High-dimensional regression

Advanced Methods for Data Analysis (36-402/36-608)

Spring 2014

1 Back to linear regression

1.1 Shortcomings

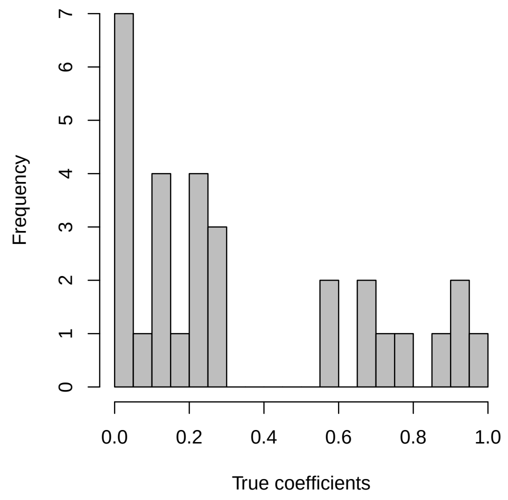
- Suppose that we are given outcome measurements $y_1, \dots y_n \in \mathbb{R}$, and corresponding predictor measurements $x_1, \dots x_n \in \mathbb{R}^p$. We know well at this point that to model y_i as a linear function of x_i, across all $i = 1, \dots n$, we can use linear regression, i.e., solve the least squares problem
$$\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^n (y_i - x_i^T \beta)^2 \quad \Longleftrightarrow \quad \min_{\beta \in \mathbb{R}^p} \|y - x\beta\|_2^2.$$
Hre $x \in \mathbb{R}^{n \times p}$ denotes the predictor matrix, and $y \in \mathbb{R}^n$ the outcome vector. Given the optimal coefficients $\hat{\beta}$, we then we make a prediction for the outcome at a future point x_0 by $\hat{\beta}^T x_0$
- There are two important shortcomings of linear regression to be aware of—and these don't even have to do with the linear model assumption! Let's even assume that the underlying model is actually close to (or exactly) linear, i.e.,
$$y_i = r(x_i) + \epsilon_i, \quad i = 1, \dots n,$$
for some underlying regression function $r(x_0)$ that is approximately (or exactly) linear in x_0. The two shortcomings:
  1. *Predictive ability:* the linear regression fit often has *low bias* but *high variance.* Recall that expected test error is a combination of these two quantities. Prediction accuracy can sometimes be improved by sacrificing some small amount of bias in order to decrease the variance
  2. *Interpretative ability:* linear regression "freely" assigns a coefficient to each predictor variable. When the number of variables p is large, we may sometimes seek, for the sake of interpretation, a smaller set of *important variables.* Hence we want to "encourage" our fitting procedure to make only a subset of the coefficients large, and others small or even zero
- These shortcomings become major problems in a *high-dimensional* regression setting, where the number of predictors p rivals—or even exceeds—the number of observations n. In fact, when $p > n$, the linear regression estimate is actually not well-defined
- Recall that the linear regression estimate can be written as
$$\hat{\beta} = (x^T x)^{-1} x^T y.$$
This is well-defined when the predictor variables, i.e., the columns of x, are linearly independent. But when $p > n$, the matrix x cannot have linearly independent columns, and $x^T x$ is not invertible

1.2 An example

- Consider a small simulation study with $n = 50$ and $p = 30$. The entries of the predictor matrix $x \in \mathbb{R}^{50\times 30}$ were all drawn i.i.d. from $N(0,1)$, so overall the predictor variables have low correlation
- We consider the underlying linear model

$$y_i = x_i^T \beta^* + \epsilon_i, \quad i = 1, \dots 50,$$

  where $\epsilon_1, \dots \epsilon_n$ are all i.i.d. $N(0,1)$, and the true coefficient vector β^* has 10 large components (between 0.5 and 1) and 20 small components (between 0 and 0.3). Here is a histogram of the components of $\beta^* \in \mathbb{R}^{30}$:



