Regularization in Linear Regression

Applied Machine Learning for Educational Data Science

true

10/20/2021

Contents

[Updated: Thu, Oct 21, 2021 - 17:46:12]

Regularization

Regularization is a general strategy to incorporate additional penalty terms into the model fitting process and used not just for regression but a variety of other types of models. The idea behind the regularization is to constrain the size of regression coefficients with the purpose of reducing their sampling variation and, hence, reducing the variance of model predictions. These constrains are typically incorporated into the loss function to be optimized. There are two commonly used regularization strategy: **ridge penalty** and **lasso penalty**. In addition, there is also **elastic net**, a mixture of these two strategies.

Ridge Regression

Ridge Penalty

Remember that we formulated the loss function for the linear regression as the sum of squared residuals across all observations. For ridge regression, we add a penalty term to this loss function and this penalty term is a function of all the regression coefficients in the model. Assuming that there are P regression coefficients in the model, the penalty term for the ridge regression would be

$$\lambda\sum_{i=1}^{P}\beta_p^2,$$

where λ is a parameter that penalizes the regression coefficients when they get larger. Therefore, when we fit a regression model with ridge penalty, the loss function to minimize becomes

$$Loss = \sum_{i=1}^{N}\epsilon_{(i)}^2 + \lambda\sum_{i=1}^{P}\beta_p^2,$$

$$Loss = SSR + \lambda\sum_{i=1}^{P}\beta_p^2.$$

Let's consider the same example from the previous class. Suppose we fit a simple linear regression model such that the readability score is the outcome (Y) and average word length is the predictor(X). Our regression model is

$$Y = \beta_0 + \beta_1 X + \epsilon,$$

and let's assume the set of coefficients are $\{\beta_0, \beta_1\}$ = {7.5,-2}, so my model is

$$Y = 7.5 - 2X + \epsilon.$$

Then, the value of the loss function when $\lambda = 0.2$ would be equal to 27.433.

```
readability_sub <- read.csv('https://raw.githubusercontent.com/uo-datasci-specialization/c4-ml-fall-202

d <-  readability_sub[,c('mean.wl','target')]

b0 = 7.5
b1 = -2

d$predicted <- b0 + b1*d$mean.wl
d$error     <- d$target - d$predicted

d
```

```
    mean.wl      target  predicted       error
1  4.603659 -2.58590836 -1.7073171 -0.87859129
2  3.830688  0.45993224 -0.1613757  0.62130790
3  4.180851 -1.07470758 -0.8617021 -0.21300545
4  4.015544 -1.81700402 -0.5310881 -1.28591594
5  4.686047 -1.81491744 -1.8720930  0.05717559
6  4.211340 -0.94968236 -0.9226804 -0.02700194
7  4.025000 -0.12103065 -0.5500000  0.42896935
8  4.443182 -2.82200582 -1.3863636 -1.43564218
9  4.089385 -0.74845172 -0.6787709 -0.06968077
10 4.156757  0.73948755 -0.8135135  1.55300107
11 4.463277 -0.96218937 -1.4265537  0.46436430
12 5.478261 -2.21514888 -3.4565217  1.24137286
13 4.770492 -1.21845136 -2.0409836  0.82253224
14 4.568966 -1.89544351 -1.6379310 -0.25751247
15 4.735751 -0.04101056 -1.9715026  1.93049203
```

```
16 4.372340 -1.83716516 -1.2446809 -0.59248431
17 4.103448 -0.18818586 -0.7068966  0.51871069
18 4.042857 -0.81739314 -0.5857143 -0.23167886
19 4.202703 -1.86307557 -0.9054054 -0.95767016
20 3.853535 -0.41630158 -0.2070707 -0.20923088
```

```
lambda = 0.2

loss <- sum((d$error)^2) + lambda*(b0^2 + b1^2)

loss
```

```
[1] 27.43364
```

Notice that when λ is equal to 0, the loss function is identical to SSR; therefore, it becomes a linear regression with no regularization. As the value of λ increases, the degree of penalty linearly increases. Technically, the λ can take any positive value between 0 and ∞.

