ADA 442: Statistical Learning

Homework 2: Comparison of different linear models

<Furkan ÖZELGE (14758028780)>

01 Mayıs, 2022

Table of Contents

ABOUT REPRODUCIBILITY 1
HOMEWORK 2 1
SOLUTIONS 27
References 27

ABOUT REPRODUCIBILITY

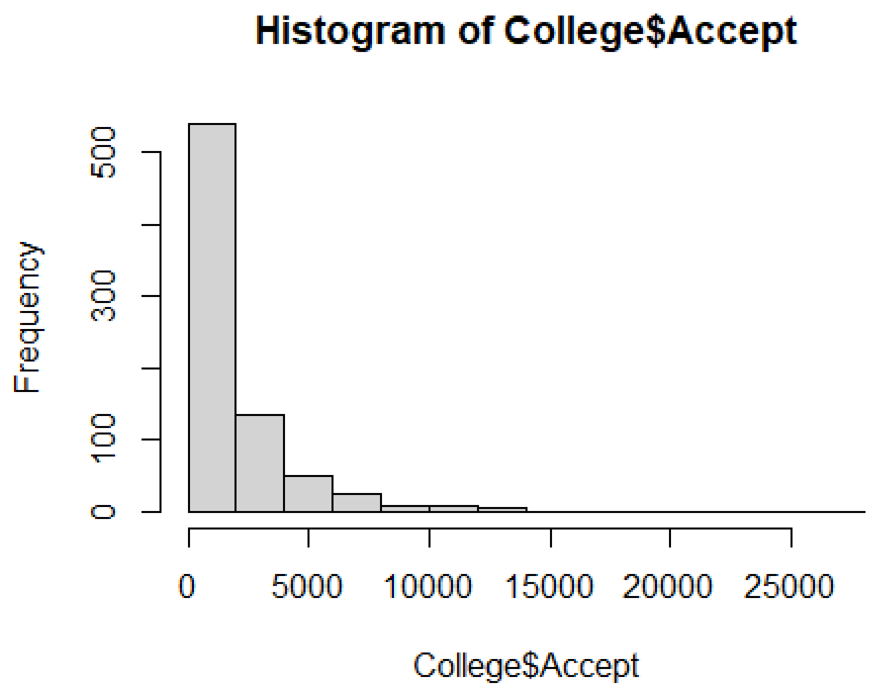
```
# FOR REPRODUCIBILITY
set.seed(28780)
# ALERT: YOU NEED TO USE YOUR STUDENT NUMBER LAST 5 DIGITS
# HERE instead of 442 MAKE SURE THAT YOU CHANGED
# BEFORE STARTING TO YOUR ANALYSIS

# THIS PART IS IMPORTANT FOR SPLITTING YOUR DATA so that
# EACH PERSON HAS DIFFERENT SPLITS AND EVEN IF YOU USE
# THE SAME DATA SET YOUR RESULTS WILL BE A BIT DIFFERENT

# ALWAYS USE 80% (TRAINING) - 20% (TESTING) SPLIT RULE in YOUR ANALYSIS

# BUT MOST IMPORTANTLY WHEN I RUN YOUR .Rmd file in my computer,
# I NEED TO SEE THE SAME RESULTS THAT YOU MENTIONED IN YOUR PDF REPORT !
```

HOMEWORK 2

You should aim to use this section to run different linear models including Ridge and Lasso in `R` and interpret the corresponding output. You will need to conduct such analyses on the available data set below (HINT: Try to focus on fitting a model to explain **Accept** variable (Number of applications accepted))

```
#install.packages("ISLR2")
library(ISLR2)
data("College")
# head(College)
summary(College) # Ranges of predictors are different !!!
```

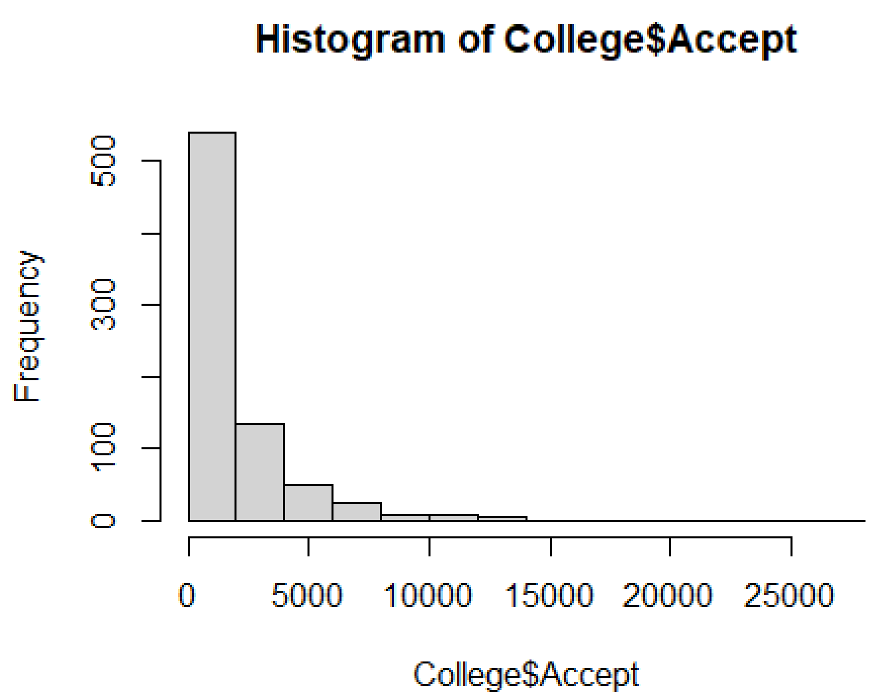
```
##  Private        Apps           Accept          Enroll       Top10perc
##  No :212   Min.   :   81   Min.   :   72   Min.   :  35   Min.   : 1.00
##  Yes:565   1st Qu.:  776   1st Qu.:  604   1st Qu.: 242   1st Qu.:15.00
##            Median : 1558   Median : 1110   Median : 434   Median :23.00
##            Mean   : 3002   Mean   : 2019   Mean   : 780   Mean   :27.56
##            3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902   3rd Qu.:35.00
##            Max.   :48094   Max.   :26330   Max.   :6392   Max.   :96.00
##    Top25perc      F.Undergrad     P.Undergrad        Outstate
##  Min.   :  9.0   Min.   :  139   Min.   :    1.0   Min.   : 2340
##  1st Qu.: 41.0   1st Qu.:  992   1st Qu.:   95.0   1st Qu.: 7320
##  Median : 54.0   Median : 1707   Median :  353.0   Median : 9990
##  Mean   : 55.8   Mean   : 3700   Mean   :  855.3   Mean   :10441
##  3rd Qu.: 69.0   3rd Qu.: 4005   3rd Qu.:  967.0   3rd Qu.:12925
##  Max.   :100.0   Max.   :31643   Max.   :21836.0   Max.   :21700
##    Room.Board       Books           Personal         PhD
##  Min.   :1780   Min.   :  96.0   Min.   : 250   Min.   :  8.00
##  1st Qu.:3597   1st Qu.: 470.0   1st Qu.: 850   1st Qu.: 62.00
##  Median :4200   Median : 500.0   Median :1200   Median : 75.00
##  Mean   :4358   Mean   : 549.4   Mean   :1341   Mean   : 72.66
##  3rd Qu.:5050   3rd Qu.: 600.0   3rd Qu.:1700   3rd Qu.: 85.00
##  Max.   :8124   Max.   :2340.0   Max.   :6800   Max.   :103.00
##     Terminal       S.F.Ratio      perc.alumni        Expend
##  Min.   : 24.0   Min.   : 2.50   Min.   : 0.00   Min.   : 3186
##  1st Qu.: 71.0   1st Qu.:11.50   1st Qu.:13.00   1st Qu.: 6751
##  Median : 82.0   Median :13.60   Median :21.00   Median : 8377
##  Mean   : 79.7   Mean   :14.09   Mean   :22.74   Mean   : 9660
##  3rd Qu.: 92.0   3rd Qu.:16.50   3rd Qu.:31.00   3rd Qu.:10830
##  Max.   :100.0   Max.   :39.80   Max.   :64.00   Max.   :56233
##    Grad.Rate
##  Min.   : 10.00
##  1st Qu.: 53.00
##  Median : 65.00
##  Mean   : 65.46
##  3rd Qu.: 78.00
##  Max.   :118.00

# response dist.
hist(College$Accept)
```



