ISLR | Chapter 6 Exercises

Marshall McQuillen

7/28/2018

Conceptual

1

- **A**. For a model with k predictors, Best Subset Selection will always have the best *training* RSS. The reason for this is, given a fixed k, there are $\binom{p}{k}$ possible models, and Best Subset Selection considers all of those $\binom{p}{k}$ possibilities.

  In Forward Stepwise Selection, of the total $\binom{p}{k}$ possible models, only the models that contain the $(k-1)$ model produced by Forward Stepwise Selection will be considered for the "best" k-variable model.

  In Backward Stepwise Selection, of the total $\binom{p}{k}$ possible models, the predictors in the "best" k-variable model *must* be a subset of the model with $(k+1)$ predictors.

  In short, Best Subset Selection will have the best (lowest) *training* RSS for a model with k predictors because it considers **all** the possible $\binom{p}{k}$ models, whereas Forward and Backward Stepwise Selection only consider a **subset** of all the possible $\binom{p}{k}$ models.

- **B**. There is no definitive answer for which subset selection method will have the lowest *testing* RSS (overfitting). If there is a large number of predictors, Best Subset Selection has the possibility of finding a model that has a low training RSS but a high testing RSS. Cross validation could be used to estimate the testing error of three models (one for Best Subset Selection, one for Forward Stepwise Selection and one for Backward Stepwise Selection) and a decision on which model has the lowest testing RSS could be made in consideration of the CV results.

- **C**.

  i. True.

  ii. True.

  iii. False, the predictors in the k-variable model identified by Backward Subset Selection are **not** a subset of the predictors in the $(k+1)$-variable model identified by Forward Subset Selection.

  iv. False, the predictors in the k-variable model identified by Forward Stepwise Selection are **not** a subset of the predictors in the $(k+1)$-variable model identified by Backward Stepwise Selection.

  v. False, the predictors in the k-variable model identified by Best Subset Selection are **not necessarily** a subset of the predictors in the $(k+1)$-variable model identified by Best Subset Selection.

2

- **A**. The lasso, relative to least squares is, *iii*, less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance.

- **B**. Ridge Regression, relative to least squares is, *iii*, less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance.

- **C**. Non-linear methods, relative to least squares are, *ii*, more flexible and hence will give improved prediction accuracy when their increase in variance is less than their decrease in bias.

3

In the (alternate) cost function for the Lasso…

$$\sum_{i=1}^{n}\left(y_k - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 \; subject\ to \sum_{j=1}^{p}|\beta_j| \leq s$$

As we increase s from 0…

- **A**. …the training RSS will, *iv*, steadily decrease. As the budget (s) for the sum of the regression coefficients increases from 0, each β_j will approach the value it would reach in ordinary least squares regression (no constraint). Therefore, without a constraint, we are letting the regression coefficients "roam freely" to reach their ordinary least squares values. Using the constraint, we are "lassoing" them in (pun intended).
- **B**. …the testing RSS will, *ii*, decrease initially, and then eventually start increasing in a U shape. When s is 0, all the regression coefficients are 0, and the "model" is simply the intercept, β_0, the mean of the response (highly biased, very low variance). As s increases from 0, the model becomes more flexible, allowing for an increasingly better fit to the data up to a point (bottom of the U). Once this point is reached, the model becomes **overly** flexible, overfitting the training data and leading to an increase in the test error (right side of U).
- **C**. …variance will, *iii*, steadily increase. As s increases from 0, the model becomes more and more flexible, and as we know, more flexible models have a higher variance and lower bias. The model will be more influenced by the data it is trained on.
- **D**. …(squared) bias will, *iv*, steadily decrease. As s increases from 0, the model becomes more and more flexible, and as we know, more flexible models have a higher variance and lower bias. The model will have a better chance of representing the *true* relationship between the predictors and the response.
- **E**. …the irreducible error will, *v*, remain constant. The irreducible error is just that, **irreducible**.

4

In the cost function for Ridge Regression…

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

…as we increase λ from 0…

- **A**. …the training RSS will, *iii*, steadily increase. As λ increases from 0, more "weight" is given to the second term in the cost function, thus penalizing large regression coefficients more and more. Increasing λ from 0 restricts the regression coefficients more and more.
- **B**. …the testing RSS will, *ii*, decrease initially, and then eventually start increasing in a U shape. As λ increases from 0, increasingly strict restrictions are put on the magnitude that the regression coefficients can grow too. This has the effect of making the model less flexible and more generalizable, **up to a point**. The left side of the U represents the model's increase in bias being less than it's decrease in variance, and the right side of the U represents the decrease in variance no longer being worth the increase in bias.
- **C**. …the variance will, *iv*, steadily decrease. As λ increases from 0, increasingly strict limits are placed on the regression coefficients, making it less flexible.