- We repeated the following 100 times:
  - Generate an outcome vector y from the linear model
  - Compute the linear regression estimate $\hat{\beta}$
  - Generate a new outcome vector y' from the linear model
  - Record the test error $1/n \sum_{i=1}^n (y_i' - x_i^T \hat{\beta})^2$

  We averaged these test errors over the 100 repetitions to get an estimate of the expected test error
- We also estimated the squared bias and variance of $x_i^T \hat{\beta}$, and averaged this over $i = 1, \dots n$. Recall that it should be the case that the expected test error $= \sigma^2$ + squared bias + variance (here $\sigma^2 = 1$)
- The results: squared bias ≈ 0.006, variance ≈ 0.627, test error $\approx 1 + 0.006 + 0.627 = 1.633$
- In other words, we can see that the bias of the linear regression estimate is essentially zero, and the variance is responsible for nearly all of the error in prediction. You will check on your homework that, for a truly linear underlying model, the bias of the linear regression estimate is exactly 0, and the variance is $p \cdot \sigma^2/n$

1.3 How can we do better?

- For a truly linear underlying model, the linear regression has expected test error $\sigma^2 + p \cdot \sigma^2/n$. The first term is the irreducible error, the second term comes entirely from the variance of the linear regression estimate (averaged over the input points). Its bias is exactly zero
- What can we learn from this? If we add another predictor variable into the mix, then it will add the same amount of variance, σ^2/n, regardless of whether its true coefficient is large or small (or zero)
- So in the last example, we were "spending" variance in trying to fit truly small coefficients—there were 20 of them, out of 30 total
- One might think therefore that we can we do better by *shrinking* small coefficients towards zero, which potentially introduces some bias, but also potentially reduces the variance. In other words, this is trying to ignore some "small details" in order to get a more stable "big picture". If done properly, this will actually work, as we'll see next

2 Ridge regression

2.1 Definition

- *Ridge regression* is like ordinary linear regression, but it shrinks the estimated coefficients towards zero. The ridge coefficients are defined by solving

$$\begin{aligned} &\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^n (y_i - x_i^T \beta)^2 + \lambda \sum_{j=1}^p \beta_j^2 \\ \iff \quad &\min_{\beta \in \mathbb{R}^p} \underbrace{\|y - x\beta\|_2^2}_{\text{Loss}} + \lambda \underbrace{\|\beta\|_2^2}_{\text{Penalty}} \end{aligned}$$

  Here $\lambda \geq 0$ is a tuning parameter, which controls the strength of the penalty term. Write $\hat{\beta}^{\text{ridge}}$ as the ridge solution. Note that:

  - When $\lambda = 0$, we get the linear regression estimate
  - When $\lambda = \infty$, we get $\hat{\beta}^{\text{ridge}} = 0$
  - For λ in between, we are balancing two ideas: fitting a linear model of y on x, and shrinking the coefficients

- When including an intercept term in the regression, we usually leave this coefficient unpenalized. Otherwise we could add some constant amount c to the vector y, and this would not result in the same solution. Hence ridge regression with intercept solves

$$\min_{\beta_0 \in \mathbb{R},\, \beta \in \mathbb{R}^p} \|y - \beta_0 \mathbb{1} - x\beta\|_2^2 + \lambda \|\beta\|_2^2,$$

  where $\mathbb{1}$ is the vector of all 1s. If we center the columns of x, then the intercept estimate ends up just being $\hat{\beta}_0 = \bar{y}$, so we usually just assume that y, x have been centered and don't include an intercept

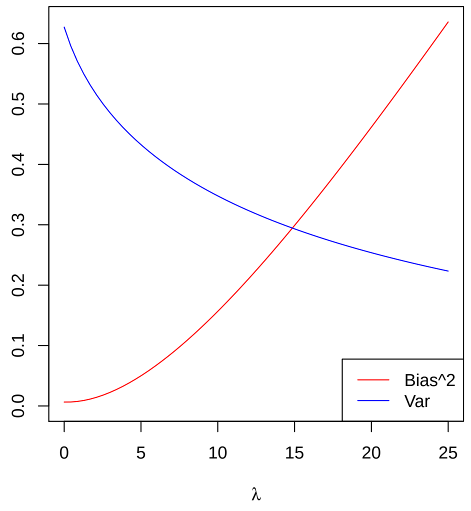
- Also, the penalty term $\|\beta\|_2^2 = \sum_{j=1}^p \beta_j^2$ is unfair is the predictor variables are not on the same scale. (Why?) Therefore, if we know that the variables are not measured in the same units, we typically scale the columns of x (to have sample variance 1), and then we perform ridge regression