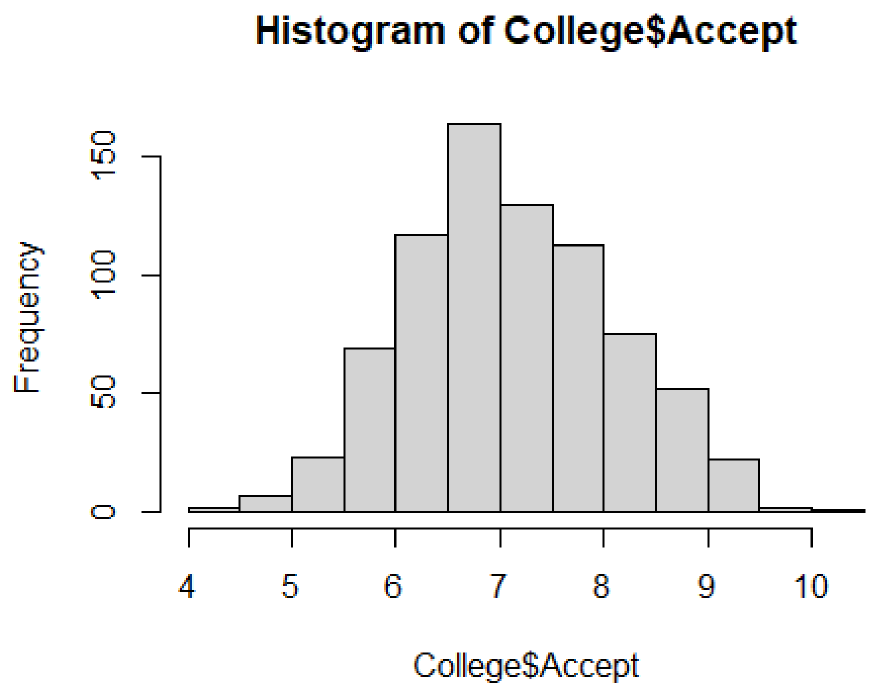
1. Consider any necessary **data-preprocessing process** on the data set (**HINT:** Ranges of predictors are different and the response variable should be approximately normal !!!)

```
#package install first.
#install.packages("ISLR2")
#load library
library(ISLR2)
#taking data and summary and take its histogram
data("College")
summary(College) # Ranges of predictors are different !!!

##  Private        Apps           Accept          Enroll       Top10perc
##  No :212   Min.   :   81   Min.   :   72   Min.   :  35   Min.   : 1.00
##  Yes:565   1st Qu.:  776   1st Qu.:  604   1st Qu.: 242   1st Qu.:15.00
##             Median : 1558   Median : 1110   Median : 434   Median :23.00
##             Mean   : 3002   Mean   : 2019   Mean   : 780   Mean   :27.56
##             3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902   3rd Qu.:35.00
##             Max.   :48094   Max.   :26330   Max.   :6392   Max.   :96.00
##    Top25perc      F.Undergrad     P.Undergrad         Outstate
##  Min.   :  9.0   Min.   :  139   Min.   :    1.0   Min.   : 2340
##  1st Qu.: 41.0   1st Qu.:  992   1st Qu.:   95.0   1st Qu.: 7320
##  Median : 54.0   Median : 1707   Median :  353.0   Median : 9990
##  Mean   : 55.8   Mean   : 3700   Mean   :  855.3   Mean   :10441
##  3rd Qu.: 69.0   3rd Qu.: 4005   3rd Qu.:  967.0   3rd Qu.:12925
##  Max.   :100.0   Max.   :31643   Max.   :21836.0   Max.   :21700
##    Room.Board       Books           Personal          PhD
##  Min.   :1780   Min.   :  96.0   Min.   : 250   Min.   :  8.00
```

```
##  1st Qu.:3597   1st Qu.: 470.0   1st Qu.: 850   1st Qu.: 62.00
##  Median :4200   Median : 500.0   Median :1200   Median : 75.00
##  Mean   :4358   Mean   : 549.4   Mean   :1341   Mean   : 72.66
##  3rd Qu.:5050   3rd Qu.: 600.0   3rd Qu.:1700   3rd Qu.: 85.00
##  Max.   :8124   Max.   :2340.0   Max.   :6800   Max.   :103.00
##     Terminal        S.F.Ratio       perc.alumni        Expend
##  Min.   : 24.0   Min.   : 2.50   Min.   : 0.00   Min.   : 3186
##  1st Qu.: 71.0   1st Qu.:11.50   1st Qu.:13.00   1st Qu.: 6751
##  Median : 82.0   Median :13.60   Median :21.00   Median : 8377
##  Mean   : 79.7   Mean   :14.09   Mean   :22.74   Mean   : 9660
##  3rd Qu.: 92.0   3rd Qu.:16.50   3rd Qu.:31.00   3rd Qu.:10830
##  Max.   :100.0   Max.   :39.80   Max.   :64.00   Max.   :56233
##    Grad.Rate
##  Min.   : 10.00
##  1st Qu.: 53.00
##  Median : 65.00
##  Mean   : 65.46
##  3rd Qu.: 78.00
##  Max.   :118.00

hist(College$Accept)
College = na.omit(College)

#  response distribution
hist(College$Accept)
```



```r
# yes =1 and no = 0 we convert variable to numeric.
College$Private = as.numeric(unclass(College$Private) - 1.0)

# We want to make better predictions, so I need to normalize the variables, a
nd I do this with log.
College[,2:18] = log(College[,2:18])
#summary and histogram
summary(College)

##     Private            Apps            Accept           Enroll     
##  Min.   :0.0000   Min.   : 4.394   Min.   : 4.277   Min.   :3.555  
##  1st Qu.:0.0000   1st Qu.: 6.654   1st Qu.: 6.404   1st Qu.:5.489  
##  Median :1.0000   Median : 7.351   Median : 7.012   Median :6.073  
##  Mean   :0.7272   Mean   : 7.427   Mean   : 7.110   Mean   :6.173  
##  3rd Qu.:1.0000   3rd Qu.: 8.195   3rd Qu.: 7.793   3rd Qu.:6.805  
##  Max.   :1.0000   Max.   :10.781   Max.   :10.178   Max.   :8.763  
##    Top10perc        Top25perc       F.Undergrad       P.Undergrad   
##  Min.   :0.000   Min.   :2.197   Min.   : 4.934   Min.   :0.000  
##  1st Qu.:2.708   1st Qu.:3.714   1st Qu.: 6.900   1st Qu.:4.554  
##  Median :3.135   Median :3.989   Median : 7.442   Median :5.866  
##  Mean   :3.114   Mean   :3.951   Mean   : 7.635   Mean   :5.691  
##  3rd Qu.:3.555   3rd Qu.:4.234   3rd Qu.: 8.295   3rd Qu.:6.874  
##  Max.   :4.564   Max.   :4.605   Max.   :10.362   Max.   :9.991  
##     Outstate        Room.Board        Books          Personal    
##  Min.   :7.758   Min.   :7.484   Min.   :4.564   Min.   :5.521  
##  1st Qu.:8.898   1st Qu.:8.188   1st Qu.:6.153   1st Qu.:6.745  
##  Median :9.209   Median :8.343   Median :6.215   Median :7.090  
##  Mean   :9.176   Mean   :8.348   Mean   :6.272   Mean   :7.085  
##  3rd Qu.:9.467   3rd Qu.:8.527   3rd Qu.:6.397   3rd Qu.:7.438  
##  Max.   :9.985   Max.   :9.003   Max.   :7.758   Max.   :8.825  
##       PhD           Terminal       S.F.Ratio       perc.alumni   
##  Min.   :2.079   Min.   :3.178   Min.   :0.9163   Min.   : -Inf  
##  1st Qu.:4.127   1st Qu.:4.263   1st Qu.:2.4423   1st Qu.:2.565  
##  Median :4.317   Median :4.407   Median :2.6101   Median :3.045  
##  Mean   :4.252   Mean   :4.358   Mean   :2.6036   Mean   : -Inf  
##  3rd Qu.:4.443   3rd Qu.:4.522   3rd Qu.:2.8034   3rd Qu.:3.434  
##  Max.   :4.635   Max.   :4.605   Max.   :3.6839   Max.   :4.159  
##      Expend          Grad.Rate    
##  Min.   : 8.067   Min.   :2.303  
##  1st Qu.: 8.817   1st Qu.:3.970  
##  Median : 9.033   Median :4.174  
##  Mean   : 9.081   Mean   :4.141  
##  3rd Qu.: 9.290   3rd Qu.:4.357  
##  Max.   :10.937   Max.   :4.771

hist(College$Accept)
```




```
# i want to find outlier's values and indexes.
out = boxplot.stats(College$Accept)$out
out_ind = which(College$Accept %in% c(out))
out_ind

## [1] 111 484

College[out_ind, "Accept"]

## [1]  4.276666 10.178464
```

