Homework 0

Yipin

September 15, 2018

Assignment 0

1 Regularization.

Using the accompanying Hitters dataset (found here (Links to an external site.)Links to an external site.), we will explore regression models to predict a player's Salary from other variables. You can use any programming languages or frameworks that you wish.

1.1 Use LASSO regression to predict Salary from the other numeric predictors (you should omit the categorical predictors).

1.1.1. Create a visualization of the coefficient trajectories

```
set.seed(123)
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.4.4
```

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

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

```
## Warning: package 'foreach' was built under R version 3.4.4
```

```
## Loaded glmnet 2.0-16
```

```
library(plotmo)
```

```
## Warning: package 'plotmo' was built under R version 3.4.4
```

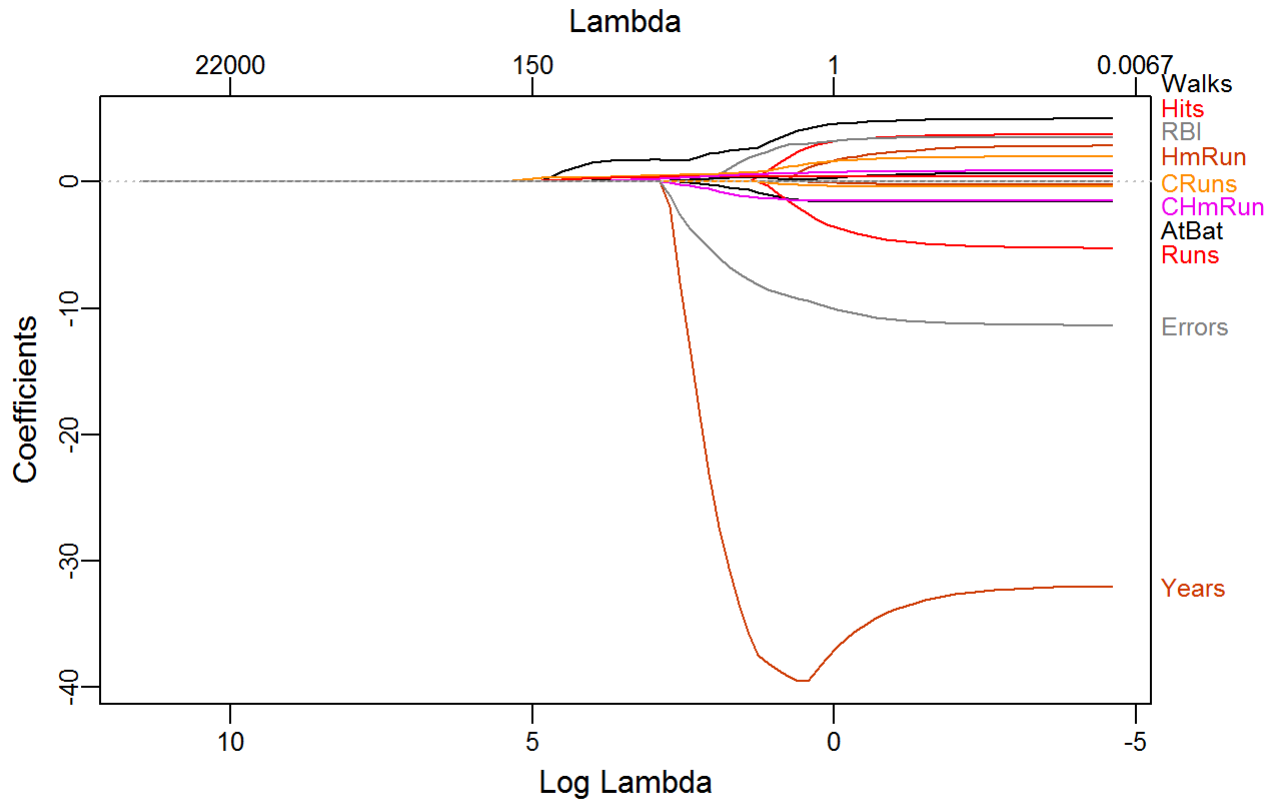
```
## Loading required package: plotrix
```

```
## Warning: package 'plotrix' was built under R version 3.4.4
```

