Homework 5

Chance Kang / A13605546 / csk025@ucsd.edu
Jayden Kim / A16271107 / s0k003@ucsd.edu
Sia Sheth / A16357789 / snsheth@ucsd.edu
Math 189
Spring 2023

Conceptual Question

1. Ridge regression adds a penalty term to the objective function of the linear regression model, which shrinks the coefficients towards zero and reduces their variance. Hence, ridge regression can be used to reduce the variance of coefficients where standard linear regression becomes unstable. A scenario would be where the sample size is too small so that the standard linear regression may have high variance. In such scenario, the ridge regression may assist in reducing variance and improving overall stablility of the model.
2. Contrast to the scenario given in the last question, a large sample size may be an indicator to avoid utilizing ridge regression. The variance of the estimates of the coefficients is likley to be small using standard linear regression. Hence, ridge regression may lead to biased estiates of the coefficient if the sample size is large. The ridge regression also limits one from assessing relationship between predictor and response because of the penalty term.

Application Question

```
library(ISLR2)
library(glmnet)
```

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

```
## Loaded glmnet 4.1-7
```

```
data(Hitters)
```

3.

a.

```
data.frame <- subset(Hitters, select = -c(League, Division, NewLeague))
data.frame <- data.frame[complete.cases(data.frame),]
dim(data.frame)
```

```
## [1] 263  17
```

b.

```
set.seed(123)
train <- sample(1:nrow(data.frame), nrow(data.frame)*0.8)
test = (-train)
```

c.

```
lr <- lm(Salary ~ ., data = data.frame, subset = train)
summary(lr)
```

```
##
## Call:
## lm(formula = Salary ~ ., data = data.frame, subset = train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -789.6 -179.9  -36.2  140.2 1915.8
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 174.45475   94.28494   1.850  0.06580 .
## AtBat        -1.81331    0.76001  -2.386  0.01800 *
## Hits          5.12107    2.92236   1.752  0.08130 .
## HmRun        -5.02689    6.92000  -0.726  0.46846
## Runs         -1.14996    3.44698  -0.334  0.73903
## RBI           2.78032    3.00883   0.924  0.35661
## Walks         6.37394    2.13035   2.992  0.00313 **
## Years       -13.27666   14.31453  -0.927  0.35483
## CAtBat       -0.28595    0.15817  -1.808  0.07218 .
## CHits         0.75525    0.81379   0.928  0.35453
## CHmRun        1.57799    1.88372   0.838  0.40323
## CRuns         1.37818    0.90003   1.531  0.12735
## CRBI          0.22551    0.84573   0.267  0.79002
## CWalks       -0.63479    0.39721  -1.598  0.11165
## PutOuts       0.21738    0.08754   2.483  0.01388 *
## Assists       0.51573    0.25583   2.016  0.04520 *
## Errors       -6.85930    4.92600  -1.392  0.16538
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 316.8 on 193 degrees of freedom
## Multiple R-squared:  0.5819, Adjusted R-squared:  0.5472
## F-statistic: 16.79 on 16 and 193 DF,  p-value: < 2.2e-16
```

d.

```
sqrt(mean((data.frame$Salary[test] - predict(lr, data.frame[test,]))^2))
```

```
## [1] 373.854
```

e. The root mean squared error is larger than the residual standard error. We did expect the root mean squared error to be larger, for the root mean squared error is the square root of the variance of the residuals whereas, the residual standard error is the square root of the residual sum of squares divided by the residual degrees of freedom. Hence, the root mean squared error takes bias and variance into account whereas the residual standard error considers the variance of the residuals. Such indicates an overfitting.

4.

```
x <- model.matrix(Salary ~ ., data.frame)[, -1]
y <- data.frame$Salary
```

a.

```
lasso.mod <- glmnet(x[train, ], y[train], alpha = 1)

set.seed(123)
cv.out <- cv.glmnet(x[train,], y[train], alpha=1)
bestlam<-cv.out$lambda.min
bestlam
```

```
## [1] 11.3676
```

Lambda chosen (bestlam) is the smallest cross-validation error.

b.

```
out <- glmnet(x, y, alpha = 1)
lasso.coef <- predict(out, type = "coefficients",
    s = bestlam)[1:17, ]
lasso.coef
```

```
##  (Intercept)        AtBat         Hits        HmRun         Runs          RBI
## -57.87962606   0.00000000   2.01735224   0.00000000   0.00000000   0.00000000
##        Walks        Years       CAtBat        CHits       CHmRun        CRuns
##   2.32400645   0.00000000   0.00000000   0.00000000   0.05757738   0.25313288
##         CRBI       CWalks      PutOuts      Assists       Errors
##   0.36566680   0.00000000   0.23677761   0.00000000  -0.50656487
```

If a variable gets zeroes, it means the variable is considered not significant and excluded from the model. Hence, the zeroed variables have no effect on the response variable of the model.

c.

```
lasso.pred <- predict(lasso.mod, s = bestlam,
    newx = x[test, ])
sqrt(mean((lasso.pred - y[test])^2))
```

```
## [1] 355.8825
```

5.

a.

```
ridge.mod <- glmnet(x, y, alpha = 0)
set.seed(123)
cv.out <- cv.glmnet(x[train,], y[train], alpha = 0)
bestlam<-cv.out$lambda.min
bestlam
```

```
## [1] 29.49919
```

As before, lambda chosen (bestlam) is the smallest cross-validation error.

b.

```
out <- glmnet(x, y, alpha = 0)
ridge.coef <- predict(out, type = "coefficients",
    s = bestlam)[1:17, ]
ridge.coef
```

```
##  (Intercept)        AtBat         Hits        HmRun         Runs          RBI
## 26.380055513 -0.694390246  2.633989400 -2.596893892  1.346924422  1.219215702
##        Walks        Years       CAtBat        CHits       CHmRun        CRuns
##  3.211205839 -8.412491370 -0.001611116  0.125378442  0.670698590  0.296284294
```

```
##         CRBI       CWalks      PutOuts      Assists       Errors
##  0.236480976 -0.222254359  0.269365801  0.166423971 -3.390186140
```

```
lr$coefficients
```

```
## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
## 174.4547466  -1.8133053   5.1210696  -5.0268864  -1.1499567   2.7803192
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##   6.3739376 -13.2766632  -0.2859545   0.7552524   1.5779930   1.3781755
##        CRBI      CWalks     PutOuts     Assists      Errors
##   0.2255121  -0.6347937   0.2173825   0.5157254  -6.8592952
```

Both of the models have relatively high positive coefficient associated to the variables Intercept, Hits, and Walks. Aside from the variable Runs, the signs of the coefficient of both models match. The maginitude of coefficients of Ridge models are relatively lower than the ones in simple linear regression.

c.

```
ridge.pred <- predict(ridge.mod, s = bestlam,
    newx = x[test, ])
sqrt(mean((ridge.pred - y[test])^2))
```

```
## [1] 307.6064
```

6.

```
sd(data.frame$Salary)
```

```
## [1] 451.1187
```

All models' RMSEs are below the standard deviation of the variable Salary from actual. However, using the LASSO model seems adequate if the purpose of the model is to find out which variables are most important for predicting a players salary. Such is because LASSO model excludes variables by setting its coefficients to 0, allowing feature selection. Hence, it is easier to deduce which variables contribute more to the Salary if LASSO model were to be used.

Contribution

Our group homework process goes as the following:
1. Each member attempts to complete the homework
2. Compare and discuss the answers
3. Complete a finalized version to submit

All members have contributed about the equal amount to complete this homework.