Linear Model Selection and Regularization II: Ridge Regression and the Lasso

Rebecca C. Steorts, Duke University

STA 325, Chapter 6 ISL

Agenda

Shrinking Regression Coefficients

- Ridge Regression
- Lasso Regression
- Application to the Hitters data set

Shrinkage Methods

The previous module covered subset selection methods involving 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 turns out that shrinking the coefficient estimates can significantly reduce their variance.

The two best-known techniques for shrinking the regression coefficients towards zero are ridge regression and the lasso.

Shrinkage Applied to Credit Data

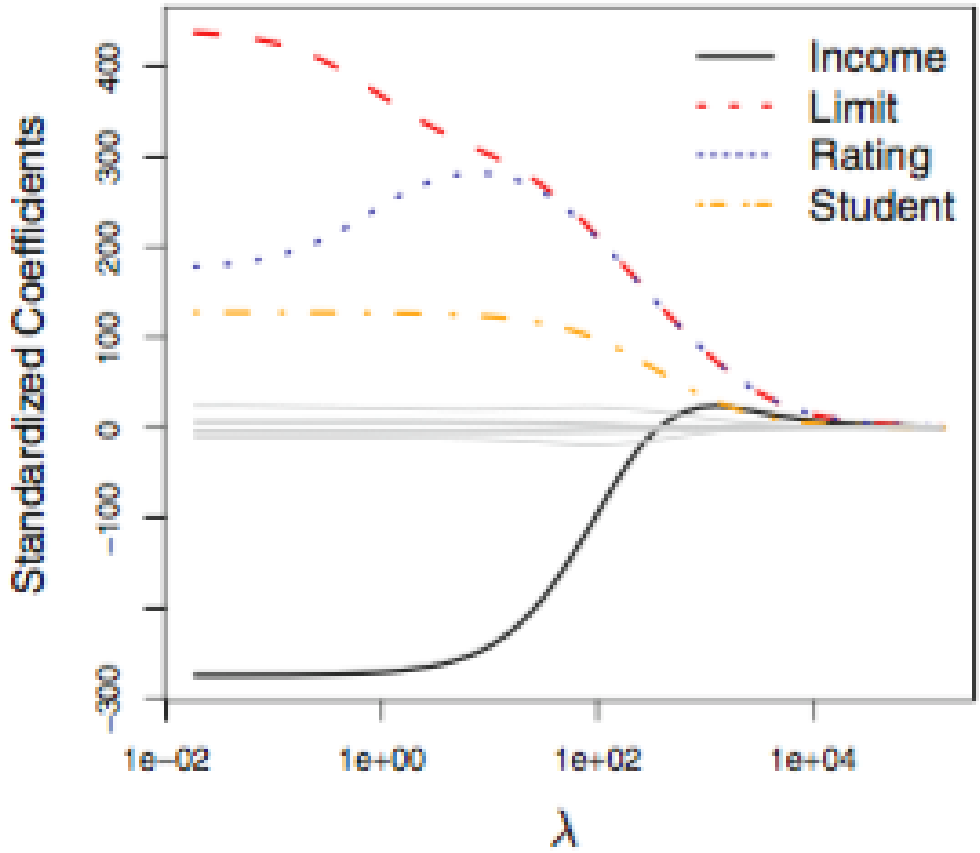


Figure 1: Each curve corresponds to the ridge regression coefficient estimate for one of the ten variables, plotted as a function of λ. For example, the black solid line represents the ridge regression estimate for the income coefficient, as λ is varied.

