BIOSTAT 274 Spring 2021 Homework 1

Due 11:59 PM 04/21/2020 (Submit to CCLE)

Zian ZHUANG

Remark. For **Computational Part**, please complete your answer in the **RMarkdown** file and summit the generated PDF and RMD files. Related packages have been loaded in setup.

Computational Part

1. (Model Selection, [ISL] 6.8, *25 pt*) In this exercise, we will generate simulated data, and will then use this data to perform model selection.

(a) Use the `rnorm` function to generate a predictor $\boldsymbol{X}$ of length $n = 100$, as well as a noise vector $\boldsymbol{\epsilon}$ of length $n = 100$.

```
set.seed(199609)
X <- rnorm(100)
epsilon <- rnorm(100)
```

(b) Generate a response vector $\boldsymbol{Y}$ of length $n = 100$ according to the model

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon,$$

where $\beta_0 = 3$, $\beta_1 = 2$, $\beta_2 = -3$, $\beta_3 = 0.3$.

```
b0 <- 3 ; b1 <- 2 ; b2 <- -3 ; b3 <- 0.3

Y <- b0 + b1*X + b2*X^2 + b3*X^3 + epsilon
```

(c) Use the `regsubsets` function from `leaps` package to perform best subset selection in order to choose the best model from the set of predictors $(X, X^2, \cdots, X^{10})$. What are the best models obtained according to C_p, BIC, and adjusted R^2, respectively? Show some plots to provide evidence for your answer, and report the coefficients of the best model obtained.

```
df <- data.frame(X,X^2,X^3,X^4,X^5,X^6,X^7,X^8,X^9,X^10,Y)
best_model <- regsubsets(Y ~ ., data = df, nvmax=10)
summary_model <- best_model %>% summary

#the best models according to C_p, BIC, and adjusted R^2
min.cp <- which.min(summary_model$cp)
min.cp
```

```
## [1] 3
```

```r
min.bic <- which.min(summary_model$bic)
min.bic
```

```
## [1] 3
```

```r
min.adjr2 <- which.max(summary_model$adjr2)
min.adjr2
```

```
## [1] 3
```

```r
# plots that show the best models
par(mfrow = c(2,2))
plot(summary_model$cp, xlab = "Number of Poly(X)",
     ylab ="Cp", type="l")
points(min.cp, summary_model$cp[min.cp], col = "red", pch = 4, lwd = 5)

plot(summary_model$bic, xlab = "Number of Poly(X)",
     ylab = "BIC", type="l")
points(min.bic, summary_model$bic[min.bic], col = "red", pch = 4, lwd = 5)

plot(summary_model$adjr2, xlab = "Number of Poly(X)",
     ylab = "Adjusted R^2", type="l")
points(min.adjr2, summary_model$adjr2[min.adjr2], col = "red", pch = 4, lwd = 5)

coef(best_model, min.cp)
```

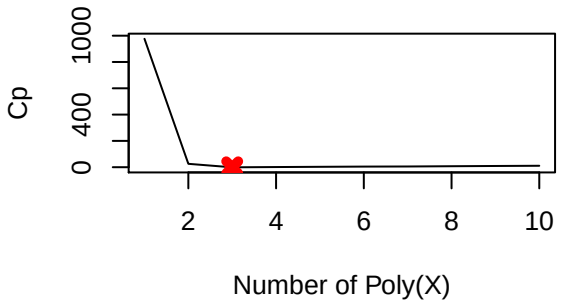
```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

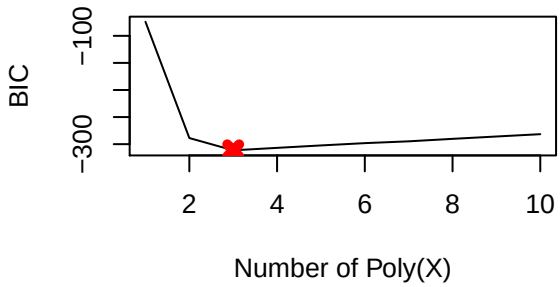
```r
coef(best_model, min.bic)
```

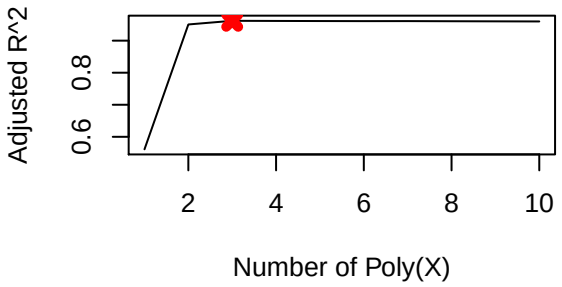
```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

```r
coef(best_model, min.adjr2)
```

```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```







(d) Repeat (c), using forward stepwise selection and also using backward stepwise selection. How does your answer compare to the results in (c)?