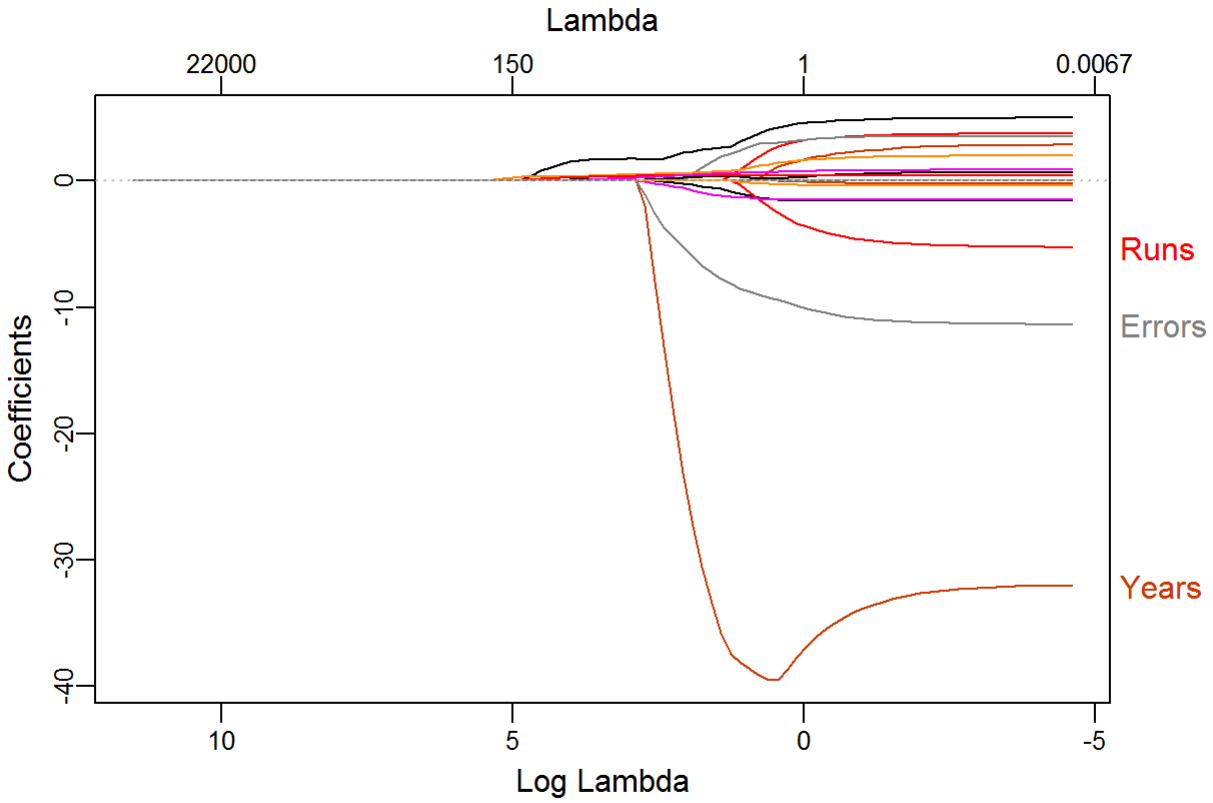
```
## Loading required package: TeachingDemos
```

```
## Warning: package 'TeachingDemos' was built under R version 3.4.4
```

```
df = read.csv("C:/Deep Learning/Homework1/Hitters.csv")
df = na.omit(df)
## remove categorical data
df = df[ , -which(names(df) %in% c("League", "Division", "X", "NewLeague"))]
grid = 10^seq(5, -2, length = 100)

x = model.matrix(Salary~., df)[,-1]
y = df$Salary

##generate training and testing dataset
train = sample(1:nrow(x),nrow(x)/2)
test = -train
y.test = y[test]

## make the trajectories
lasso.mod = glmnet(x[train,],y[train],alpha = 1, lambda = grid)
plot_glmnet(lasso.mod)
```




```
plot_glmnet(lasso.mod, label=3)
```