Shrinkage Applied to Credit Data

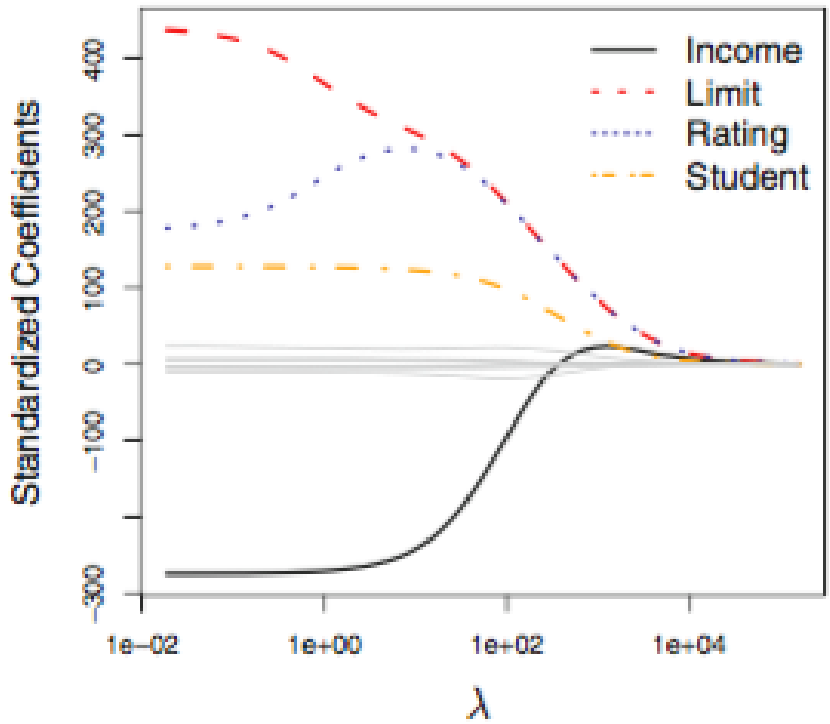


Figure 2: At the left-hand side of the plot, λ is essentially zero, and the corresponding ridge coefficient estimates are the same as the least squares estimates. As λ increases, the ridge coefficient estimates shrink towards zero. When λ is extremely large, all of the ridge coefficient estimates are basically zero (the null model that contains no predictors). Note that the income, limit, rating, and student variables are displayed in distinct colors, since these variables tend to have by far the largest coefficient estimates.

Ridge applied to Credit data

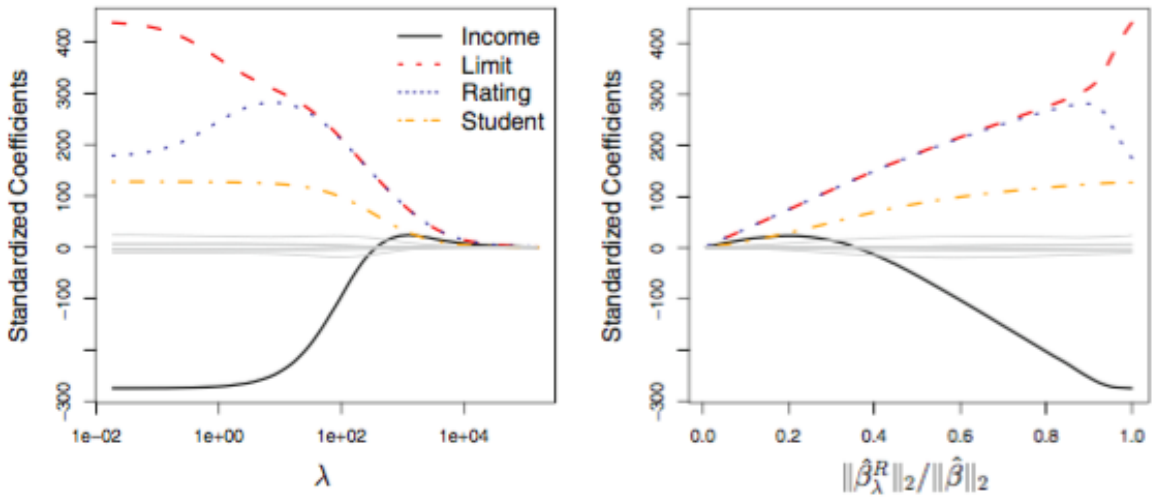


Figure 3: The right-hand panel displays the same ridge coefficient estimates as the left-hand panel, but instead of displaying λ on the x-axis, we display $||\hat{\beta}^R_\lambda||_2/||\hat{\beta}||_2$, where $\hat{\beta}$ denotes the vector of least squares coefficient estimates.

- $||\hat{\beta}||_2 = \sqrt{\sum_j \beta_j^2}$ and is the ℓ_2 norm of a vector.
- Here, it measures the distance of β from zero.