2. Fit a **multiple linear regression model** after partitioning your data set into training and testing (you can apply 80-20 % rule). After fitting the model, **make predictions on testing data** and compare with the original observations.

```
#ikinci soru

# Data partitioning %80 %20 rate.
trainIndex = sample(seq_len(nrow(College)), round(0.8*nrow(College)))
# my train data
trainData = College[trainIndex, ]
# my test data
testData = College[-trainIndex, ]
#dimension
dim(trainData)

## [1] 622  18

dim(testData)
```

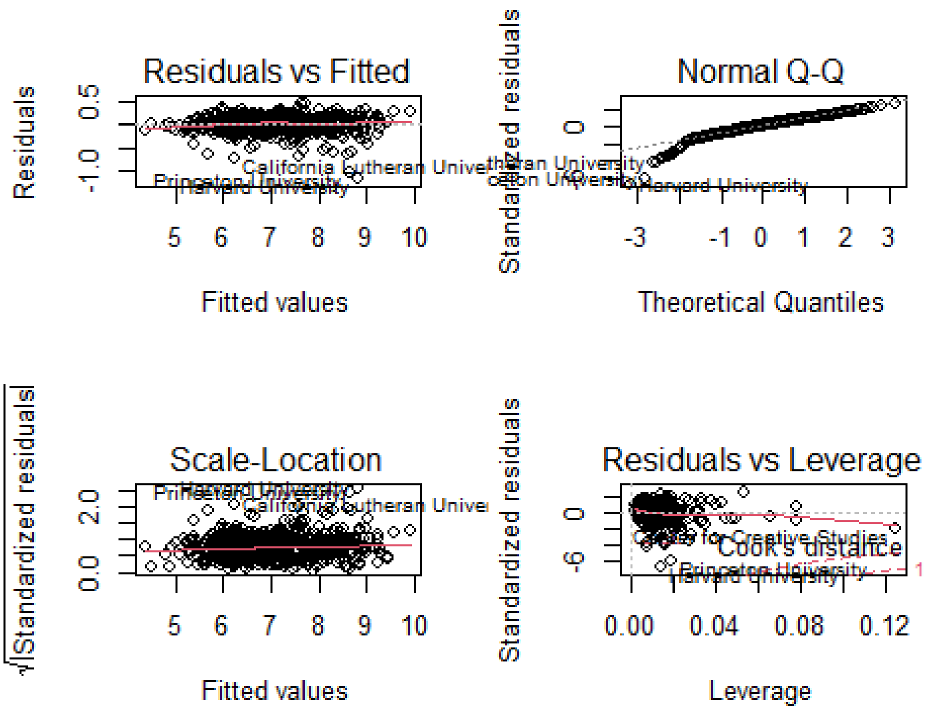
```
## [1] 155  18
```

When I normalized the response value, I got inaccurate exaggeration results, but we have to evolve all the data we have to the normal distribution to produce more accurate and confident estimates and realistic P values that require data preprocessing. I can achieve this using 0 and 1. This is the method I will use. Thanks to 0 and 1, we discover two distant and different values in our searches. Grubbs.test() allows us to use the Grubbs test in R. We use the Grubbs test to determine whether the smallest or largest value of a data set is an outlier.

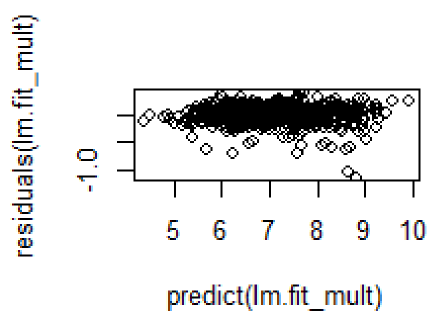
```
# multiple linear regression model

lm.fit_mult = lm(trainData$Accept ~ trainData$Apps + trainData$Enroll + train
Data$Top10perc + trainData$Outstate + trainData$Books + trainData$S.F.Ratio ,
data = trainData)
summary(lm.fit_mult)

## 
## Call:
## lm(formula = trainData$Accept ~ trainData$Apps + trainData$Enroll + 
##     trainData$Top10perc + trainData$Outstate + trainData$Books + 
##     trainData$S.F.Ratio, data = trainData)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -1.14086 -0.08433  0.01860  0.11254  0.41983 
## 
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)    
## (Intercept)         -0.72878    0.29900  -2.437  0.01508 *  
## trainData$Apps       0.61180    0.01978  30.923  < 2e-16 ***
## trainData$Enroll     0.36557    0.02259  16.186  < 2e-16 ***
## trainData$Top10perc -0.08121    0.01310  -6.199 1.04e-09 ***
## trainData$Outstate   0.16426    0.02422   6.783 2.77e-11 ***
## trainData$Books     -0.06580    0.02634  -2.498  0.01276 *  
## trainData$S.F.Ratio  0.07725    0.02840   2.720  0.00672 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1723 on 615 degrees of freedom
## Multiple R-squared:  0.9701, Adjusted R-squared:  0.9698 
## F-statistic:  3324 on 6 and 615 DF,  p-value: < 2.2e-16

# fitted model's predict
Pred = predict(lm.fit_mult, type = "response")
```

3. Using the `plot` command, comment on the **validity of the assumption of the model** that you fit in Question 2 (Note before using the `plot` command you may wish to specify a 2x2 graphics window using `par(mfrow = c(2, 2))`).

```
par(mfrow = c(2, 2))
plot(lm.fit_mult)
```




```
plot(predict(lm.fit_mult), residuals(lm.fit_mult))
```



The difference between the train value and my guess is very small. This shows that I have a successful prediction. Our line is straight, and there are no trends. Hence the Residual vs Fitted plog is a perfect selection. Residuals vs Leverage plot is over 0.5. At the same time, there is no perfect trend in the Scale-Location plot. In addition, according to my Q-Q chart, the data showed a normal distribution. There is only a small tail. This is the part that we want expendable.

4. Consider **the subset selection** idea to understand which of the variables are selected mostly when you implement; **i) best subset**, **ii) forward stepwise** and **iii) backward stepwise** algorithms. Try to figure out **optimal numbers in each selection algorithm**, by considering the **minimum BIC** performance metric!

```
#package installs
#install.packages("caret")
#install.packages("lattice")
#install.packages("ggplot2")
#install.packages("tidyverse")
#import libraries
library(caret)

## Zorunlu paket yükleniyor: ggplot2

## Zorunlu paket yükleniyor: lattice

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.
3.1 --

## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflict
s() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()

train_control = trainControl(method = "cv",number = 10)

model = train(Accept ~ Apps + Enroll + Top10perc + Outstate + Books + S.F.Rat
io, trainData,
               method = "lm",
               trControl = train_control)

summary(model)

##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max 
## -1.14086 -0.08433  0.01860  0.11254  0.41983 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -0.72878    0.29900  -2.437  0.01508 *  
## Apps         0.61180    0.01978  30.923  < 2e-16 ***
## Enroll       0.36557    0.02259  16.186  < 2e-16 ***
## Top10perc   -0.08121    0.01310  -6.199 1.04e-09 ***
## Outstate     0.16426    0.02422   6.783 2.77e-11 ***
## Books       -0.06580    0.02634  -2.498  0.01276 *  
## S.F.Ratio    0.07725    0.02840   2.720  0.00672 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1723 on 615 degrees of freedom
## Multiple R-squared:  0.9701, Adjusted R-squared:  0.9698 
## F-statistic:  3324 on 6 and 615 DF,  p-value: < 2.2e-16
```

