimate for one of the ten variables, plotted as a function of λ.
- The right-hand panel displays the same ridge coefficient estimates as the left-hand panel, but instead of displaying λ on the x-axis, we now display $\|\hat{\beta}_\lambda^R\|_2/\|\hat{\beta}\|_2$, where $\hat{\beta}$ denotes the vector of least squares coefficient estimates.
- The notation $\|\beta\|_2$ denotes the ℓ_2 norm (pronounced "ell 2") of a vector, and is defined as $\|\beta\|_2 = \sqrt{\sum_{j=1}^{p} {\beta_j}^2}$.

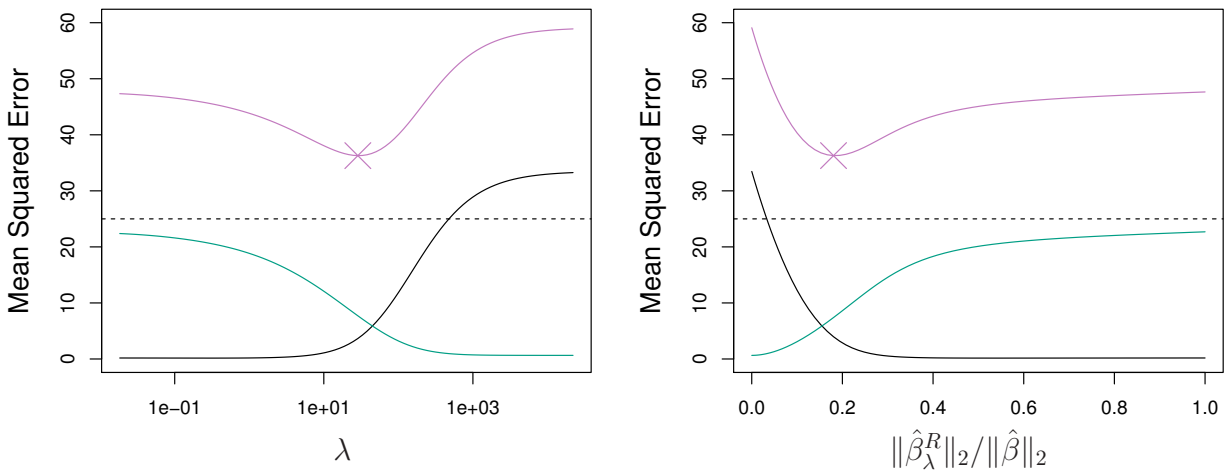
Ridge regression: scaling of predictors

- The standard least squares coefficient estimates are *scale equivariant*: multiplying X_j by a constant c simply leads to a scaling of the least squares coefficient estimates by a factor of $1/c$. In other words, regardless of how the jth predictor is scaled, $X_j\hat{\beta}_j$ will remain the same.
- In contrast, the ridge regression coefficient estimates can change *substantially* when multiplying a given predictor by a constant, due to the sum of squared coefficients term in the penalty part of the ridge regression objective function.
- Therefore, it is best to apply ridge regression after *standardizing the predictors*, using the formula

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

Why Does Ridge Regression Improve Over Least Squares?

The Bias-Variance tradeoff



Simulated data with $n = 50$ observations, $p = 45$ predictors, all having nonzero coefficients. Squared bias (black), variance (green), and test mean squared error (purple) for the ridge regression predictions on a simulated data set, as a function of λ and $\|\hat{\beta}^R_\lambda\|_2/\|\hat{\beta}\|_2$. The horizontal dashed lines indicate the minimum possible MSE. The purple crosses indicate the ridge regression models for which the MSE is smallest.

The variance of prediction errors

Let the prediction of a linear model be

$$\widehat{y}_s = x_s b,$$

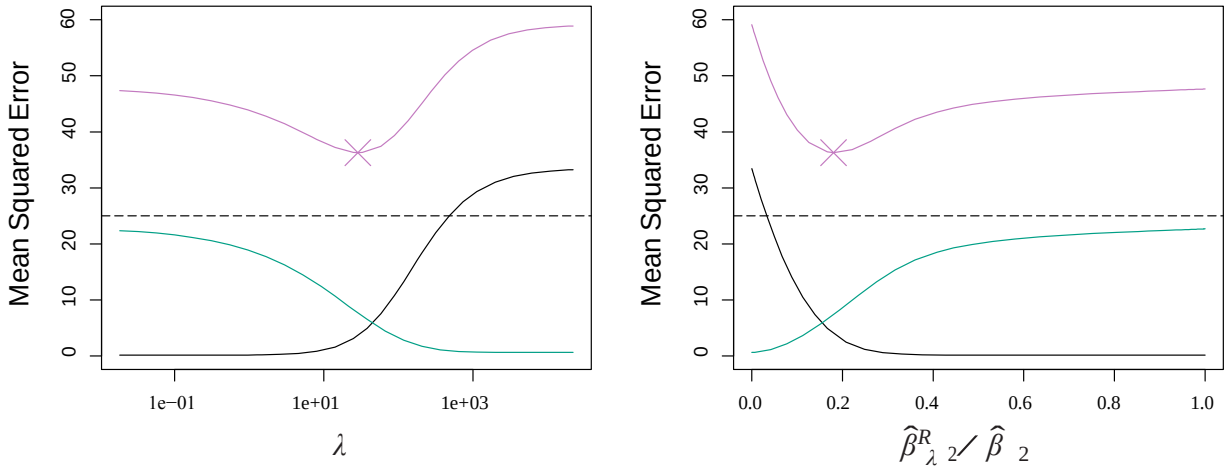
where x_s is a row vector of observables and b is a vector of estimates. Denote $E(b) = b_e$. The prediction error has variance

$$\begin{aligned} E\left[y_s - x_s b\right]^2 &= E\left[x_s\beta + \varepsilon_s - x_s b\right]^2 \\ &= E\left[\varepsilon_s^2\right] + E\left[x_s\left(\beta - b_e\right)\right]^2 + E\left[x_s\left(b - b_e\right)\right]^2 \\ &= \sigma^2 + \underbrace{E\left[x_s\left(\beta - b_e\right)\right]^2}_{bias} + x_s Var\left(b\right) x_s' \end{aligned}$$

1. The OLS estimator is lexicographic, it sets bias to zero and then minimizes the remaining bias.
2. ML estimators achieve lower variance, but bias the β-estimates.
3. $Var\left(b\right)$ increases the variance in predictions. This falls
   - if we exclude x-variables, in particular those imprecisely estimated and highly correlated with other x-variables,
   - or if we scale the b-coefficients towards zero.

Why Does Ridge Regression Improve Over Least Squares?

The Bias-Variance tradeoff



Gauss-Markov: Least Squares is the minimum variance, linear ***unbiased*** *estimator.*

Ridge Regression prediction has lower variance, but is biased.

