Stats101c_hw5

Takao

2022-11-15

Takao Oba

Stats 101C HW 5

Q1) In this Question, we will predict the amount of money expend in college using the other variables in the College data set. Split the data set into a 70% training set and 30% testing set. Use set.seed(1128)

```
temp <- read.csv("/Users/takaooba/Downloads/College Fall 2021.csv")

college <- temp[, -c(1,2)]
head(college)
```

```
##   Private  Apps Accept Enroll Top10perc Top25perc F.Undergrad P.Undergrad
## 1     Yes  1758   1485    419        27        58        2041         174
## 2      No 14463   6166   1757        60        94        8544         671
## 3     Yes   838    651    159        11        25         654         162
## 4     Yes  1127    884    308        30        64        1310         766
## 5     Yes   735    423    366        20        48        2448         707
## 6     Yes   504    482    185        10        36         550          84
##   Outstate Room.Board Books Personal PhD Terminal S.F.Ratio perc.alumni
## 1    12040       4100   600     1100  92       96      13.2          17
## 2     6550       4598   700     1000  83      100      18.0          15
## 3     8640       3700   400     1915  62       62      12.2          13
## 4    11718       7398   450     1800  73       87      16.4          33
## 5     9210       3782   700     1000  49       51      39.8          15
## 6     9130       3322   450     1450  46       51      12.6          25
##   Grad.Rate Expend
## 1        72   9060
## 2        80   8055
## 3        48   7634
## 4        76   8871
## 5        34   6562
## 6        54   8686
```

```r
dim(college)
```

```
## [1] 2000   18
```

```r
# Splitting 70% training and 30% testing
set.seed(1128)
test.i <- sample(1:2000, 600, replace = F)
college.test <- college[test.i,]

college.train <- college[-test.i,]
```

a) Fit a full multiple linear model using least squares on the training set, and report the MSE obtained using both data sets (training and testing).

```r
college.model <- lm(Expend ~ ., data = college.train)
summary(college.model)
```

```
##
## Call:
## lm(formula = Expend ~ ., data = college.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9501.7 -1401.5  -330.9   854.2 29479.8
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5690.96704  815.75143   6.976 4.68e-12 ***
## PrivateYes  -509.84234  294.92523  -1.729 0.084083 .
## Apps           0.81070    0.07684  10.551  < 2e-16 ***
## Accept        -1.08476    0.14955  -7.253 6.75e-13 ***
## Enroll         0.96925    0.41746   2.322 0.020390 *
## Top10perc    118.71611   11.03958  10.754  < 2e-16 ***
## Top25perc    -61.16818    8.70870  -7.024 3.38e-12 ***
## F.Undergrad   -0.10797    0.07397  -1.460 0.144575
## P.Undergrad   -0.06055    0.07456  -0.812 0.416932
## Outstate       0.55630    0.03768  14.765  < 2e-16 ***
## Room.Board     0.02744    0.10672   0.257 0.797134
## Books          1.26521    0.48786   2.593 0.009604 **
## Personal       0.23805    0.13346   1.784 0.074700 .
## PhD           -4.16541    9.09178  -0.458 0.646916
## Terminal      33.12020   10.10478   3.278 0.001073 **
## S.F.Ratio   -291.15598   24.29938 -11.982  < 2e-16 ***
## perc.alumni   12.06622    8.48524   1.422 0.155245
## Grad.Rate    -22.24596    6.23650  -3.567 0.000373 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2875 on 1382 degrees of freedom
## Multiple R-squared:  0.6972, Adjusted R-squared:  0.6934
## F-statistic: 187.2 on 17 and 1382 DF,  p-value: < 2.2e-16
```

The MSE obtained using both data sets

```r
pred.train <- predict(college.model, college.train)
pred.test <- predict(college.model, college.test)


# MSE for the training data set
sum((college.train$Expend - pred.train)^2)/length(pred.train)
```

```
## [1] 8160418
```

```r
# MSE for the testing data set
sum((college.test$Expend - pred.test)^2)/length(pred.test)
```

```
## [1] 10174615
```

b) Fit a ridge regression model on the training set, with lambda chosen by cross-validation. Report the MSE obtained using both data sets (training and testing).

```r
set.seed(1128)
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.1.2

## Loading required package: Matrix

## Loaded glmnet 4.1-4
```

```r
# Training data
x = model.matrix(Expend ~ ., data = college.train)
x.test <- model.matrix(Expend ~., data= college.test)
y = college.train$Expend

i = seq(10, -2, length = 100)
lambda.v = 10^i


model.ridge <- cv.glmnet(x, y, alpha = 0, lambda = lambda.v, type.measure = 'mse', keep=TRUE)
lambda.ridge = model.ridge$lambda.min
lambda.id <- which(model.ridge$lambda == model.ridge$lambda.min)
model.ridge
```

```
##
## Call:  cv.glmnet(x = x, y = y, lambda = lambda.v, type.measure = "mse",      keep = TRUE, alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure      SE Nonzero
## min   14.2    74 8490594 1358392      17
## 1se 1629.8    57 9696330 1682395      17
```

```
summary(model.ridge)
```

```
##             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            7 -none- call
## name            1 -none- character
## glmnet.fit     12 elnet  list
## fit.preval 140000 -none- numeric
## foldid       1400 -none- numeric
## lambda.min      1 -none- numeric
## lambda.1se      1 -none- numeric
## index           2 -none- numeric
```

```
mod1 <- glmnet(x,y, alpha = 0, lambda = lambda.v)
mod1.train <- predict(mod1, s = lambda.ridge, newx = x)
mod1.test <- predict(mod1, s = lambda.ridge, newx = x.test)

# Training
mean((mod1.train - college.train$Expend)^2)
```

```
## [1] 8163692
```

```
# Testing
mean((mod1.test - college.test$Expend)^2)
```

```
## [1] 10157872
```

```
# mse <- function(x,y){mean((x-y)^2)}
# mse.1 <- mse(model.ridge$fit[, lambda.id], y)
# mse.1
```

Training MSE is 8163692 and testing MSE is 8163692.

```
# set.seed(1128)
# # Testing
# x = model.matrix(Expend ~ ., data = college.test)
# y = college.test$Expend
#
# i = seq(10, -2, length = 100)
# lambda.v = 10^i
#
# model.ridge <- cv.glmnet(x, y, alpha = 0, lambda = lambda.v, type.measure = 'mse', keep=TRUE)
# lambda.lasso = model.ridge$lambda.min
# lambda.id <- which(model.ridge$lambda == model.ridge$lambda.min)
# model.ridge
# summary(model.ridge)
```

```
#
# mse <- function(x,y){mean((x-y)^2)}
# mse.1 <- mse(model.ridge$fit[, lambda.id], y)
# mse.1
```

c) Fit a lasso model on the training set, with lambda chosen by cross validation. Report the MSE obtained using both data sets (training and testing), along with the number of non-zero coefficient estimates.

```
set.seed(1128)
# Training data
x = model.matrix(Expend ~ ., data = college.train)
x.test = model.matrix(Expend ~ ., data = college.test)
y = college.train$Expend

i = seq(10, -2, length = 100)
lambda.v = 10^i

model.lasso <- cv.glmnet(x, y, alpha = 1, lambda = lambda.v, type.measure = 'mse', keep=TRUE)
lambda.lasso = model.lasso$lambda.min
lambda.id <- which(model.lasso$lambda == model.lasso$lambda.min)
model.lasso
```

```
##
## Call:  cv.glmnet(x = x, y = y, lambda = lambda.v, type.measure = "mse",      keep = TRUE, alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure      SE Nonzero
## min    0.5    86 8494226 1352101      17
## 1se  305.4    63 9714282 1637899       7
```

```
summary(model.lasso)
```

```
##             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             7 -none- call
## name             1 -none- character
## glmnet.fit      12 elnet  list
## fit.preval  140000 -none- numeric
## foldid        1400 -none- numeric
## lambda.min       1 -none- numeric
## lambda.1se       1 -none- numeric
## index            2 -none- numeric
```

```r
mod1 <- glmnet(x,y, alpha = 1, lambda = lambda.v)
mod1.train <- predict(mod1, s = lambda.lasso, newx = x)
mod1.test <- predict(mod1, s = lambda.lasso, newx = x.test)


# Training
mean((mod1.train - college.train$Expend)^2)
```

```
## [1] 8160593
```

```r
# Testing
mean((mod1.test - college.test$Expend)^2)
```

```
## [1] 10170911
```

```r
# mse <- function(x,y){mean((x-y)^2)}
# mse.1 <- mse(model.lasso$fit[, lambda.id], y)
# mse.1
```

