ridgeRegressionAndLASSO

October 28, 2019

1 Ridge Regression

Our goal is to learn about Ridge Regression. This technique is closely related to standard linear regression, but adds an extra twist. The twist involves including an additional term in our minimization of sums of squares that encourages coefficients to shrink.

The idea behind ridge regression is that influencing coefficients to shrink towards zero will also discourage overfitting to our training data. If we can prevent overfitting than we can better generalize our model to yet uncollected data.

1.1 Mechanics

We saw in previous lectures that finding optimal coefficients in a linear regression is the same as minimizing the sum squares error. More concretely, give a set of observations $(x, y)_1, (x, y)_2, \cdots, (x, y)_N$ we can write our probabilistic model as

$$p(y|x) \sim N(\beta_0 + \beta' x, \sigma^2)$$

where β_0 is an intercept.

The optimal β parameters are found by minimizing the following function of β

$$\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \beta' x_i \right)^2 \right\}$$

Ridge regression adds an additional term to the above optimization problem

$$\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \beta' x_i \right)^2 \right\} + \lambda \sum_{m=1}^{M} \beta_m^2$$

The λ parameter is a choice on part of the investigator. Typically, λ is chosen by cross-validation.

This additional term penalizes coefficients related to covariates. Ridge regression does **not** penalize the size of the intercept.

1.2 Optimal β

We can solve our optimization problem by taking partial derivatives and finding the β that zeros out these derivatives.

$$f(\beta) = \sum_{i=1}^{N} \left(y_i - \beta_0 - \beta' x_i\right)^2 + \lambda \sum_{m=1}^{M} \beta_m^2$$

The partial derivative with respect to β_m is

$$\frac{df}{d\beta_m} = \sum_{i=1}^{N} -2x_m \left(y_i - \beta_0 - \beta' x_i\right) + 2\lambda\beta_m \tag{1}$$

We set the derivative equal to zero and solve for β_m.

$$\sum_{i=1}^{N} x_{im} \left(y_i - \beta_0 - \beta' x_i\right) - \lambda\beta_m = 0 \tag{2}$$

$$\sum_{i=1}^{N} x_{im} y_i - \sum_{i=1}^{N} x_{im}(\beta_0 + \beta' x_i) - \lambda\beta_m = 0 \tag{3}$$

$$\sum_{i=1}^{N} x_{im} y_i = \sum_{i=1}^{N} x_{im}(\beta_0 + \beta' x_i) + \lambda\beta_m \tag{4}$$

$$\tag{5}$$

$$x_m^{'} y = x_m^{'}(X + \lambda 1_m)\beta \tag{6}$$

where 1_m is a vector of zeros everywhere and a 1 in the mth position. The set of all optimal betas is then

$$X' y = (X' X + \lambda I)\beta$$

and the optimal β for ridge regression is

$$\beta_{\mathrm{RR}} = (X' X + \lambda I)^{-1} X' y$$

This vector of βs does not include the intercept.

2 LASSO

LASSO works similarly to ridge regression. The goal again is to optimize the typical SSE with an additional term

$$\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \beta' x_i \right)^2 \right\} + \lambda \sum_{m=1}^{M} |\beta_m|$$

The additional term is the sum of the absolute value of βs, not including the intercept.

Though a subtle difference, we can not solve for the optimal βs exactly. A numerical solver capable of convex optimization can provide an estimate of the optimal βs for lasso.

The absolute value constraint put on the sum of coefficients is stronger than the squared coefficients in ridge regression. This additional pressure to shrink estimates serves as a natural way to, among many candidate coefficients, find a subset that best predict the target variable y.

2.1 Example data

The example data set is a collection of information about baseball players. The data contains, for each baseball player, information about their performance and how much money they make.