Ridge Regression Interpretation

- Ridge regression shrinks the direction in X along the first principal component the least. This has highest variance and hence can be estimated most precisely.

The Lasso

- Ridge regression does have one obvious disadvantage: unlike subset selection, which will generally select models that involve just a subset of the variables, ridge regression will include all p predictors in the final model
- The *Lasso* is a relatively recent alternative to ridge regression that overcomes this disadvantage. The lasso coefficients, $\hat{\beta}_\lambda^L$, minimize the quantity

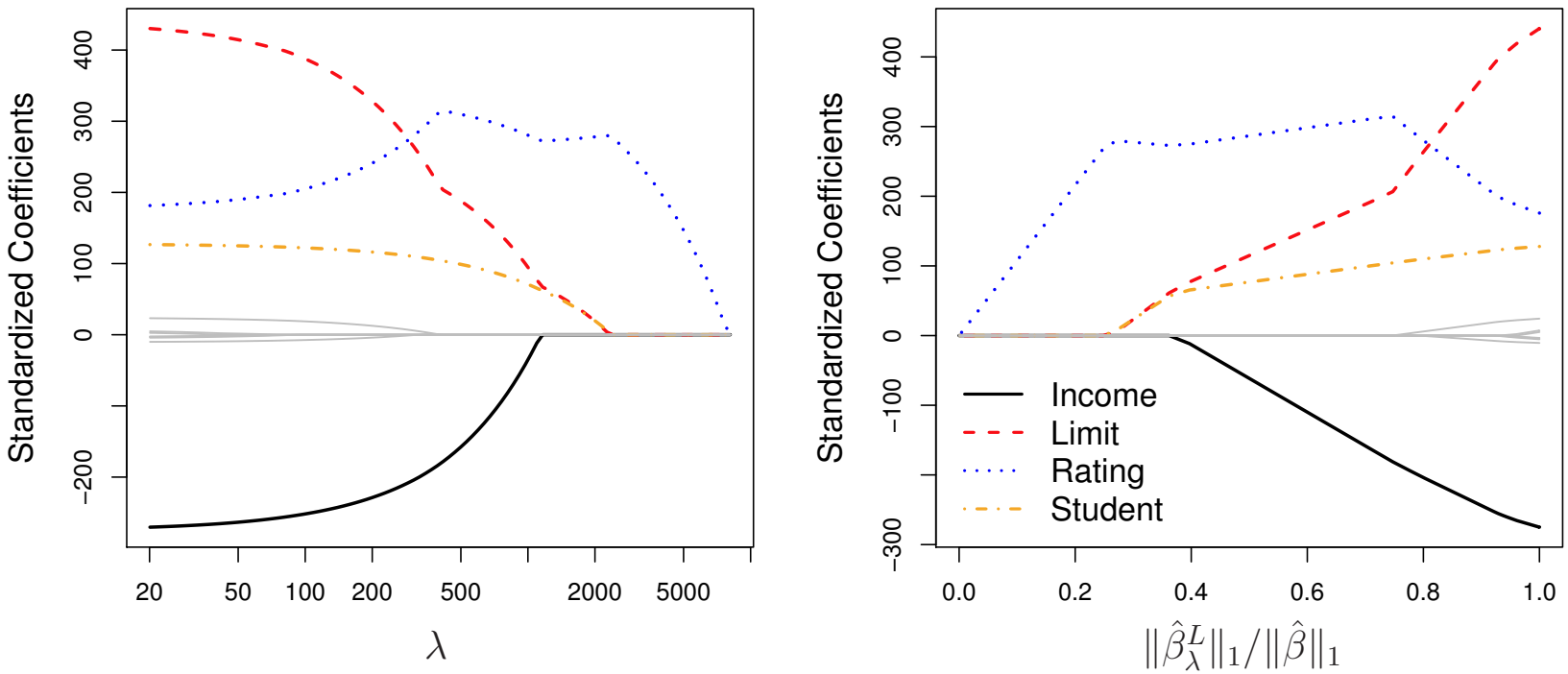
$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|.$$

- In statistical parlance, the lasso uses an ℓ_1 (pronounced "ell 1") penalty instead of an ℓ_2 penalty. The ℓ_1 norm of a coefficient vector β is given by $\|\beta\|_1 = \sum|\beta_j|$.

The Lasso: continued

- As with ridge regression, the lasso shrinks the coefficient estimates towards zero.
- However, in the case of the lasso, the ℓ_1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large.
- Hence, much like best subset selection, the lasso performs *variable selection*.
- We say that the lasso yields *sparse* models — that is, models that involve only a subset of the variables.
- As in ridge regression, selecting a good value of λ for the lasso is critical; cross-validation is again the method of choice.

Example: Credit dataset



The Variable Selection Property of the Lasso

Why is it that the lasso, unlike ridge regression, results in coefficient estimates that are exactly equal to zero?

One can show that the lasso and ridge regression coefficient estimates solve the problems