Ridge applied to Credit data

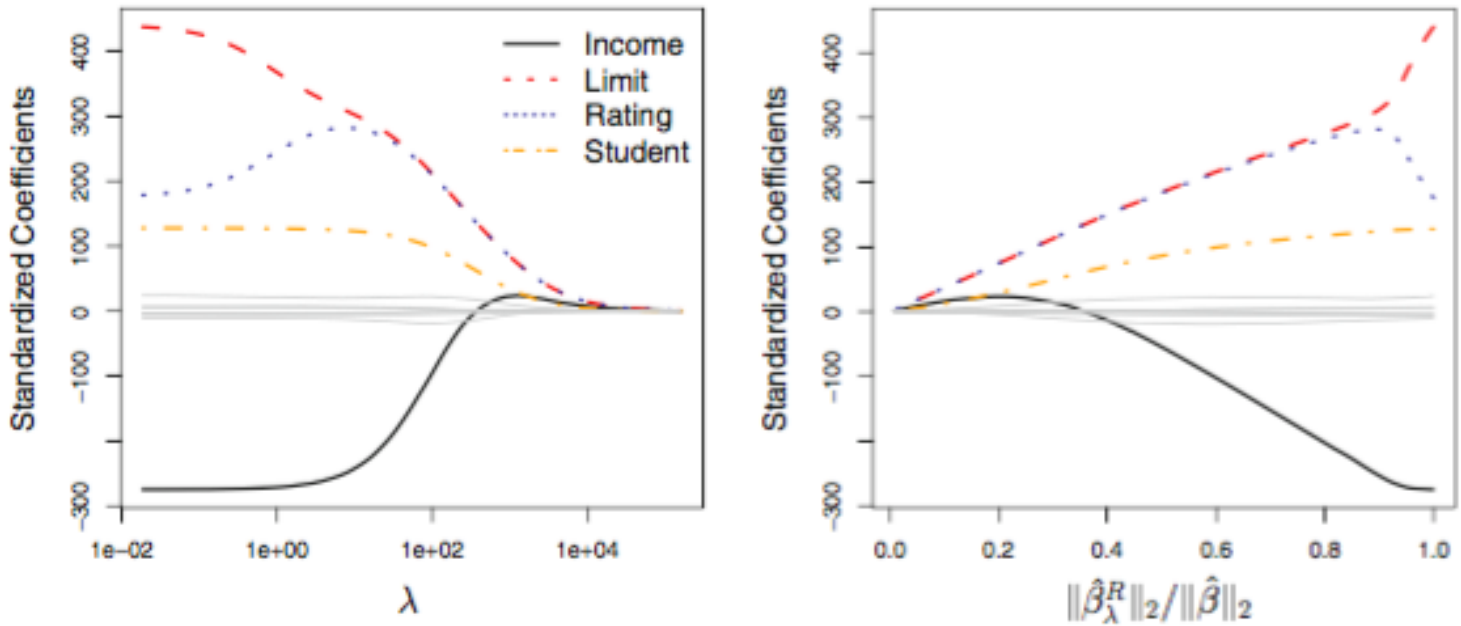


Figure 4: We can think of the x-axis in the right-hand panel as the amount that the ridge regression coefficient estimates have been shrunken towards zero; a small value indicates that they have been shrunken very close to zero.

Ridge Regression

Recall that in Chapter 3 that the least squares fitting procedure estimates $\beta_0, \ldots, \beta_p$ using the values that minimize

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

Ridge regression is very similar to **least squares**, except that the coefficients are estimated by minimizing a slightly different quantity.

Ridge Regression

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

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

where λ is a **tuning parameter** to be determined separately.

Ridge Regression

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

Equation 2 trades two different criteria.

1. As with least squares, ridge regression seeks coefficient estimates that fit the data well, by making the RSS small.
2. However, the second term, $\lambda\sum_{j=1}^{p}\beta_j^2$ called a shrinkage penalty, is small when $\beta_1,\ldots,\beta_p$ are close to zero.

- This shrinks 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.

Ridge Regression

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

- When $\lambda=0$, the penalty term has no effect, and ridge regression will produce the least squares estimates.
- However, as $\lambda\to\infty$, the impact of the shrinkage penalty grows, and the ridge regression coefficient estimates will approach zero.
- Unlike least squares, which generates only one set of coefficient estimates, ridge regression will produce a different set of coefficient estimates, $\hat{\beta}_\lambda^R$, for each value of λ.
- Selecting a good value for λ is critical (to be discussed).

Shrinkage penalty

- In equation 2, the shrinkage penalty is applied to $\beta_1, \ldots, \beta_p$, but not to the intercept β_0.
- We want to shrink the estimated association of each variable with the response.
- However, we **do not want to shrink the intercept**, which is simply a measure of the mean value of the response when $x_{i1} = x_{i2} = \ldots = x_{ip} = 0$.
- If we assume that the variables – that is, the columns of the data matrix X—have been centered to have mean zero before ridge regression is performed then the estimated intercept will take the form

$$\hat{\beta}_0 = \bar{y} = \sum_{i=1}^{n} y_i / n.$$

Why Does Ridge Regression Improve Over Least Squares?

- Ridge regression's advantage over least squares is rooted in the bias-variance trade-off.
- As λ increases, the flexibility of the ridge regression fit decreases, leading to **decreased variance** but **increased bias**.

Why Does Ridge Regression Improve Over Least Squares?

- In situations where the relationship between the response and the predictors is **close to linear**, the least squares estimates will have **low bias** but **may have high variance**.
- This means that a small change in the training data can cause a large change in the least squares coefficient estimates.
- In particular, when the number of variables p is almost as large as the number of observations n, as in the example, the least squares estimates will be extremely variable.
- And if $p > n$, then the least squares estimates do not even have a unique solution, whereas ridge regression can still perform well by trading off a small increase in bias for a large decrease in variance.

Ridge regression works best in situations where the least squares estimates have **high variance**.

Computational Advantages of Ridge Regression

Ridge regression also has substantial computational advantages over best subset selection, which requires searching through 2^p models.

Even for moderate values of p, such a search can be computationally infeasible.