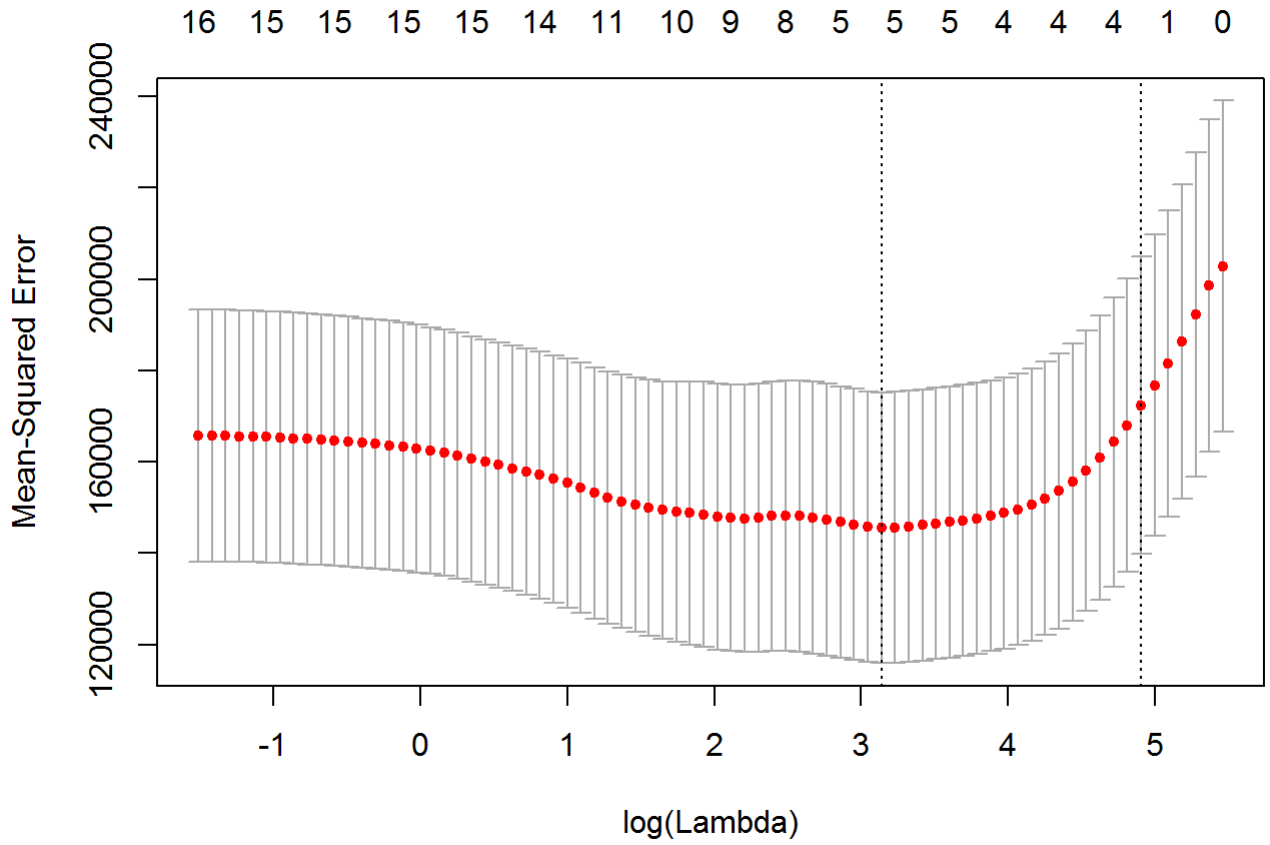


1.1.2. Comment on which are the final three predictors that remain in the model

The last three predictors are “Runs”, “Years” and “Errors”

1.1.3. Use cross-validation to find the optimal value of the regularization penalty

```
cv.lasso = cv.glmnet(x[train,],y[train], alpha = 1)
plot(cv.lasso)
```




```
bestlam = cv.lasso$lambda.min
cat("Best lambda is", bestlam)
```

```
## Best lambda is 23.03503
```

```
lasso.pred = predict(lasso.mod, s = bestlam, newx = x[test,])
cat("\nMSE for the test dataset is", mean((lasso.pred - y[test])^2))
```

```
##
## MSE for the test dataset is 115520.6
```

This plot shows that with a larger lambda, we can push more coefficients to 0, however, at the cost of a larger MSE.