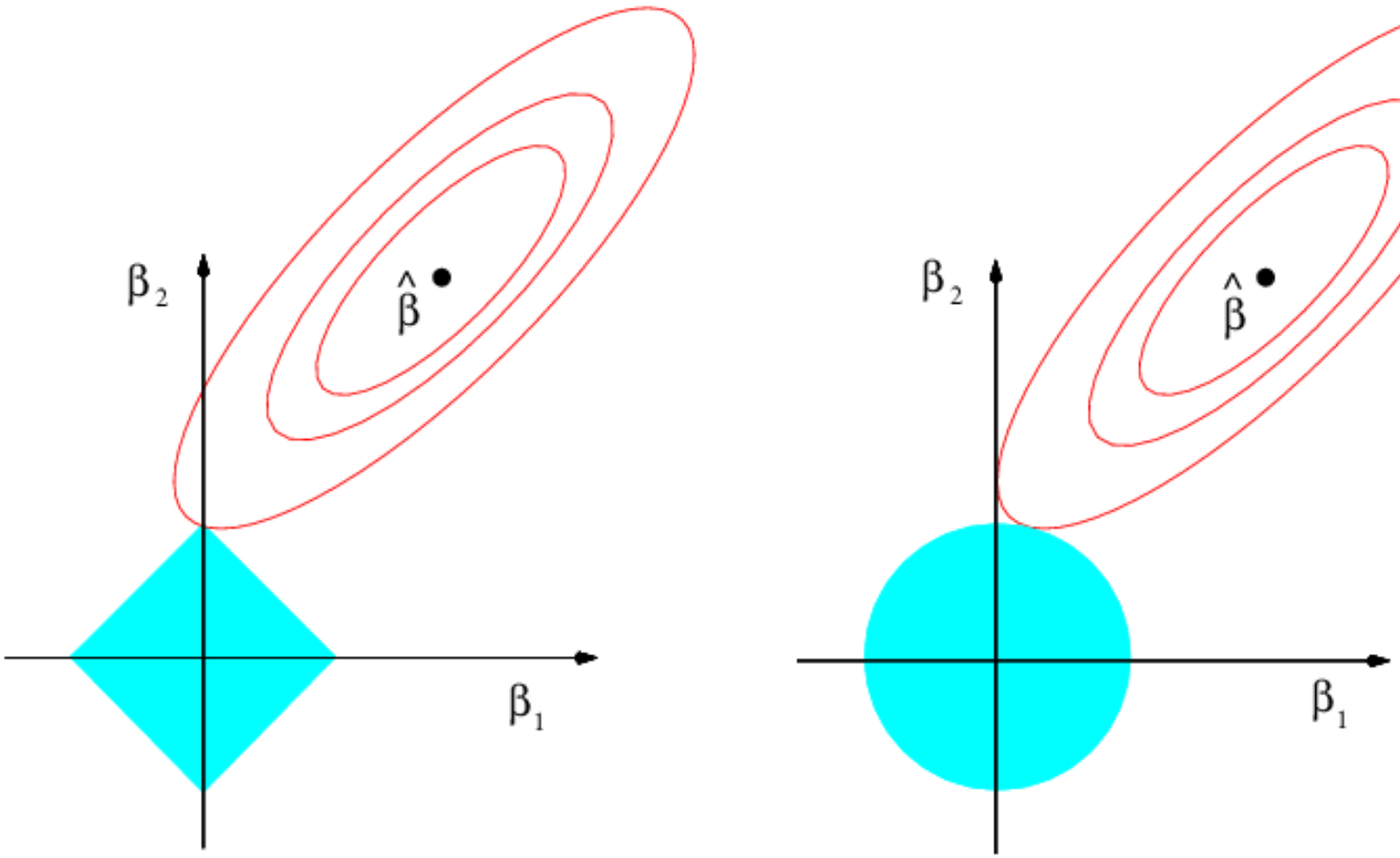
$$\underset{\beta}{\text{minimize}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

and

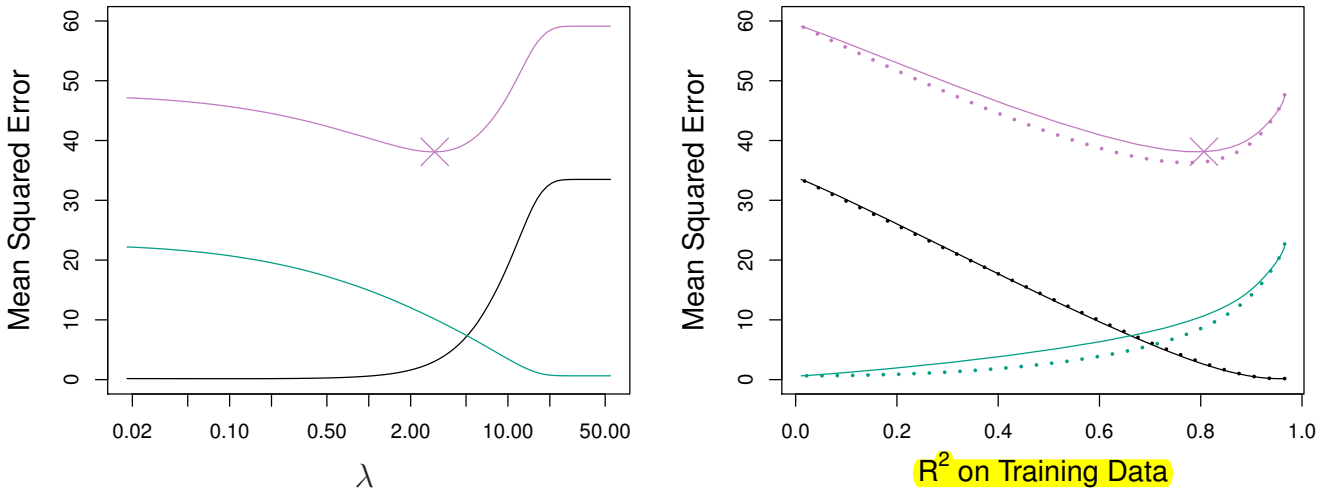
$$\underset{\beta}{\text{minimize}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s,$$

respectively.

The Lasso Picture

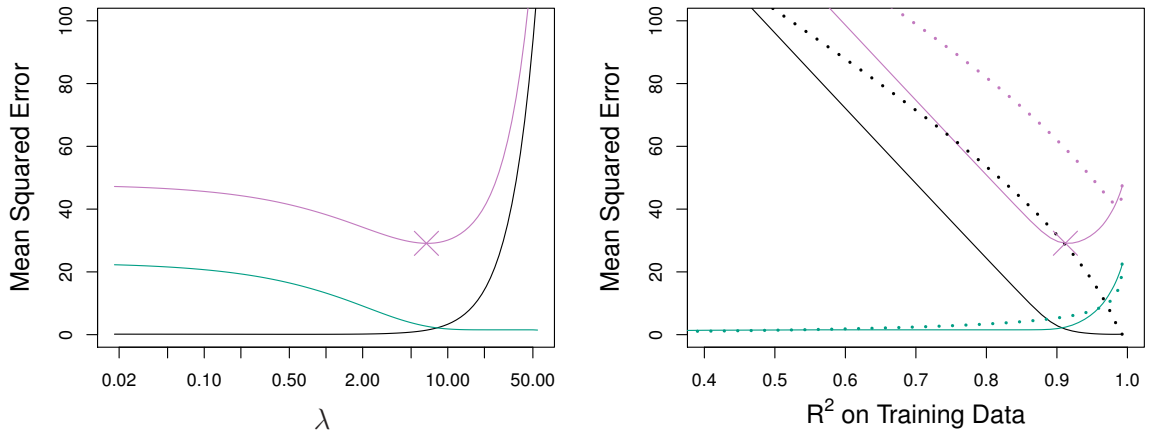


Comparing the Lasso and Ridge Regression



Left: *Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso on simulated data set of Slide 32.* *Right:* *Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dashed). Both are plotted against their* R^2 *on the training data, as a common form of indexing. The crosses in both plots indicate the lasso model for which the MSE is smallest.*

Comparing the Lasso and Ridge Regression: continued



Left: *Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso. The simulated data is similar to that in Slide 38, except that now only two predictors are related to the response.* *Right:* *Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dashed). Both are plotted against their R^2 on the training data, as a common form of indexing. The crosses in both plots indicate the lasso model for which the MSE is smallest.*