Training MSE is 8160593 and testing MSE is 10170911.

```r
# set.seed(1128)
# # Testing
# x = model.matrix(Expend ~ ., data = college.test)
# y = college.test$Expend
#
# i = seq(10, -2, length = 100)
# lambda.v = 10^i
#
# model.ridge <- cv.glmnet(x, y, alpha = 1, lambda = lambda.v, type.measure = 'mse', keep=TRUE)
# lambda.lasso = model.ridge$lambda.min
# lambda.id <- which(model.ridge$lambda == model.ridge$lambda.min)
# model.ridge
# summary(model.ridge)
#
#
#
# mse <- function(x,y){mean((x-y)^2)}
# mse.1 <- mse(model.ridge$fit[, lambda.id], y)
# mse.1
```

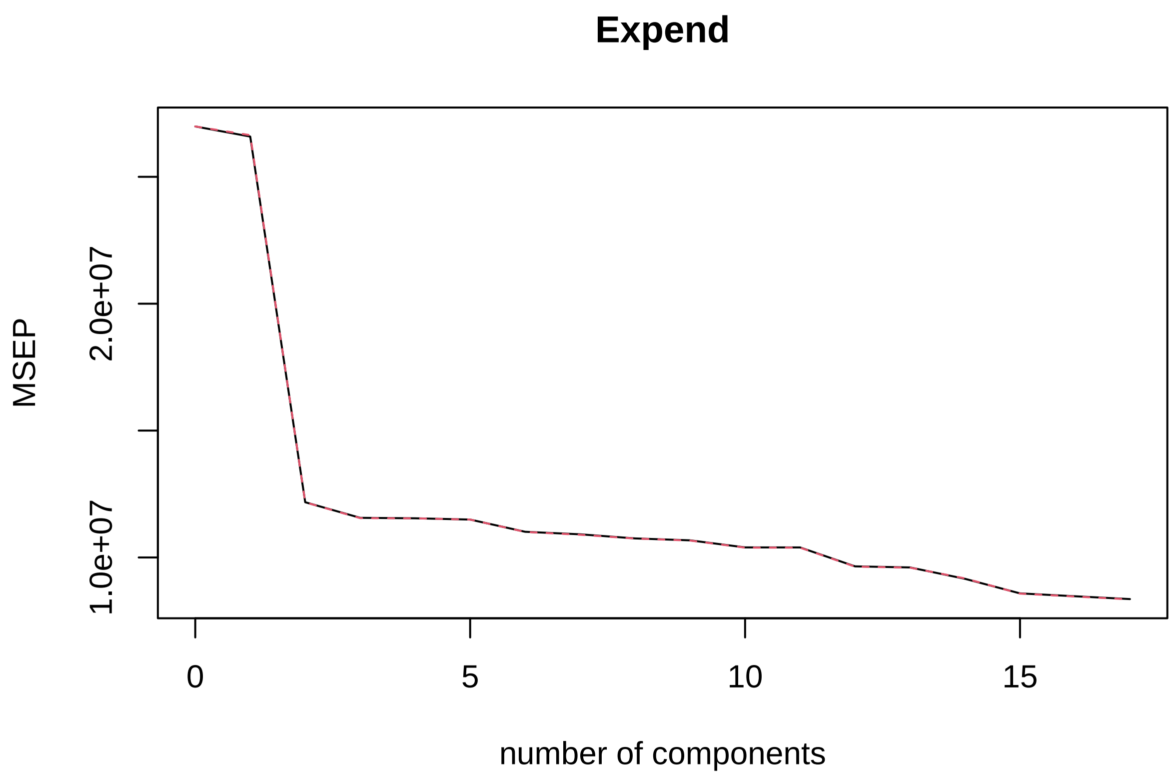
Q2) Use the same data sets in Question 1 to:

a) Fit a PCR model on the training set, with M principal components chosen by cross validation. Report the MSE obtained using both data sets (training and testing). along with the value of M principal components selected by cross-validation. Report the amount of variation explained in the X matrix by those M principal component.

```r
# install.packages("pls")
library(pls)
```

```
## Warning: package 'pls' was built under R version 4.1.2

##
## Attaching package: 'pls'

## The following object is masked from 'package:stats':
##
##     loadings
```

```r
pcr.college <-pcr(Expend ~., data = college.train, scale = TRUE, validation = "CV")

validationplot(pcr.college, val.type = "MSEP")
```



From the graph above, we can see that the value of M principal components by cross validation is 2. We will further complete our computations.

Next, we will report the MSE obtained using both data sets (training and testing)

```r
# Testing
pcr.pred.test <- predict(pcr.college, college.test, ncomp = 7)

mean((pcr.pred.test - college.test$Expend)^2)
```

```
## [1] 12321549
```

```r
# Training
pcr.pred.train <- predict(pcr.college, college.train, ncomp = 7)
mean((pcr.pred.train - college.train$Expend)^2)
```

```
## [1] 10794544
```

Now that we have assessed the MSEs of both data sets, we will look into the amount of variation explained in the X matrix by those M principal component.

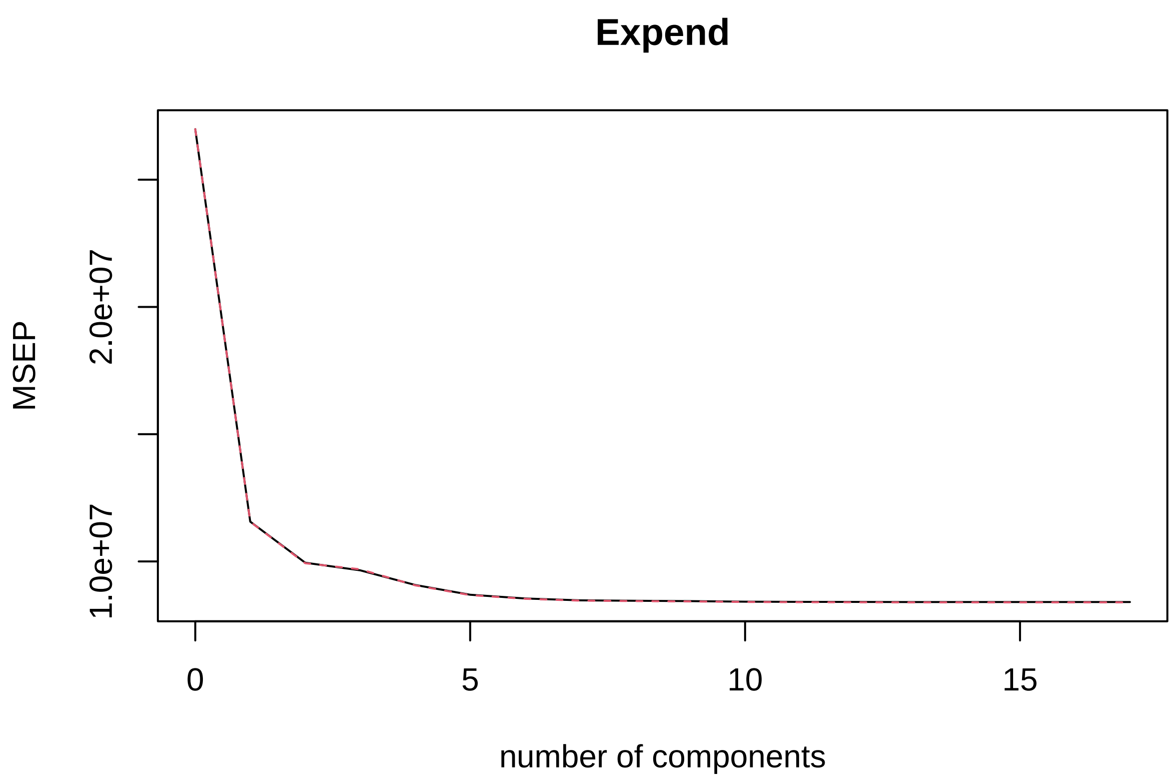
```r
summary(pcr.college)
```

```
## Data:    X dimension: 1400 17
##  Y dimension: 1400 1
## Fit method: svdpc
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            5195     5156     3489     3400     3398     3390     3318
## adjCV         5195     5161     3489     3399     3397     3390     3318
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        3303     3279     3267      3224      3224      3106      3099
## adjCV     3301     3278     3266      3223      3223      3105      3098
##        14 comps  15 comps  16 comps  17 comps
## CV         3026      2930      2910      2891
## adjCV      3029      2928      2909      2889
##
## TRAINING: % variance explained
##         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X        30.870    59.79    66.49    72.28    77.71    82.34    85.77    89.02
## Expend    1.662    55.00    57.29    57.36    57.53    59.47    59.94    60.56
##         9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X         91.91     94.31     96.28     97.46     98.40     99.04     99.64
## Expend    60.96     62.03     62.06     64.85     65.02     66.55     68.75
##         16 comps  17 comps
## X          99.87    100.00
## Expend     69.33     69.72
```

Using 85% as our threshold for the amount of variation in the X matrix, we will have that the variation explained is with 7 components and 59.94% of the variation explained.

b) Fit a PLS model on the training set, with M principal components chosen by cross validation. Report the MSE obtained using both data sets (training and testing), along with the value of M principal components selected by cross-validation. Report the amount of variation explained in the X matrix by those M principal component. Note: Use 85% as your threshold for the amount of variation in the X matrix

```
pls.college <- plsr(Expend ~ ., data = college.train, scale = TRUE, validation = "CV")

validationplot(pls.college, val.type = "MSEP")
```



From the graph above, we can see that the value of M principal components by cross validation is 2. We will further complete our computations.