- **D**. …the (squared) bias will, *iii*, steadily increase. As λ increases from 0, increasingly strict limits are placed on the regression coefficients, making it less flexible. This will make it increasingly harder for the model to estimate the true relationship between the response and the predictors.
- **E**. …the irreducible error will, v, remain constant. The irreducible error is just that, **irreducible**.

5

- **A**. The optimization problem for Ridge Regression is:

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

  Written out when $n = p = 0$, the equation comes to:

$$(y_1 - \beta_0 - \beta_1 x_{11} - \beta_2 x_{12})^2 + (y_2 - \beta_0 - \beta_1 x_{21} - \beta_2 x_{22})^2 + \lambda\left(\beta_1^2 + \beta_2^2\right)$$

  When $\beta_0 = 0$, a small simplification can be made such that the above equation becomes:

$$(y_1 - \beta_1 x_{11} - \beta_2 x_{12})^2 + (y_2 - \beta_1 x_{21} - \beta_2 x_{22})^2 + \lambda\left(\beta_1^2 + \beta_2^2\right)$$

- **B**. Given that $x_{11} = x_{12}$ and $x_{21} = x_{22}$, I will refer to these as simply x_1 and x_2 respectively where applicable. A small rewrite of the preceding equation gives:

$$f(x) = (y_1 - \beta_1 x_1 - \beta_2 x_1)^2 + (y_2 - \beta_1 x_2 - \beta_2 x_2)^2 + \lambda\left(\beta_1^2 + \beta_2^2\right)$$

  Given that the problem above is one of *minimization*, taking the derivative is the first step in showing that $\beta_1 = \beta_2$.

  Calculating the derivative with respect to β_1 can be broken down into the derivative of the three separate terms in the above equation:

$$\frac{\partial}{\partial\beta_1}f(x) = \frac{\partial}{\partial\beta_1}(y_1 - \beta_1 x_1 - \beta_2 x_1)^2 + \frac{\partial}{\partial\beta_1}(y_2 - \beta_1 x_2 - \beta_2 x_2)^2 + \frac{\partial}{\partial\beta_1}\left(\lambda\beta_1^2 + \lambda\beta_2^2\right)$$

  First Term Derivative

$$\frac{\partial}{\partial\beta_1}(y_1 - \beta_1 x_1 - \beta_2 x_1)^2 = 2(y_1 - \beta_1 x_1 - \beta_2 x_1)\cdot(-x_1) = 2\left(-y_1 x_1 + \beta_1 x_1^2 + \beta_2 x_1^2\right)$$

  Second Term Derivative

$$\frac{\partial}{\partial\beta_1}(y_2 - \beta_1 x_2 - \beta_2 x_2)^2 = 2(y_2 - \beta_1 x_2 - \beta_2 x_2)\cdot(-x_2) = 2\left(-y_2 x_2 + \beta_1 x_2^2 + \beta_2 x_2^2\right)$$

  Third Term Derivative

$$\frac{\partial}{\partial\beta_1}\left(\lambda\beta_1^2 + \lambda\beta_2^2\right) = 2\lambda\beta_1$$

Bringing this all together, the derivative of the full equation comes out to:

$$\frac{\partial}{\partial \beta_1} f(x) = 2\left(-y_1 x_1 + \beta_1 x_1^2 + \beta_2 x_1^2\right) + 2\left(-y_2 x_2 + \beta_1 x_2^2 + \beta_2 x_2^2\right) + 2\lambda\beta_1$$

Setting the derivative equal to 0 and solving for β_1:

$$2\left(-y_1 x_1 + \beta_1 x_1^2 + \beta_2 x_1^2\right) + 2\left(-y_2 x_2 + \beta_1 x_2^2 + \beta_2 x_2^2\right) + 2\lambda\beta_1 = 0$$

Divide by 2:

$$\left(-y_1 x_1 + \beta_1 x_1^2 + \beta_2 x_1^2\right) + \left(-y_2 x_2 + \beta_1 x_2^2 + \beta_2 x_2^2\right) + \lambda\beta_1 = 0$$

Group β_1 terms and β_2 terms together:

$$\left(\beta_1 x_1^2 + \beta_1 x_2^2 + \lambda\beta_1\right) + \beta_2 x_1^2 + \beta_2 x_2^2 - y_1 x_1 - y_2 x_2 = 0$$