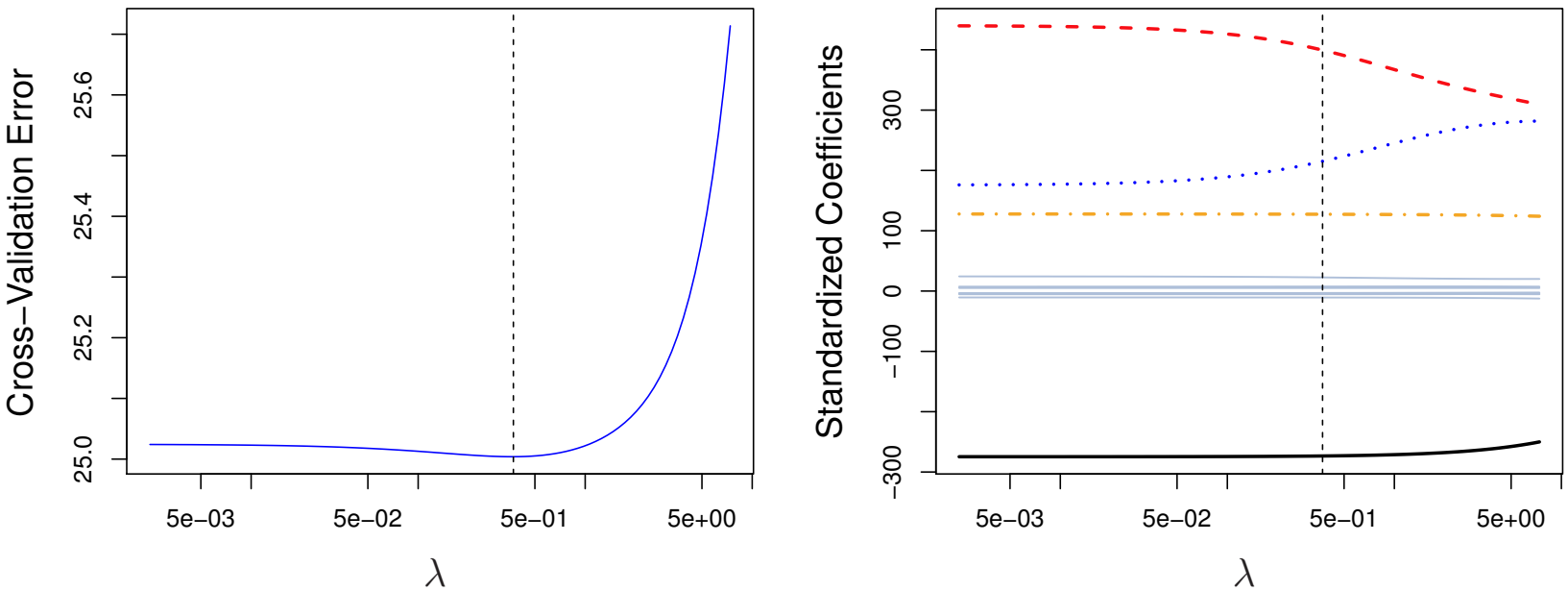
Conclusions

- These two examples illustrate that neither ridge regression nor the lasso will universally dominate the other.
- In general, one might expect the lasso to perform better when the response is a function of only a relatively small number of predictors.
- However, the number of predictors that is related to the response is never known *a priori* for real data sets.
- A technique such as cross-validation can be used in order to determine which approach is better on a particular data set.

Selecting the Tuning Parameter for Ridge Regression and Lasso

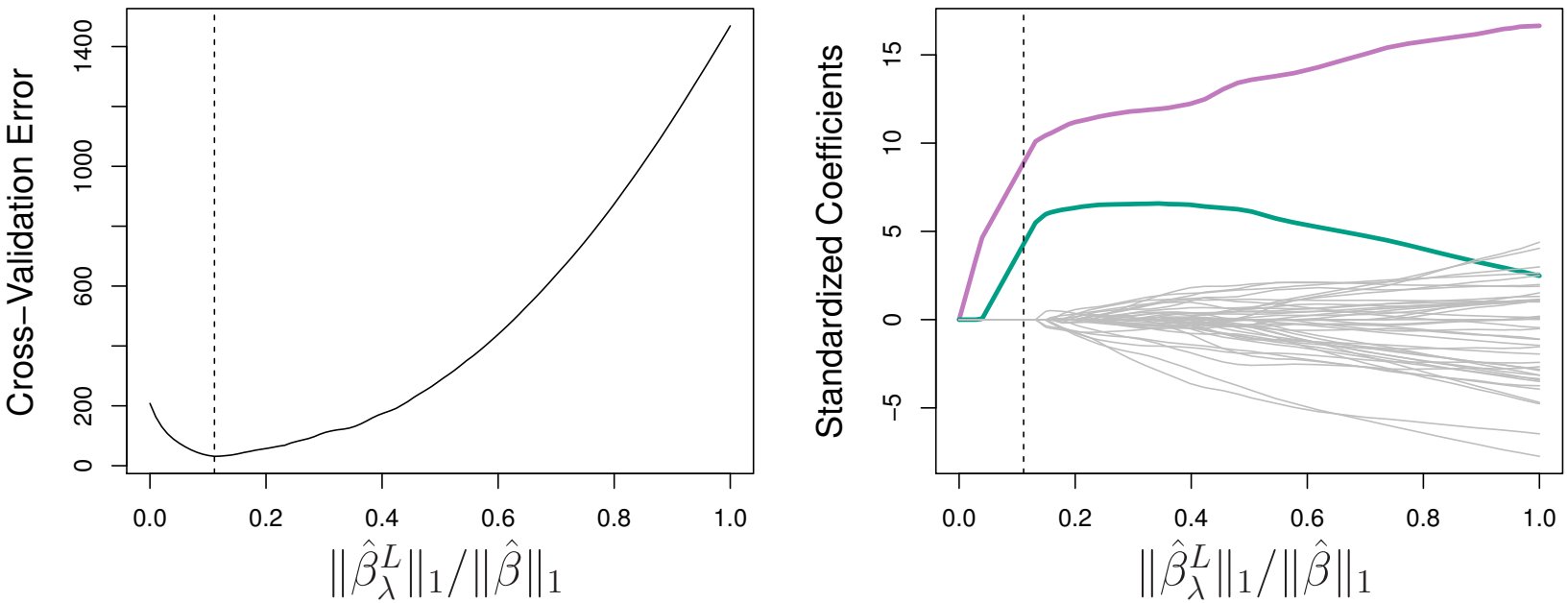
- As for subset selection, for ridge regression and lasso we require a method to determine which of the models under consideration is best.
- That is, we require a method selecting a value for the tuning parameter λ or equivalently, the value of the constraint s.
- *Cross-validation* provides a simple way to tackle this problem. We choose a grid of λ values, and compute the cross-validation error rate for each value of λ.
- We then select the tuning parameter value for which the cross-validation error is smallest.
- Finally, the model is re-fit using all of the available observations and the selected value of the tuning parameter.