2.2 Bias and variance

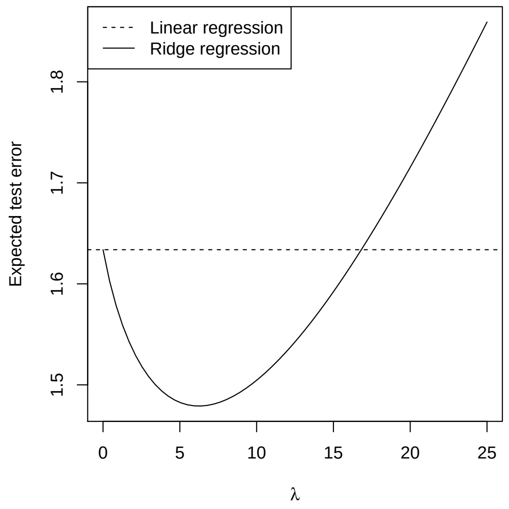
- The bias and variance are not quite as simple to write down for ridge regression as they were for linear regression, but closed-form expressions are still possible. The general trend is:
  - The bias increases as λ (amount of shrinkage) increases
  - The variance decreases as λ (amount of shrinkage) increases

  Think: what is the bias at $\lambda = 0$? The variance at $\lambda = \infty$?

- Back to the example we examined previously ($n = 50$, $p = 30$, 10 large true coefficients, 20 small). Here is the bias and variance of ridge regression estimates across different λ values:



- Expected test error for this example setup:



Linear regression:
Squared bias ≈ 0.006
Variance ≈ 0.627
Test error $\approx 1 + 0.006 + 0.627 = 1.633$

Ridge regression, at its best:
Squared bias ≈ 0.077
Variance ≈ 0.403
Test error $\approx 1 + 0.077 + 0.403 = 1.480$

- What you may (should) be thinking now: yes, OK, but this only work for some values of λ—so how would we choose λ in practice? This is actually a challenging question; we'll talk about this a bit later, but as you can imagine, one way to do this involves cross-validation

- Another question: what happens when we none of the true coefficients are small? In other words, if all the true coefficients are moderate or large, is it still helpful to shrink the coefficient estimates? The answer is (perhaps surprisingly) still "yes". But the advantage of ridge regression here is less dramatic, and the corresponding range for good values of λ is smaller

2.3 Variable selection

- To the other extreme (of a subset of small coefficients), suppose that there is a group of true coefficients that are identically zero. That is, that the mean outcome doesn't depend on these predictors at all, so they are completely extraneous
- The problem of picking out the relevant variables from a larger set is called *variable selection.* In the linear model setting, this means estimating some coefficients to be exactly zero. Aside from predictive accuracy, this can be very important for the purposes of model interpretation
- So how does ridge regression perform if a group of the true coefficients was exactly zero? The answer depends whether on we are interested in prediction or interpretation. In terms of prediction, the answer is effectively exactly the same as what happens with a group of small true coefficients—there is no real difference
- But for interpretation purposes, ridge regression does not provide as much help as we'd like. This is because it shrinks components of its estimate toward zero, but never sets these components to be zero exactly (unless $\lambda = \infty$, in which case all components are zero). So strictly speaking, ridge regression does not perform variable selection

3 Lasso regression

3.1 Definition

- *Lasso regression*[1] also solves a penalized least squares problem, but a different one:

$$\begin{aligned} &\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^n (y_i - x_i^T \beta) + \lambda \sum_{j=1}^p |\beta_j| \\ \iff \quad &\min_{\beta \in \mathbb{R}^p} \underbrace{\|y - x\beta\|_2^2}_{\text{Loss}} + \lambda \underbrace{\|\beta\|_1}_{\text{Penalty}} . \end{aligned}$$

  The difference between lasso and ridge regression is that the latter uses a (squared) ℓ_2 penalty $\|\beta\|_2^2$, while the former uses an ℓ_1 penalty $\|\beta\|_1$. Even though these problems look similar, their solutions behave very differently!