```
# forward stepwise selection.
best_model_f <- regsubsets(Y~., data=df, nvmax=10, method='forward')
summary_model_f <- best_model_f %>% summary

# refer to C_p, BIC, and adjusted R^2
min.cp <- which.min(summary_model_f$cp)
min.cp
```

```
## [1] 3
```

```
min.bic <- which.min(summary_model_f$bic)
min.bic
```

```
## [1] 3
```

```
min.adjr2 <- which.max(summary_model_f$adjr2)
min.adjr2
```

```
## [1] 3
```

```
coef(best_model, min.cp)
```

```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

```
coef(best_model, min.bic)
```

```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

```
coef(best_model, min.adjr2)
```

```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

```
# backward stepwise selection.
best_model_b = regsubsets(Y~., data=df, nvmax=10, method='backward')
summary_model_b <- best_model_f %>% summary

#the best models according to C_p, BIC, and adjusted R^2
min.cp <- which.min(summary_model_b$cp)
min.cp
```

```
## [1] 3
```

```
min.bic <- which.min(summary_model_b$bic)
min.bic
```

```
## [1] 3
```

```
min.adjr2 <- which.max(summary_model_b$adjr2)
min.adjr2
```

```
## [1] 3
```

```
coef(best_model, min.cp)
```

```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

```
coef(best_model, min.bic)
```

```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

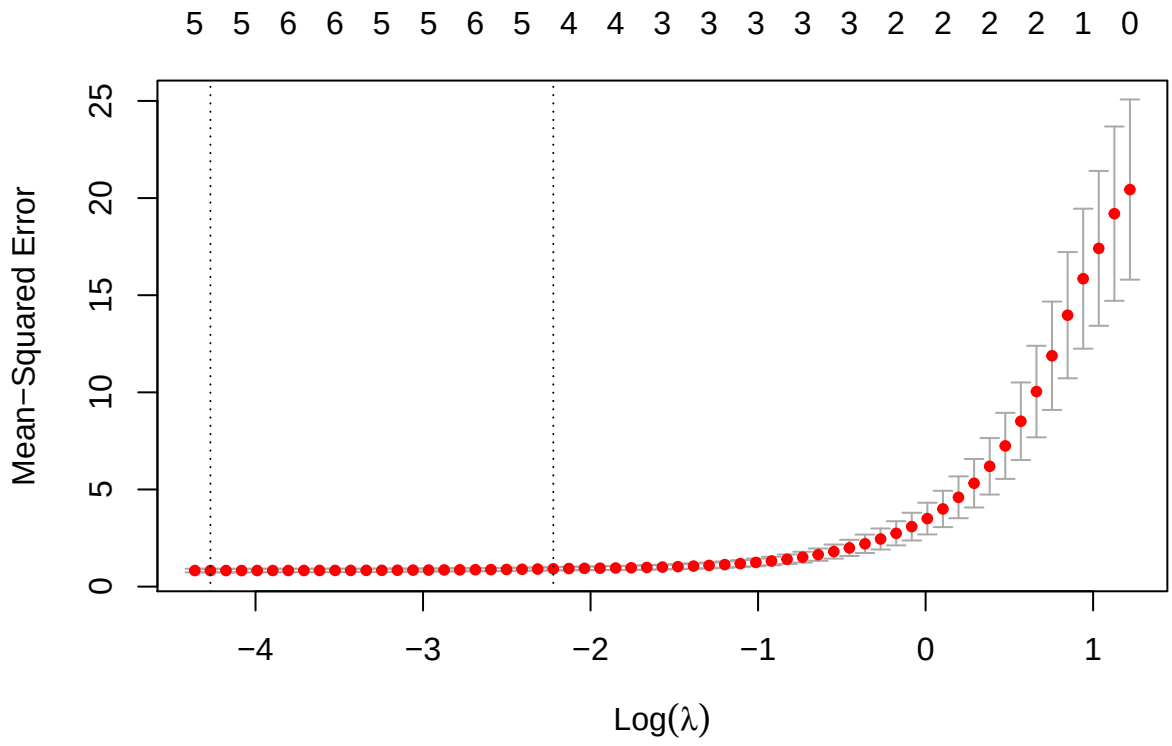
```
coef(best_model, min.adjr2)
```

```
##  (Intercept)            X          X.2          X.9
##  3.140245094  2.630517280 -3.063234290  0.001100575
```

