Lab #15 - Ridge and LASSO

Econ 224

October 30th, 2018

Introduction

In this lab you will work through Section 6.6 of ISL and record your code and results in an RMarkdown document. I have added section headings below to help you organize your results. You do not have to submit this lab, so you don't have to type up a detailed description of what you've done. However, I'd suggest that you write down some notes for your own future reference. These will be helpful on the problem set. You do not need to follow the code in ISL exactly: feel free to use your preferred coding style.

You will need the `ISLR` package for the lab, so please install it if you have not done so already. This lab uses the `Hitters` dataset: in particular, we will try to predict a baseball player's `Salary` in a given year using performance statistics from the preceding year. Make sure to read the documentation file for `Hitters` before proceeding. You will also need to use the package `glmnet` so make sure to install it before proceeding.

Ridge Regression

Work through section 6.6.1 of ISL and add your code and results below.

```r
library(glmnet)
library(ISLR)

#-------------- Remove missing values
Hitters <- na.omit(Hitters)

#-------------- Design matrix x without intercept
x <- model.matrix(Salary ~ ., Hitters)[,-1]
y <- Hitters$Salary

#-------------- Grid of values for lambda from 1e11 to 1e-1
lam_grid <- 10^seq(10, -2, length = 100)

#-------------- alpha = 0 in glmnet gives ridge regression
# (glmnet standardizes x by default)
ridge_fits <- glmnet(x, y, alpha = 0, lambda = lam_grid)

#-------------- use coef to display fitted coefficients
ridge_fits$lambda[50] # lambda = 11497.57
```

```
[1] 11497.57
```

```r
coef(ridge_fits)[,50]
```

```
  (Intercept)         AtBat          Hits         HmRun          Runs
407.356050200   0.036957182   0.138180344   0.524629976   0.230701523
          RBI         Walks         Years        CAtBat         CHits
  0.239841459   0.289618741   1.107702929   0.003131815   0.011653637
```

```
       CHmRun         CRuns          CRBI        CWalks       LeagueN 
  0.087545670   0.023379882   0.024138320   0.025015421   0.085028114 
    DivisionW       PutOuts       Assists        Errors    NewLeagueN 
 -6.215440973   0.016482577   0.002612988  -0.020502690   0.301433531 
```

```
ridge_fits$lambda[60] # lambda = 11497.57
```

```
[1] 705.4802
```

```
coef(ridge_fits)[,60]
```

```
 (Intercept)        AtBat         Hits        HmRun         Runs 
 54.32519950   0.11211115   0.65622409   1.17980910   0.93769713 
         RBI        Walks        Years       CAtBat        CHits 
  0.84718546   1.31987948   2.59640425   0.01083413   0.04674557 
      CHmRun        CRuns         CRBI       CWalks      LeagueN 
  0.33777318   0.09355528   0.09780402   0.07189612  13.68370191 
   DivisionW      PutOuts      Assists       Errors   NewLeagueN 
-54.65877750   0.11852289   0.01606037  -0.70358655   8.61181213 
```

```
#-------------- Compare l2 norms with different values of lambda
ridge_coefs <- coef(ridge_fits)[-1,]
get_l2_norm <- function(x) sqrt(sum(x^2))
l2_norms <- apply(ridge_coefs, 2, get_l2_norm)
l2_norms[c(50, 60)]
```

```
      s49       s59 
 6.360612 57.110014 
```

```
#-------------- predict.glmnet() to get ridge coefs for lambda = 50
# (this is a new value of lambda)
predict(ridge_fits, s = 50, type = 'coefficients')
```

```
20 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept)  4.876610e+01
AtBat       -3.580999e-01
Hits         1.969359e+00
HmRun       -1.278248e+00
Runs         1.145892e+00
RBI          8.038292e-01
Walks        2.716186e+00
Years       -6.218319e+00
CAtBat       5.447837e-03
CHits        1.064895e-01
CHmRun       6.244860e-01
CRuns        2.214985e-01
CRBI         2.186914e-01
CWalks      -1.500245e-01
LeagueN      4.592589e+01
DivisionW   -1.182011e+02
```

```
PutOuts      2.502322e-01
Assists      1.215665e-01
Errors      -3.278600e+00
NewLeagueN  -9.496680e+00
```

```r
#-------------- Create training and test sets
set.seed(1)
train_indices <- sample(1:nrow(x), floor(nrow(x)/2))
test_indices <- -(train_indices)
x_train <- x[train_indices,]
y_train <- y[train_indices]
x_test <- x[test_indices,]
y_test <- y[test_indices]

#-------------- Fit ridge on training set
ridge_train <- glmnet(x_train, y_train, alpha = 0, lambda = lam_grid,
                      thresh = 1e-12)

#-------------- Calculate MSE on test set with lambda = 4
ridge_pred1 <- predict(ridge_train, s = 4, newx = x_test)
mean((ridge_pred1 - y_test)^2)
```

```
[1] 101036.8
```

```r
#-------------- Compare to MSE of "null model" with only intercept, or huge lambda
mean((y_test - mean(y_train))^2)
```

```
[1] 193253.1
```