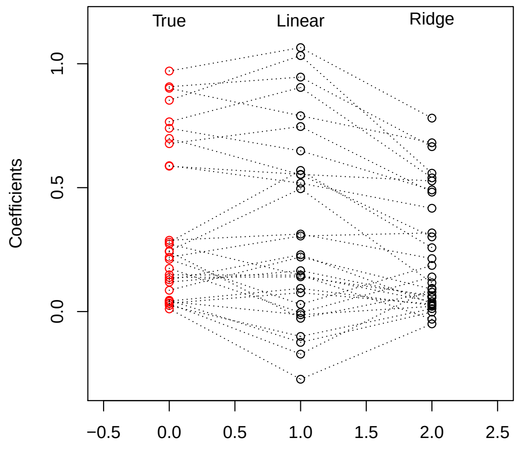
- Write $\hat{\beta}^{\text{lasso}}$ for the lasso solution. As before, the tuning parameter λ controls the strength of the penalty, and we get $\hat{\beta}^{\text{lasso}}$ = the linear regression estimate when $\lambda = 0$, and $\hat{\beta}^{\text{lasso}} = 0$ when $\lambda = \infty$. Again, for λ in between these two extremes, we are balancing two ideas: fitting a linear model of y on x, and shrinking the coefficients. But the nature of the ℓ_1 penalty is such that some coefficients are shrunken to *zero exactly*
- This is what makes the lasso substantially different from ridge regression: it is able to perform variable selection in the linear model. As λ increases, more coefficients are set to zero (less variables are selected), and among the nonzero coefficients, more shrinkage is employed

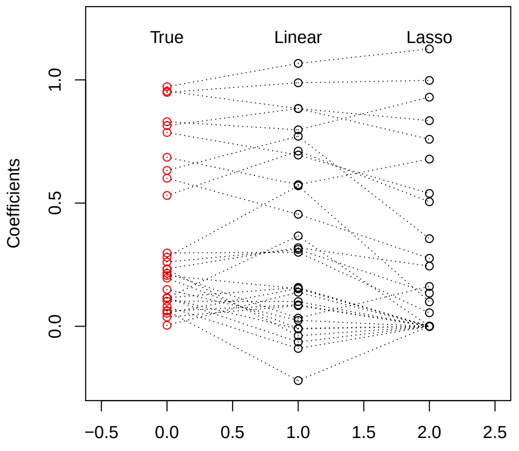
[1]Note that the name "lasso" is actually an acronym for: Least Absolute Selection and Shrinkage Operator.

- As was the case with ridge regression, centering y, x has the same effect as including an (unpenalized) intercept term in the lasso problem. And, if we know that the variables are not on the same scale to begin with, then we would scale the columns of x before fitting the lasso estimate

3.2 Bias and variance

- The lasso not only set coefficients to zero exactly, but it also shrinks the nonzero coefficients less than does ridge regression. (Think of the absolute value versus squared penalties.) On the running example ($n = 50$, $p = 30$, 10 large true coefficients, 20 small), here is a visual comparison of the ridge regression and lasso estimates:

