Regularized linear regression models - Ridge and Lasso regression

In this exercise, we will predict the number of applications received using the other variables in the College data set.

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.6.3
```

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

```
## Loaded glmnet 4.1-1
```

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.6.3
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
### dataset from the ISLR package
df <- College
df <- na.omit(df)
names(df)
```

```
##  [1] "Private"     "Apps"        "Accept"      "Enroll"      "Top10perc"
##  [6] "Top25perc"   "F.Undergrad" "P.Undergrad" "Outstate"    "Room.Board"
## [11] "Books"       "Personal"    "PhD"         "Terminal"    "S.F.Ratio"
## [16] "perc.alumni" "Expend"      "Grad.Rate"
```

```
### create training and test dataset
set.seed(1000) # allows reproducibilty
index <- sample(1:nrow(df),0.8*nrow(df)) # use random sample (80%) as training data
train <- df[index,] # training dataset
test <- df[-index,] # test dataset
```

```
### scale numeric features
cols <- c("Private","Accept","Enroll","Top10perc","Top25perc",
          "F.Undergrad","P.Undergrad","Outstate","Room.Board",
          "Books","Personal","PhD","Terminal","S.F.Ratio",
          "perc.alumni","Expend","Grad.Rate")
pre_proc <- preProcess(train[,cols], method =c("center", "scale"))
train[,cols] <- predict(pre_proc,train[,cols])
test[,cols] <- predict(pre_proc, test[,cols])
summary(train) # confirm that the mean of the predictor variables is zero
```

```
##  Private        Apps           Accept            Enroll
##  No :165   Min.   :   81   Min.   :-0.7706   Min.   :-0.7728
##  Yes:456   1st Qu.:  713   1st Qu.:-0.5735   1st Qu.:-0.5752
##            Median : 1561   Median :-0.3601   Median :-0.3693
```

```
##            Mean   : 3002   Mean   : 0.0000   Mean   : 0.0000
##            3rd Qu.: 3580   3rd Qu.: 0.1197   3rd Qu.: 0.1074
##            Max.   :48094   Max.   : 9.5826   Max.   : 5.8612
##    Top10perc          Top25perc         F.Undergrad        P.Undergrad
##  Min.   :-1.5080   Min.   :-2.3556   Min.   :-0.71486   Min.   :-0.6571
##  1st Qu.:-0.7207   1st Qu.:-0.7520   1st Qu.:-0.54761   1st Qu.:-0.5854
##  Median :-0.2145   Median :-0.1005   Median :-0.40283   Median :-0.3685
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.4603   3rd Qu.: 0.7012   3rd Qu.: 0.04455   3rd Qu.: 0.1395
##  Max.   : 3.8347   Max.   : 2.2046   Max.   : 5.60266   Max.   : 7.7710
##     Outstate          Room.Board          Books             Personal
##  Min.   :-2.0113   Min.   :-2.3352   Min.   :-2.7029   Min.   :-1.5697
##  1st Qu.:-0.7951   1st Qu.:-0.7204   1st Qu.:-0.5789   1st Qu.:-0.7098
##  Median :-0.1203   Median :-0.1452   Median :-0.2789   Median :-0.1851
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.6312   3rd Qu.: 0.6568   3rd Qu.: 0.3211   3rd Qu.: 0.4854
##  Max.   : 2.7744   Max.   : 2.7860   Max.   :10.7612   Max.   : 7.9771
##       PhD              Terminal          S.F.Ratio         perc.alumni
##  Min.   :-4.0514   Min.   :-3.7711   Min.   :-3.0163   Min.   :-1.8247
##  1st Qu.:-0.6338   1st Qu.:-0.5955   1st Qu.:-0.6671   1st Qu.:-0.7753
##  Median : 0.1119   Median : 0.1478   Median :-0.1189   Median :-0.1296
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.7954   3rd Qu.: 0.8234   3rd Qu.: 0.6381   3rd Qu.: 0.6776
##  Max.   : 1.8517   Max.   : 1.3640   Max.   : 3.5877   Max.   : 3.3413
##      Expend           Grad.Rate
##  Min.   :-1.1938   Min.   :-3.271345
##  1st Qu.:-0.5442   1st Qu.:-0.714646
##  Median :-0.2363   Median :-0.001149
##  Mean   : 0.0000   Mean   : 0.000000
##  3rd Qu.: 0.2092   3rd Qu.: 0.712348
##  Max.   : 8.4994   Max.   : 2.079885
```