We found that both forward and backward method provided same results as part (c).

(e) Now fit a LASSO model with `glmnet` function from `glmnet` package to the simulated data, again using $(X, X^2, \cdots, X^{10})$ as predictors. Use cross-validation to select the optimal value of λ. Create plots of the cross-validation error as a function of λ. Report the resulting coefficient estimates, and discuss the results obtained.

```
# Lasso model
lasso = glmnet(df[, -11]%>%as.matrix, df[, 11], alpha=1)
# cross-validation
cv.out = cv.glmnet(df[, -11]%>%as.matrix, df[, 11], alpha=1)
# cross-validation error
plot(cv.out)
```




```
# the optimal value of lambda
best_lam = cv.out$lambda.min
# Coefficients from lasso model with best lambda.
predict(lasso, s=best_lam, type="coefficients")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                            1
## (Intercept)  3.1110109901
## X            2.5502953100
## X.2         -3.0253733926
## X.3          0.0287526190
## X.4          .
## X.5          .
## X.6          .
## X.7          0.0023727142
## X.8          .
## X.9          0.0005832285
## X.10         .
```

According the results, we see that estimated coefficients of b1~b3 from lasso model with best lambda are largely consistent with the true value of b1~b3. Nevertheless, lasso model included X^7 and X^9 by mistake, with small coefficients.

(f) Now generate a response vector Y according to the model

$$Y = \beta_0 + \beta_7 X^7 + \epsilon,$$

where $\beta_7 = 7$, and perform best subset selection and the LASSO. Discuss the results obtained.

```
# generate y2
b7 <- 7
Y2 <- b0 + b7*X^7 + epsilon

df <- data.frame(X,X^2,X^3,X^4,X^5,X^6,X^7,X^8,X^9,X^10,Y2)

# best subset selection
best_model <- regsubsets(Y2 ~ ., data=df, nvmax=10)
summary_model <- summary(best_model)

# refer to C_p, BIC, and adjusted R^2
min.cp <- which.min(summary_model$cp)
min.bic <- which.min(summary_model$bic)
min.adjr2 <- which.max(summary_model$adjr2)

# check results
coef(best_model, min.cp)
```

```
## (Intercept)           X         X.5         X.7
##   3.0764616   0.4000575  -0.1098626   7.0172760
```