Factor out β_1 and β_2:

$$\beta_1\left(x_1^2 + x_2^2 + \lambda\right) + \beta_2\left(x_1^2 + x_2^2\right) - y_1 x_1 - y_2 x_2 = 0$$

Add $y_1 x_1$ and $y_2 x_2$ to both sides of the equation:

$$\beta_1\left(x_1^2 + x_2^2 + \lambda\right) + \beta_2\left(x_1^2 + x_2^2\right) = y_1 x_1 + y_2 x_2$$

Repeating the same process, this time taking the derivative **with respect to** β_2, would yield:

$$\beta_2\left(x_1^2 + x_2^2 + \lambda\right) + \beta_1\left(x_1^2 + x_2^2\right) = y_1 x_1 + y_2 x_2$$

Using substitution, we can set the two equations equal to each other:

$$\beta_1\left(x_1^2 + x_2^2 + \lambda\right) + \beta_2\left(x_1^2 + x_2^2\right) = \beta_2\left(x_1^2 + x_2^2 + \lambda\right) + \beta_1\left(x_1^2 + x_2^2\right)$$

Factoring λ out into it's own term:

$$\beta_1\left(x_1^2 + x_2^2\right) + \beta_1\lambda + \beta_2\left(x_1^2 + x_2^2\right) = \beta_2\left(x_1^2 + x_2^2\right) + \beta_2\lambda + \beta_1\left(x_1^2 + x_2^2\right)$$

Subract $\beta_1\left(x_1^2 + x_2^2\right)$ and $\beta_2\left(x_1^2 + x_2^2\right)$ then divide by λ:

$$\beta_1\lambda = \beta_2\lambda \quad thus \quad \beta_1 = \beta_2$$

- **C**. The lasso optimization problem is similar to that of Ridge Regression, with a small change to the third term in the equation:

$$f(x) = \left(y_1 - \beta_1 x_1 - \beta_2 x_1\right)^2 + \left(y_2 - \beta_1 x_2 - \beta_2 x_2\right)^2 + \lambda\left(|\beta_1| + |\beta_2|\right)$$

- **D**. Taking a visual approach to an explanation as to why there are many β_1's and β_2's that solve the Lasso starts with the image of the error contour plots and the shaded constraint regions from Chapter 6, reproduced below.

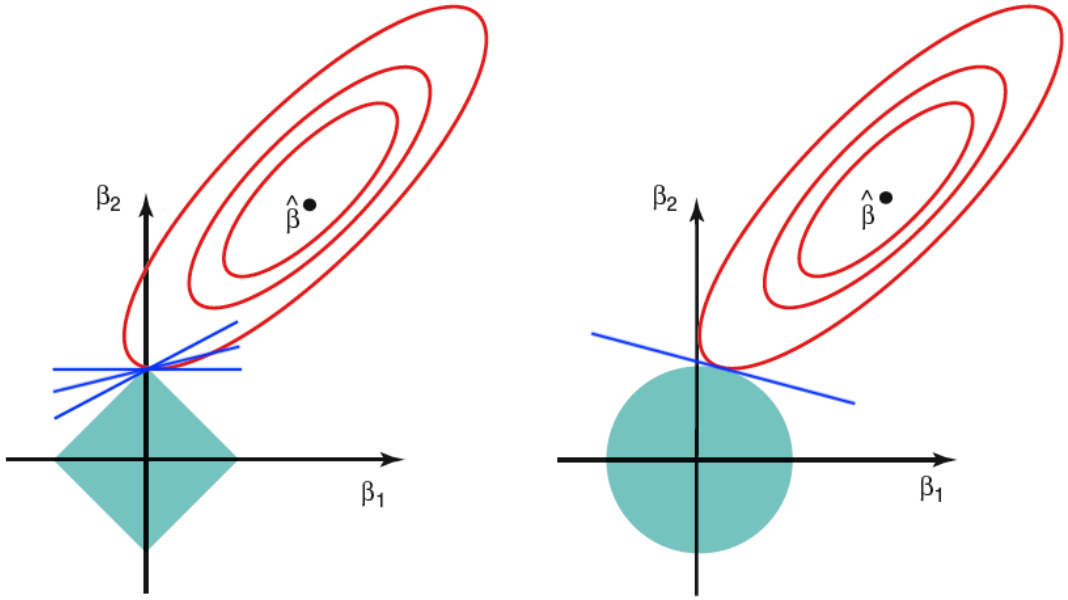


FIGURE 6.7. *Contours of the error and constraint functions for the lasso* (left) *and ridge regression* (right). *The solid blue areas are the constraint regions,* $|\beta_1| + |\beta_2| \leq s$ *and* $\beta_1^2 + \beta_2^2 \leq s$, *while the red ellipses are the contours of the RSS.*