Next, we will report the MSE obtained using both data sets (training and testing)

```
# Testing
pls.pred.test <- predict(pls.college, college.test, ncomp = 10)
mean((pls.pred.test - college.test$Expend)^2)
```

```
## [1] 10189596
```

```r
# Training
pls.pred.train <- predict(pls.college, college.train, ncomp = 10)
mean((pls.pred.train - college.train$Expend)^2)
```

```
## [1] 8177409
```

Now that we have assessed the MSEs of both data sets, we will look into the amount of variation explained in the X matrix by those M principal component.

```r
summary(pls.college)
```

```
## Data:    X dimension: 1400 17
##  Y dimension: 1400 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            5195     3400     3154     3107     3012     2948     2923
## adjCV         5195     3400     3151     3113     3010     2946     2921
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        2910     2908     2905      2901      2900      2900      2899
## adjCV     2908     2906     2903      2900      2898      2898      2897
##        14 comps  15 comps  16 comps  17 comps
## CV         2899      2899      2899      2899
## adjCV      2897      2897      2897      2897
##
## TRAINING: % variance explained
##         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X          28.9    35.75    63.11    68.62    71.83    74.83    77.76    81.22
## Expend     57.5    63.80    64.72    67.28    68.69    69.17    69.39    69.50
##         9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X         84.03     86.55     90.02     92.86     94.30     96.19     97.96
## Expend    69.57     69.65     69.70     69.70     69.71     69.71     69.72
##         16 comps  17 comps
## X          99.30    100.00
## Expend     69.72     69.72
```

Using 85% as our threshold for the amount of variation in the X matrix, we will have that the variation explained is with 10 components and 69.65% of the variation explained.

Q3) This question relates to the College data sets in question 1.

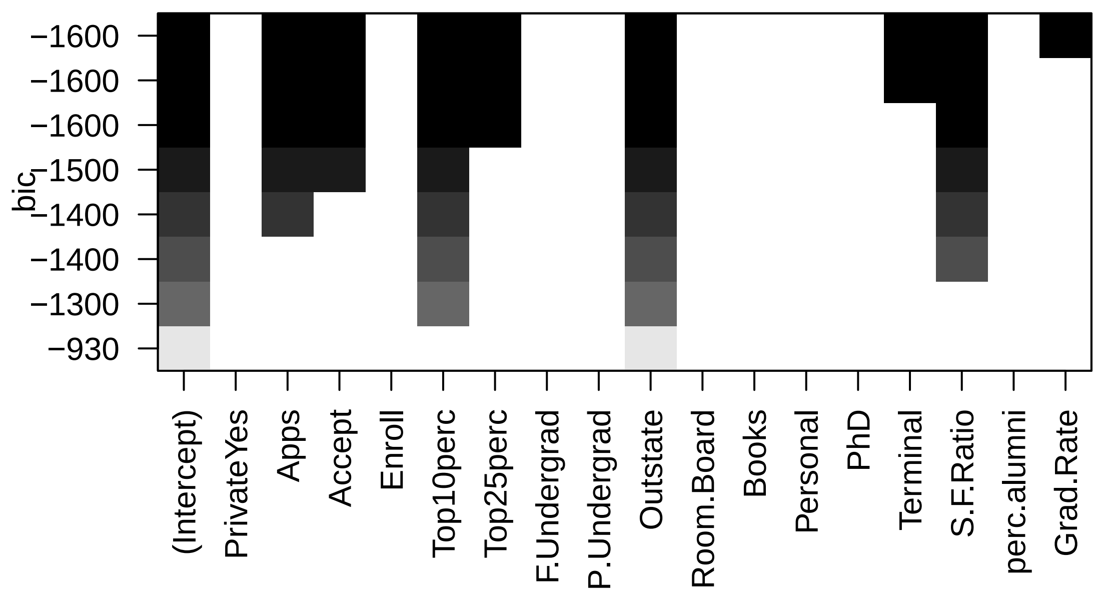
(a) Using "Expend" as the response and the other variables as predictors, perform backward stepwise selection (Choose BIC as your criteria) on the training set in order to identify a satisfactory model that uses just a subset of the predictors.

```r
library(leaps)

regfit.bck <- regsubsets(Expend ~ ., data = college.train, method = "backward")
summary(regfit.bck)
```

```
## Subset selection object
## Call: regsubsets.formula(Expend ~ ., data = college.train, method = "backward")
## 17 Variables  (and intercept)
##             Forced in Forced out
## PrivateYes      FALSE      FALSE
## Apps            FALSE      FALSE
## Accept          FALSE      FALSE
## Enroll          FALSE      FALSE
## Top10perc       FALSE      FALSE
## Top25perc       FALSE      FALSE
## F.Undergrad     FALSE      FALSE
## P.Undergrad     FALSE      FALSE
## Outstate        FALSE      FALSE
## Room.Board      FALSE      FALSE
## Books           FALSE      FALSE
## Personal        FALSE      FALSE
## PhD             FALSE      FALSE
## Terminal        FALSE      FALSE
## S.F.Ratio       FALSE      FALSE
## perc.alumni     FALSE      FALSE
## Grad.Rate       FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: backward
##          PrivateYes Apps Accept Enroll Top10perc Top25perc F.Undergrad
## 1  ( 1 ) " "        " "  " "    " "    " "       " "       " "
## 2  ( 1 ) " "        " "  " "    " "    "*"       " "       " "
## 3  ( 1 ) " "        " "  " "    " "    "*"       " "       " "
## 4  ( 1 ) " "        "*"  " "    " "    "*"       " "       " "
## 5  ( 1 ) " "        "*"  "*"    " "    "*"       " "       " "
## 6  ( 1 ) " "        "*"  "*"    " "    "*"       "*"       " "
## 7  ( 1 ) " "        "*"  "*"    " "    "*"       "*"       " "
## 8  ( 1 ) " "        "*"  "*"    " "    "*"       "*"       " "
##          P.Undergrad Outstate Room.Board Books Personal PhD Terminal S.F.Ratio
## 1  ( 1 ) " "         "*"      " "        " "   " "      " " " "      " "
## 2  ( 1 ) " "         "*"      " "        " "   " "      " " " "      " "
## 3  ( 1 ) " "         "*"      " "        " "   " "      " " " "      "*"
## 4  ( 1 ) " "         "*"      " "        " "   " "      " " " "      "*"
## 5  ( 1 ) " "         "*"      " "        " "   " "      " " " "      "*"
## 6  ( 1 ) " "         "*"      " "        " "   " "      " " " "      "*"
## 7  ( 1 ) " "         "*"      " "        " "   " "      " " "*"      "*"
## 8  ( 1 ) " "         "*"      " "        " "   " "      " " "*"      "*"
##          perc.alumni Grad.Rate
## 1  ( 1 ) " "         " "
## 2  ( 1 ) " "         " "
## 3  ( 1 ) " "         " "
## 4  ( 1 ) " "         " "
## 5  ( 1 ) " "         " "
## 6  ( 1 ) " "         " "
```

```
## 7  ( 1 ) " "         " "
## 8  ( 1 ) " "         "*"
```

```r
plot(regfit.bck, scale = "bic")
```




```r
# out <- summary(regsubsets(Expend ~ ., data = college.train, method = "backward"))
# qplot(1:10, out$bic) + geom_line()
```