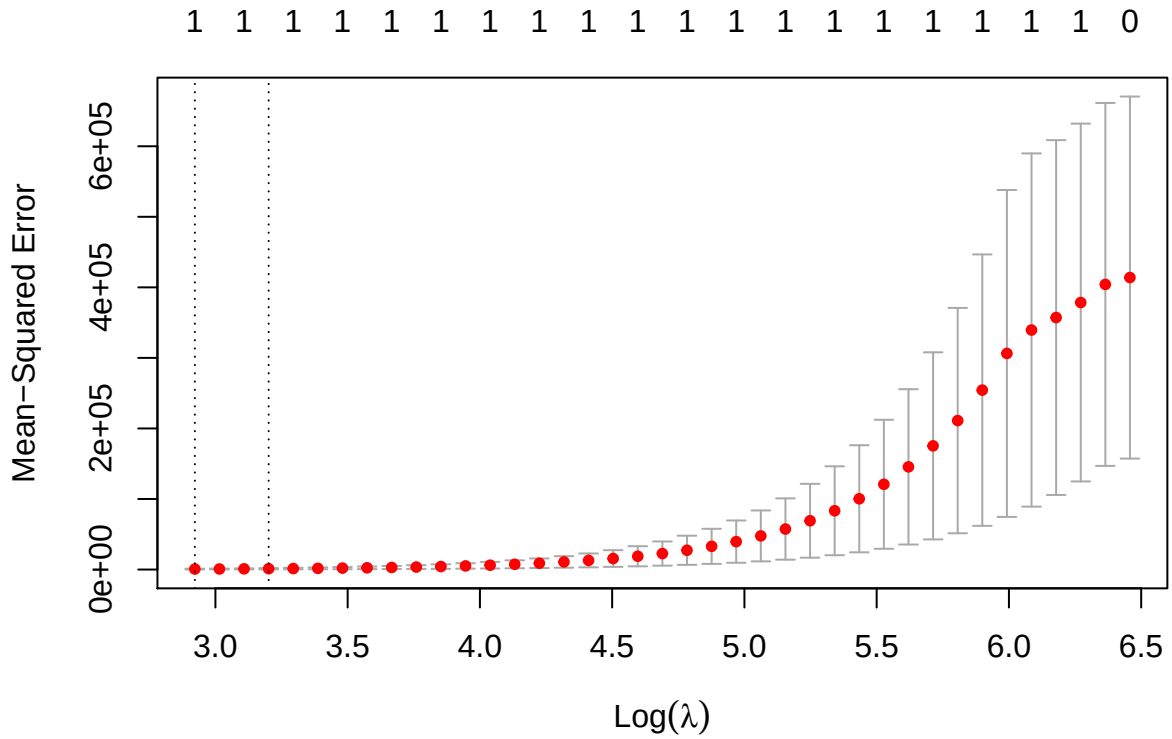
```
coef(best_model, min.bic)
```

```
## (Intercept)         X.7
##    3.092703    7.000839
```

```
coef(best_model, min.adjr2)
```

```
## (Intercept)           X         X.5         X.7
##   3.0764616   0.4000575  -0.1098626   7.0172760
```

```
# LASSO model
lasso = glmnet(df[, -11]%>%as.matrix, df[, 11], alpha=1)
# cross-validation
cv.out = cv.glmnet(df[, -11]%>%as.matrix, df[, 11], alpha=1)
# cross-validation error
plot(cv.out)
```




```
# the optimal value of lambda
best_lam = cv.out$lambda.min
# Coefficients from lasso model with best lambda.
predict(lasso, s=best_lam, type="coefficients")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) 4.506576
## X           .
## X.2         .
## X.3         .
## X.4         .
```

```
## X.5          .
## X.6          .
## X.7          6.796761
## X.8          .
## X.9          .
## X.10         .
```

According to the results, we found that the best subset selection (referring to the BIC value) and Lasso regression provides the relatively more accurate estimates than that of best subset selection (referring to C_p and adjusted R^2).

2. (Prediction, [ISL] 6.9, *20 pt*) In this exercise, we will predict the number of applications received (`Apps`) using the other variables in the `College` data set from `ISLR` package.

(a) Randomly split the data set into equal sized training set and test set (1:1).

```
data(College)

set.seed(199609)
train_id <- sample(dim(College)[1], dim(College)[1]/2)

train_set <- College[train_id, ]
test_set <- College[-train_id, ]
```