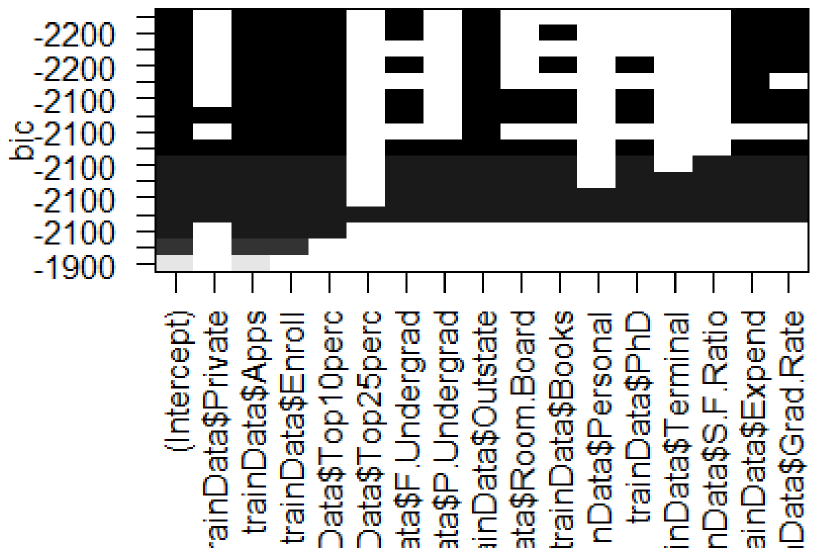
We will be able to build a good model.

```
#install.packages("leaps")
library(leaps)

# predictors using for linear model fitting
regfit.full = regsubsets(trainData$Accept ~ trainData$Private + trainData$App
s + trainData$Enroll + trainData$Top10perc + trainData$Top25perc + trainData$
F.Undergrad + trainData$P.Undergrad + trainData$Outstate + trainData$Room.Boa
rd + trainData$Books + trainData$Personal + trainData$PhD + trainData$Termina
l + trainData$S.F.Ratio  + trainData$Expend + trainData$Grad.Rate, data = tra
inData, nvmax = 18, method = "exhaustive")
summary(regfit.full)

## Subset selection object
## Call: regsubsets.formula(trainData$Accept ~ trainData$Private + trainData$
Apps + 
##     trainData$Enroll + trainData$Top10perc + trainData$Top25perc + 
##     trainData$F.Undergrad + trainData$P.Undergrad + trainData$Outstate + 
##     trainData$Room.Board + trainData$Books + trainData$Personal + 
##     trainData$PhD + trainData$Terminal + trainData$S.F.Ratio + 
##     trainData$Expend + trainData$Grad.Rate, data = trainData, 
##     nvmax = 18, method = "exhaustive")
## 16 Variables  (and intercept)
##                       Forced in Forced out
## trainData$Private         FALSE      FALSE
## trainData$Apps            FALSE      FALSE
## trainData$Enroll          FALSE      FALSE
## trainData$Top10perc       FALSE      FALSE
## trainData$Top25perc       FALSE      FALSE
## trainData$F.Undergrad     FALSE      FALSE
## trainData$P.Undergrad     FALSE      FALSE
```

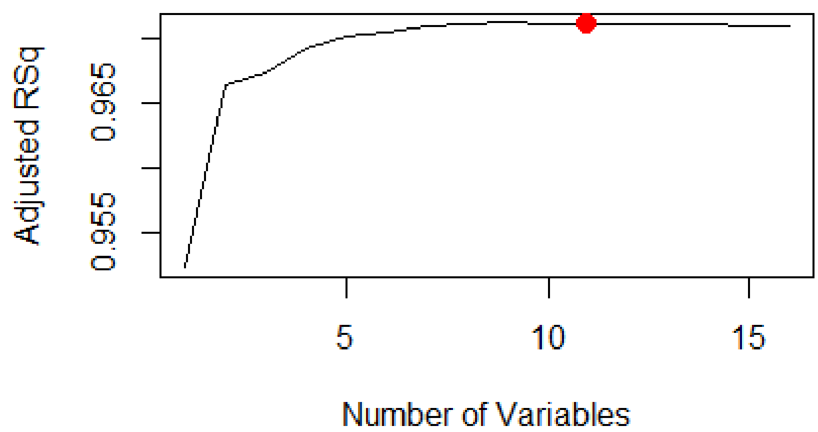
```
## trainData$Outstate        FALSE      FALSE
## trainData$Room.Board      FALSE      FALSE
## trainData$Books           FALSE      FALSE
## trainData$Personal        FALSE      FALSE
## trainData$PhD             FALSE      FALSE
## trainData$Terminal        FALSE      FALSE
## trainData$S.F.Ratio       FALSE      FALSE
## trainData$Expend          FALSE      FALSE
## trainData$Grad.Rate       FALSE      FALSE
## 1 subsets of each size up to 16
## Selection Algorithm: exhaustive
##           trainData$Private trainData$Apps trainData$Enroll trainData$Top1
0perc
## 1  ( 1 )  " "               "*"            " "              " "
## 2  ( 1 )  " "               "*"            "*"              " "
## 3  ( 1 )  " "               "*"            "*"              "*"
## 4  ( 1 )  " "               "*"            "*"              "*"
## 5  ( 1 )  " "               "*"            "*"              "*"
## 6  ( 1 )  " "               "*"            "*"              "*"
## 7  ( 1 )  " "               "*"            "*"              "*"
## 8  ( 1 )  " "               "*"            "*"              "*"
## 9  ( 1 )  " "               "*"            "*"              "*"
## 10  ( 1 ) " "               "*"            "*"              "*"
## 11  ( 1 ) "*"               "*"            "*"              "*"
## 12  ( 1 ) "*"               "*"            "*"              "*"
## 13  ( 1 ) "*"               "*"            "*"              "*"
## 14  ( 1 ) "*"               "*"            "*"              "*"
## 15  ( 1 ) "*"               "*"            "*"              "*"
## 16  ( 1 ) "*"               "*"            "*"              "*"
##           trainData$Top25perc trainData$F.Undergrad trainData$P.Undergrad
## 1  ( 1 )  " "                 " "                   " "
## 2  ( 1 )  " "                 " "                   " "
## 3  ( 1 )  " "                 " "                   " "
## 4  ( 1 )  " "                 " "                   " "
## 5  ( 1 )  " "                 " "                   " "
## 6  ( 1 )  " "                 " "                   " "
## 7  ( 1 )  " "                 "*"                   " "
## 8  ( 1 )  " "                 "*"                   " "
## 9  ( 1 )  " "                 "*"                   " "
## 10  ( 1 ) " "                 "*"                   " "
## 11  ( 1 ) " "                 "*"                   " "
## 12  ( 1 ) " "                 "*"                   "*"
## 13  ( 1 ) " "                 "*"                   "*"
## 14  ( 1 ) " "                 "*"                   "*"
## 15  ( 1 ) " "                 "*"                   "*"
## 16  ( 1 ) "*"                 "*"                   "*"
##           trainData$Outstate trainData$Room.Board trainData$Books
## 1  ( 1 )  " "                " "                  " "
## 2  ( 1 )  " "                " "                  " "
## 3  ( 1 )  " "                " "                  " "
```

```
## 4  ( 1 )  "*"                 " "                     " "
## 5  ( 1 )  "*"                 " "                     " "
## 6  ( 1 )  "*"                 " "                     " "
## 7  ( 1 )  "*"                 " "                     " "
## 8  ( 1 )  "*"                 " "                     "*"
## 9  ( 1 )  "*"                 " "                     "*"
## 10  ( 1 ) "*"                 "*"                     "*"
## 11  ( 1 ) "*"                 "*"                     "*"
## 12  ( 1 ) "*"                 "*"                     "*"
## 13  ( 1 ) "*"                 "*"                     "*"
## 14  ( 1 ) "*"                 "*"                     "*"
## 15  ( 1 ) "*"                 "*"                     "*"
## 16  ( 1 ) "*"                 "*"                     "*"
##           trainData$Personal trainData$PhD trainData$Terminal
## 1  ( 1 )  " "                " "           " "
## 2  ( 1 )  " "                " "           " "
## 3  ( 1 )  " "                " "           " "
## 4  ( 1 )  " "                " "           " "
## 5  ( 1 )  " "                " "           " "
## 6  ( 1 )  " "                " "           " "
## 7  ( 1 )  " "                " "           " "
## 8  ( 1 )  " "                " "           " "
## 9  ( 1 )  " "                "*"           " "
## 10  ( 1 ) " "                "*"           " "
## 11  ( 1 ) " "                "*"           " "
## 12  ( 1 ) " "                "*"           " "
## 13  ( 1 ) " "                "*"           " "
## 14  ( 1 ) " "                "*"           "*"
## 15  ( 1 ) "*"                "*"           "*"
## 16  ( 1 ) "*"                "*"           "*"
##           trainData$S.F.Ratio trainData$Expend trainData$Grad.Rate
## 1  ( 1 )  " "                 " "              " "
## 2  ( 1 )  " "                 " "              " "
## 3  ( 1 )  " "                 " "              " "
## 4  ( 1 )  " "                 " "              " "
## 5  ( 1 )  " "                 "*"              " "
## 6  ( 1 )  " "                 "*"              "*"
## 7  ( 1 )  " "                 "*"              "*"
## 8  ( 1 )  " "                 "*"              "*"
## 9  ( 1 )  " "                 "*"              "*"
## 10  ( 1 ) " "                 "*"              "*"
## 11  ( 1 ) " "                 "*"              "*"
## 12  ( 1 ) " "                 "*"              "*"
## 13  ( 1 ) "*"                 "*"              "*"
## 14  ( 1 ) "*"                 "*"              "*"
## 15  ( 1 ) "*"                 "*"              "*"
## 16  ( 1 ) "*"                 "*"              "*"
```