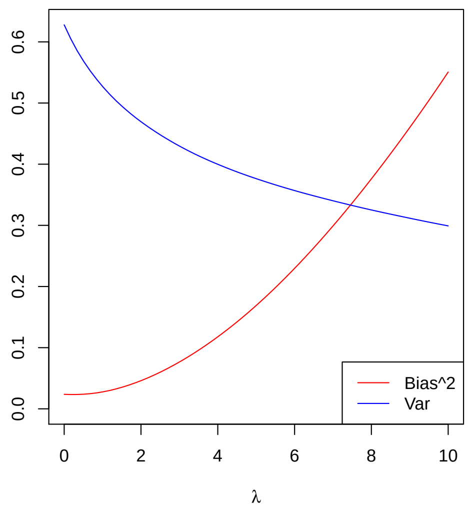




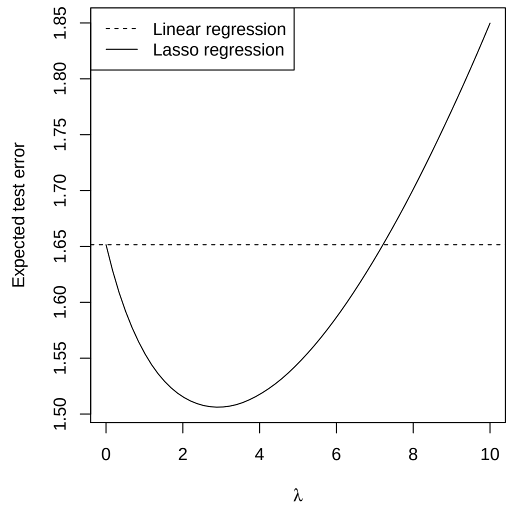
The ridge and lasso estimates have been tuned to have the same degrees of freedom

- The bias and variance of lasso regression estimates behave in the same general manner as they did for ridge regression. The bias increases as λ (amount of shrinkage) increases, and the the variance decreases as λ (amount of shrinkage) increases

- In the running example, here is the bias and variance of lasso estimates across different λ values:



- And here is the expected test error for this example setup:



Linear regression:
Squared bias ≈ 0.006
Variance ≈ 0.627
Test error $\approx 1 + 0.006 + 0.627 = 1.633$

Lasso regression, at its best:
Squared bias ≈ 0.072
Variance ≈ 0.434
Test error $\approx 1 + 0.072 + 0.434 = 1.506$

4 Degrees of freedom

- Recall that degrees of freedom is a way of quantifying the effective number of parameters used by a fitting procedure. Given $y \sim N(\mu, \sigma^2 I)$, and fitted values $\hat{y} = (\hat{y}_1, \ldots \hat{y}_n)$ coming from a fitting procedure, recall that we define

$$\mathrm{df}(\hat{y}) = \frac{1}{\sigma^2} \sum_{i=1}^{n} \mathrm{Cov}(\hat{y}_i, y_i) = \frac{1}{\sigma^2} \mathrm{tr}\Big(\mathrm{Cov}(\hat{y}, y)\Big).$$

  E.g., we might be interested in the degrees of freedom of $\hat{y}^{\mathrm{ridge}} = x\hat{\beta}^{\mathrm{ridge}}$ or $\hat{y}^{\mathrm{lasso}} = x\hat{\beta}^{\mathrm{lasso}}$ (each for a fixed value of the tuning parameter λ)

- Fortuitously, we can calculate degrees of freedom exactly for ridge regression, and we can calculate an unbiased estimate of degrees of freedom for the lasso. For ridge regression, this is

$$\mathrm{df}(\hat{y}^{\mathrm{ridge}}) = \mathrm{tr}\Big(x(x^T x + \lambda I)^{-1} x^T\Big),$$

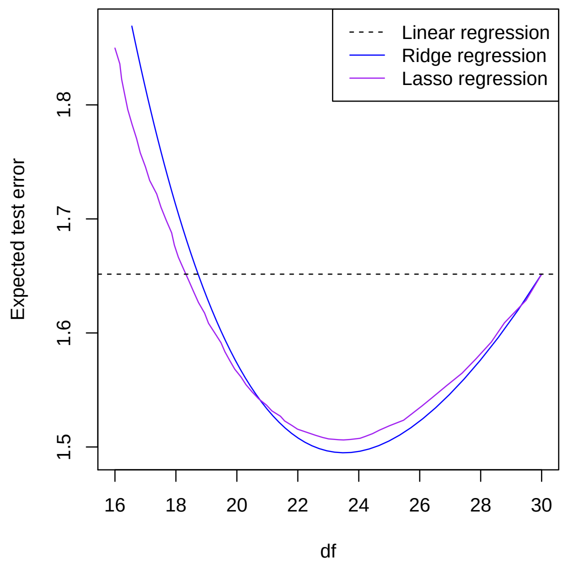
  where I is the $p \times p$ identity matrix. For the lasso, we have