(b) Fit a linear model using least squares on the training set, and report the test error obtained.

```
# linear model
modl <- lm(Apps~., data = train_set)
summary(modl)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5178.9  -426.6   -21.1   326.7  8064.7
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -9.474e+02  6.275e+02  -1.510 0.131915
## PrivateYes  -5.111e+02  1.955e+02  -2.615 0.009292 **
## Accept       1.732e+00  4.983e-02  34.758  < 2e-16 ***
## Enroll      -9.015e-01  2.619e-01  -3.442 0.000643 ***
## Top10perc    4.058e+01  8.031e+00   5.053 6.83e-07 ***
## Top25perc   -8.851e+00  6.211e+00  -1.425 0.155032
## F.Undergrad -4.512e-02  5.089e-02  -0.887 0.375813
## P.Undergrad  1.081e-01  5.363e-02   2.016 0.044481 *
## Outstate    -1.191e-01  2.750e-02  -4.330 1.92e-05 ***
## Room.Board   1.427e-01  7.034e-02   2.029 0.043152 *
## Books        4.959e-01  4.088e-01   1.213 0.225879
## Personal     4.192e-03  8.420e-02   0.050 0.960316
```

```
## PhD         -1.077e+01  7.351e+00  -1.465 0.143682
## Terminal     5.666e+00  8.112e+00   0.698 0.485316
## S.F.Ratio    2.220e+01  2.170e+01   1.023 0.307091
## perc.alumni  8.559e+00  5.910e+00   1.448 0.148396
## Expend       6.065e-02  1.501e-02   4.041 6.47e-05 ***
## Grad.Rate    7.809e+00  4.091e+00   1.909 0.057033 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1025 on 370 degrees of freedom
## Multiple R-squared:  0.9422, Adjusted R-squared:  0.9396
## F-statistic:   355 on 17 and 370 DF,  p-value: < 2.2e-16
```

```
pred1 <- predict(modl, newdata = test_set[,-2], type = "response")

# test error
MAE(pred1, College$Apps[-train_id])
```

```
## [1] 622.9599
```

```
MSE(pred1, College$Apps[-train_id])
```

```
## [1] 1287249
```

```
RMSE(pred1, College$Apps[-train_id])
```

```
## [1] 1134.57
```

```
R2(pred1, College$Apps[-train_id], form = "traditional")
```

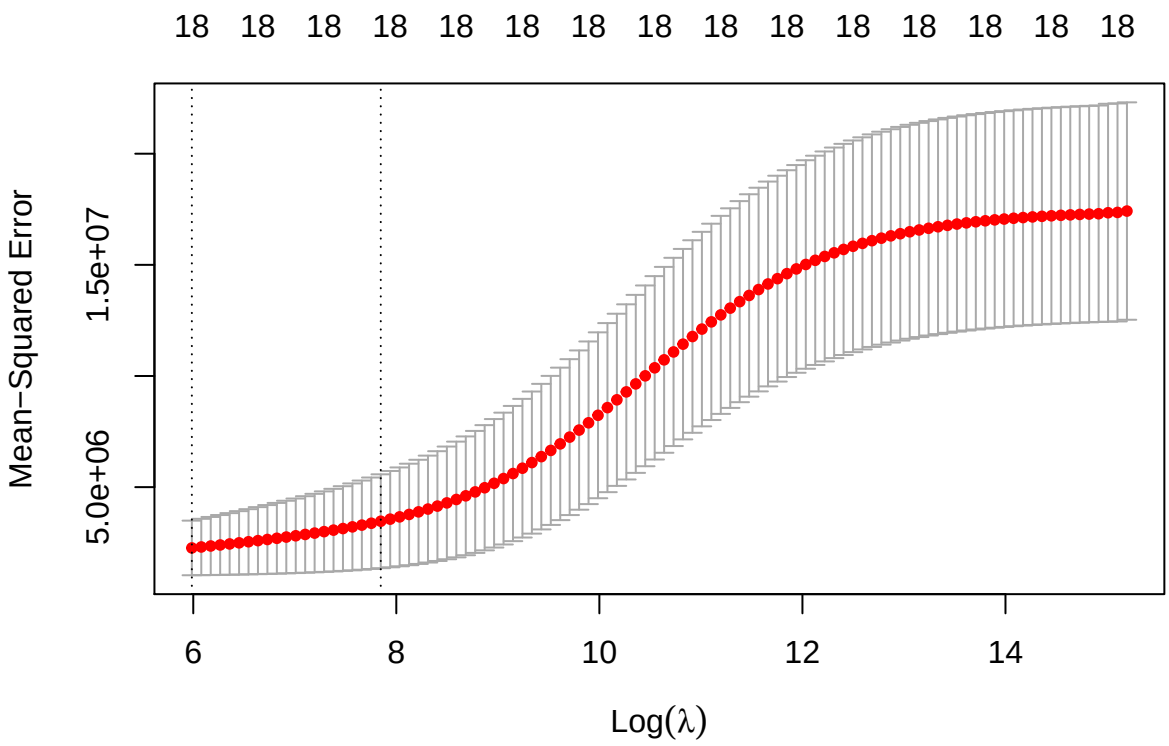
```
## [1] 0.8974923
```

```
err1 <- data.frame(MAE = MAE(pred1, College$Apps[-train_id]),
                   MSE = MSE(pred1, College$Apps[-train_id]),
                   RMSE = RMSE(pred1, College$Apps[-train_id]),
                   R2 = R2(pred1, College$Apps[-train_id], form = "traditional"))
```