1.1.4. How many predictors are left in that model?

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

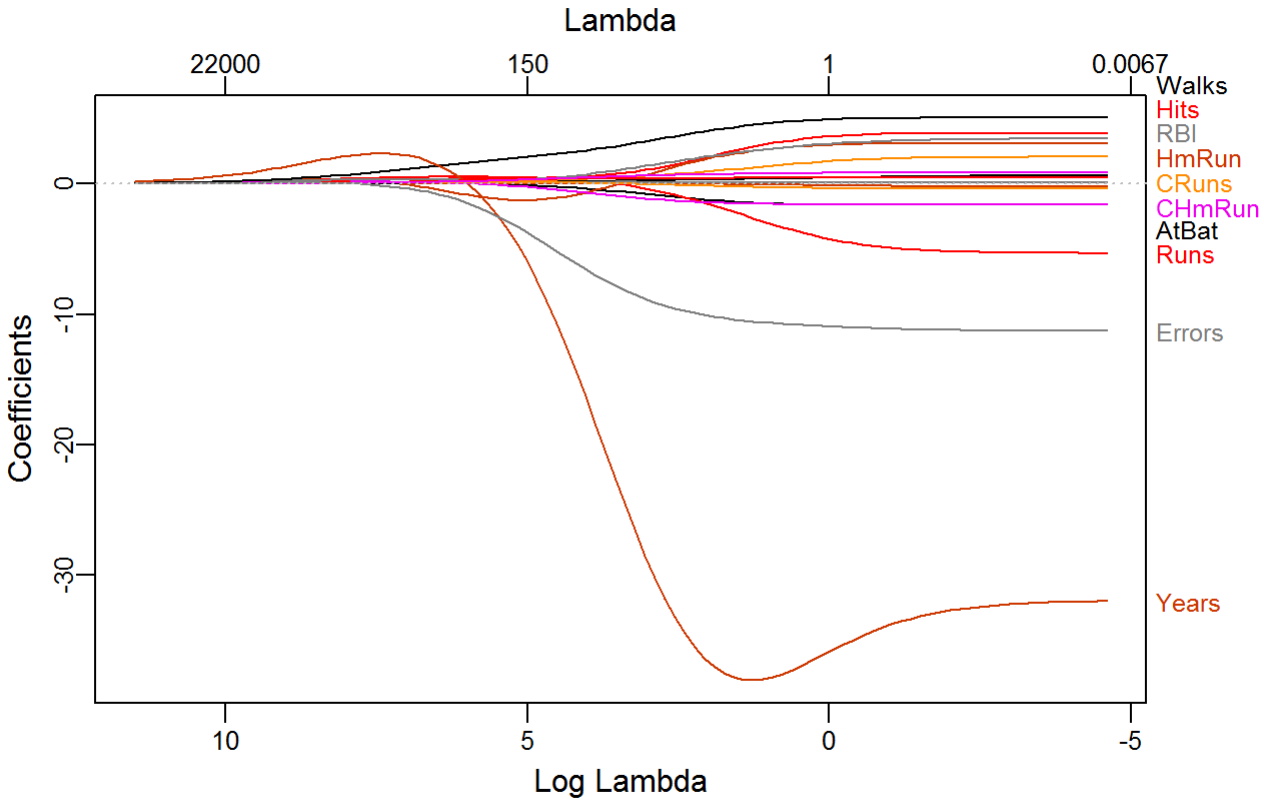
```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept) -2.399046e+01
## AtBat        .
## Hits         1.872269e+00
## HmRun        .
## Runs         .
## RBI          .
## Walks        2.205588e+00
## Years        .
## CAtBat       .
## CHits        .
## CHmRun       3.543539e-04
## CRuns        2.331654e-01
## CRBI         3.730225e-01
## CWalks       .
## PutOuts      2.058705e-01
## Assists      .
## Errors       .
```

6 predictors are left in that model.