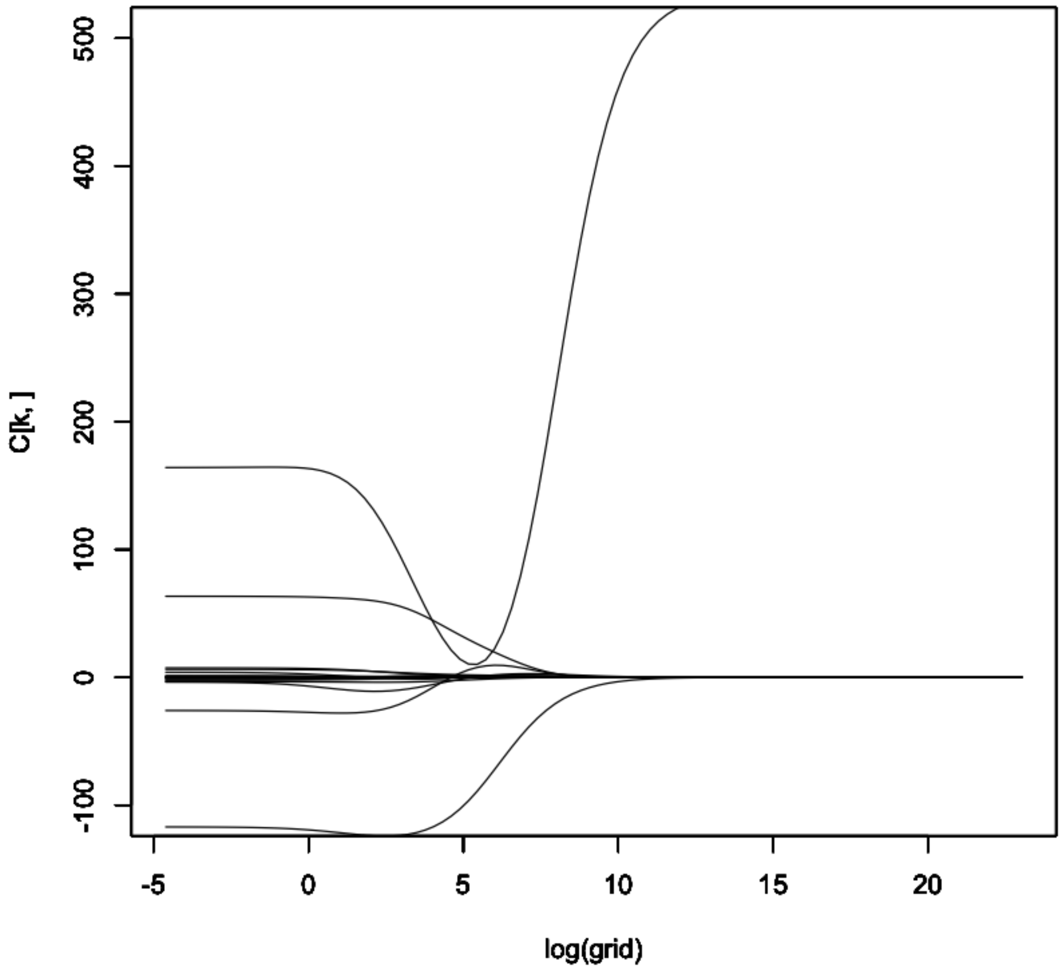
Our goal will be to understand the relationship between a baseball player's performance and their salary.

To generalize from this subset to other baseball players we did not collect data on, ridge regression will be used. Ridge regression will penalize over emphasizing specific covariates that could cause us to overfit to our training data. We can also fit a LASSO model to our training data. LASSO will shrink covariate estimates to zero if they do not substantially contribute to reducing the SSE. LASSO aims to create a more parsimonious model, and by doing so, create a model that can better generalize from training to testing data.

[56]:
```
library(ISLR)
library(glmnet)
library(dplyr)
library(tidyr)

# Hitters dataset
Hitters = na.omit(Hitters)
print(head(Hitters))

# Take covariates except Salary
x = model.matrix(Salary~., Hitters)[,-1]

# Take salary variable
y = Hitters$Salary

# Ridge regression can be fit with the glmnet function.
```

```r
# This function takes a matrix of covariates, the target variable.
# Setting alpha=0 specifies Ridge regression and Lambda corresponds the the
 ↪amount of covariate "shrinkage"
grid = 10^seq(10, -2, length = 100)
ridge_mod = glmnet(x, y, alpha = 0, lambda = grid)

C = coef(ridge_mod)
for (k in 1:nrow(C)){
    plot( log(grid),  C[k,], type='l', ylim=c(-100,500))
    par(new=TRUE)
}
```

```
                  AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun
-Alan Ashby         315   81     7   24  38    39    14   3449   835     69
-Alvin Davis        479  130    18   66  72    76     3   1624   457     63
-Andre Dawson       496  141    20   65  78    37    11   5628  1575    225
-Andres Galarraga   321   87    10   39  42    30     2    396   101     12
-Alfredo Griffin    594  169     4   74  51    35    11   4408  1133     19
-Al Newman          185   37     1   23   8    21     2    214    42      1
                  CRuns CRBI CWalks League Division PutOuts Assists Errors
-Alan Ashby         321  414    375      N        W     632      43     10
-Alvin Davis        224  266    263      A        W     880      82     14
-Andre Dawson       828  838    354      N        E     200      11      3
-Andres Galarraga    48   46     33      N        E     805      40      4
-Alfredo Griffin    501  336    194      A        W     282     421     25
-Al Newman           30    9     24      N        E      76     127      7
                  Salary NewLeague
-Alan Ashby        475.0         N
-Alvin Davis       480.0         A
-Andre Dawson      500.0         N
-Andres Galarraga   91.5         N
-Alfredo Griffin   750.0         A
-Al Newman          70.0         A
```