As we did in the previous lecture, imagine that we computed the loss function with the ridge penalty term for every possible combination of the intercept (β_0) and the slope (β_1). Let's say the plausible range for the intercept is from -10 to 10 and the plausible range for the slope is from -2 to 2. Now, we also have to think different values of λ because the surface we try to minimize is dependent on the value λ and different values of λ yield different estimates of β_0 and and β_1.

You can try a number of different values for λ using the shiny app at this link and explore how the loss function value and coefficient estimates change for different values of λ. Note that when λ is equal to zero, it should be equivalent of what we have seen in the earlier lecture. Try values of 1, 5, 10, 50, and 100.

Below is also a demonstration of what happens to loss function and the regression coefficients for increasing levels of λ.

Model Estimation

Matrix Solution The matrix solution we learned before for regression without regularization can also be applied to estimate the coeffients from ridge regression given the λ value. Given that

- $\mathbf{Y}$ is an N x 1 column vector of observed values for the outcome variable,
- $\mathbf{X}$ is an N x (P+1) **design matrix* for the set of predictor variables including an intercept term,
- $\boldsymbol{\beta}$ is an (P+1) x 1 column vector of regression coefficients,
- $\mathbf{I}$ is a (P+1) x (P+1) identity matrix,
- and λ is positive real-valued number,

the set of ridge regression coefficients can be estimated using the following matrix operation.

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathbf{T}}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^{\mathbf{T}}\mathbf{Y}$$

Now, suppose we want to predict the readability score by using the two predictors, the average word length (X_1) and number of sentences (X_2). Our model will be

$$Y_{(i)} = \beta_0 + \beta_1 X_{1(i)} + \beta_2 X_{2(i)} + \epsilon_{(i)}.$$

If we estimate the ridge regression coefficients by using $\lambda = .5$, the estimates would be $\{\beta_0, \beta_1, \beta_2\} =$ {0.277,-.593,0.097}.

```
Y <-  as.matrix(readability_sub$target)
X <-  as.matrix(cbind(1,readability_sub$mean.wl,readability_sub$sents))

lambda <- 0.5

beta <- solve(t(X)%*%X + lambda*diag(ncol(X)))%*%t(X)%*%Y

beta
```

```
            [,1]
[1,]  0.27693153
[2,] -0.59327091
[3,]  0.09692781
```

If we change the value of λ to 2, then we will get a different set of estimates for the regression coefficients.

```
Y <-  as.matrix(readability_sub$target)
X <-  as.matrix(cbind(1,readability_sub$mean.wl,readability_sub$sents))

lambda <- 2

beta <- solve(t(X)%*%X + lambda*diag(ncol(X)))%*%t(X)%*%Y

beta
```

```
             [,1]
[1,]  0.006012867
[2,] -0.526374942
[3,]  0.095845692
```

We can manipulate the value of λ from 0 to 100 with increments of .1 and calculate the regression coefficients for every possible value of λ. Note the regression coefficients will shrink towards zero, but will never be exactly equal to zero in ridge regression.