$$\widehat{\mathrm{df}}(\hat{y}^{\mathrm{lasso}}) = \# \text{ of nonzero components in } \hat{\beta}^{\mathrm{lasso}}$$

  as an unbiased estimate of its degrees of freedom, i.e.,

$$\mathrm{df}(\hat{y}^{\mathrm{lasso}}) = \mathbb{E}\Big[\# \text{ of nonzero components in } \hat{\beta}^{\mathrm{lasso}}\Big]$$

- Using this, we can now, e.g., reparametrize the error plots from the running example to compare ridge regression and the lasso over their range of tuning parameter values



Pretty similar performance here. But remember, with, e.g., 24 degrees of freedom, the lasso is only using 24 variables (on average), whereas ridge regression is still using all of them

5 Cross-validation and the one standard error rule

- We've already discussed cross-validation as a simple, effective way to estimate the expected test error of a method. In our context, we can use it to choose appropriate tuning parameter values for ridge regression and the lasso. To recap, we would divide the samples (x_i, y_i), $i = 1, \dots n$ into K folds, and compute

  $$\mathrm{CVErr}(\hat{\beta}_\lambda^{\mathrm{ridge}}) = \frac{1}{K} \sum_{k=1}^{K} \mathrm{CV}_k\Big((\hat{\beta}_\lambda^{\mathrm{ridge}})^{-(k)}\Big),$$

  over a grid of λ values. Here $(\hat{\beta}_\lambda^{\mathrm{ridge}})^{-(k)}$ denote the ridge regression coefficients, using a tuning parameter value λ, and fit on all samples except those in the kth fold. Also, $\mathrm{CV}_k((\hat{\beta}_\lambda^{\mathrm{ridge}})^{-(k)})$ denotes the test error incurred when we use these coefficients to predict the outcomes of points in the kth fold

  We would do the same for the lasso, i.e., compute

  $$\mathrm{CVErr}(\hat{\beta}_\lambda^{\mathrm{lasso}}) = \frac{1}{K} \sum_{k=1}^{K} \mathrm{CV}_k\Big((\hat{\beta}_\lambda^{\mathrm{lasso}})^{-(k)}\Big),$$

  over its own grid of λ values

- The usual error rule would then choose the tuning parameter, for each method, to minimize the CV error:

  $$\hat{\lambda}^{\mathrm{ridge}} = \underset{\lambda \in \{\lambda_1, \dots \lambda_m\}}{\mathrm{argmin}} \; \mathrm{CVErr}(\hat{\beta}_\lambda^{\mathrm{ridge}}),$$

  and

  $$\hat{\lambda}^{\mathrm{lasso}} = \underset{\lambda \in \{\lambda_1, \dots \lambda_m\}}{\mathrm{argmin}} \; \mathrm{CVErr}(\hat{\beta}_\lambda^{\mathrm{lasso}})$$

- This rule often selects a tuning parameter value that yields good predictive performance (by construction). However, the rule often ends up selecting models that are larger than desirable for interpretation purposes, say, when studying the lasso

- We can achieve smaller, simpler models with comparable predictive performance by using a simple device called the *one standard error rule*. First, recall that we can compute cross-validation standard errors as