We assess which predictors to examine through looking at the predictors where the bar touches the top of the graph which are Apps, Accept, Top10perc, Top25perc, Outstate, Terminal, S.F.Ratio, Grad.Rate

```r
# The satisfactory model is below
college.train.1 <- college.train[,c(2,3,5,6,9,14,15,17,18)]
college.model <- lm(Expend ~ ., data = college.train.1)
summary(college.model)
```

```
##
## Call:
## lm(formula = Expend ~ ., data = college.train.1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10166.6  -1437.4   -258.8    793.6  29474.3
##
```

```
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6630.17593  664.30790   9.981  < 2e-16 ***
## Apps           0.82166    0.07418  11.077  < 2e-16 ***
## Accept        -0.91879    0.10855  -8.464  < 2e-16 ***
## Top10perc    127.59278   10.51691  12.132  < 2e-16 ***
## Top25perc    -63.46323    8.66851  -7.321 4.15e-13 ***
## Outstate       0.52331    0.02873  18.216  < 2e-16 ***
## Terminal      33.81667    6.57410   5.144 3.08e-07 ***
## S.F.Ratio   -292.72328   23.63735 -12.384  < 2e-16 ***
## Grad.Rate    -23.29867    5.89937  -3.949 8.23e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2891 on 1391 degrees of freedom
## Multiple R-squared:  0.6919, Adjusted R-squared:  0.6901
## F-statistic: 390.4 on 8 and 1391 DF,  p-value: < 2.2e-16
```

To find the MSE of the backwards stepwise BIC model, we have

```
pred.train <- predict(college.model, college.train)
pred.test <- predict(college.model, college.test)


# MSE for the training data set
sum((college.train$Expend - pred.train)^2)/length(pred.train)
```

```
## [1] 8302748
```

```
# MSE for the testing data set
sum((college.test$Expend - pred.test)^2)/length(pred.test)
```

```
## [1] 10200996
```

The MSE for training is 8302748 and the MSE for testing is 10200996

(b) Fit a GAM on the training data, using "Expend" as the response and the features selected in the previous step (Part a) as your predictors. Plot the results, and explain your findings.

```
library(splines)
library(ISLR)
# install.packages("gam")
library(gam)
```

```
## Warning: package 'gam' was built under R version 4.1.2
```

```
## Loading required package: foreach
```

```
## Warning: package 'foreach' was built under R version 4.1.2
```

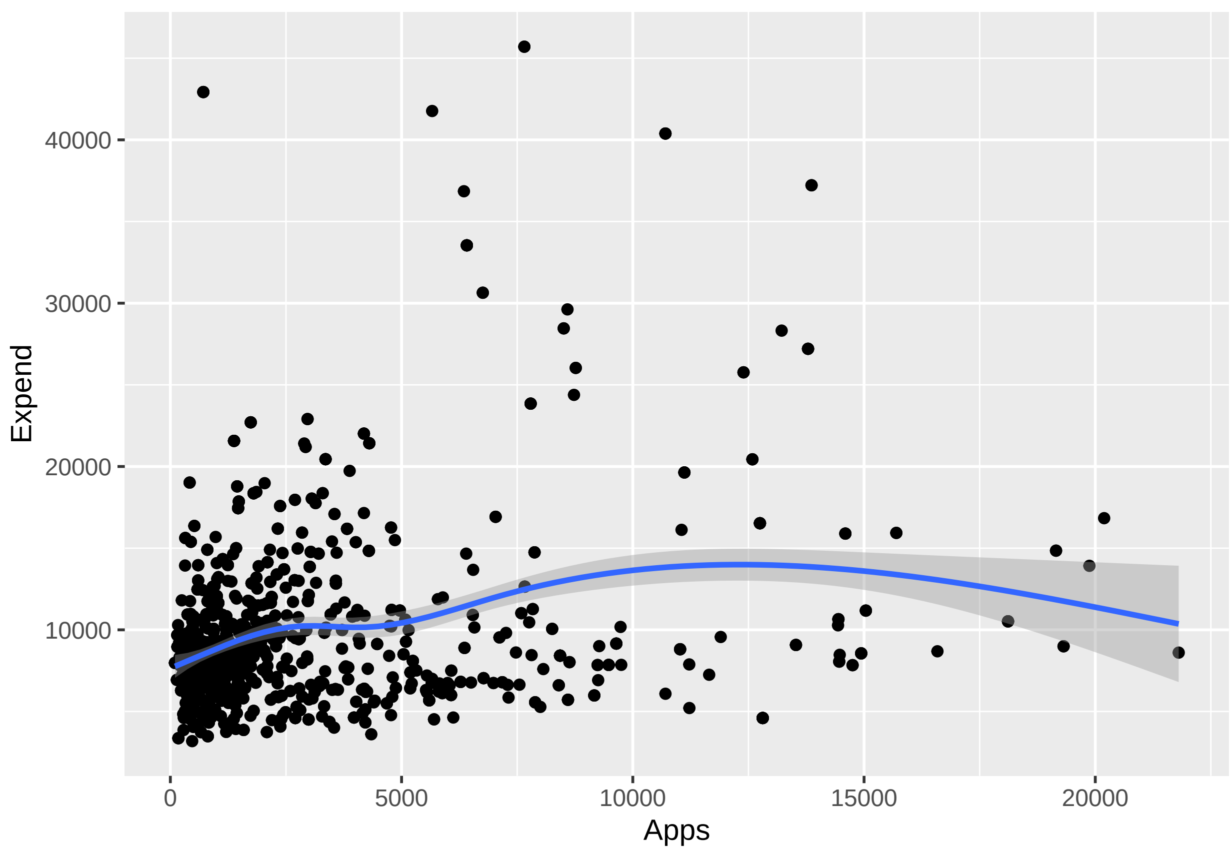
```
## Loaded gam 1.22
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.2
```