In contrast, for any fixed value of λ, ridge regression only fits a single model, and the model-fitting procedure can be performed quite quickly.

One can show that the computations required to solve equation 2, simultaneously for all values of λ, are almost identical to those for fitting a model using least squares.

The Lasso

- Ridge regression has a major disadvantage: it puts all p predictors in the final model.
- This may not be a problem for prediction accuracy, but it can create a **challenge in model interpretation** in settings in which the number of variables p is quite large.
- The **lasso** is a relatively recent alternative to ridge regression that overcomes this disadvantage.

The Lasso

The lasso coefficients $\hat{\beta}_\lambda^L$ minimize the quantity

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS}\lambda\sum_{j=1}^{p}|\beta_j| \qquad (3)$$

Comparing equation 2 to equation 3, we see that the ridge and lasso regression have similar formulations.

The Lasso

The only difference is that the β_j^2 penalty in the ridge regression penality (2) has been replaced by $|\beta_j|$ in the lasso penalty (3).

In statistics, the lasso uses an ℓ_1 penalty instead of an ℓ_2 penalty.

The ℓ_1 norm of a coefficient vector β is given by $||\beta||_1 = \sum_j |\beta_j|$

The Lasso for Variable Selection

- 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.
- Much like best subset selection, the lasso performs variable selection.

The Lasso for Variable Selection

- As a result, models generated from the lasso are generally much easier to interpret than those produced by ridge regression.
- 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 and CV is recommended.

Another Formulation for Ridge Regression and the Lasso

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

$$min_{\beta}\{\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij})^2\} \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s \qquad (4)$$

and

$$min_{\beta}\{\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij})^2\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s \qquad (5)$$

For every value of λ, there is some s such that equations 3 and 4 will give the same lasso coefficient estimates.

For every value of λ, there is some s such that equations 2 and 5 will give the same ridge coefficient estimates.

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?

Figure 5 illustrates the situation.

The Variable Selection Property of the Lasso

The least squares solution is marked as $\hat{\beta}$, while the blue diamond and circle represent the lasso and ridge regression constraints in 5 and **??**.

The Variable Selection Property of the Lasso

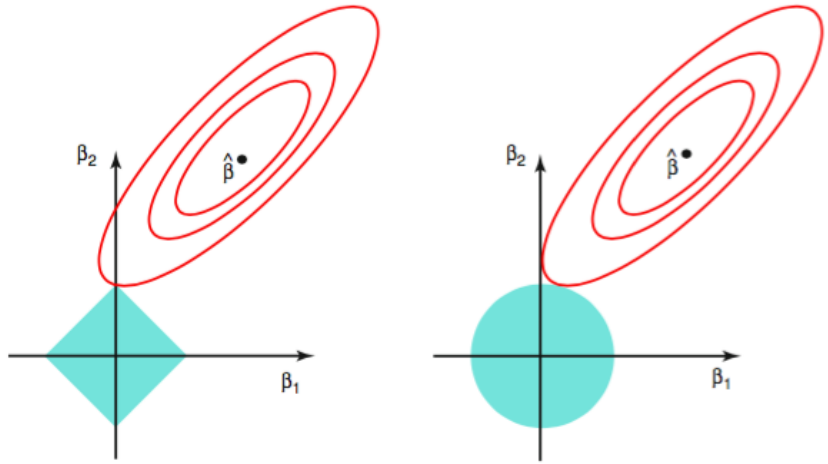


Figure 5: Contours of the error and constraint functions for the lasso (left) and ridge regression (right). The solid teal areas are the constraints regions, while the red ellipses are the countours of the RSS.

The ellipses that are centered around $\hat{\beta}$ represent regions of constant RSS.

In other words, all of the points on a given ellipse share a common value of the RSS.

As the ellipses expand away from the least squares coefficient estimates, the RSS increases.

Comparing the Lasso and Ridge Regression

It is clear that the lasso has a major advantage over ridge regression, in that it produces simpler and more interpretable models that involve only a subset of the predictors.

However, which method leads to better prediction accuracy?

In general, neither ridge regression nor the lasso will universally dominate the other.

Comparing the Lasso and Ridge Regression

1. One might expect the lasso to perform better in a setting where a relatively small number of predictors have substantial coefficients, and the remaining predictors have coefficients that are very small or that equal zero.
2. Ridge regression will perform better when the response is a function of many predictors, all with coefficients of roughly equal size.

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

Implementing ridge regression and the lasso requires a method for selecting a value for the tuning parameter λ.

Cross-validation provides a simple way to tackle this problem.

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.

Ridge and Lasso Applied to Hitters

We will use the glmnet package in order to perform ridge regression and the lasso.

The main function in this package is glmnet(), which can be used to fit ridge regression models, lasso models, and more. T

We must pass in an x **matrix** as well as a y **vector**, and we **do not use** the y x syntax.