Figure 1:

6

- **A**. Considering the Ridge Regression loss function where $p = 1$, the equation...

$$\sum_{j=1}^{p} (y_j - \beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

...can be re-written as the following.

$$f(\beta_j) = (y_j - \beta_j)^2 + \lambda \beta_j^2$$

A quick derivation shows that equation 6.12 is solved by 6.14:

$$\frac{\partial f(\beta_j)}{\partial \beta_j} = 2\,(y_j - \beta_j)\,(-1) + 2\lambda\beta_j$$

$$2\,(y_j - \beta_j)\,(-1) + 2\lambda\beta_j = 0$$

$$-2\,(y_j - \beta_j) + 2\lambda\beta_j = 0$$

$$-2y_j + 2\beta_j + 2\lambda\beta_j = 0$$

$$-y_j + \beta_j + \lambda\beta_j = 0$$

$$-y_j + \beta_j(1 + \lambda) = 0$$

$$\beta_j(1 + \lambda) = y_j$$

$$\beta_j = \frac{y_j}{(1 + \lambda)}$$

The following code holds $y_j = \lambda = 1$ and plots a sequence of β_j estimates in blue. The minimum of those estimates is circled in red, and the minimum estimated by the equation above is circled in green, both encompassing 0.5

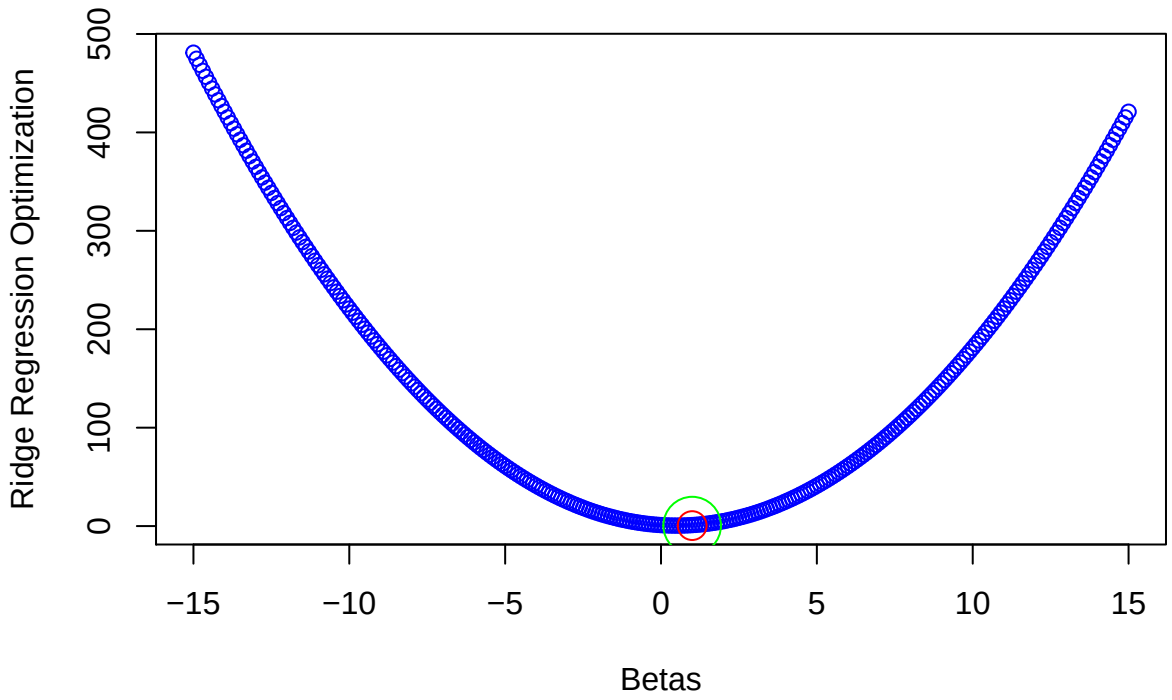
```
ridge <- function(y, lambda, beta) {
    return((y - beta)^2 + lambda*beta^2)
}

y = 1
lambda = 1
betas <- seq(-15, 15, 0.1)

f.x <- ridge(y, lambda, betas)

function_estimate <- f.x[which.min(f.x)]
equation_estimate <- y/(lambda + 1)

plot(x = betas,
     y = f.x,
     col = "blue",
     xlab = 'Betas',
     ylab = "Ridge Regression Optimization")
points(function_estimate, col = 'red', cex = 2)
points(equation_estimate, col = 'green', cex = 4)
```




```
function_estimate == equation_estimate
```

```
## [1] TRUE
```