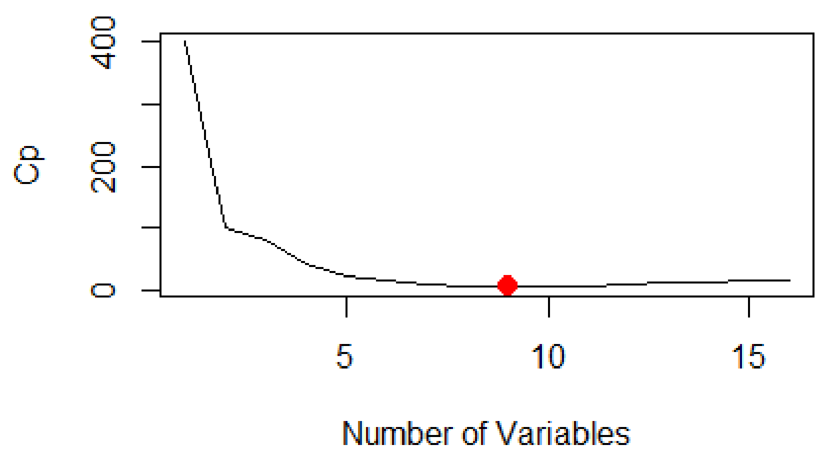
```
plot(regfit.full)
```



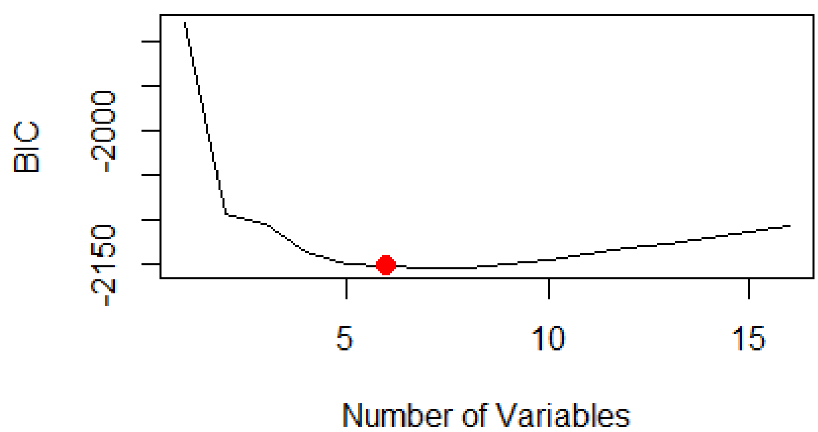

```
reg.summary = summary(regfit.full)
paste(data.frame(
  Adj.R2 = which.max(reg.summary$adjr2),
  CP = which.min(reg.summary$cp),
  BIC = which.min(reg.summary$bic)
))

## [1] "9" "9" "7"

# which.max(reg.summary$adjr2)
plot(reg.summary$adjr2 , xlab = "Number of Variables", ylab = "Adjusted RSq",
type = "l")
points (11, reg.summary$adjr2[11] , col = "red", cex = 2, pch = 20)
```




```r
# which.min(reg.summary$cp)
plot(reg.summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
points (9, reg.summary$cp[9] , col = "red", cex = 2, pch = 20)
```



```
# which.min(reg.summary$bic)
plot(reg.summary$bic , xlab = "Number of Variables", ylab = "BIC", type = "l"
)
points (6, reg.summary$bic [6], col = "red", cex = 2, pch = 20)
```



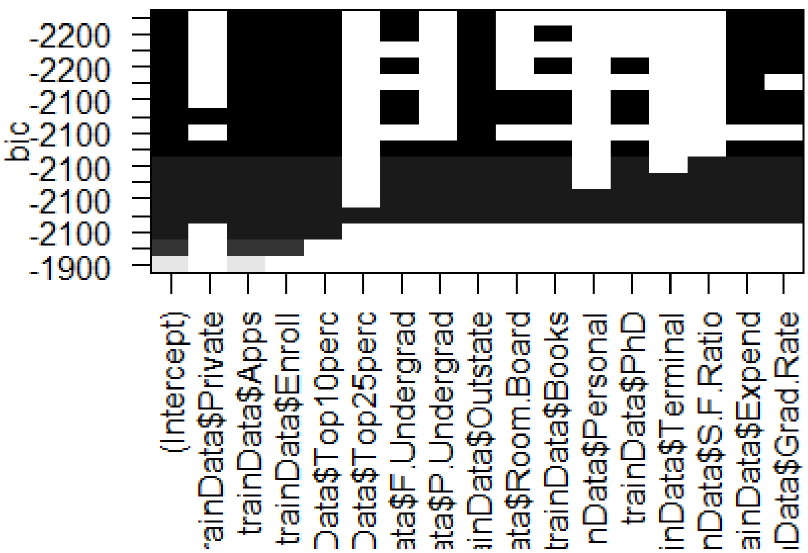
I need a better approach because when I approach with the Best Model approach, different results appear according to the value I measure.

```
# fit linear model using the predictors
regfit.fwd = regsubsets(trainData$Accept ~ trainData$Private + trainData$Apps
+ trainData$Enroll + trainData$Top10perc + trainData$Top25perc + trainData$F.
Undergrad + trainData$P.Undergrad + trainData$Outstate + trainData$Room.Board
+ trainData$Books + trainData$Personal + trainData$PhD + trainData$Terminal +
trainData$S.F.Ratio  + trainData$Expend + trainData$Grad.Rate, data = trainDa
ta, nvmax = 18, method = "forward")
summary(regfit.fwd)

## Subset selection object
## Call: regsubsets.formula(trainData$Accept ~ trainData$Private + trainData$
Apps +
##     trainData$Enroll + trainData$Top10perc + trainData$Top25perc +
##     trainData$F.Undergrad + trainData$P.Undergrad + trainData$Outstate +
##     trainData$Room.Board + trainData$Books + trainData$Personal +
##     trainData$PhD + trainData$Terminal + trainData$S.F.Ratio +
##     trainData$Expend + trainData$Grad.Rate, data = trainData,
##     nvmax = 18, method = "forward")
## 16 Variables  (and intercept)
```

```
##                       Forced in Forced out
## trainData$Private         FALSE      FALSE
## trainData$Apps            FALSE      FALSE
## trainData$Enroll          FALSE      FALSE
## trainData$Top10perc       FALSE      FALSE
## trainData$Top25perc       FALSE      FALSE
## trainData$F.Undergrad     FALSE      FALSE
## trainData$P.Undergrad     FALSE      FALSE
## trainData$Outstate        FALSE      FALSE
## trainData$Room.Board      FALSE      FALSE
## trainData$Books           FALSE      FALSE
## trainData$Personal        FALSE      FALSE
## trainData$PhD             FALSE      FALSE
## trainData$Terminal        FALSE      FALSE
## trainData$S.F.Ratio       FALSE      FALSE
## trainData$Expend          FALSE      FALSE
## trainData$Grad.Rate       FALSE      FALSE
## 1 subsets of each size up to 16
## Selection Algorithm: forward
##            trainData$Private trainData$Apps trainData$Enroll trainData$Top1
0perc
## 1  ( 1 )  " "               "*"            " "              " "
## 2  ( 1 )  " "               "*"            "*"              " "
## 3  ( 1 )  " "               "*"            "*"              "*"
## 4  ( 1 )  " "               "*"            "*"              "*"
## 5  ( 1 )  " "               "*"            "*"              "*"
## 6  ( 1 )  " "               "*"            "*"              "*"
## 7  ( 1 )  " "               "*"            "*"              "*"
## 8  ( 1 )  " "               "*"            "*"              "*"
## 9  ( 1 )  " "               "*"            "*"              "*"
## 10  ( 1 ) " "               "*"            "*"              "*"
## 11  ( 1 ) "*"               "*"            "*"              "*"
## 12  ( 1 ) "*"               "*"            "*"              "*"
## 13  ( 1 ) "*"               "*"            "*"              "*"
## 14  ( 1 ) "*"               "*"            "*"              "*"
## 15  ( 1 ) "*"               "*"            "*"              "*"
## 16  ( 1 ) "*"               "*"            "*"              "*"
##            trainData$Top25perc trainData$F.Undergrad trainData$P.Undergrad
## 1  ( 1 )  " "                 " "                   " "
## 2  ( 1 )  " "                 " "                   " "
## 3  ( 1 )  " "                 " "                   " "
## 4  ( 1 )  " "                 " "                   " "
## 5  ( 1 )  " "                 " "                   " "
## 6  ( 1 )  " "                 " "                   " "
## 7  ( 1 )  " "                 "*"                   " "
## 8  ( 1 )  " "                 "*"                   " "
## 9  ( 1 )  " "                 "*"                   " "
## 10  ( 1 ) " "                 "*"                   " "
## 11  ( 1 ) " "                 "*"                   " "
## 12  ( 1 ) " "                 "*"                   "*"
```

```
## 13  ( 1 ) " "                 "*"                   "*"
## 14  ( 1 ) " "                 "*"                   "*"
## 15  ( 1 ) " "                 "*"                   "*"
## 16  ( 1 ) "*"                 "*"                   "*"
##           trainData$Outstate trainData$Room.Board trainData$Books
## 1  ( 1 )  " "                " "                  " "
## 2  ( 1 )  " "                " "                  " "
## 3  ( 1 )  " "                " "                  " "
## 4  ( 1 )  "*"                " "                  " "
## 5  ( 1 )  "*"                " "                  " "
## 6  ( 1 )  "*"                " "                  " "
## 7  ( 1 )  "*"                " "                  " "
## 8  ( 1 )  "*"                " "                  "*"
## 9  ( 1 )  "*"                " "                  "*"
## 10  ( 1 ) "*"                "*"                  "*"
## 11  ( 1 ) "*"                "*"                  "*"
## 12  ( 1 ) "*"                "*"                  "*"
## 13  ( 1 ) "*"                "*"                  "*"
## 14  ( 1 ) "*"                "*"                  "*"
## 15  ( 1 ) "*"                "*"                  "*"
## 16  ( 1 ) "*"                "*"                  "*"
##           trainData$Personal trainData$PhD trainData$Terminal
## 1  ( 1 )  " "                " "           " "
## 2  ( 1 )  " "                " "           " "
## 3  ( 1 )  " "                " "           " "
## 4  ( 1 )  " "                " "           " "
## 5  ( 1 )  " "                " "           " "
## 6  ( 1 )  " "                " "           " "
## 7  ( 1 )  " "                " "           " "
## 8  ( 1 )  " "                " "           " "
## 9  ( 1 )  " "                "*"           " "
## 10  ( 1 ) " "                "*"           " "
## 11  ( 1 ) " "                "*"           " "
## 12  ( 1 ) " "                "*"           " "
## 13  ( 1 ) " "                "*"           " "
## 14  ( 1 ) " "                "*"           "*"
## 15  ( 1 ) "*"                "*"           "*"
## 16  ( 1 ) "*"                "*"           "*"
##           trainData$S.F.Ratio trainData$Expend trainData$Grad.Rate
## 1  ( 1 )  " "                 " "              " "
## 2  ( 1 )  " "                 " "              " "
## 3  ( 1 )  " "                 " "              " "
## 4  ( 1 )  " "                 " "              " "
## 5  ( 1 )  " "                 "*"              " "
## 6  ( 1 )  " "                 "*"              "*"
## 7  ( 1 )  " "                 "*"              "*"
## 8  ( 1 )  " "                 "*"              "*"
## 9  ( 1 )  " "                 "*"              "*"
## 10  ( 1 ) " "                 "*"              "*"
## 11  ( 1 ) " "                 "*"              "*"
```

```
## 12  ( 1 ) " "                    "*"                "*"
## 13  ( 1 ) "*"                    "*"                "*"
## 14  ( 1 ) "*"                    "*"                "*"
## 15  ( 1 ) "*"                    "*"                "*"
## 16  ( 1 ) "*"                    "*"                "*"

plot(regfit.fwd)
```