Make sure to remove missing observations!

Ridge and Lasso Applied to Hitters

```
library(ISLR)
# remove missing values
Hitters=na.omit(Hitters)
# x must be a _matrix
x=model.matrix(Salary~.,Hitters)[,-1]
# y must be a _vector
y=Hitters$Salary
```

The model.matrix() function is particularly useful for creating x.

It produces a matrix corresponding to the 19 predictors and automatically transforms any qualitative variables into dummy variables.

Ridge Applied to Hitters

The glmnet() function has an alpha argument that determines what type of model is fit.

If alpha=0 then a ridge regression model is fit.

If alpha=1 then a lasso model is fit.

We first fit a ridge regression model.

Ridge Applied to Hitters

- By default the glmnet() function performs ridge regression for an automatically selected range of λ values.
- However, here we have chosen to implement the function over a grid of values ranging from $\lambda = 1010\lambda$ to $\lambda = 10^{-2}$, essentially covering the full range of scenarios from the null model containing only the intercept, to the least squares fit.
- By default, the glmnet() function standardizes the variables so that they are on the same scale.

Ridge Applied to Hitters

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-10
```

```
grid=10^seq(10,-2,length=100)
ridge.mod=glmnet(x,y,alpha=0,lambda=grid)
```

Ridge Applied to Hitters

Associated with each value of λ is a vector of ridge regression coefficients, stored in a matrix that can be accessed by coef()

```
dim(coef(ridge.mod))
```

```
## [1]  20 100
```

This is a 20×100 matrix, with 20 rows (one for each predictor, plus an intercept) and 100 columns (one for each value of λ).

Ridge Applied to Hitters

We expect the coefficient estimates to be much smaller, in terms of ℓ_2 norm, when a large value of λ is used, as compared to when a small value of λ is used.

These are the coefficients when $\lambda = 11,498$, along with their ℓ_2 norm:

Ridge Applied to Hitters

```
ridge.mod$lambda [50]
```

```
## [1] 11497.57
```

```
coef(ridge.mod)[,50]
```

```
##   (Intercept)         AtBat          Hits         HmRun
## 407.356050200   0.036957182   0.138180344   0.524629976
##          RBI         Walks         Years        CAtBat
##   0.239841459   0.289618741   1.107702929   0.003131815
##       CHmRun         CRuns          CRBI        CWalks
##   0.087545670   0.023379882   0.024138320   0.025015421
##    DivisionW       PutOuts       Assists        Errors
##  -6.215440973   0.016482577   0.002612988  -0.020502690
```

```
sqrt(sum(coef(ridge.mod)[-1,50]^2))
```

```
## [1] 6.360612
```

Ridge Applied to Hitters

In contrast, here are the coefficients when $\lambda = 705$, along with their ℓ_2 norm.

Note the much larger ℓ_2 norm: of the coefficients associated with this smaller value of λ.

```
ridge.mod$lambda [60]
```

```
## [1] 705.4802
```

```
coef(ridge.mod)[,60]
```

```
##  (Intercept)        AtBat         Hits        HmRun
##  54.32519950   0.11211115   0.65622409   1.17980910   0.
##          RBI        Walks        Years       CAtBat
##   0.84718546   1.31987948   2.59640425   0.01083413   0.
##       CHmRun        CRuns         CRBI       CWalks
##   0.33777318   0.09355528   0.09780402   0.07189612  13.
##    DivisionW      PutOuts      Assists       Errors Ne
```

Ridge Prediction for Hitters

We can use the predict() function for a number of purposes.

For instance, we can obtain the ridge regression coefficients for a new value of λ, say 50:

```
predict(ridge.mod,s=50,type="coefficients")[1:5,]
```

```
## (Intercept)       AtBat        Hits       HmRun           R
##  48.7661033  -0.3580999   1.9693593  -1.2782480   1.1458
```

Ridge and Lasso for Hitters

We now split the samples into a training set and a test set in order to estimate the test error of ridge regression and the lasso.

We set up the training and test data as follows: Randomly choose a subset of numbers between 1 and n; these can then be used as the indices for the training observations.

```
set.seed (1)
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.test=y[test]
```

Ridge regression for Hitters

1. Fit a ridge regression model on the training set,
2. Evaluate its MSE on the test set, using $\lambda = 4$.
3. Use the predict() function again. This time we get predictions for a **test set**, by replacing type="coefficients" with the newx argument.

```
ridge.mod=glmnet(x[train,],y[train],alpha=0,
    lambda=grid, thresh =1e-12)
ridge.pred=predict(ridge.mod,s=4,newx=x[test,])
mean((ridge.pred-y.test)^2)
```

```
## [1] 101036.8
```

The test MSE is 101,037.

Ridge regression for Hitters

If we had instead simply fit a model with just an intercept, we would have predicted each test observation using the mean of the training observations. In that case, we could compute the test set MSE like this:

```
mean((mean(y[train])-y.test)^2)[1]
```

```
## [1] 193253.1
```

We could also get the same result by fitting a ridge regression model with a very large value of λ.