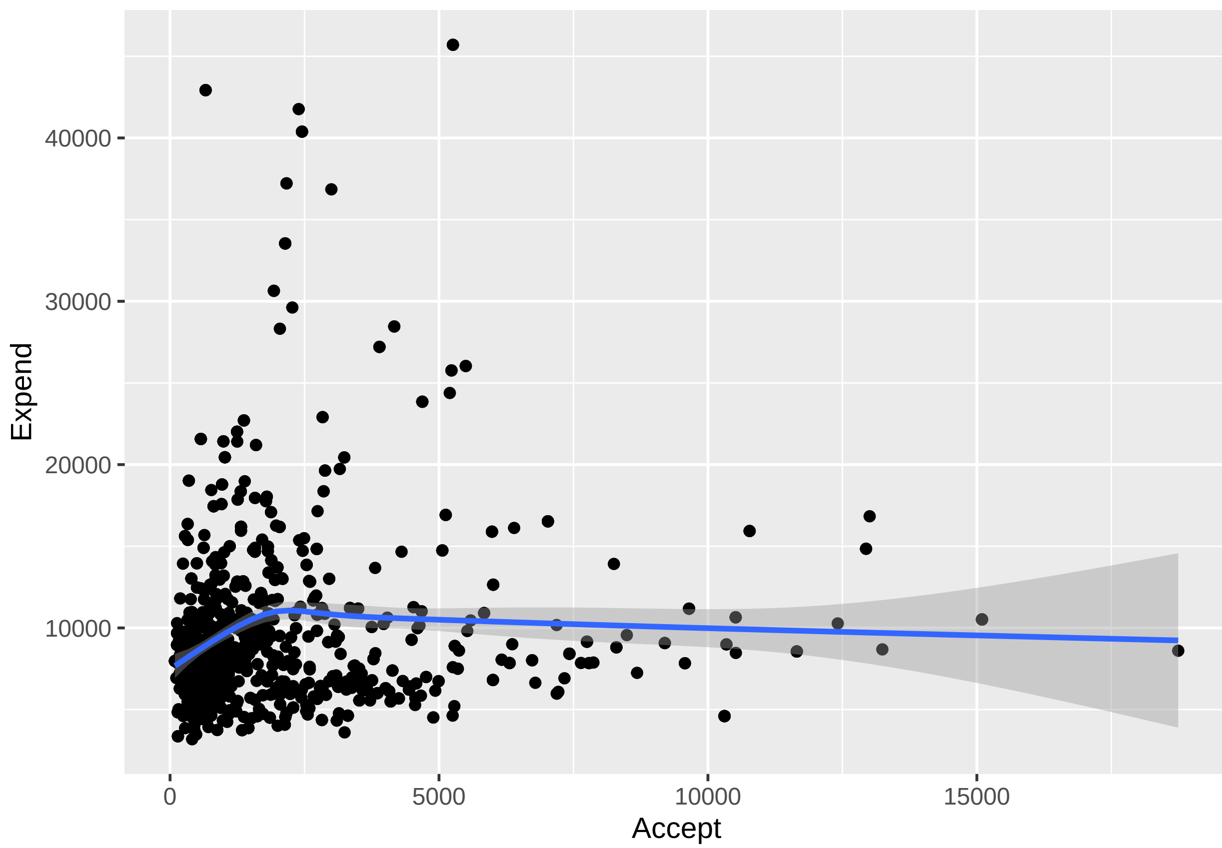
```
ggplot(data= college.train.1, aes(Apps, Expend)) + geom_point() + geom_smooth()
```

```
## 'geom_smooth()' using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



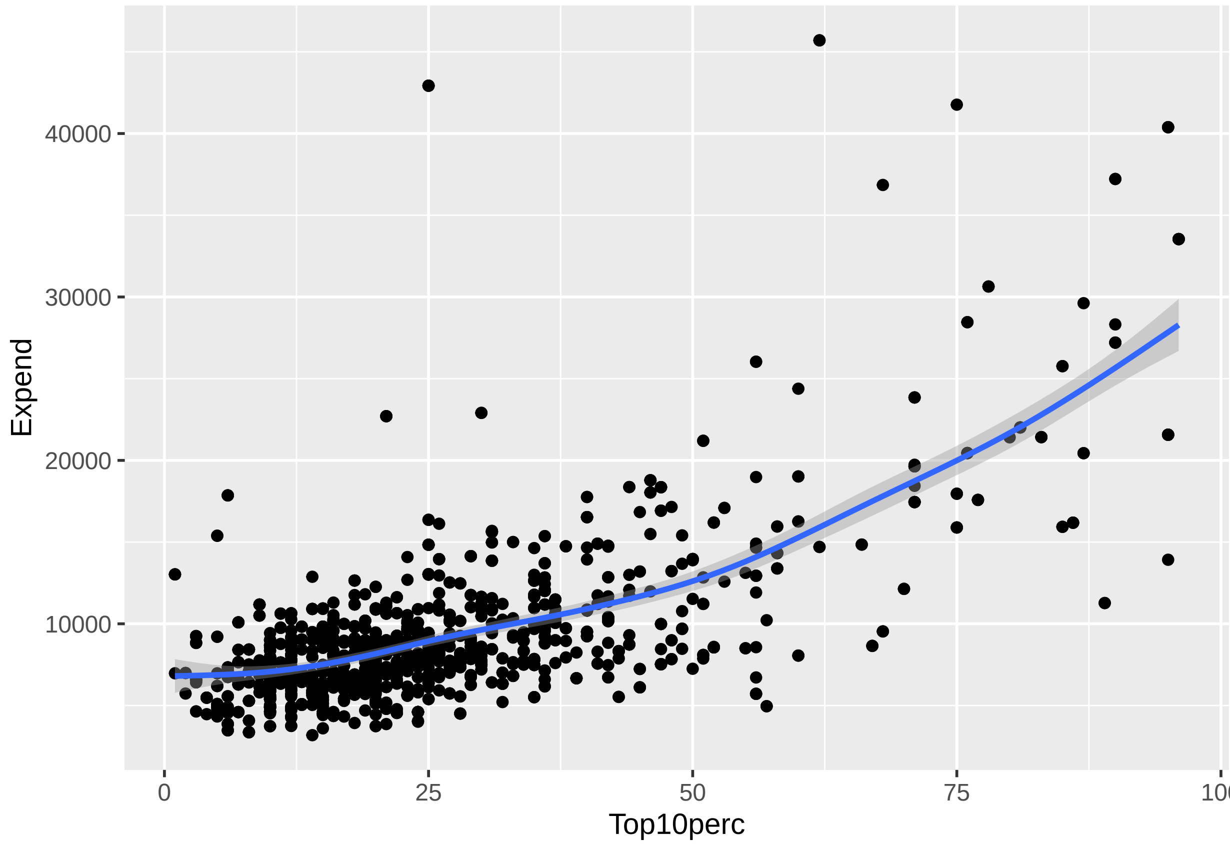

```
ggplot(data= college.train.1, aes(Accept, Expend)) + geom_point() + geom_smooth()
```

```
## 'geom_smooth()' using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



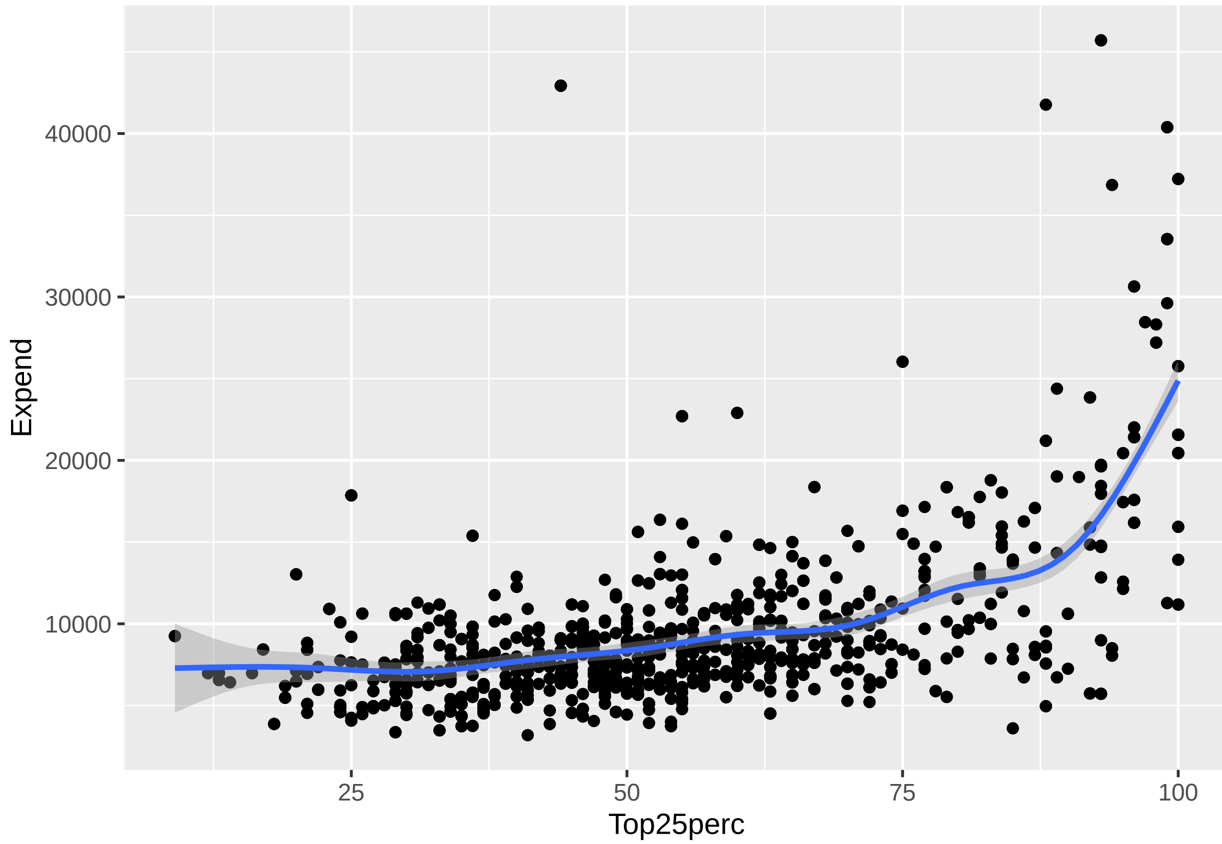

```
ggplot(data= college.train.1, aes(Top10perc, Expend)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



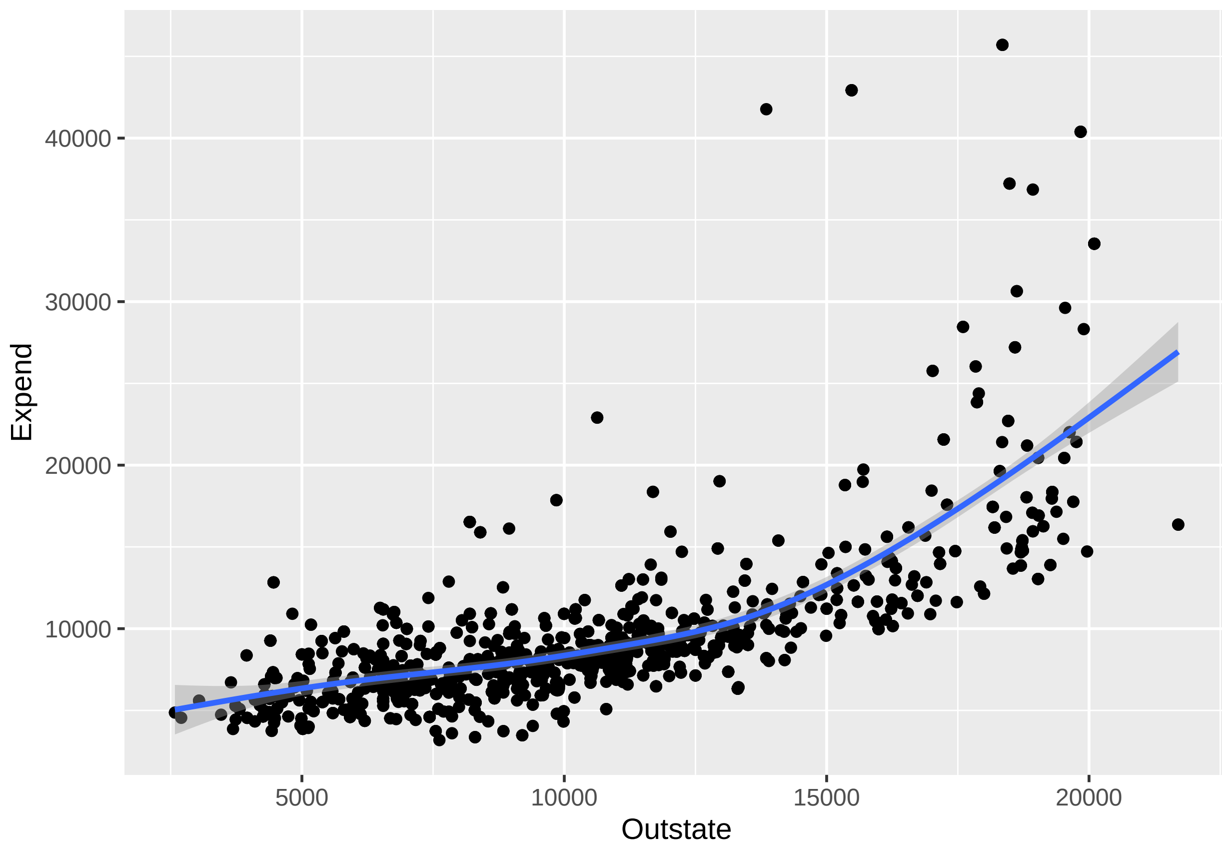