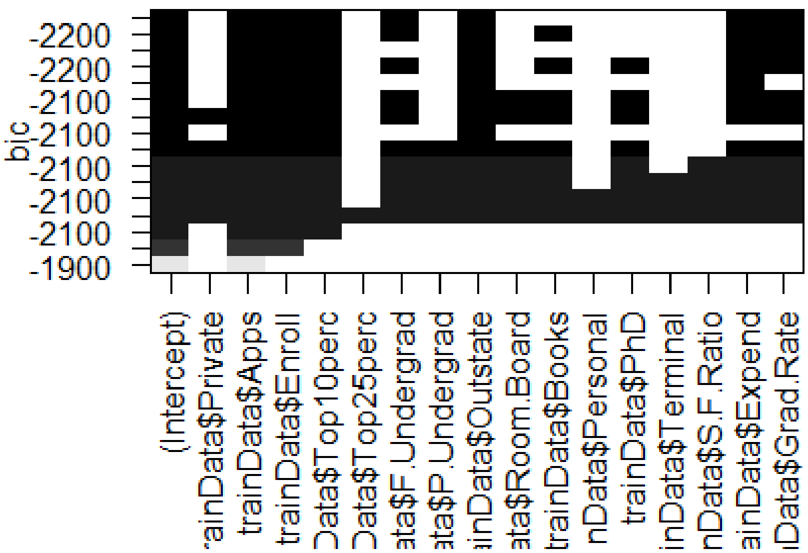
Forward

```
# fitting linear model with predictors
regfit.bwd = regsubsets(trainData$Accept ~ trainData$Private + trainData$Apps
+ trainData$Enroll + trainData$Top10perc + trainData$Top25perc + trainData$F.
Undergrad + trainData$P.Undergrad + trainData$Outstate + trainData$Room.Board
+ trainData$Books + trainData$Personal + trainData$PhD + trainData$Terminal +
trainData$S.F.Ratio  + trainData$Expend + trainData$Grad.Rate, data = trainDa
ta, nvmax = 18, method = "backward")
summary(regfit.bwd)

## Subset selection object
## Call: regsubsets.formula(trainData$Accept ~ trainData$Private + trainData$
Apps +
##     trainData$Enroll + trainData$Top10perc + trainData$Top25perc +
##     trainData$F.Undergrad + trainData$P.Undergrad + trainData$Outstate +
##     trainData$Room.Board + trainData$Books + trainData$Personal +
##     trainData$PhD + trainData$Terminal + trainData$S.F.Ratio +
##     trainData$Expend + trainData$Grad.Rate, data = trainData,
##     nvmax = 18, method = "backward")
## 16 Variables  (and intercept)
```

```
##                        Forced in Forced out
## trainData$Private         FALSE      FALSE
## trainData$Apps            FALSE      FALSE
## trainData$Enroll          FALSE      FALSE
## trainData$Top10perc       FALSE      FALSE
## trainData$Top25perc       FALSE      FALSE
## trainData$F.Undergrad     FALSE      FALSE
## trainData$P.Undergrad     FALSE      FALSE
## trainData$Outstate        FALSE      FALSE
## trainData$Room.Board      FALSE      FALSE
## trainData$Books           FALSE      FALSE
## trainData$Personal        FALSE      FALSE
## trainData$PhD             FALSE      FALSE
## trainData$Terminal        FALSE      FALSE
## trainData$S.F.Ratio       FALSE      FALSE
## trainData$Expend          FALSE      FALSE
## trainData$Grad.Rate       FALSE      FALSE
## 1 subsets of each size up to 16
## Selection Algorithm: backward
##           trainData$Private trainData$Apps trainData$Enroll trainData$Top1
0perc
## 1  ( 1 )  " "               "*"            " "              " "
## 2  ( 1 )  " "               "*"            "*"              " "
## 3  ( 1 )  " "               "*"            "*"              "*"
## 4  ( 1 )  " "               "*"            "*"              "*"
## 5  ( 1 )  " "               "*"            "*"              "*"
## 6  ( 1 )  " "               "*"            "*"              "*"
## 7  ( 1 )  " "               "*"            "*"              "*"
## 8  ( 1 )  " "               "*"            "*"              "*"
## 9  ( 1 )  " "               "*"            "*"              "*"
## 10  ( 1 ) " "               "*"            "*"              "*"
## 11  ( 1 ) "*"               "*"            "*"              "*"
## 12  ( 1 ) "*"               "*"            "*"              "*"
## 13  ( 1 ) "*"               "*"            "*"              "*"
## 14  ( 1 ) "*"               "*"            "*"              "*"
## 15  ( 1 ) "*"               "*"            "*"              "*"
## 16  ( 1 ) "*"               "*"            "*"              "*"
##           trainData$Top25perc trainData$F.Undergrad trainData$P.Undergrad
## 1  ( 1 )  " "                 " "                   " "
## 2  ( 1 )  " "                 " "                   " "
## 3  ( 1 )  " "                 " "                   " "
## 4  ( 1 )  " "                 " "                   " "
## 5  ( 1 )  " "                 " "                   " "
## 6  ( 1 )  " "                 " "                   " "
## 7  ( 1 )  " "                 "*"                   " "
## 8  ( 1 )  " "                 "*"                   " "
## 9  ( 1 )  " "                 "*"                   " "
## 10  ( 1 ) " "                 "*"                   " "
## 11  ( 1 ) " "                 "*"                   " "
## 12  ( 1 ) " "                 "*"                   "*"
```

```
## 13  ( 1 ) " "                  "*"                    "*"
## 14  ( 1 ) " "                  "*"                    "*"
## 15  ( 1 ) " "                  "*"                    "*"
## 16  ( 1 ) "*"                  "*"                    "*"
##           trainData$Outstate trainData$Room.Board trainData$Books
## 1  ( 1 )  " "                " "                  " "
## 2  ( 1 )  " "                " "                  " "
## 3  ( 1 )  " "                " "                  " "
## 4  ( 1 )  "*"                " "                  " "
## 5  ( 1 )  "*"                " "                  " "
## 6  ( 1 )  "*"                " "                  " "
## 7  ( 1 )  "*"                " "                  " "
## 8  ( 1 )  "*"                " "                  "*"
## 9  ( 1 )  "*"                " "                  "*"
## 10  ( 1 ) "*"                "*"                  "*"
## 11  ( 1 ) "*"                "*"                  "*"
## 12  ( 1 ) "*"                "*"                  "*"
## 13  ( 1 ) "*"                "*"                  "*"
## 14  ( 1 ) "*"                "*"                  "*"
## 15  ( 1 ) "*"                "*"                  "*"
## 16  ( 1 ) "*"                "*"                  "*"
##           trainData$Personal trainData$PhD trainData$Terminal
## 1  ( 1 )  " "                " "           " "
## 2  ( 1 )  " "                " "           " "
## 3  ( 1 )  " "                " "           " "
## 4  ( 1 )  " "                " "           " "
## 5  ( 1 )  " "                " "           " "
## 6  ( 1 )  " "                " "           " "
## 7  ( 1 )  " "                " "           " "
## 8  ( 1 )  " "                " "           " "
## 9  ( 1 )  " "                "*"           " "
## 10  ( 1 ) " "                "*"           " "
## 11  ( 1 ) " "                "*"           " "
## 12  ( 1 ) " "                "*"           " "
## 13  ( 1 ) " "                "*"           " "
## 14  ( 1 ) " "                "*"           "*"
## 15  ( 1 ) "*"                "*"           "*"
## 16  ( 1 ) "*"                "*"           "*"
##           trainData$S.F.Ratio trainData$Expend trainData$Grad.Rate
## 1  ( 1 )  " "                 " "              " "
## 2  ( 1 )  " "                 " "              " "
## 3  ( 1 )  " "                 " "              " "
## 4  ( 1 )  " "                 " "              " "
## 5  ( 1 )  " "                 "*"              " "
## 6  ( 1 )  " "                 "*"              "*"
## 7  ( 1 )  " "                 "*"              "*"
## 8  ( 1 )  " "                 "*"              "*"
## 9  ( 1 )  " "                 "*"              "*"
## 10  ( 1 ) " "                 "*"              "*"
## 11  ( 1 ) " "                 "*"              "*"
```

```
## 12  ( 1 ) " "                       "*"               "*"
## 13  ( 1 ) "*"                       "*"               "*"
## 14  ( 1 ) "*"                       "*"               "*"
## 15  ( 1 ) "*"                       "*"               "*"
## 16  ( 1 ) "*"                       "*"               "*"

plot(regfit.bwd)
```