```
ridge.pred=predict(ridge.mod,s=1e10,newx=x[test,])
mean((ridge.pred-y.test)^2)
```

```
## [1] 193253.1
```

Ridge regression for Hitters

So fitting a ridge regression model with $\lambda = 4$ leads to a much lower test MSE than fitting a model with just an intercept.

We now check whether there is any benefit to performing ridge regression with $\lambda = 4$ instead of just performing least squares regression.

Recall that least squares is simply ridge regression with $\lambda = 0$

```
ridge.pred=predict(ridge.mod,s=0,newx=x[test,],exact=T)
mean((ridge.pred-y.test)^2)
```

```
## [1] 114783.1
```

```
lm(y~x, subset=train)
```

```
##
## Call:
## lm(formula = y ~ x, subset = train)
##
```

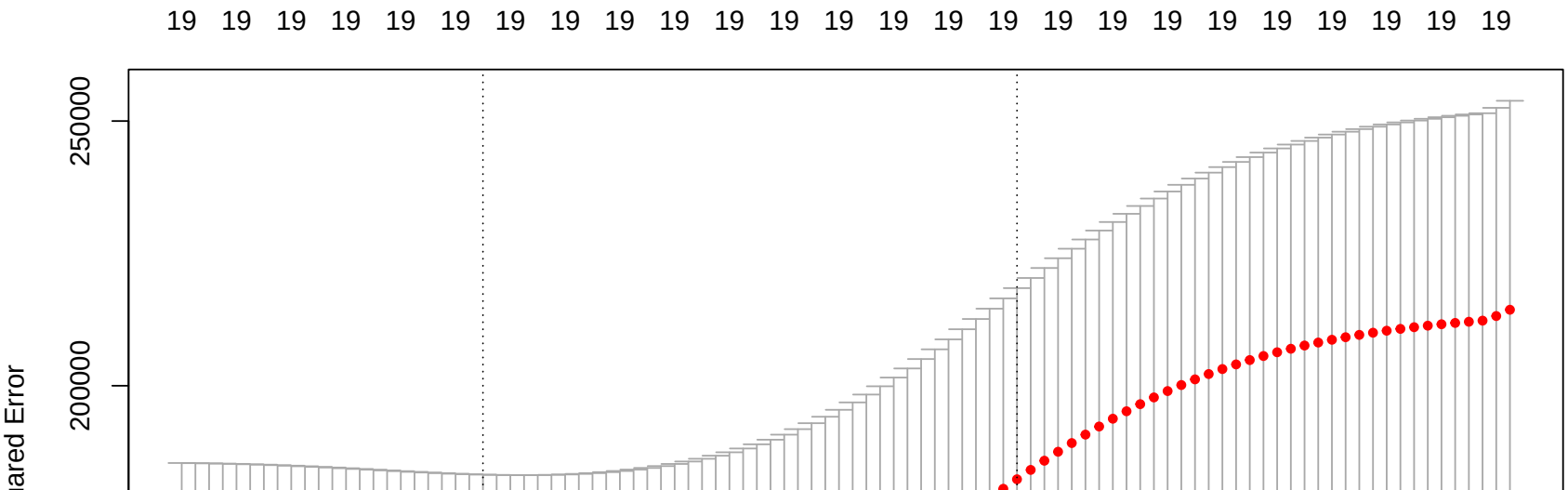
Ridge regression for Hitters

In general, instead of arbitrarily choosing $\lambda = 4$, it would be better to use cross-validation to choose the tuning parameter λ.

We can do this using the built-in cross-validation function, cv.glmnet().

By default, the function performs ten-fold cross-validation, though this can be changed using the argument folds.

```
set.seed (1)
cv.out=cv.glmnet(x[train,],y[train],alpha=0)
plot(cv.out)
```



Ridge regression for Hitters

Therefore, we see that the value of λ that results in the smallest CV error is 212.

What is the test MSE associated with this value of λ?