```
ggplot(data= college.train.1, aes(Top25perc, Expend)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



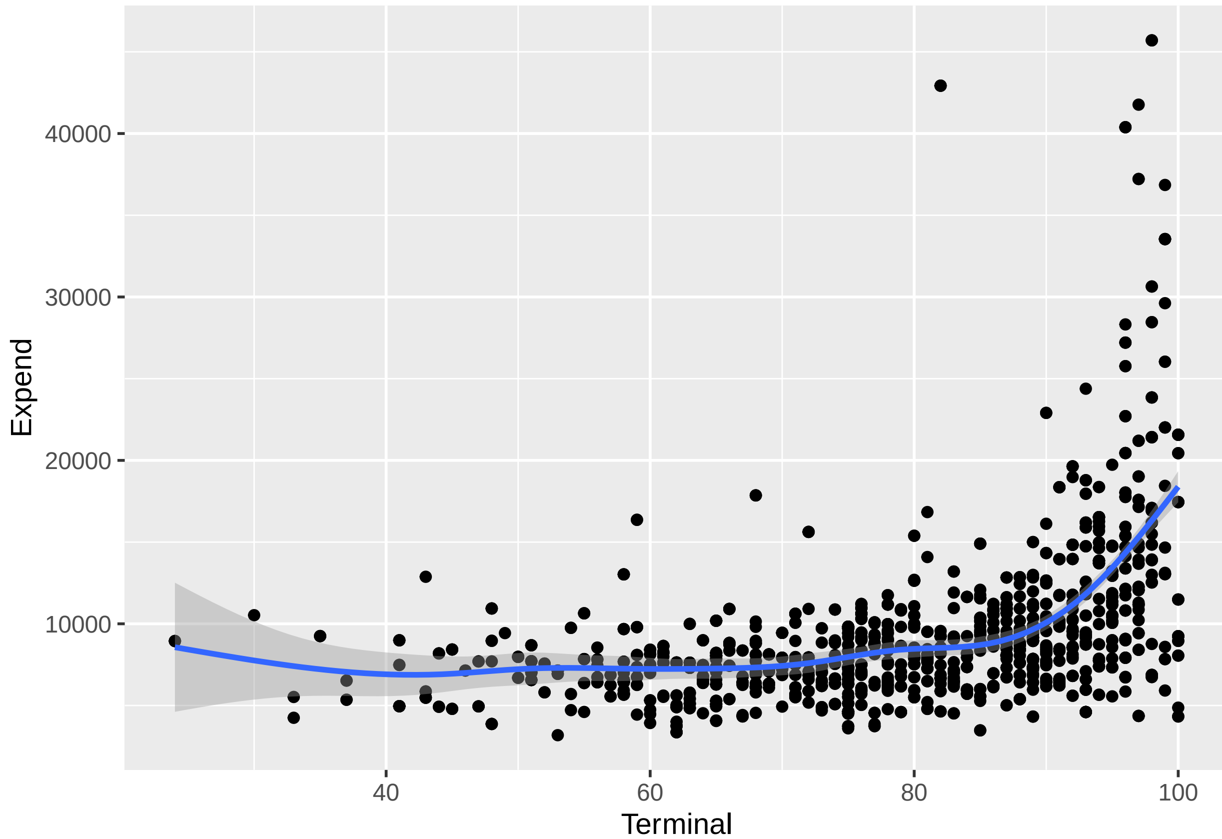

```r
ggplot(data= college.train.1, aes(Outstate, Expend)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



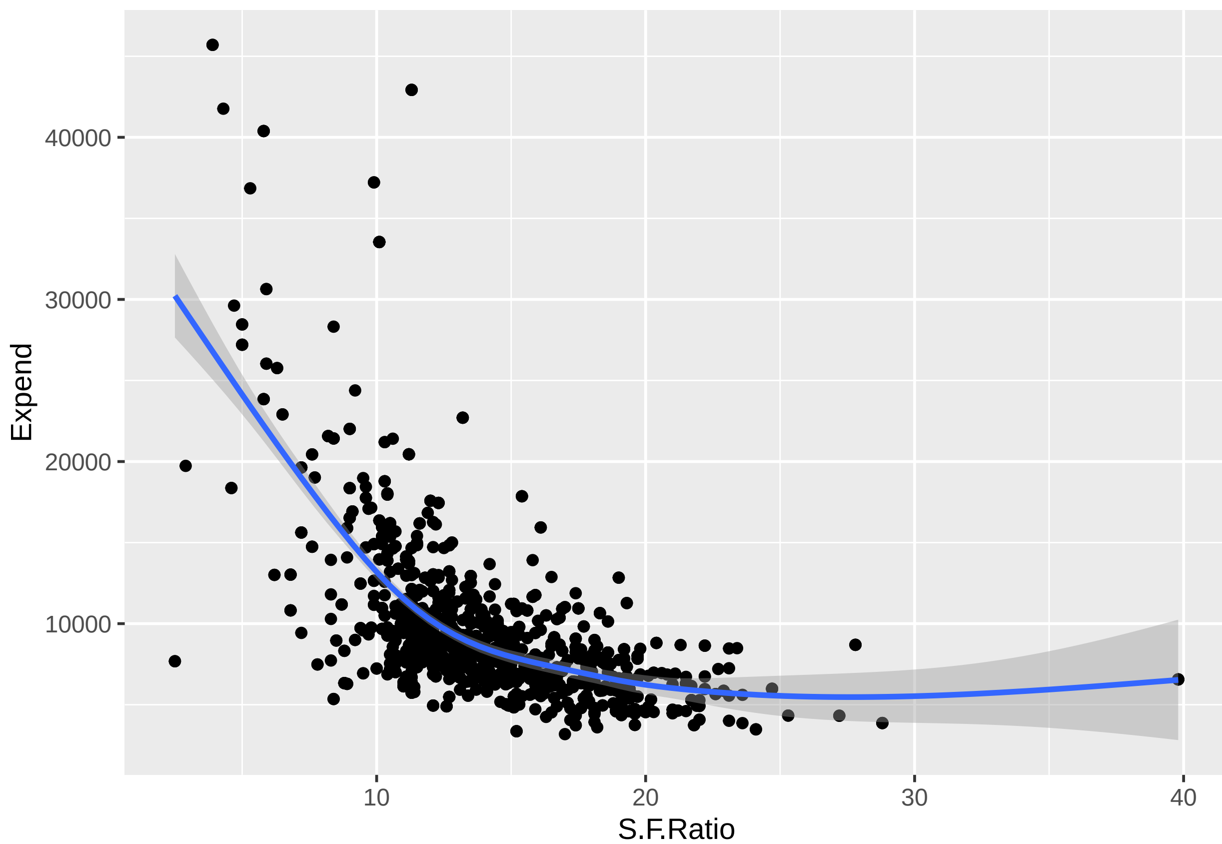