Credit data example



Left: *Cross-validation errors that result from applying ridge regression to the* `Credit` *data set with various values of* λ.
Right: *The coefficient estimates as a function of* λ. *The vertical dashed lines indicates the value of* λ *selected by cross-validation.*

Simulated data example



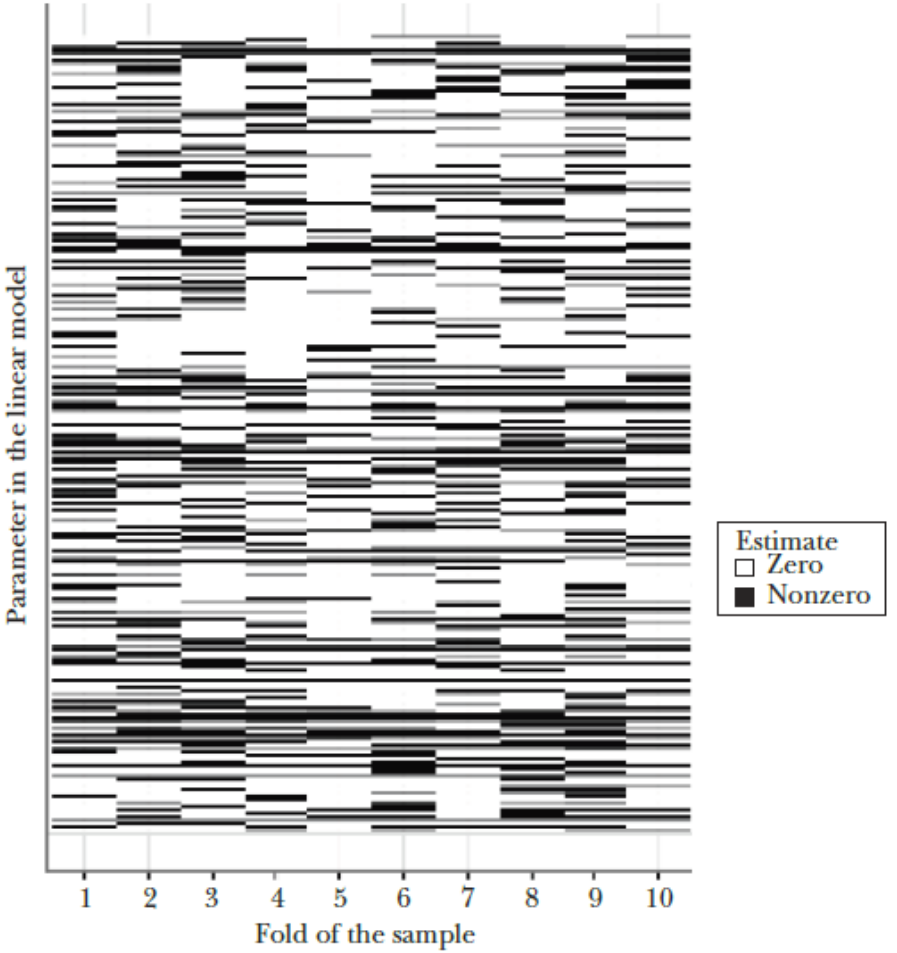
Left: *Ten-fold cross-validation MSE for the lasso, applied to the sparse simulated data set from Slide 39.* Right: *The corresponding lasso coefficient estimates are displayed. The vertical dashed lines indicate the lasso fit for which the cross-validation error is smallest.*

What is Lasso not good for?

- Learning about the underlying model.
  - LASSO produces a sparse prediction function: many coefficients are zero.
  - This selection is sensitive to multicollinearity. The prediction is stable but not the choice of included regressors.

Figure 2

Selected Coefficients (Nonzero Estimates) across Ten LASSO Regressions



Related Estimators

Least angle regression algorithm to compute entire Lasso path.

1. Standardize all p predictors. Start with $r=y-\bar{y}$, $\beta 1, \beta 2, \ldots \beta p=0$.
2. Find the predictor xj most correlated with r. Move βj from 0 towards its OLS coefficient on r until some other predictor xk has as much correlation with the current residual as xj..
3. Move xj and xk in the direction defined by their joint least squares coefficient on the current residual until some other competitor xl has as much correlation with the current residual.
4. Continue in this way until all predictors have been entered.

Elastic net penalty:

$$\alpha\beta_j^2 + (1-\alpha)|\beta_j|$$

Dimension Reduction Methods

- The methods that we have discussed so far in this chapter have involved fitting linear regression models, via least squares or a shrunken approach, using the original predictors, $X_1, X_2, \ldots, X_p$.
- We now explore a class of approaches that *transform* the predictors and then fit a least squares model using the transformed variables. We will refer to these techniques as *dimension reduction* methods.

Dimension Reduction Methods: details

- Let $Z_1, Z_2, \ldots, Z_M$ represent $M < p$ *linear combinations* of our original p predictors. That is,

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j \tag{1}$$

  for some constants $\phi_{m1}, \ldots, \phi_{mp}$.
- We can then fit the linear regression model,

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i, \quad i = 1, \ldots, n, \tag{2}$$

  using ordinary least squares.
- Note that in model (2), the regression coefficients are given by $\theta_0, \theta_1, \ldots, \theta_M$. If the constants $\phi_{m1}, \ldots, \phi_{mp}$ are chosen wisely, then such dimension reduction approaches can often outperform OLS regression.

- Notice that from definition (1),

$$\sum_{m=1}^{M} \theta_m z_{im} = \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \phi_{mj} x_{ij} = \sum_{j=1}^{p} \sum_{m=1}^{M} \theta_m \phi_{mj} x_{ij} = \sum_{j=1}^{p} \beta_j x_{ij},$$