(c) Fit a ridge regression model on the training set, with λ chosen by 5-fold cross-validation. Report the test error obtained.

```
# prepare data
train_set <- model.matrix(Apps~.-1, data=College[train_id, ])
test_set <- model.matrix(Apps~.-1, data=College[-train_id, ])

# ridge regression with cross-validation
modr_cv <- cv.glmnet(train_set %>% as.matrix,
                     College$Apps[train_id] %>% as.matrix,
                     nfolds = 5, alpha = 0)
plot(modr_cv)
```




```
lambda <- modr_cv$lambda.min
pred2 <- predict(modr_cv, s = lambda, newx = test_set)
summary(modr_cv)
```

```
##            Length Class  Mode
## lambda     100    -none- numeric
## cvm        100    -none- numeric
## cvsd       100    -none- numeric
## cvup       100    -none- numeric
## cvlo       100    -none- numeric
## nzero      100    -none- numeric
## call         5    -none- call
## name         1    -none- character
## glmnet.fit  12    elnet  list
## lambda.min   1    -none- numeric
## lambda.1se   1    -none- numeric
## index        2    -none- numeric
```

```
# test error
MAE(pred2, College$Apps[-train_id])
```

```
## [1] 631.6212
```

```
MSE(pred2, College$Apps[-train_id])
```

```
## [1] 1173883
```

```
RMSE(pred2, College$Apps[-train_id])
```

```
## [1] 1083.459
```

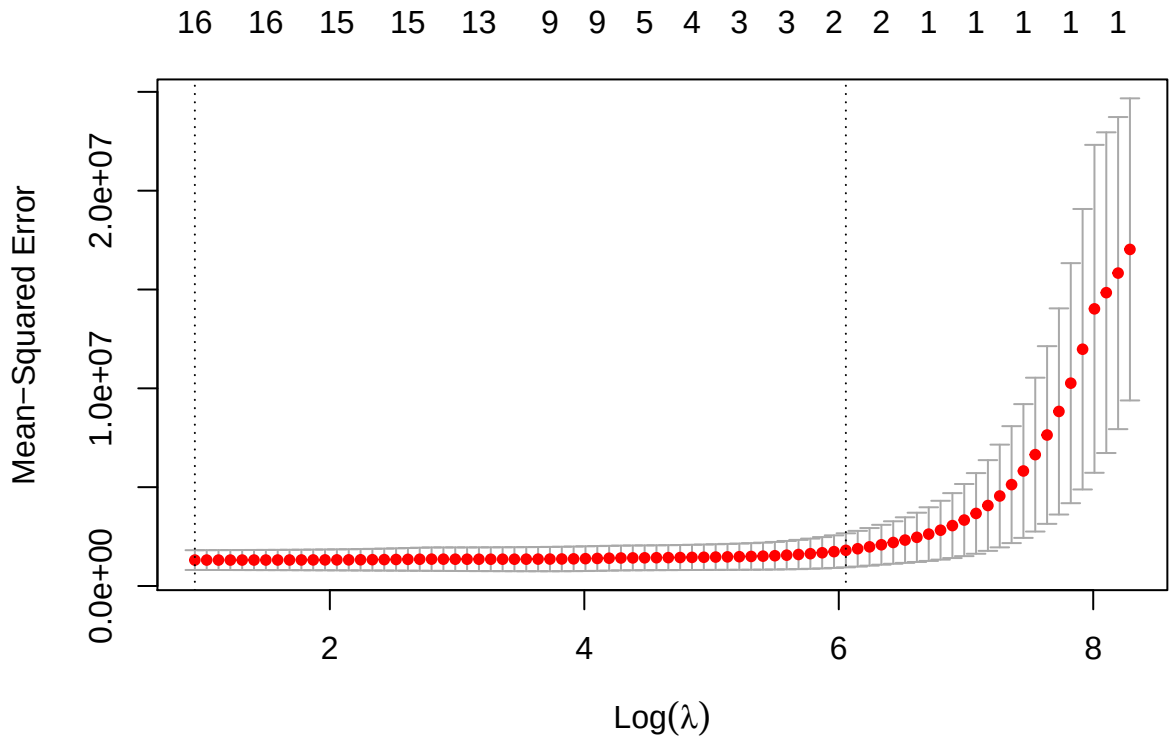
```
R2(pred2, College$Apps[-train_id], form = "traditional")
```

```
## [1] 0.90652
```

```
err2 <- data.frame(MAE = MAE(pred2, College$Apps[-train_id]),
                   MSE = MSE(pred2, College$Apps[-train_id]),
                   RMSE = RMSE(pred2, College$Apps[-train_id]),
                   R2 = R2(pred2, College$Apps[-train_id], form = "traditional"))
```