```
ggplot(data= college.train.1, aes(Terminal, Expend)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



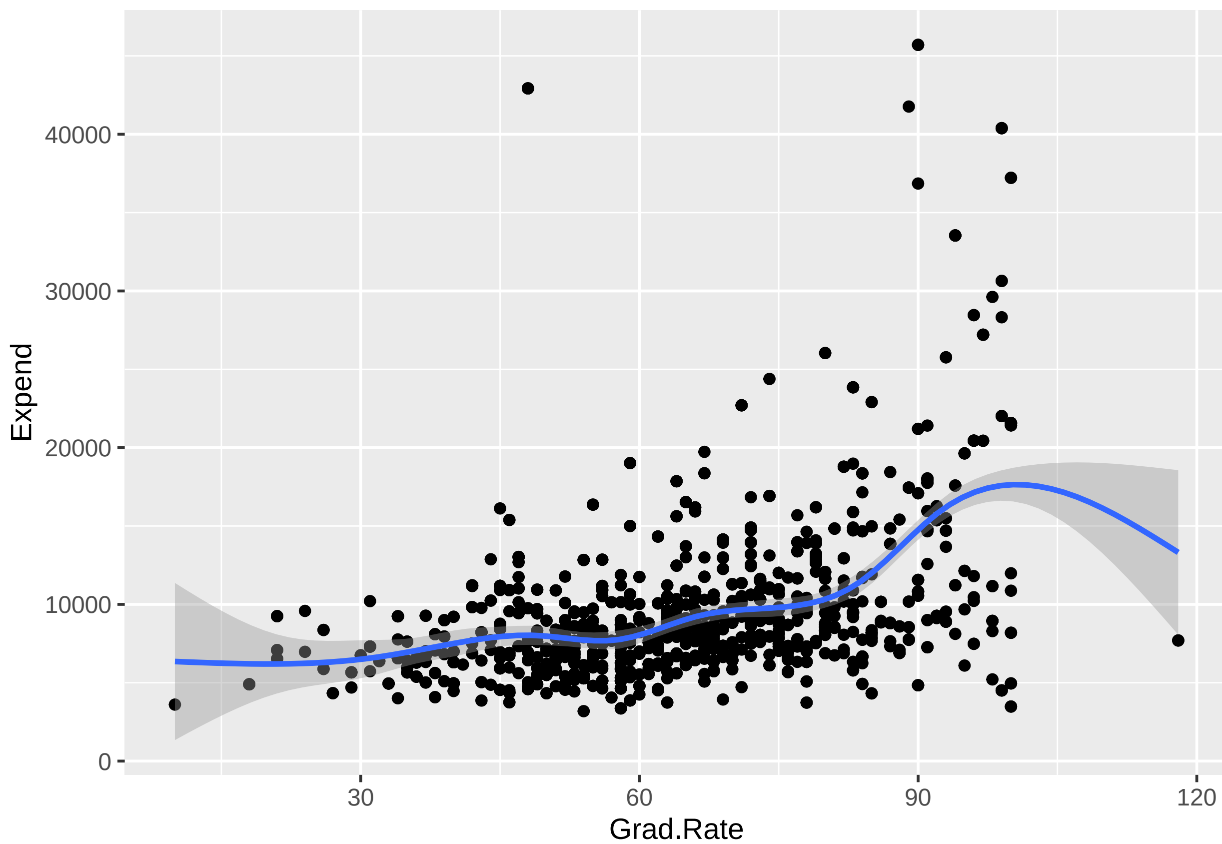

```r
ggplot(data= college.train.1, aes(S.F.Ratio, Expend)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```




```
ggplot(data= college.train.1, aes(Grad.Rate, Expend)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```




```
gam0 <- lm(Expend ~ bs(Apps, 4) + Accept + Top10perc + bs(Top25perc, 4) + Outstate + bs(Terminal, 4) + 
summary(gam0)
```

```
##
## Call:
## lm(formula = Expend ~ bs(Apps, 4) + Accept + Top10perc + bs(Top25perc,
##     4) + Outstate + bs(Terminal, 4) + bs(S.F.Ratio, 4) + bs(Grad.Rate,
##     4), data = college.train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13447.8  -1126.0   -296.1    756.8  29939.3
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.569e+04  2.185e+03   7.180 1.13e-12 ***
## bs(Apps, 4)1       8.653e+02  5.204e+02   1.663  0.09663 .
## bs(Apps, 4)2       5.133e+03  1.047e+03   4.905 1.05e-06 ***
## bs(Apps, 4)3       8.705e+03  1.747e+03   4.984 7.00e-07 ***
## bs(Apps, 4)4       7.117e+03  2.363e+03   3.012  0.00264 **
## Accept            -4.744e-01  1.102e-01  -4.305 1.79e-05 ***
## Top10perc          6.254e+01  1.160e+01   5.391 8.22e-08 ***
## bs(Top25perc, 4)1 -1.932e+03  1.199e+03  -1.612  0.10728
## bs(Top25perc, 4)2 -1.477e+03  7.993e+02  -1.848  0.06475 .
## bs(Top25perc, 4)3 -4.713e+03  1.144e+03  -4.121 4.00e-05 ***
```

```
## bs(Top25perc, 4)4 -5.378e+02  1.231e+03  -0.437  0.66220    
## Outstate            4.985e-01  2.844e-02  17.527  < 2e-16 ***
## bs(Terminal, 4)1  -2.985e+03  1.920e+03  -1.554  0.12037    
## bs(Terminal, 4)2  -2.672e+02  1.071e+03  -0.250  0.80295    
## bs(Terminal, 4)3  -5.012e+02  1.319e+03  -0.380  0.70408    
## bs(Terminal, 4)4   1.203e+03  1.226e+03   0.981  0.32659    
## bs(S.F.Ratio, 4)1 -9.980e+03  1.385e+03  -7.206 9.46e-13 ***
## bs(S.F.Ratio, 4)2 -1.716e+04  1.201e+03 -14.289  < 2e-16 ***
## bs(S.F.Ratio, 4)3 -1.486e+04  1.982e+03  -7.498 1.16e-13 ***
## bs(S.F.Ratio, 4)4 -1.462e+04  1.712e+03  -8.539  < 2e-16 ***
## bs(Grad.Rate, 4)1  4.054e+03  1.929e+03   2.102  0.03577 *  
## bs(Grad.Rate, 4)2  9.817e+02  1.168e+03   0.840  0.40097    
## bs(Grad.Rate, 4)3  1.365e+03  1.780e+03   0.767  0.44343    
## bs(Grad.Rate, 4)4  2.576e+03  1.874e+03   1.375  0.16941    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 2668 on 1376 degrees of freedom
## Multiple R-squared:  0.7404, Adjusted R-squared:  0.7361 
## F-statistic: 170.7 on 23 and 1376 DF,  p-value: < 2.2e-16
```