Backward

```
coef(regfit.fwd, 16)

##           (Intercept)     trainData$Private        trainData$Apps
##          0.4553910734          0.0232613937          0.6446664747
##      trainData$Enroll   trainData$Top10perc   trainData$Top25perc
##          0.4267193205         -0.0645569498         -0.0008951659
## trainData$F.Undergrad trainData$P.Undergrad    trainData$Outstate
##         -0.0831557358          0.0045059567          0.2232522531
##  trainData$Room.Board       trainData$Books    trainData$Personal
##         -0.0411221527         -0.0492307860          0.0004354508
##         trainData$PhD    trainData$Terminal    trainData$S.F.Ratio
##          0.0603805754         -0.0032112944          0.0035595869
##      trainData$Expend   trainData$Grad.Rate
##         -0.1292680417         -0.1070560027

coef(regfit.bwd, 16)

##           (Intercept)     trainData$Private        trainData$Apps
##          0.4553910734          0.0232613937          0.6446664747
##      trainData$Enroll   trainData$Top10perc   trainData$Top25perc
```

```
##          0.4267193205         -0.0645569498         -0.0008951659
## trainData$F.Undergrad trainData$P.Undergrad    trainData$Outstate
##         -0.0831557358          0.0045059567          0.2232522531
##   trainData$Room.Board       trainData$Books     trainData$Personal
##         -0.0411221527         -0.0492307860          0.0004354508
##         trainData$PhD    trainData$Terminal   trainData$S.F.Ratio
##          0.0603805754         -0.0032112944          0.0035595869
##      trainData$Expend   trainData$Grad.Rate
##         -0.1292680417         -0.1070560027
```

The comparision result is like this.

5. Fit a **ridge regression** model on the training set by **using the all predictors**, with λ parameter chosen by **cross-validation** beforehand. After building the model, report the test error obtained.

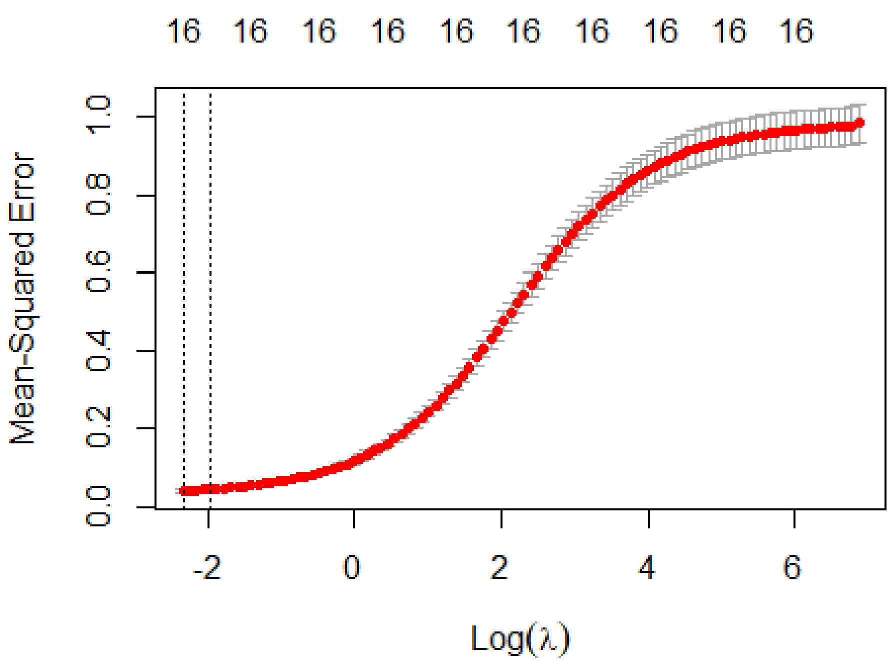
```
# Data proprocessing for Ridge Regression.
x = model.matrix(Accept ~., trainData)[,-1]
y = trainData$Accept
y = y[is.na(y) == FALSE]


#  Ridge Regression
#install.packages("glmnet")
library(glmnet)

## Zorunlu paket yükleniyor: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-4

grid = 10^seq(10, -2, length = 100)
ridge.mod = glmnet(x, y, alpha = 0, lambda = grid, standardize = FALSE)

summary(ridge.mod)

##           Length Class     Mode
## a0         100   -none-    numeric
## beta      1700   dgCMatrix S4
## df         100   -none-    numeric
## dim          2   -none-    numeric
## lambda     100   -none-    numeric
## dev.ratio  100   -none-    numeric
## nulldev      1   -none-    numeric
## npasses      1   -none-    numeric
## jerr         1   -none-    numeric
```

```
## offset       1   -none-    logical
## call         6   -none-    call
## nobs         1   -none-    numeric
```

```r
# k-fold cross-validation for  find optimal lambda value
cv_model = cv.glmnet(x, y, alpha = 0)

# optimal lambda value that minimizes test MSE
best_lambda = cv_model$lambda.min
best_lambda
```

```
## [1] 0.09666183
```

```r
# Produce plot of test MSE by lambda value
plot(cv_model)
```