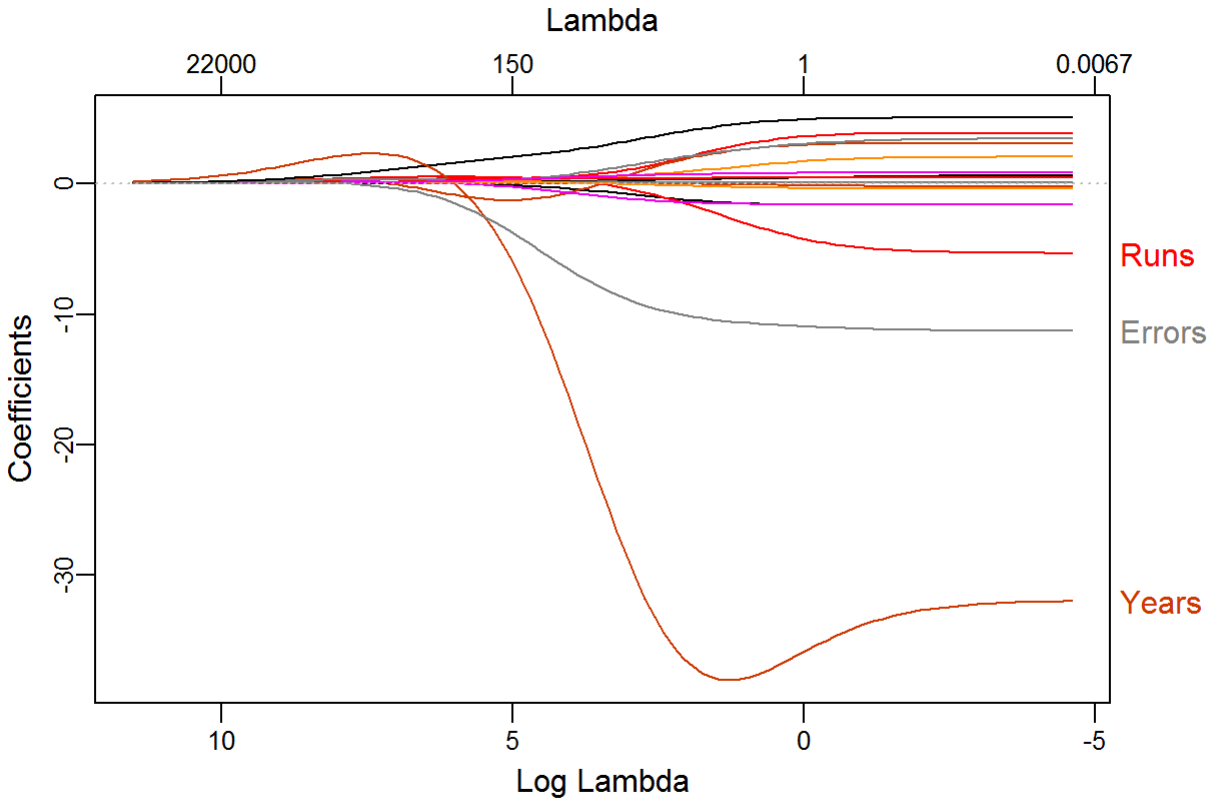
1.2 Repeat with Ridge Regression.

1.2.1 Visualize the coefficient trajectories

```
ridge.mod = glmnet(x[train,],y[train],alpha = 0, lambda = grid)
plot_glmnet(ridge.mod)
```




```
plot_glmnet(ridge.mod, label=3)
```



The last three predictors are also Runs, Errors and Years.

1.2.2 Use cross-validation to find the optimal value of the regularization penalty

```
cv.lasso = cv.glmnet(x[train,],y[train], alpha = 0)
bestlam = cv.lasso$lambda.min
cat("Best lambda is", bestlam,"\n")
```

```
## Best lambda is 350.112
```

```
ridge.out = glmnet(x,y, alpha = 0, lambda = grid)
ridge.coef = predict(ridge.out, type = "coefficients", s = bestlam)
print(ridge.coef)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept) -10.56131640
## AtBat         0.07635173
## Hits          0.84721575
## HmRun         0.46421411
## Runs          1.07881681
## RBI           0.91040622
## Walks         1.63826327
## Years         1.51429627
## CAtBat        0.01123927
## CHits         0.05635496
## CHmRun        0.39079801
## CRuns         0.11415871
## CRBI          0.11811442
## CWalks        0.05982305
## PutOuts       0.16509612
## Assists       0.03053209
## Errors       -1.22456904
```

```
ridge.pred = predict(ridge.mod, s = bestlam, newx = x[test,])
cat("\nMSE for the test dataset is", mean((ridge.pred - y[test])^2))
```

```
##
## MSE for the test dataset is 110602.1
```

The L2 regularization does not push the coefficients to 0. Instead, it will make them very small. It also decreases the variance with a slight increase in bias.

2 Short Answer.

2.1 Explain in your own words the bias-variance tradeoff

When training the machine learning models, we cannot minimize the bias and the variance at the same time. When we try to reduce the bias, the variance increases. When the variance decreases, the bias also increases.