(d) Fit a LASSO model on the training set, with λ chosen by 5-fold cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates.

```
# LASSO regression with cross-validation
modr_cv <- cv.glmnet(train_set %>% as.matrix,
                     College$Apps[train_id] %>% as.matrix,
                     nfolds = 5, alpha = 1)
plot(modr_cv)
```



```
lambda <- modr_cv$lambda.min
pred3 <- predict(modr_cv, s = lambda, newx = test_set)

# test error
MAE(pred3, College$Apps[-train_id])
```

```
## [1] 612.0778
```

```
MSE(pred3, College$Apps[-train_id])
```

```
## [1] 1272646
```

```
RMSE(pred3, College$Apps[-train_id])
```

```
## [1] 1128.116
```

```
R2(pred3, College$Apps[-train_id], form = "traditional")
```

```
## [1] 0.8986552
```

```
err3 <- data.frame(MAE = MAE(pred3, College$Apps[-train_id]),
                   MSE = MSE(pred3, College$Apps[-train_id]),
                   RMSE = RMSE(pred3, College$Apps[-train_id]),
                   R2 = R2(pred3, College$Apps[-train_id], form = "traditional"))

# the number of non-zero coefficient estimates (intercept term included)
coef <- predict(modr_cv, s=best_lam, type="coefficients")
coef[which(coef != 0)] %>% length
```

```
## [1] 14
```

(e) Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these three approaches?

```
data.frame(Model = c("Linear", "Ridge", "Lasso"),
           rbind(err1, err2, err3) %>% round(2))
```

```
##    Model    MAE     MSE    RMSE   R2
## 1 Linear 622.96 1287249 1134.57 0.90
## 2  Ridge 631.62 1173883 1083.46 0.91
## 3  Lasso 612.08 1272646 1128.12 0.90
```

We used `MAE`, `MSE`, `RMSE` and `R-squared` to access the model test error.

- MAE (Mean absolute error) represents the difference between the original and predicted values extracted by averaged the absolute difference over the data set.
- MSE (Mean Squared Error) represents the difference between the original and predicted values extracted by squared the average difference over the data set.

- RMSE (Root Mean Squared Error) is the error rate by the square root of MSE.
- R-squared (Coefficient of determination) represents the coefficient of how well the values fit compared to the original values. The value from 0 to 1 interpreted as percentages. The higher the value is, the better the model is. source

Generally speaking, all of models showed high R2 ($\geqslant 0.9$), indicating that the high accuracy of the predictions. According to four test errors, we found that Ridge regression provides the lowerst `MSE`, `RMSE` and the highest `R-squared`, which suggested that Ridge regression performed the best. Nevertheless, the test errors differences are very small.