Ridge_Regression.R

vijaykalmath

2022-01-04

```
# Ridge Regression and Lasso Regression

# GLMNET package is used for Lasso and Ridge Regression

# set seed for reproducibility
set.seed(1)

library(ISLR)  # For the data
library(glmnet)
```

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

```
## Loaded glmnet 4.1-3
```

```
# Omitting NA
Hitters = na.omit(Hitters)

# Model Matrix produces matrix with 19 predictors but also transforms any qualitative variables into du

x = model.matrix(Salary~.,data=Hitters)[,-1]

y = Hitters$Salary

# Creating Exponential space gridspace for lambda or regularization factor
grid= 10^seq(10,-2,length=100)

# Alpha = 0 -> Ridge Regression , Alpha = 1 -> Lasso Regression

ridge.mod = glmnet(x,y,alpha=0,lambda = grid)

# 100 Models with corresponding Coefficients
dim(coef(ridge.mod))
```

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

```
# Define Train and test set
train <- sample(1:nrow(x), nrow(x) / 2)
test <- (-train)
```

```
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] 142199.2
```

```
# With a single dimensional predictor , intercept only , we get the MSE of the constant mean predictor
mean((mean(y[train]) - y[test])^2)
```

```
## [1] 224669.9
```

```
# By making the lambda value very high , we can force the all the coeff to become closed to 0.

ridge.pred <- predict(ridge.mod, s = 1e10, newx = x[test, ])

mean((ridge.pred - y[test])^2)
```

```
## [1] 224669.8
```

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

mean((ridge.pred - y[test])^2)
```

```
## [1] 168588.6
```

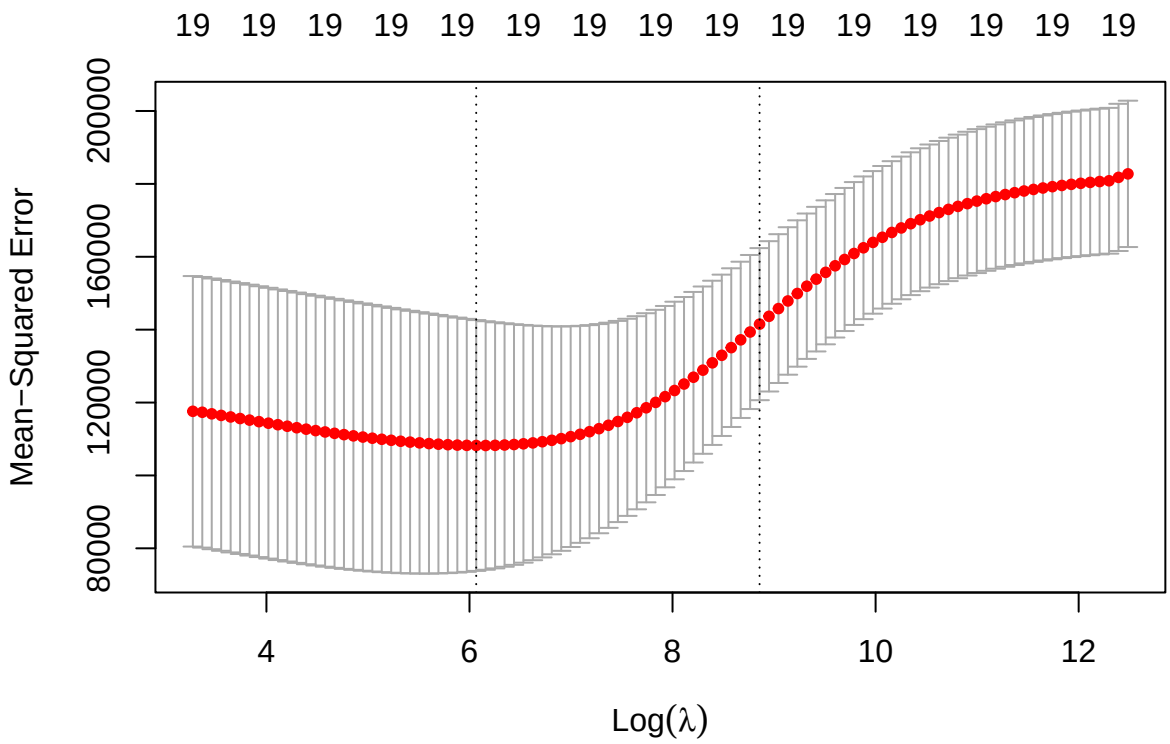
```
lm(y ~ x, subset = train)
```

```
##
## Call:
## lm(formula = y ~ x, subset = train)
##
## Coefficients:
## (Intercept)       xAtBat        xHits       xHmRun        xRuns         xRBI
##    274.0145      -0.3521      -1.6377       5.8145       1.5424       1.1243
##      xWalks       xYears      xCAtBat       xCHits      xCHmRun       xCRuns
##      3.7287     -16.3773      -0.6412       3.1632       3.4008      -0.9739
##       xCRBI      xCWalks     xLeagueN   xDivisionW     xPutOuts     xAssists
##     -0.6005       0.3379     119.1486    -144.0831       0.1976       0.6804
##     xErrors  xNewLeagueN
##     -4.7128     -71.0951
```

```
predict(ridge.mod, s = 0, exact = T, type = "coefficients",
        x = x[train, ], y = y[train])[1:20, ]
```

```
##  (Intercept)        AtBat         Hits        HmRun         Runs          RBI
##  274.0200994   -0.3521900   -1.6371383    5.8146692    1.5423361    1.1241837
##        Walks        Years       CAtBat        CHits       CHmRun        CRuns
##    3.7288406  -16.3795195   -0.6411235    3.1629444    3.4005281   -0.9739405
##         CRBI       CWalks      LeagueN    DivisionW      PutOuts      Assists
##   -0.6003976    0.3378422  119.1434637 -144.0853061    0.1976300    0.6804200
##       Errors   NewLeagueN
##   -4.7127879  -71.0898914
```

```
cv.out <- cv.glmnet(x[train, ], y[train], alpha = 0)
plot(cv.out)
```




```
bestlam <- cv.out$lambda.min

bestlam
```

```
## [1] 431.0623
```

```
ridge.pred <- predict(ridge.mod, s = bestlam,
                      newx = x[test, ])

mean((ridge.pred - y[test])^2)
```

```
## [1] 138796
```

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

```
##  (Intercept)        AtBat         Hits        HmRun         Runs          RBI
##  24.88203033   0.09571573   0.77893014   0.85488974   1.02412478   0.87575813
##        Walks        Years       CAtBat        CHits       CHmRun        CRuns
##   1.51400163   1.93680144   0.01128321   0.05331300   0.38052103   0.10651815
##         CRBI       CWalks      LeagueN    DivisionW      PutOuts      Assists
##   0.11215778   0.06298811  18.94995835 -70.27086710   0.14893846   0.02329106
##       Errors   NewLeagueN
##  -1.09701573   9.45875165
```