```
ridge.pred=predict(ridge.mod,s=bestlam ,newx=x[test,])
mean((ridge.pred-y.test)^2)
```

```
## [1] 96015.51
```

This represents a further improvement over the test MSE that we got using $\lambda = 4$.

Finally, we refit our ridge regression model on the full data set, using the value of λ chosen by CV, and examine the coefficient estimates.

Final Ridge on Hitters Data, $\lambda = 212$

```
out=glmnet(x,y,alpha=0)
predict(out,type="coefficients",s=bestlam)[1:20,]
```

```
##  (Intercept)        AtBat         Hits        HmRun
##   9.88487157   0.03143991   1.00882875   0.13927624   1.
##          RBI        Walks        Years       CAtBat
##   0.87318990   1.80410229   0.13074381   0.01113978   0.
##       CHmRun        CRuns         CRBI       CWalks
##   0.45158546   0.12900049   0.13737712   0.02908572  27.
##    DivisionW      PutOuts      Assists       Errors   Ne
## -91.63411299   0.19149252   0.04254536  -1.81244470   7.
```

As expected, none of the coefficients are zero—ridge regression does not perform variable selection!

The Lasso on Hitters

We saw that ridge regression with a wise choice of λ can outperform least squares as well as the null model on the Hitters data set.

We now ask whether the lasso can yield either a more accurate or a more interpretable model than ridge regression.

In order to fit a lasso model, we once again use the glmnet() function; however, this time we use the argument $\alpha = 1$.

Other than that change, we proceed just as we did in fitting a ridge model.

The Lasso on Hitters

```
lasso.mod=glmnet(x[train, ],
                 y[train], alpha=1, lambda=grid)
pdf(file="figures/lasso.pdf")
plot(lasso.mod)
dev.off()
```

```
## pdf
##   2
```

The Lasso on Hitters

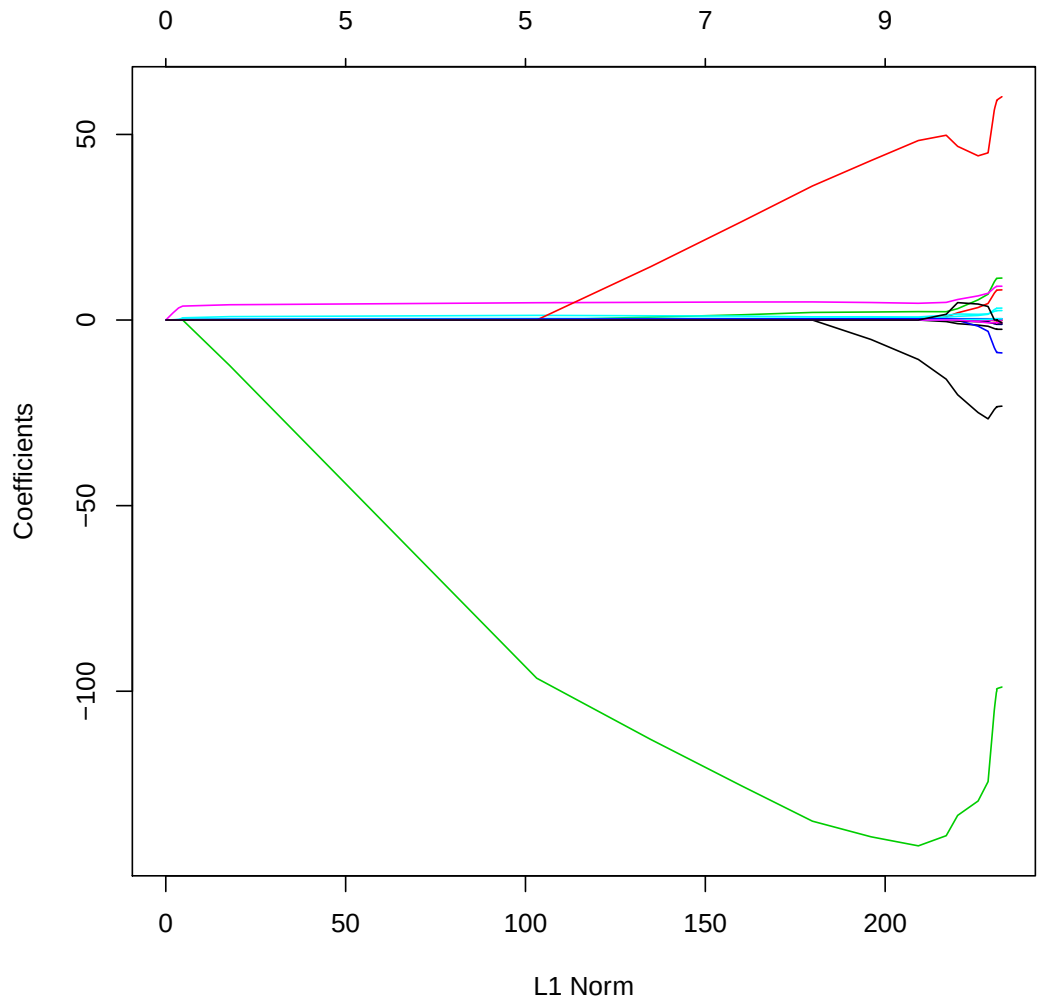


Figure 6: Depending on the choice of tuning parameter, some of the coefficients will be exactly equal to zero.

The Lasso on Hitters

We now perform cross-validation and compute the associated test error.