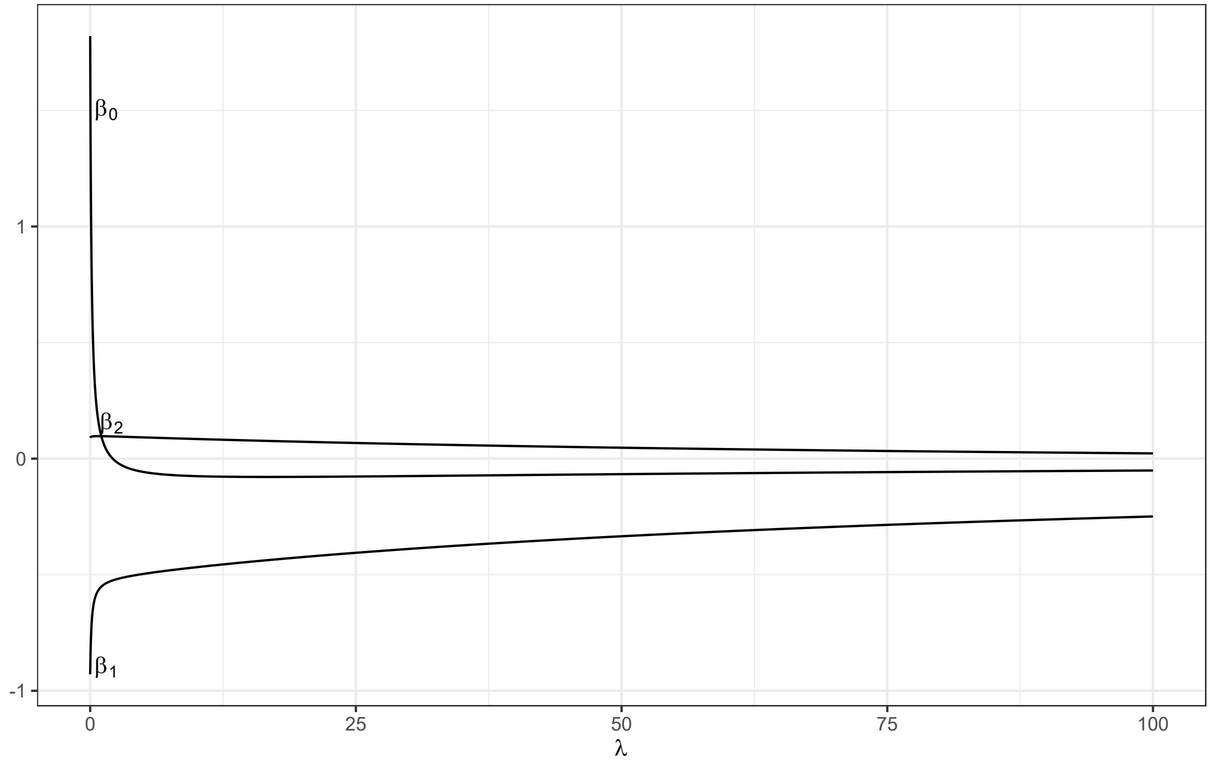
```
Y <-  as.matrix(readability_sub$target)
X <-  as.matrix(cbind(1,readability_sub$mean.wl,readability_sub$sents))

lambda <- seq(0,100,.01)

beta     <- data.frame(matrix(nrow=length(lambda),ncol=4))
beta[,1] <- lambda

for(i in 1:length(lambda)){
  beta[i,2:4] <- t(solve(t(X)%*%X + lambda[i]*diag(ncol(X)))%*%t(X)%*%Y)
}

ggplot(data = beta)+
  geom_line(aes(x=X1,y=X2))+
  geom_line(aes(x=X1,y=X3))+
  geom_line(aes(x=X1,y=X4))+
  xlab(expression(lambda))+
  ylab('')+
  theme_bw()+
  annotate(geom='text',x=1.5,y=1.5,label=expression(beta[0]))+
  annotate(geom='text',x=2,y=.15,label=expression(beta[2]))+
  annotate(geom='text',x=1.5,y=-.9,label=expression(beta[1]))
```



Ridge Regression with Standardized Variables We didn't consider a very important issue while we discussed the model estimation. This issue is not necessarily important if you only one predictor. However, it is critial whenever you have more than one predictor. Different variables have different scales and therefore the magnitude of the regression coefficients for different variables will be dependent on the scales of the variables. A regression coefficient for a variable with a range from 0 to 100 will be very different than a regression coefficient for a variable with a range from 0 to 1.

`glmnet()` function

Tuning the Hyperparameter λ without Cross-validation

The λ parameter in ridge regression is called a **hyperparameter**. In the context of machine learning, the parameters in a model can be classified into two types: parameters and hyperparameters. The parameters of the model are typically estimated from data and not set by us. In the context of regularized regression, regression coefficients, $\{\beta_0, \beta_1, ..., \beta_P\}$,

Tuning the Hyperparameter λ with 10-fold Cross-validation

Using Ridge Regression to Predict Readability Scores

Lasso Regression

Lasso Penalty

Elastic Net