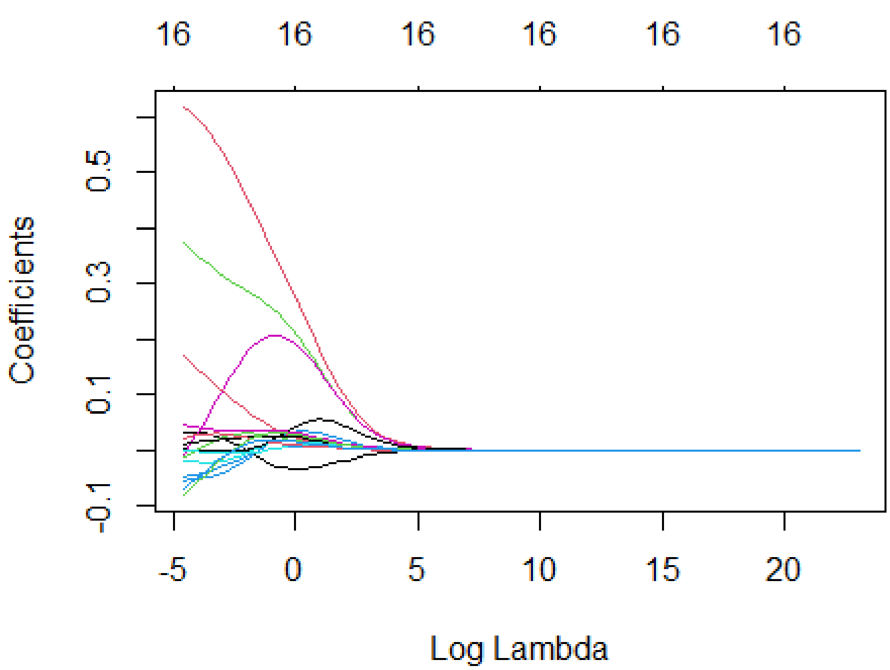



```r
# Find coefficients of best model
best_model = glmnet(x, y, alpha = 0, lambda = best_lambda)
coef(best_model)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                        s0
## (Intercept) -1.6783588964
## Private     -0.0159387674
## Apps         0.4399069626
## Enroll       0.3333481485
## Top10perc   -0.0341357581
## Top25perc   -0.0232162038
```

```
## F.Undergrad  0.1548036570
## P.Undergrad -0.0015296537
## Outstate     0.1717689582
## Room.Board   0.0623260257
## Books       -0.0333249516
## Personal     0.0009192791
## PhD          0.0653770820
## Terminal     0.0435275964
## S.F.Ratio    0.0723805736
## perc.alumni  .
## Expend      -0.0027377056
## Grad.Rate   -0.0049253058
```

```r
# Produce Ridge trace plot
plot(ridge.mod, xvar = "lambda")
```

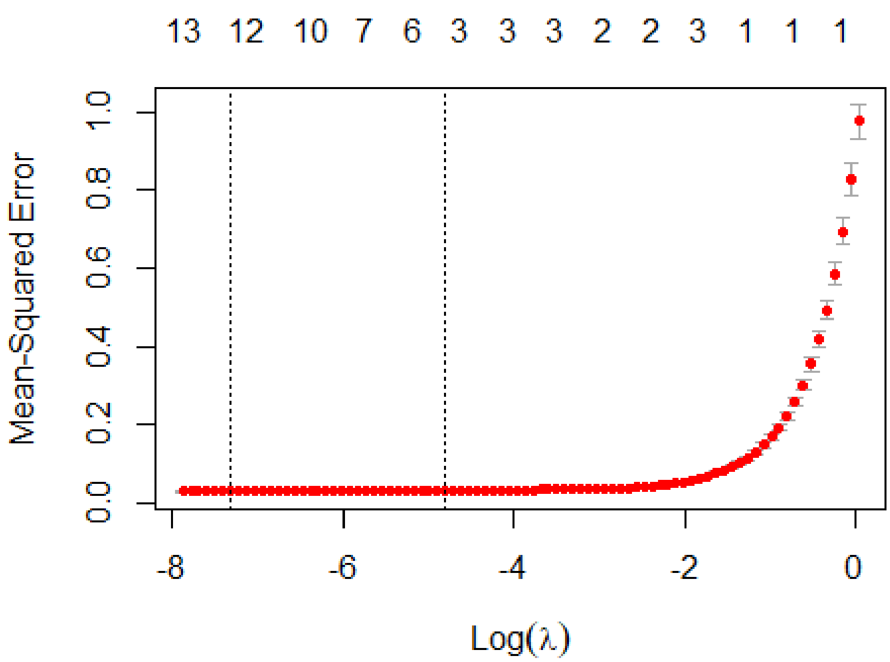



```r
# Use fitted best model to make predictions
y_predicted = predict(ridge.mod, s = best_lambda, newx = x)

# Find SST and SSE
sst = sum((y - mean(y))^2)
sse = sum((y_predicted - y)^2)

# Find R-Squared
rsq = 1 - sse/sst
rsq
```

```
## [1] 0.9628762
```

6. Fit a **LASSO regression** model on the training set by **using the all predictors**, with λ parameter chosen by **cross-validation** beforehand. After building the model, report the test error obtained.

```
# Data proprocessing for Lasso Regression.
x = model.matrix(Accept ~., trainData)[,-1]
y = trainData$Accept
y = y[is.na(y) == FALSE]
# Perform k-fold cross-validation to find optimal lambda value
cv_model = cv.glmnet(x, y, alpha = 1, standardize = FALSE)
# finding optimal lambda value that minimizes test MSE
best_lambda = cv_model$lambda.min
best_lambda

## [1] 0.0006684653

# Produce plot of test MSE by lambda value
plot(cv_model)
```




```
# Find coefficients of best model
best_model = glmnet(x, y, alpha = 1, lambda = best_lambda)
coef(best_model)

## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept)  0.399270661
```

```
## Private      0.021933450
## Apps         0.641155698
## Enroll       0.408293392
## Top10perc   -0.061031540
## Top25perc   -0.005142735
## F.Undergrad -0.060984929
## P.Undergrad  0.002184467
## Outstate     0.213687954
## Room.Board  -0.031139478
## Books       -0.048835300
## Personal     .
## PhD          0.052063717
## Terminal     .
## S.F.Ratio    0.002686435
## perc.alumni  .
## Expend      -0.123985746
## Grad.Rate   -0.100454520
```

```r
# Use fitted best model to make predictions
y_predicted = predict(best_model, s = best_lambda, newx = x)
# finding SST and SSE
sst = sum((y - mean(y))^2)
sse = sum((y_predicted - y)^2)
# finding R-Squared
rsq = 1 - sse/sst
rsq
```

```
## [1] 0.9716514
```

7. Comment on the above obtained results. How accurately can we predict the number of college applications received (Accept variable)? In terms of test error calculations you derived, is there much difference among the above-considered linear models ? **Which one is more preferable** ?

Ridge Regression has a better R-squared value. Therefore, Ridge Regression should be preferred. The differences between Ridge Regression and Lasso Regression are Ridge regression, which reduces all of our coefficients towards zero and works in this way. But Lasso Regression tries to set all coefficients to 0. Therefore, it has the ability to remove estimators from the model.

SOLUTIONS

- MAKE SURE THAT ALL NECESSARY PACKAGES ARE ALREADY INSTALLED and READY TO USE
- You can use as many as Rcode chunks you want. In the final output, both Rcodes and your ouputs including your comments should appear in an order
- Use the given R-code chunk below to make your calculations and summarize your result thereafter by adding comments on it,

References

Give a list of the available sources that you used while preparing your home-work (If you use other resources, you can make a list here for checking & reproducibility).

For instance;

- https://www.statlearning.com/
- https://lms.tedu.edu.tr/
- https://www.statisticshowto.com/