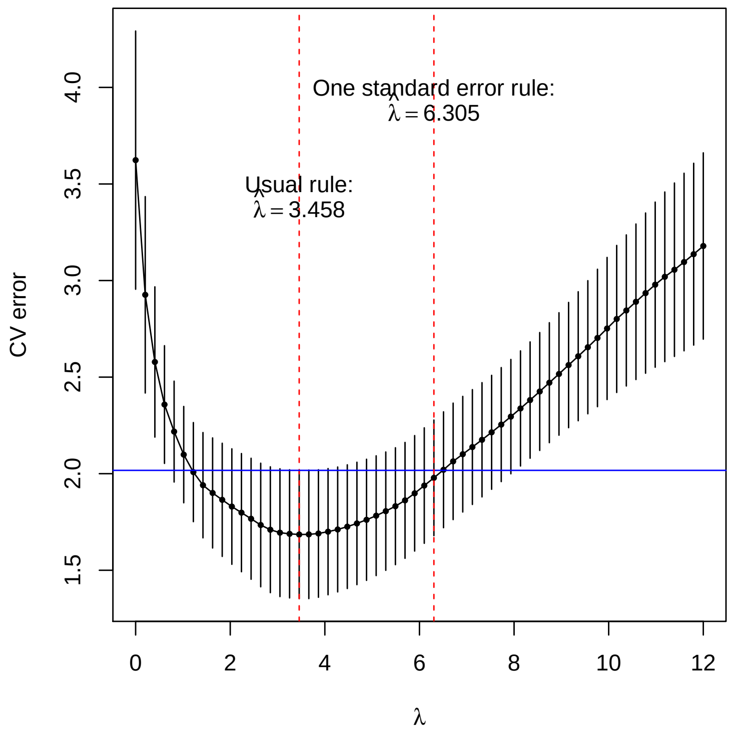
$$\mathrm{SE}\Big(\mathrm{CVErr}(\hat\beta_\lambda^{\mathrm{lasso}})\Big) = \frac{1}{\sqrt{K}} \cdot \mathrm{sd}\Big\{\mathrm{CV}_1\Big((\hat\beta_\lambda^{\mathrm{lasso}})^{-(1)}\Big), \ldots \mathrm{CV}_K\Big((\hat\beta_\lambda^{\mathrm{lasso}})^{-(K)}\Big)\Big\},$$

  where $\mathrm{sd}(\cdot)$ denotes the sample variance operator. Now, instead of choosing the tuning parameter value that minimizes the CV error curve, the one standard error rule chooses the tuning parameter value corresponding to the simplest model whose CV error is within one standard error of the minimum. I.e., if λ_0 denotes the minimizer of the CV error curve, then the one standard error rule selects the value

$$\hat\lambda^{\mathrm{lasso}} = \max\Big\{\lambda \in \{\lambda_1, \ldots \lambda_m\} \ : \ \mathrm{CVErr}(\hat\beta_\lambda^{\mathrm{lasso}}) \le \mathrm{CVErr}(\hat\beta_{\lambda_0}^{\mathrm{lasso}}) + \mathrm{SE}\Big(\mathrm{CVErr}(\hat\beta_{\lambda_0}^{\mathrm{lasso}})\Big)\Big\}.$$

  In words: starting at the minimizer, we move in the direction of model simplicity (smaller degrees of freedom) until the CV error exceeds one standard error from its minimum value

- It helps to see a picture. Here is the one standard error rule in action, on our running example for the lasso ($n = 50$, $p = 30$, 10 truly large coefficients, 20 zero coefficients):



In this example, the one standard error rule chooses a model with 15 variables, whereas the usual rule chooses a model with 20

6 Regularization in general

- In what we just learned, the penalty terms—squared ℓ_2 for ridge regression, and ℓ_1 for the lasso—are called forms of *regularization.* They make the estimate more regular, by shrinking it toward zero and hence decreasing its variance

- This type of regularization is often employed well beyond the linear model setting. Virtually anytime we have an estimate defined by optimizing a certain criterion, we can regularize this estimate in a high-dimensional scenario by adding a penalty term. Like what you saw here, adding an squared ℓ_2 penalty term will shrink components of the estimate but not set them to zero; adding an ℓ_1 penalty term will both shrink and set to zero exactly. Such practice, of regularizing parameter estimates, is often done, e.g., in high-dimensional maximum likelihood settings

- Finally, regularization is not limited to squared ℓ_2 and ℓ_1 penalties, nor is it philosophically limited to the idea of shrinking the components of parameter estimates toward zero. Across various problems, various types of regularization may be appropriate and can be employed. Recall, e.g., the smoothing spline problem in nonparametric regression, where we used a regularization term to encourage the fitted function to be smooth in its second derivative