```
set.seed(1)
cv.out=cv.glmnet(x[train ,],y[train],alpha=1)
pdf(file="figures/lasso-cv.pdf")
plot(cv.out)
dev.off()
```

```
## pdf
##   2
```

```
(bestlam=cv.out$lambda.min)
```

```
## [1] 16.78016
```

```
lasso.pred=predict(lasso.mod,
                   s=bestlam ,newx=x[test,])
mean((lasso.pred-y.test)^2)
```

The Lasso on Hitters

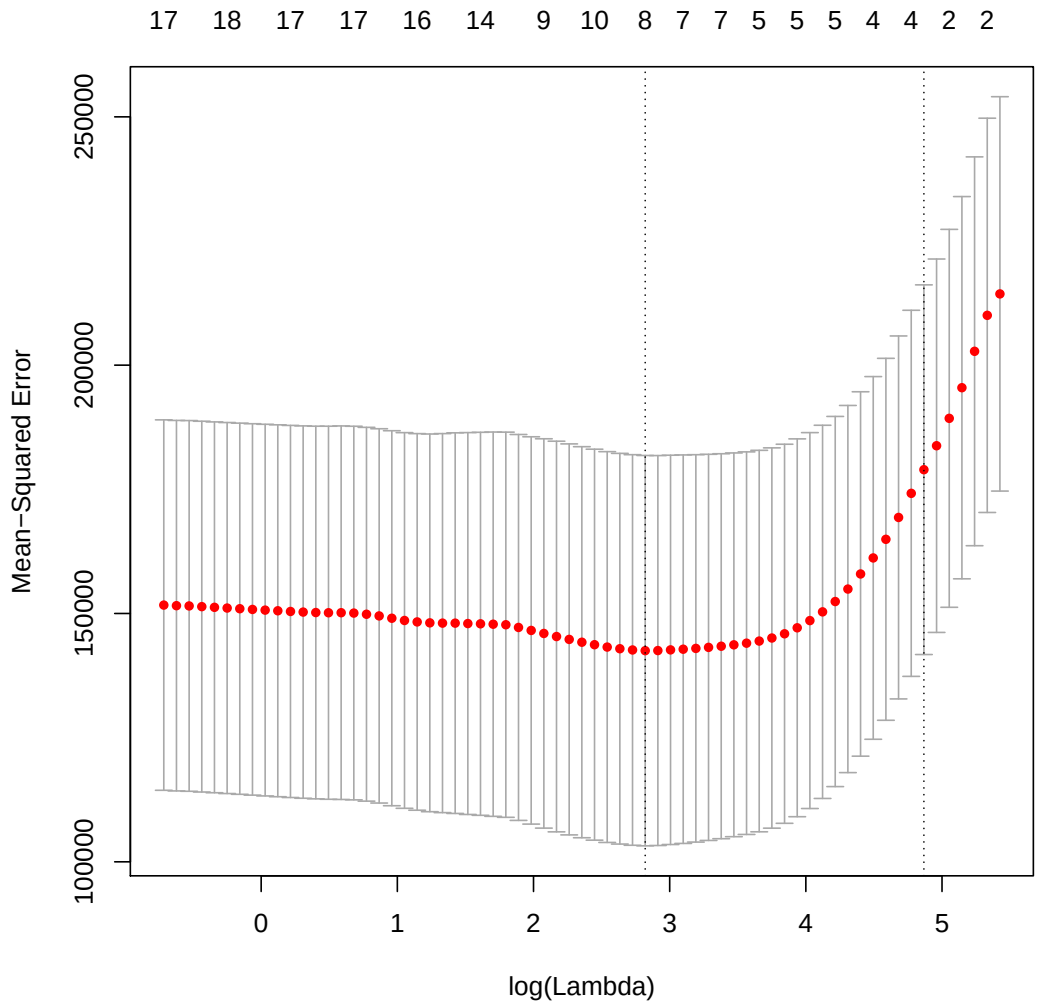


Figure 7: MSE via CV versus log-λ.

The Lasso on Hitters

However, the lasso has a substantial advantage over ridge regression in that the resulting coefficient estimates are sparse.

Next, we see that 12 of the 19 coefficient estimates are exactly zero.

So the lasso model with λ chosen by cross-validation contains only seven variables.

The Lasso on Hitters

```
out=glmnet(x,y,alpha=1,lambda=grid)
lasso.coef=predict(out,type="coefficients",
                   s=bestlam)[1:10,]
lasso.coef
```

```
## (Intercept)       AtBat        Hits       HmRun           R
##   18.539484    0.000000    1.873539    0.000000    0.000
##       Walks       Years      CAtBat       CHits
##    2.217844    0.000000    0.000000    0.000000
```

```
lasso.coef[lasso.coef!=0]
```

```
## (Intercept)        Hits       Walks
##   18.539484    1.873539    2.217844
```