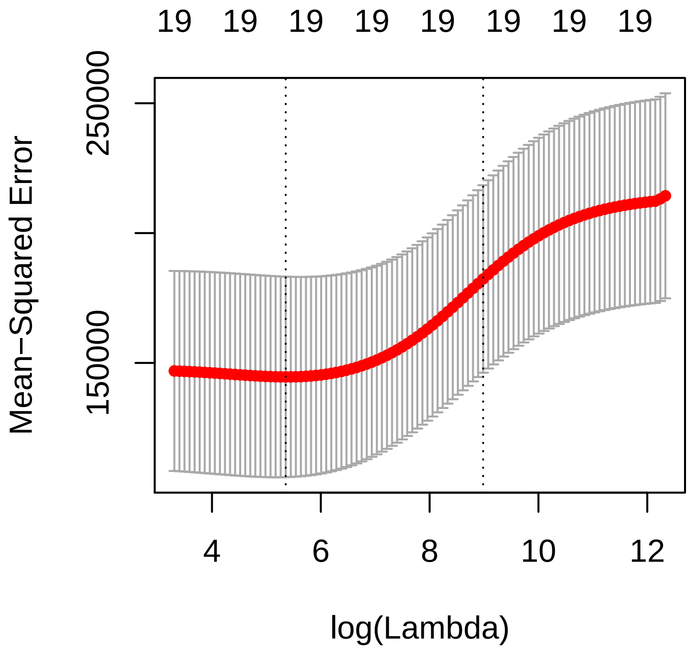
```r
ridge_pred2 <- predict(ridge_train, s = 1e10, newx = x_test)
mean((ridge_pred2 - y_test)^2)
```

```
[1] 193253.1
```

```r
#-------------- Compare to "exact" OLS predictions
# (the code in the book doesn't work: need to specify x and y)
ridge_pred3 <- predict(ridge_train, x = x_train, y = y_train, s = 0,
                       newx = x_test, exact = TRUE)
mean((ridge_pred3 - y_test)^2)
```

```
[1] 114783.1
```

```r
#-------------- Cross-validation for ridge
# (defaults to 10-fold)
set.seed(1)
cv_ridge <- cv.glmnet(x_train, y_train, alpha = 0)
plot(cv_ridge)
```



```
#-------------- Best lambda and associated MSE (according to CV)
best_lam_ridge <- cv_ridge$lambda.min
best_lam_ridge
```

```
[1] 211.7416
```

```
ridge_pred4 <- predict(ridge_train, s = best_lam_ridge, newx = x_test)
mean((ridge_pred4 - y_test)^2)
```

```
[1] 96015.51
```

```
#-------------- Re-fit model with full dataset
ridge_full <- glmnet(x, y, alpha = 0)
predict(ridge_full, type = 'coefficients', s = best_lam_ridge)
```