Linear regression model

```
df.lm <- lm(Apps~., data = train)
summary(df.lm)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5041.0  -396.9   -22.7   295.0  7444.4
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3268.386    117.018  27.931  < 2e-16 ***
## PrivateYes  -362.861    149.276  -2.431  0.01536 *
## Accept      4155.578    108.175  38.415  < 2e-16 ***
## Enroll      -936.076    185.155  -5.056 5.70e-07 ***
## Top10perc    674.121    108.159   6.233 8.62e-10 ***
## Top25perc   -192.270     97.427  -1.973  0.04890 *
## F.Undergrad  255.720    170.665   1.498  0.13456
## P.Undergrad   63.258     57.755   1.095  0.27384
```

```
## Outstate    -422.843     83.991  -5.034 6.34e-07 ***
## Room.Board   167.751     58.699   2.858  0.00441 **
## Books         -7.064     43.253  -0.163  0.87032
## Personal      55.965     46.575   1.202  0.22999
## PhD         -144.835     91.271  -1.587  0.11306
## Terminal     -32.592     89.922  -0.362  0.71714
## S.F.Ratio     64.001     58.003   1.103  0.27029
## perc.alumni   51.702     56.026   0.923  0.35647
## Expend       541.182     71.940   7.523 1.96e-13 ***
## Grad.Rate    145.460     56.706   2.565  0.01055 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1021 on 603 degrees of freedom
## Multiple R-squared:  0.9358, Adjusted R-squared:  0.934
## F-statistic: 517.3 on 17 and 603 DF,  p-value: < 2.2e-16
```

The linear regression model is given as: *Apps = 3268.4 - 362.7(Private[Yes]) + 4155.6(Accept) - 936.1(enroll) + 674.1(Top10perc) -192.3(Top25perc) - 422.8(Outstate) + 167.8(Room.Board) + 541.2(expend) + 145.5(Grad.Rate) + e*

```
## Predictions and performance of linear regression model
predictions <- predict(df.lm,test)
RMSE <- RMSE(predictions, test$Apps)
MSE <- RMSE(predictions, test$Apps)**2
data.frame(RMSE,MSE)
```

```
##       RMSE     MSE
## 1 1154.578 1333050
```

Regularized linear models

```
### generate training and test data sets
set.seed(1000)
x_train <- model.matrix(Apps~.,train)[,-1] #predictor variables
y_train <- train$Apps # response variables
x_test <- model.matrix(Apps~.,test)[,-1]
y_test <- test$Apps
```

Ridge regression

```
cv_ridge <- cv.glmnet(x_train, y_train, alpha = 0)
cv_ridge$lambda.min
```

```
## [1] 376.4657
```

```
ridge_model <- glmnet(x_train, y_train, alpha = 0, lambda = cv_ridge$lambda.min)
coef(ridge_model)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## (Intercept) 3333.51812
## PrivateYes  -451.56086
## Accept      2621.31766
## Enroll       356.81632
## Top10perc    351.70396
## Top25perc     34.90233
## F.Undergrad  332.25086
```

```
## P.Undergrad   74.56092
## Outstate    -116.93109
## Room.Board   219.99112
## Books         18.18758
## Personal      25.56750
## PhD          -56.71783
## Terminal     -75.86231
## S.F.Ratio     47.22635
## perc.alumni  -70.78744
## Expend       468.97075
## Grad.Rate    184.29180
```

The ridge regression model is given as: *Apps = 3001.9 + 2636.2(Accept) + 377.2(Enroll) + 342.2(Top10perc) + 35.1(Top25perc) + 369.8(F.Undergrad) + 86.4(P.Undergrad) - 181.7(Outstate) + 198.5(Room.Board) + 12.2(Books) + 31.4(Personal) - 33.7(PhD) - 54.6(Terminal) + 75.1(S.F.ratio) -85.5(perc.alumni) + 482.8(Expend) +175(Grad.Rate) + e*

```
## Predictions and performance of ridge regression model
predictions <- predict(ridge_model,x_test)
RMSE <- RMSE(predictions, y_test)
MSE <- RMSE(predictions, y_test)**2
data.frame(RMSE, MSE)
```

```
##       RMSE     MSE
## 1 1137.996 1295035
```

Lasso regression

```
cv_lasso <- cv.glmnet(x_train, y_train, alpha = 1)
cv_lasso$lambda.min
```

```
## [1] 20.5636
```

```
lasso_model <- glmnet(x_train, y_train, alpha = 1, lambda = cv_lasso$lambda.min)
coef(lasso_model)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept) 3213.18407
## PrivateYes  -287.68488
## Accept      3853.20287
## Enroll      -344.94137
## Top10perc    434.11553
## Top25perc      .
## F.Undergrad    .
## P.Undergrad   27.32058
## Outstate    -294.19118
## Room.Board   127.81755
## Books          .
## Personal      30.18076
## PhD          -88.31285
## Terminal     -36.26458
## S.F.Ratio     24.30726
## perc.alumni    .
## Expend       495.09082
## Grad.Rate     97.82362
```

The lasso regression model is given as: *Apps = 3001.9 + 3840(Accept) - 282.5(Enroll) + 416.8(Top10perc) +*

31.5(P.Undergrad) - 337.5(Outstate) + 112.7(Room.Board) + 31.1(Personal)- 65.9(PhD) - 21.3(Terminal) +39.7(S.F.Ratio) + 503.5(Expend) + 87.9(Grad.Rate) + e

```
## Predictions and performance of the lasso regression model
predictions <- predict(lasso_model,x_test)
RMSE <- RMSE(predictions, y_test)
MSE <- RMSE(predictions, y_test)**2
data.frame(RMSE, MSE)
```

```
##       RMSE     MSE
## 1 1157.458 1339710
```

The model performance assessment results show that ridge regression is the best moel compared to the linear and lasso regression models. This is because ridge regression does not discrad any of the predictor variables but rather reduces the coefficients of the predictor variables depending on their significance to the response.