  where

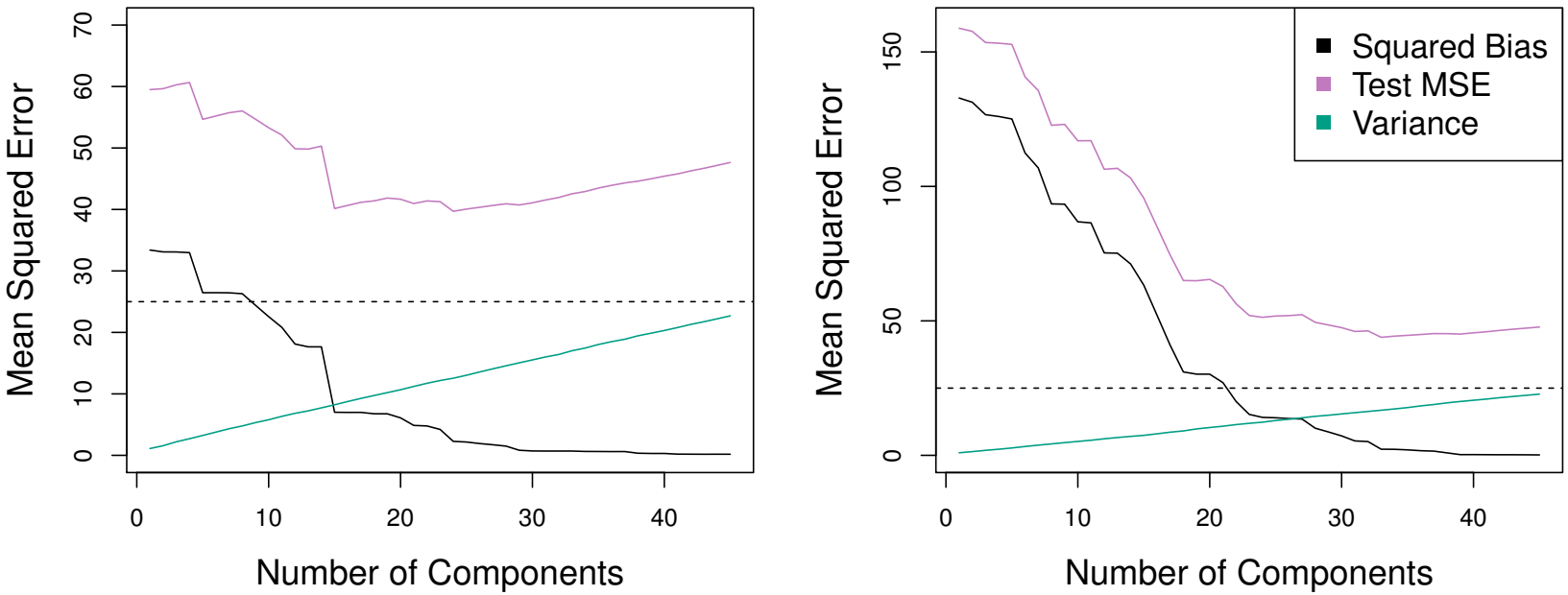
$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{mj}. \tag{3}$$

- Hence model (2) can be thought of as a special case of the original linear regression model.
- Dimension reduction serves to constrain the estimated β_j coefficients, since now they must take the form (3).
- Can win in the bias-variance tradeoff.

Principal Components Regression

- Here we apply principal components analysis (PCA) (discussed in Chapter 10 of the text) to define the linear combinations of the predictors, for use in our regression.
- The first principal component is that (normalized) linear combination of the variables with the largest variance.
- The second principal component has largest variance, subject to being uncorrelated with the first.
- And so on.
- Hence with many correlated original variables, we replace them with a small set of principal components that capture their joint variation.

Application to Principal Components Regression



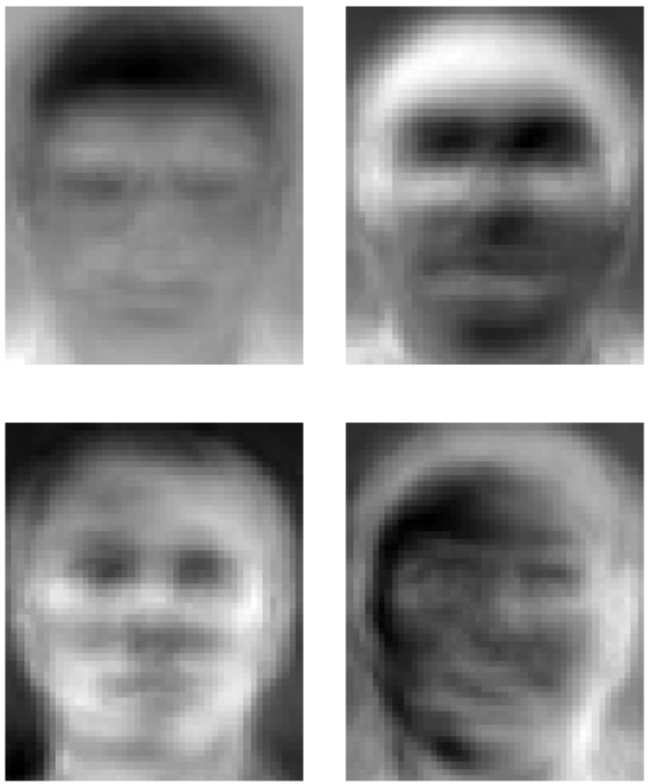
PCR was applied to two simulated data sets. The black, green, and purple lines correspond to squared bias, variance, and test mean squared error, respectively. Left: *Simulated data from slide 32.* Right: *Simulated data from slide 39.*

Face Recognition

- Large set of digitized images of human faces is taken under the same lighting conditions.
- The images are normalized to line up the eyes and mouths.
- A picture is a vector of e.g. 100x100 pixels (x-variables) with a value of 1-16 grayscale.
- The eigenvectors of the covariance matrix of the face image vectors are then extracted.
- These eigenvectors are called eigenfaces.

Eigenfaces

- The principal eigenface looks like a bland androgynous average human face
  http://en.wikipedia.org/wiki/Image:Eigenfaces.png