2.2 What role does regularization play in this tradeoff?

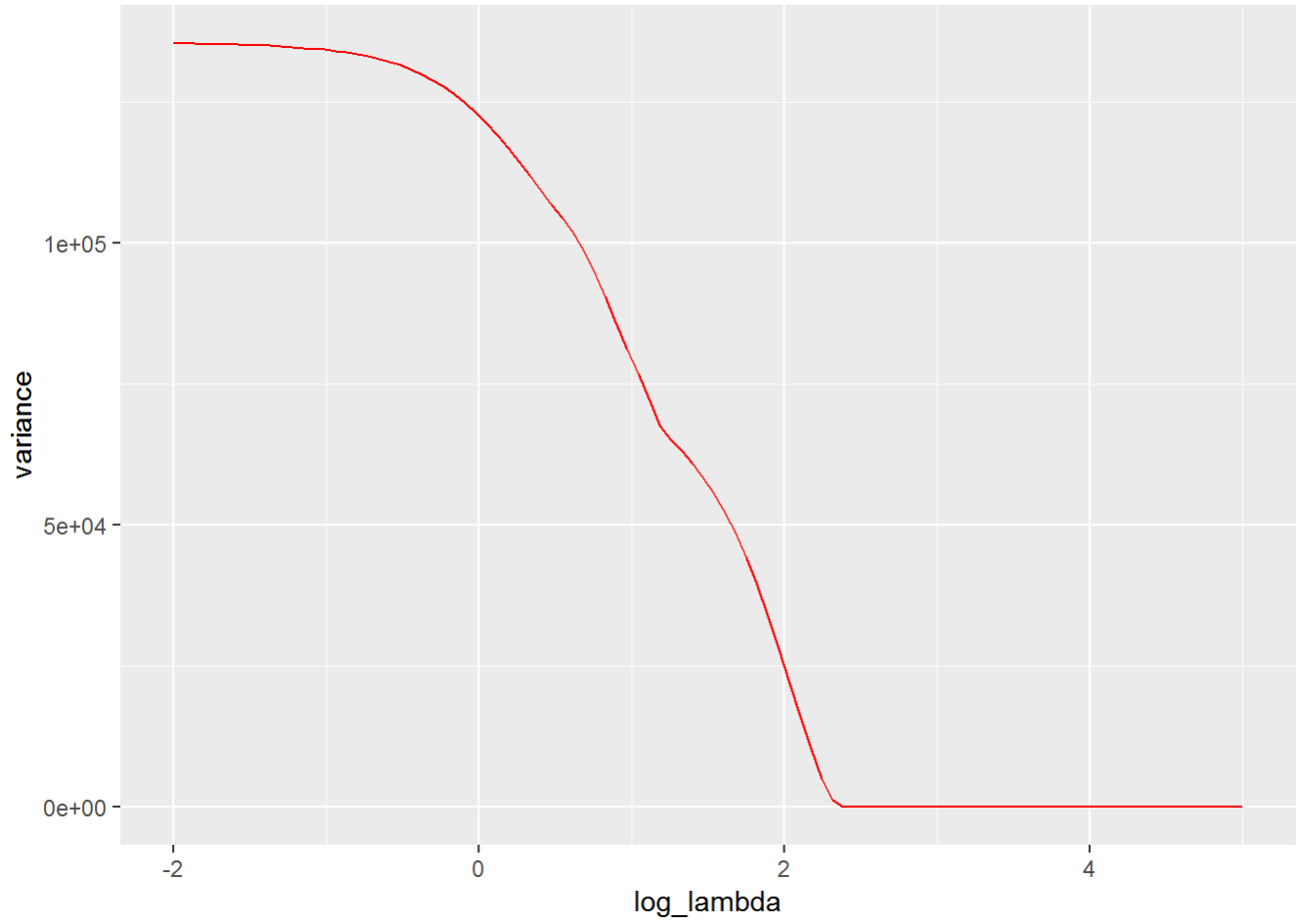
The regularization is a useful way for us to find the midpoint of bias-variance tradeoff. It will put a penalty towards coefficients of predictors that are not significant. It can reduce the variance when the bias increases a little bit.