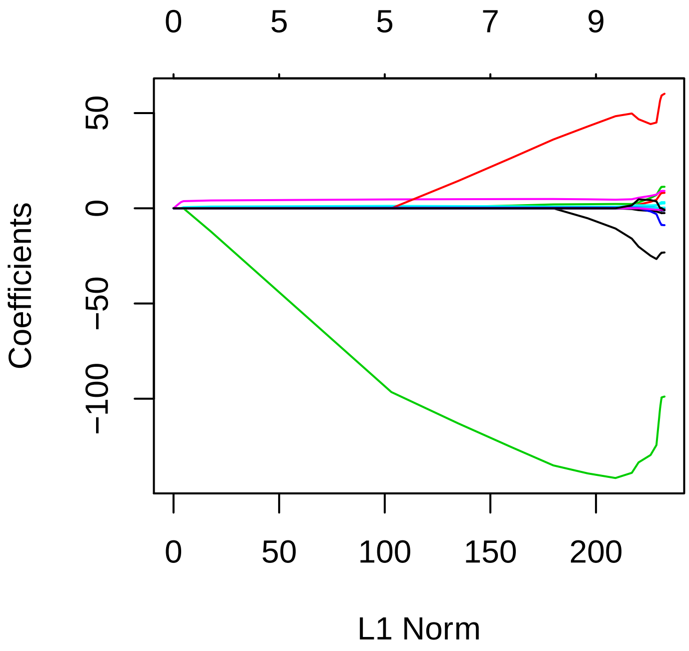
```
20 x 1 sparse Matrix of class "dgCMatrix"
                       1
(Intercept)   9.88487157
AtBat         0.03143991
Hits          1.00882875
HmRun         0.13927624
Runs          1.11320781
RBI           0.87318990
Walks         1.80410229
Years         0.13074383
CAtBat        0.01113978
CHits         0.06489843
CHmRun        0.45158546
CRuns         0.12900049
CRBI          0.13737712
CWalks        0.02908572
```

```
LeagueN      27.18227527
DivisionW   -91.63411282
PutOuts       0.19149252
Assists       0.04254536
Errors       -1.81244470
NewLeagueN    7.21208394
```

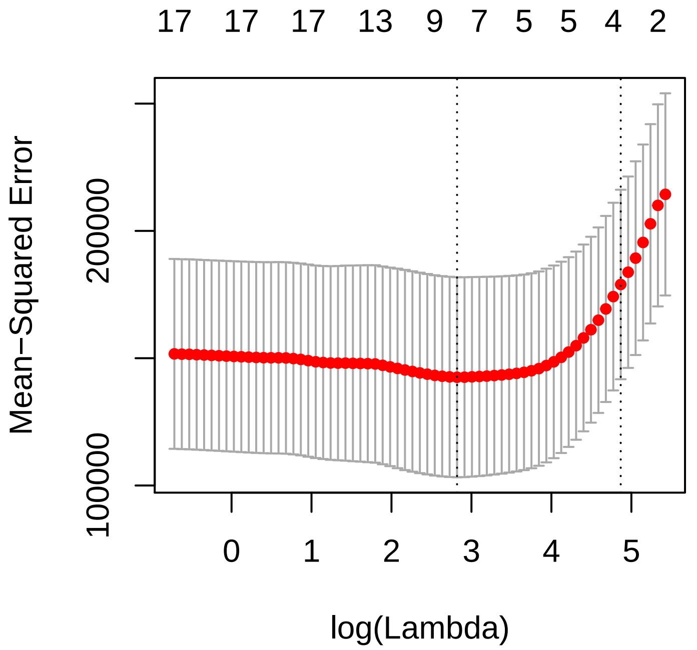
The Lasso

Work through section 6.6.2 of ISL and add your code and results below.

```
#--------------- Fit LASSO to training data
# (set alpha = 1 for LASSO)
lasso_train <- glmnet(x_train, y_train, alpha = 1, lambda = lam_grid)
plot(lasso_train)
```



```
#--------------- Cross-validation to choose lambda for LASSO
set.seed(1) # re-set the seed to get the same folds as for ridge
cv_lasso <- cv.glmnet(x_train, y_train, alpha = 1)
plot(cv_lasso)
```



```
best_lam_lasso <- cv_lasso$lambda.min
lasso_pred <- predict(lasso_train, s = best_lam_lasso, newx = x_test)
mean((lasso_pred - y_test)^2)
```

```
[1] 100743.4
```

```
#--------------- Re-fit LASSO with full dataset
lasso_full <- glmnet(x, y, alpha = 1, lambda = lam_grid)
predict(lasso_full, type = 'coefficients', s = best_lam_lasso)
```

```
20 x 1 sparse Matrix of class "dgCMatrix"
                       1
(Intercept)   18.5394844
AtBat          .
Hits           1.8735390
HmRun          .
Runs           .
RBI            .
Walks          2.2178444
Years          .
CAtBat         .
CHits          .
CHmRun         .
CRuns          0.2071252
CRBI           0.4130132
CWalks         .
LeagueN        3.2666677
DivisionW   -103.4845458
PutOuts        0.2204284
Assists        .
Errors         .
NewLeagueN     .
```