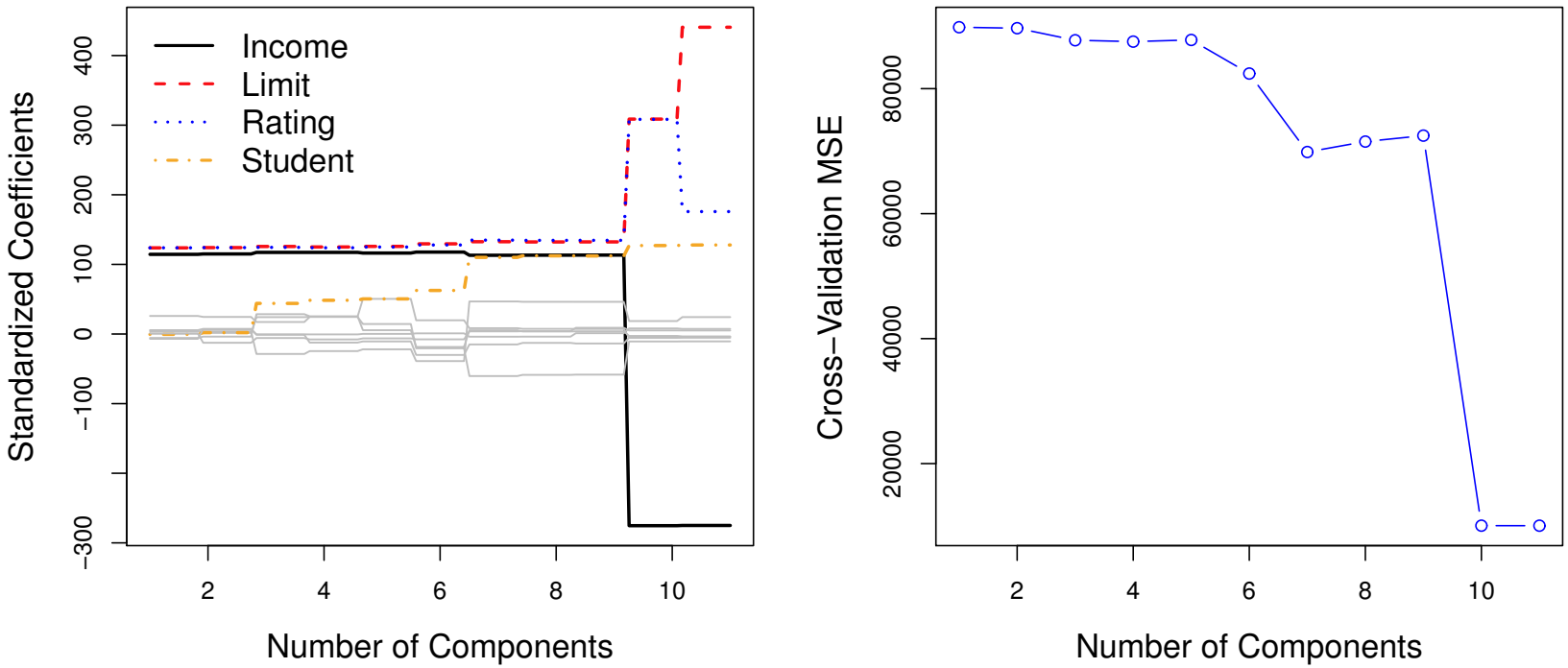


Face Recognition

- A new face can be projected onto the collection of eigenfaces, producing a vector of weight coefficients θ.
- Match with the face of minimum distance ||θ-θm||
- Most faces identified using a projection on 100-150 eigenfaces

```
coefficients =
 {-6.85693, 23.7498, -11.4515, -3.43352, 5.24749, -7.1615,
  8.09015, -9.7205, -0.660834, -2.4148, -10.3942, 3.33424,
  2.94988, -2.75981, 3.02687, -2.4499, -2.09885, -5.98832,
  -4.22564, -0.65014, 2.20144, -5.43782, -9.61821, -3.25227,
  7.49413, -0.145002, 7.61483, -0.696994, -3.7731, 3.23569,
  -1.78853, 0.0400116, -3.86804, -2.02456, 2.20949, -1.86902,
  1.23445, 0.140996, 0.698304, -0.420466, 2.30691, 3.70434,
  1.02417, 0.382809, 0.413049, -0.994902, 0.754145, 0.363418,
  -0.383865, 1.46379, 1.96381, -2.90388, -2.33381, -0.438939,
  -0.30523, -0.105925, 0.665962, -0.729409, -1.28977, 0.150497,
  0.645343, 0.30724, -1.04942, 1.0462, -0.60808, 0.333288,
  1.09659, -1.38876, 0.33875, 0.278604, 1.0632, -0.0446148,
```

Choosing the number of directions M



Left: *PCR standardized coefficient estimates on the* `Credit` *data set for different values of* M. Right: *The* 10*-fold cross validation MSE obtained using PCR, as a function of* M.

Partial Least Squares

- PCR identifies linear combinations, or *directions*, that best represent the predictors $X_1, \ldots, X_p$.
- These directions are identified in an *unsupervised* way, since the response Y is not used to help determine the principal component directions.
- That is, the response does not *supervise* the identification of the principal components.
- Consequently, PCR suffers from a potentially serious drawback: there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response.

Partial Least Squares: continued

- Like PCR, PLS is a dimension reduction method, which first identifies a new set of features $Z_1, \ldots, Z_M$ that are linear combinations of the original features, and then fits a linear model via OLS using these M new features.
- But unlike PCR, PLS identifies these new features in a supervised way – that is, it makes use of the response Y in order to identify new features that not only approximate the old features well, but also that *are related to the response*.
- Roughly speaking, the PLS approach attempts to find directions that help explain both the response and the predictors.

Details of Partial Least Squares

- After standardizing the p predictors, PLS computes the first direction Z_1 by setting each ϕ_{1j} in (1) equal to the coefficient from the simple linear regression of Y onto X_j.
- One can show that this coefficient is proportional to the correlation between Y and X_j.
- Hence, in computing $Z_1 = \sum_{j=1}^{p} \phi_{1j} X_j$, PLS places the highest weight on the variables that are most strongly related to the response.
- Subsequent directions are found by taking residuals and then repeating the above prescription.

High-Dimensional Data: p>>n

- Cp, AIC and BIC are not appropriate
  because estimating $\hat{\sigma}$ is problematic.
- The test error tends to increase as *p* increases, unless the
  additional features are truly associated with the response.
  - Costly to throw in 1 million unrelated predictors..
- Extreme multicollinearity:
  any variable can be written as linear combination of the others.
  - Hard to interpret coefficients.
  - Selected model probably not unique.