2.3 Make reference to your findings in number (1) to describe models of high/low bias and variance

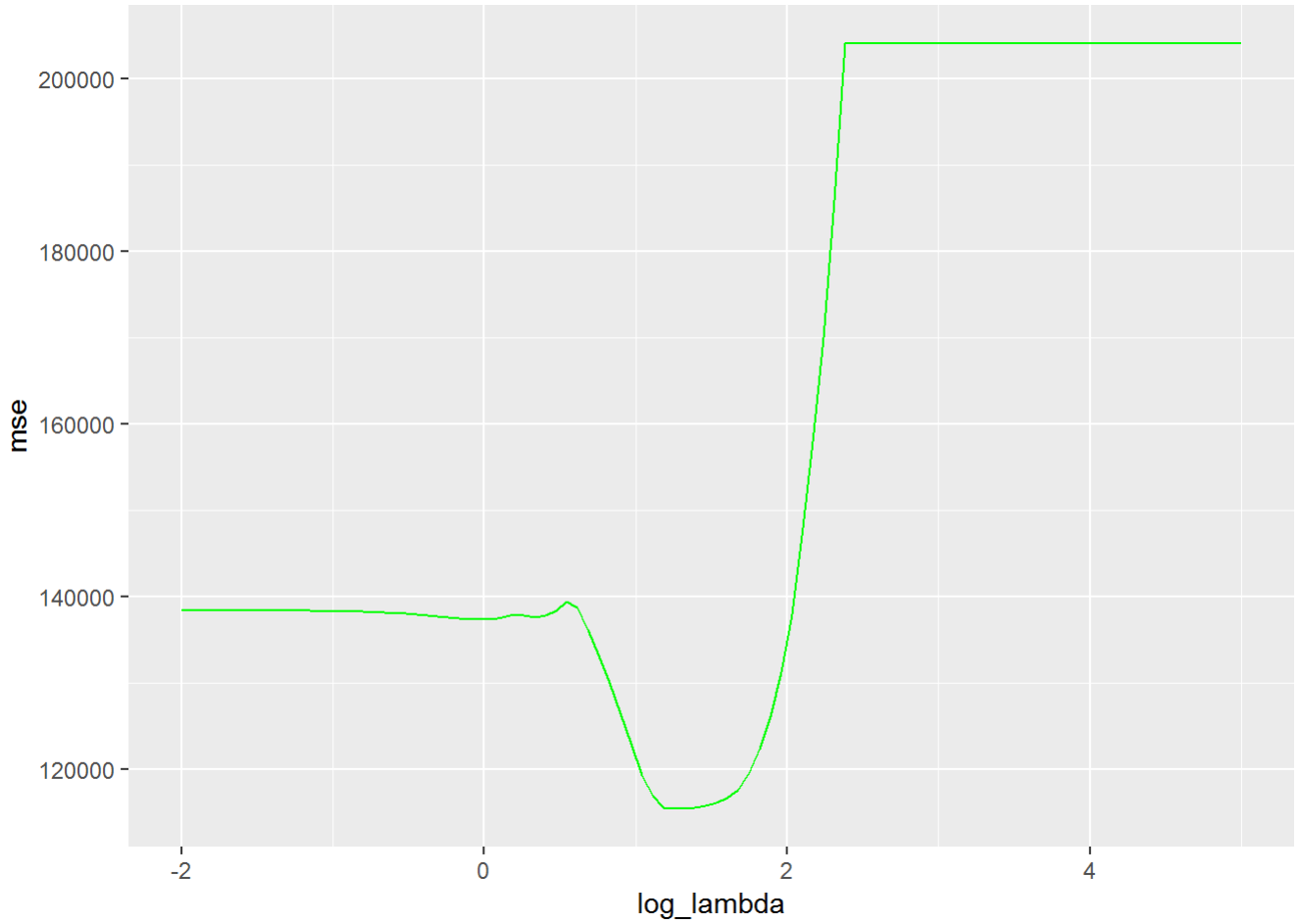
```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
grid2 = 10^seq(5, -2, length = 100)
variance = numeric(100)
mse = numeric(100)
for(i in 1:100){
  lambda <- grid2[i]
  lasso.pred = predict(lasso.mod, s = lambda, newx = x[test,])
  variance[i] = var(lasso.pred)
  mse[i] = mean((lasso.pred - y[test])^2)
}

df2 <- data.frame(log(grid2, base = 10), mse, variance)
colnames(df2)[colnames(df2) == "log.grid2..base...10."] <-  "log_lambda"
ggplot(df2)+
  geom_line(aes(log_lambda,variance),color = "red")
```




```
ggplot(df2)+
  geom_line(aes(log_lambda,mse),color = "green")
```



From the plot, we can see that, with the increase of lambda, we can decrease variance. However, the MSE might increase, this reveals the bias-variance tradeoff. In this case, when lambda is around 1.4, which is the best lambda, we can have the minimal MSE.

Moreover, because a fixed bias is much easier for us to adjust, sometimes we can utilize the tradeoff to get an excellent result, with a minimal variance and a large but fixed bias.