[63]:
```
cat("Lambda")
ridge_mod$lambda[60]
coef(ridge_mod)[,60]

cat("SSE")
sqrt(sum(coef(ridge_mod)[-1,60]^2))

cat("Lambda")
ridge_mod$lambda[50]
coef(ridge_mod)[,50]

cat("SSE")
sqrt(sum(coef(ridge_mod)[-1,50]^2))
```

Lambda

705.480231071865

(Intercept) 54.3251995018372 **AtBat** 0.112111145878249 **Hits** 0.656224085323628 **HmRun** 1.17980909638777 **Runs** 0.937697128927054 **RBI** 0.847185458771521 **Walks** 1.31987948048781 **Years** 2.59640424574253 **CAtBat** 0.0108341254432856 **CHits** 0.0467455700054452 **CHmRun** 0.337773183143353 **CRuns** 0.0935552830000676 **CRBI** 0.0978040232271687 **CWalks** 0.0718961166304866 **LeagueN** 13.6837019095343 **DivisionW** -54.658777504592 **PutOuts** 0.118522894134745 **Assists** 0.01606037317599 **Errors** -0.703586547290985 **NewLeagueN** 8.61181213448926

SSE

57.110014262533

Lambda

11497.5699539774

(Intercept) 407.356050200416 **AtBat** 0.0369571817501359 **Hits** 0.138180343807892 **HmRun** 0.524629975886911 **Runs** 0.230701522621179 **RBI** 0.239841458504058 **Walks** 0.289618741049884 **Years** 1.10770292908555 **CAtBat** 0.00313181522151328 **CHits** 0.0116536373557531 **CHmRun** 0.0875456697555949 **CRuns** 0.0233798823693758 **CRBI** 0.0241383203685686 **CWalks** 0.0250154205993732 **LeagueN** 0.0850281135625444 **DivisionW** -6.21544097273146 **PutOuts** 0.0164825767604547 **Assists** 0.00261298804528183 **Errors** -0.0205026903654579 **NewLeagueN** 0.301433531372699

SSE

6.36061242142791

[67]:

```
# The same process used for Ridge Regression can be used to fit LASSO.
# The only change is specifying alpha=1.
grid = 10^seq(10, -2, length = 100)
ridge_mod = glmnet(x, y, alpha = 1, lambda = grid)

cat("Lambda")
ridge_mod$lambda[90]
coef(ridge_mod)[,90]
sqrt(sum(coef(ridge_mod)[-1,90]^2))

cat("Lambda")
ridge_mod$lambda[80]
coef(ridge_mod)[,80]
sqrt(sum(coef(ridge_mod)[-1,80]^2))

cat("Lambda")
ridge_mod$lambda[70]
coef(ridge_mod)[,70]
sqrt(sum(coef(ridge_mod)[-1,70]^2))
```

`Lambda`

0.162975083462064

(Intercept) 162.471571163125 **AtBat** -2.00725495341728 **Hits** 7.46834345075141 **HmRun** 3.50180353190197 **Runs** -2.15880439279739 **RBI** -0.741855229071078 **Walks** 6.15950199852769 **Years** -4.69367835398105 **CAtBat** -0.140363573272626 **CHits** 0.0843854865595957 **CHmRun** -0.00859358349740949 **CRuns** 1.39671552949042 **CRBI** 0.724102431969445 **CWalks** -0.810053300830832 **LeagueN** 59.7052960522582 **DivisionW** -116.71196728165 **PutOuts** 0.283100309617026 **Assists** 0.354242405196464 **Errors** -3.25345107822638 **NewLeagueN** -22.0755201272487

133.510043789335

`Lambda`

2.65608778294668

(Intercept) 124.089487297287 **AtBat** -1.56009839061161 **Hits** 5.69316850453147 **HmRun** 0 **Runs** 0 **RBI** 0 **Walks** 4.7505395162242 **Years** -9.51802414366672 **CAtBat** 0 **CHits** 0 **CHmRun** 0.519161053589954 **CRuns** 0.660407436641157 **CRBI** 0.391541539043815 **CWalks** -0.53268682707415 **LeagueN** 32.112549338136 **DivisionW** -119.25835400136 **PutOuts** 0.272620661250992 **Assists** 0.174816439652482 **Errors** -2.05674316297772 **NewLeagueN** 0

124.125964686996

`Lambda`

43.2876128108306

(Intercept) 74.7044627184386 **AtBat** 0 **Hits** 1.64328660554985 **HmRun** 0 **Runs** 0 **RBI** 0 **Walks** 1.89863540344793 **Years** 0 **CAtBat** 0 **CHits** 0 **CHmRun** 0 **CRuns** 0.183705603033279 **CRBI** 0.374319642602473 **CWalks** 0 **LeagueN** 0 **DivisionW** -55.4369608623154 **PutOuts** 0.151945324305305 **Assists** 0 **Errors** 0 **NewLeagueN** 0

55.4955744833595