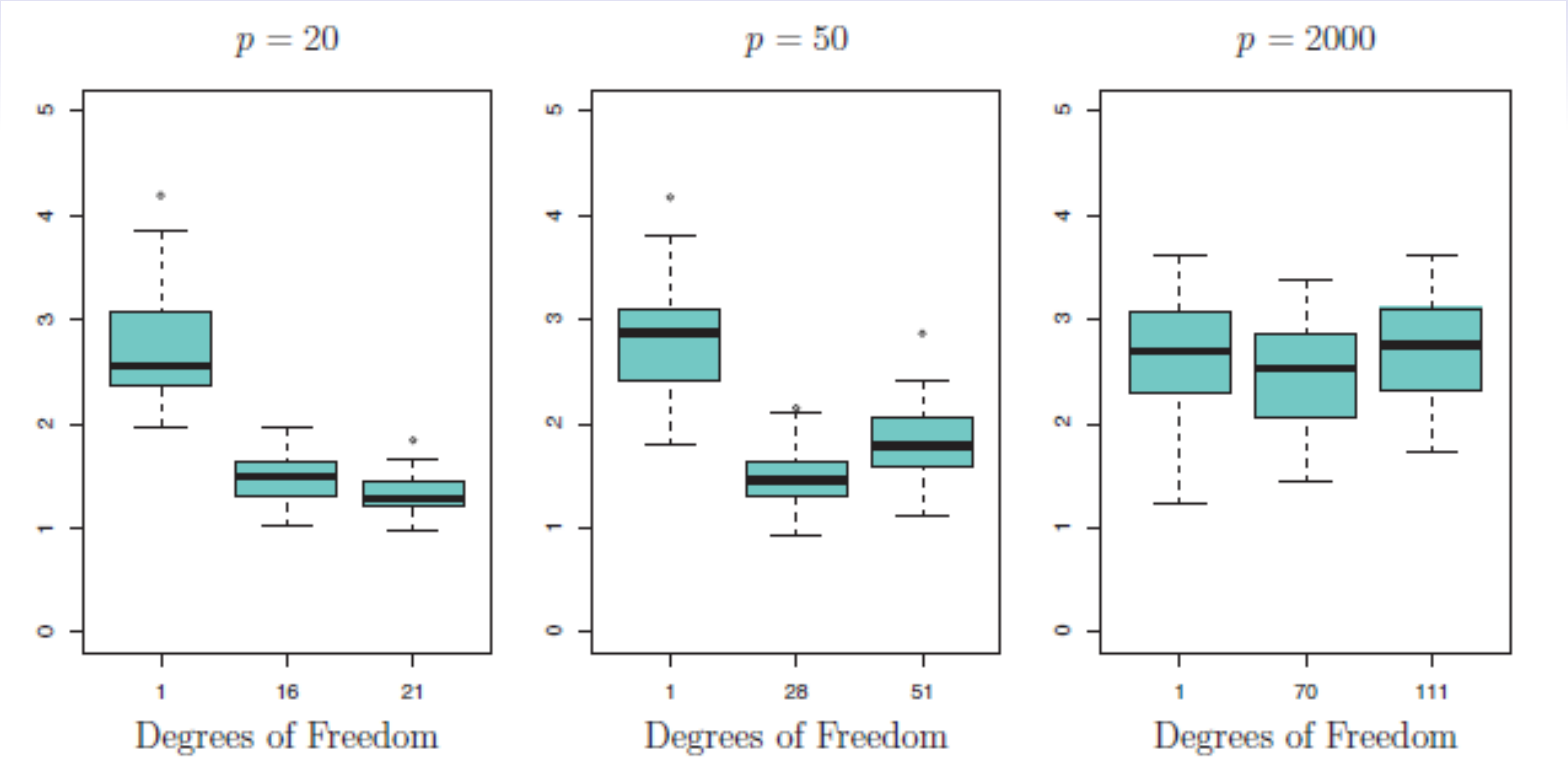


FIGURE 6.24. *The lasso was performed with $n = 100$ observations and three values of p, the number of features. Of the p features, 20 were associated with the response. The boxplots show the test MSEs that result using three different values of the tuning parameter λ in (6.7). For ease of interpretation, rather than reporting λ, the* degrees of freedom *are reported; for the lasso this turns out to be simply the number of estimated non-zero coefficients. When $p = 20$, the lowest test MSE was obtained with the smallest amount of regularization. When $p = 50$, the lowest test MSE was achieved when there is a substantial amount of regularization. When $p = 2{,}000$ the lasso performed poorly regardless of the amount of regularization, due to the fact that only 20 of the 2,000 features truly are associated with the outcome.*

Model selection and growth regressions

Background

In 1997, Xavier Sala-i-Martin published the article "I just ran two million regressions" where he tested the robustness of growth regressions by running growth regressions with various permutations of variables. His article was an early application of model selection methods to cross-country growth data.

In this exercise, you will perform a similar analysis using modern statistical learning methods. The difference is that your aim will be prediction rather than parameter robustness.

Data

There are two datasets connected to this exercise: "*growthdata*92_02" and "*growthdata*02_11". They are provided both in csv-form on Athena. The datasets cover the periods 1992-2002 and 2002-2011 respectively. [1] The datasets contain 112 countries and both have the same variables: country code (iso3), average growth over the covered time period (growth), and 27 country characteristics at the start of the period. The country characteristics and their sources are described in the excel file "*vardescriptions.xlsx*" which you can find on Mondo.

Exercise 1: Within-time period prediction

In the first section, you will only use the 1992-2002 dataset. Remove 20% of the dataset to use for testing. For the remaining 80% of the dataset, implement the prediction methods which have been taught in class: subset selection, lasso, ridge regression and principal component regression. Use 10-fold cross validation to select the best model for each methodology. Comment on the variables that are included as important when this is applicable.

This analysis will be redone with new data in Exerice 2 and 3 so it is helpful if the code allows reproduction.

As benchmark for testing the performance of your estimators, also create a naive prediction where you gess that each country in the test sample will

[1] One time period is 10 years and one is 9 years as the Penn World Table GDP data ends at 2011 and some political variables for former Soviet republics start at 1992.

have the mean growth in the training sample, as well as a "kitchen sink" regression where you throw in all variables.

Test the root mean square error of the different prediction methods, as well as of the naive predictor and the kitchen sink regression. Which was the best predictor out of sample? How did they compare to the naive estimators?

Exercise 2: Out of sample prediction

In Problem 1 we did an out of sample prediction on 20% on the dataset from the same time period. This is interesting to try out the estimators, but the relevance of the prediction exercise is limited as we used the $1992 - 2002$ growth data in the training data set, so this analysis could only been done post-2002, and then we could just check Penn World Table for the true results in the test sample.

A more interesting exercise is to test the predictions genuinely out of sample. In the $2002 - 2011$ dataset we have exactly the same variables as in the 1992-2002 dataset. This means that we can use our estimated predictors from the $1992 - 2002$ dataset to predict growth in $2002 - 2011$.

Load the $2002 - 2011$ dataset. Use the preferred estimator from you prediction exercise and calculate the predicted average growth rate using the initial state of countries in 2002 (note that you should redo the prediction method from Problem 1 using all 1992-2002 data including the 20% test set). Do the same for the naive estimator and the kitchen sink regression. Calculate the root mean squared error. What do you find? Is the fit as good as when you did it within the same time-period?

Exercise 3: Testing for changing data generating process

Redo the analysis in Problem 1 using the 2002-2011 data. Compare the results from the subset selection exercises: does the algorithm pick up the same variables? If you do the analysis with the same variables, are the parameter estimates stable across the two sets?

Discuss what, if anything, this teaches us about the results in Exercise 2. How is this related to the Lucas critique?

Software

On the homepage of *An Introduction to Statistical Learning* (`http://www-bcf.usc.edu/~gareth/ISL/` you can find very helpful code on implementing the various model selection methods. They are in Chapter 6 labs. The details of

how the code implementation relates to the theory is discussed in Chapter 6 in the book.

In Stata, check https://www.stata.com/features/lasso/ for Ridge (Elastic net) and Lasso commands. For subset selection, check Stata stepwise command.