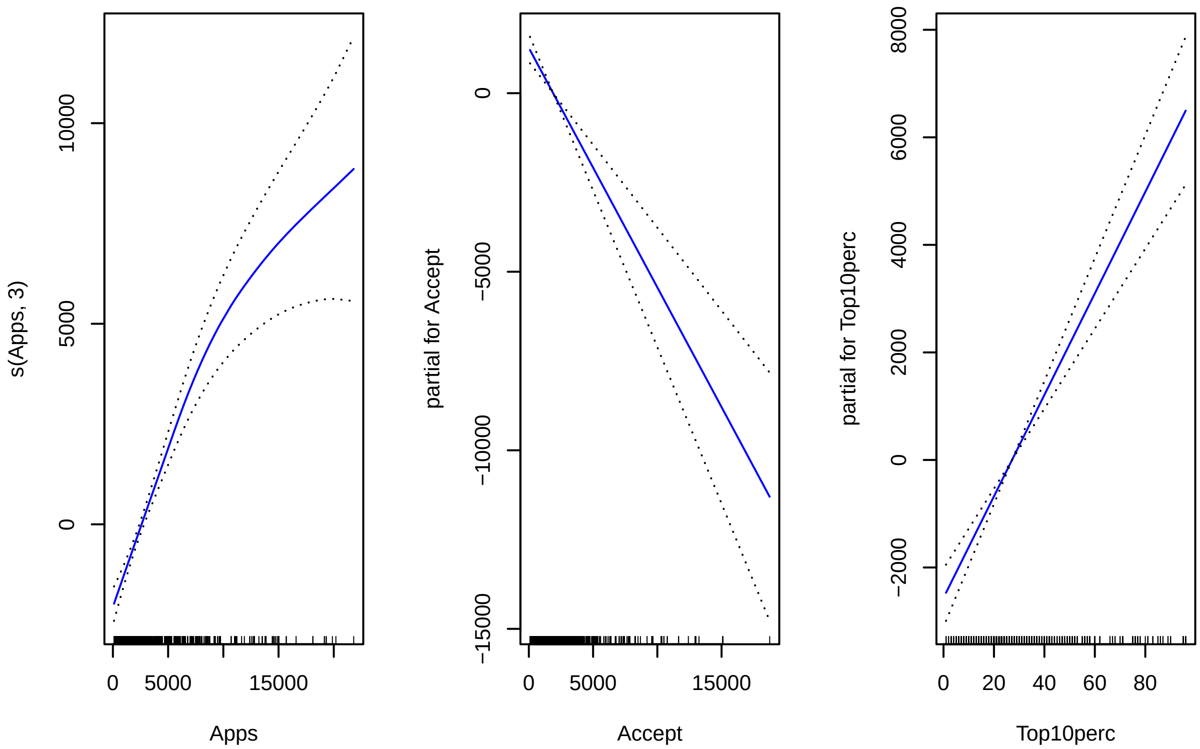
The lowest p-value polynomials are 3 for Apps, 3 for Top25perc, 1 for Terminal, 2 for S.F.Ratio, and 1 for Grad.Rate

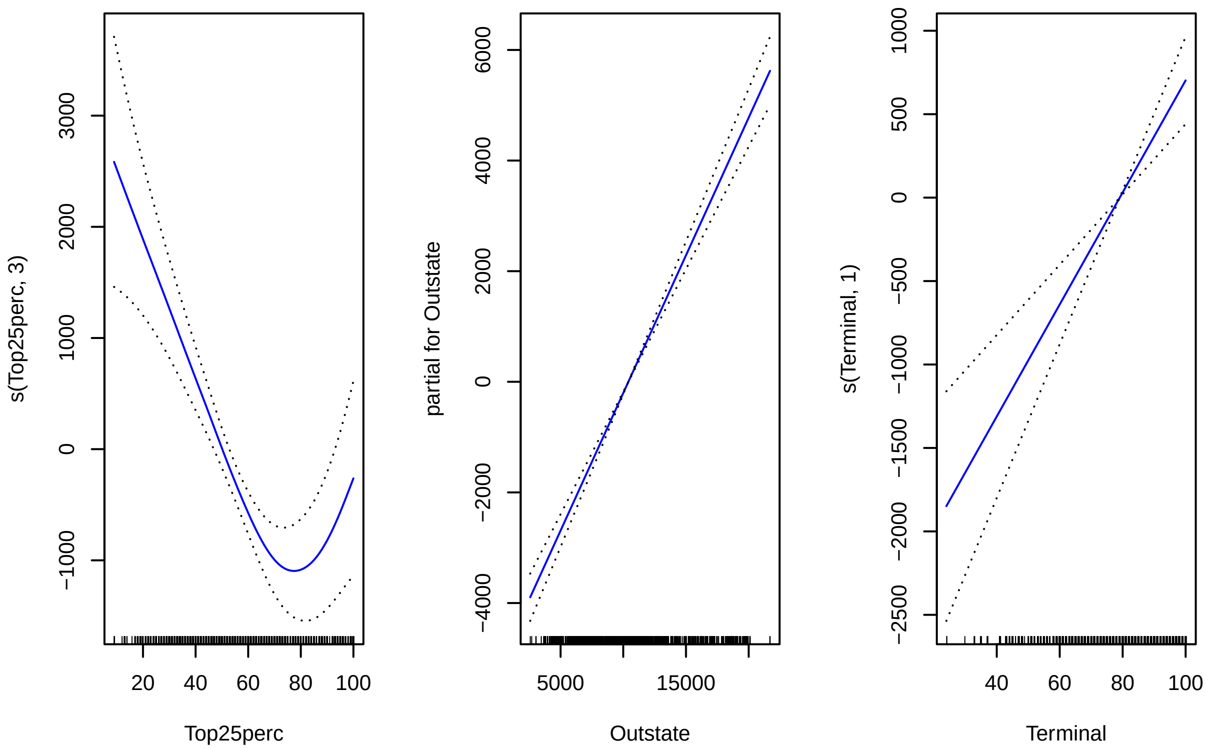
(c) Evaluate the model obtained on the testing data set, and explain the results obtained.

We will determine what coefficients to use for from the previous summary output. For the non linear components Apps, Top25perc, Terminal, S.F.Ratio, and Grad.Rate, we will utilize the polynomial with the lowest p-value. Thus, we will have the following:

```r
gam1 <- gam(Expend ~ s(Apps,3) + Accept + Top10perc + s(Top25perc,3) + Outstate + s(Terminal,1) + s(S.F

# gam1 <- gam(Expend ~ s(Apps) + s(Accept) + s(Top10perc) + s(Top25perc) + s(Outstate) + s(Terminal) + 


par(mfrow = c(1,3))
plot(gam1, se = TRUE, col = "blue")
```

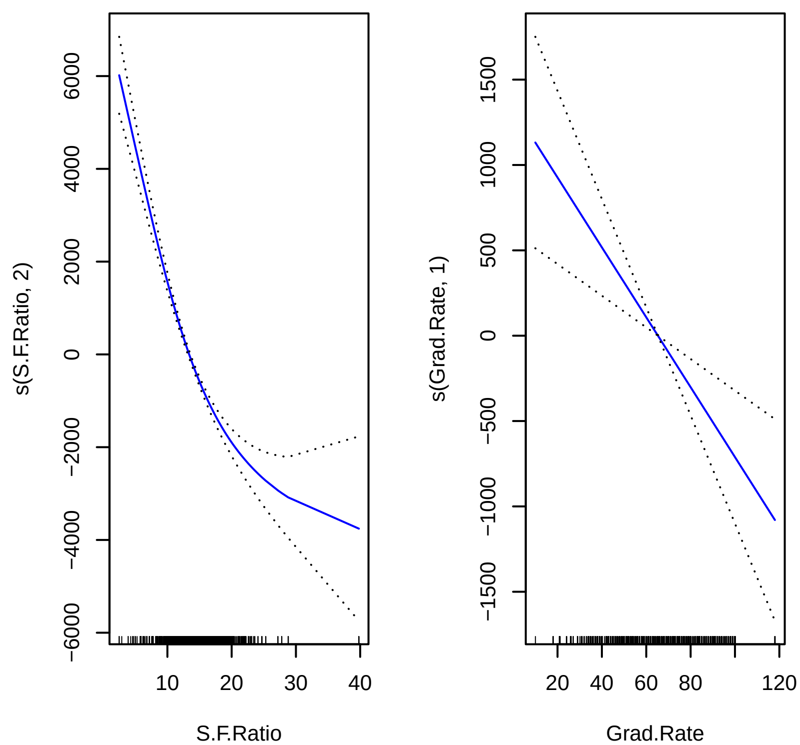
```
## Warning in pf(nl.chisq/nldf, nldf, rdf): NaNs produced
```


s(Apps, 3)
10000
5000
0
0
5000
15000
Apps
partial for Accept
0
−5000
−10000
−15000
0
5000
15000
Accept
partial for Top10perc
8000
6000
4000
2000
0
−2000
0
20
40
60
80
Top10perc




```r
pred <- predict(gam1,newdata = college.test)


# We will evaluate the model obtained on the testing data set by looking at the MSE.
# The MSE can be determined through below
MSE <- mean((college.test$Expend - pred)^2)
MSE
```

```
## [1] 8938132
```




```
# fit <- lm(Expend ~ bs(Apps, knots = c(25,40,60)), data = college)
# dim(bs(college$Expend, knots = c(25,40, 60)))

# fit.1 = lm(Expend~Apps, data = college)
# fit.2 = lm(Expend~poly(Apps,2), data = college)
# fit.3 = lm(Expend~poly(Apps,3), data = college)
# fit.4 = lm(Expend~poly(Apps,4), data = college)
# fit.5 = lm(Expend~poly(Apps,5), data = college)
# anova(fit.1,fit.2,fit.3,fit.4,fit.5)
```

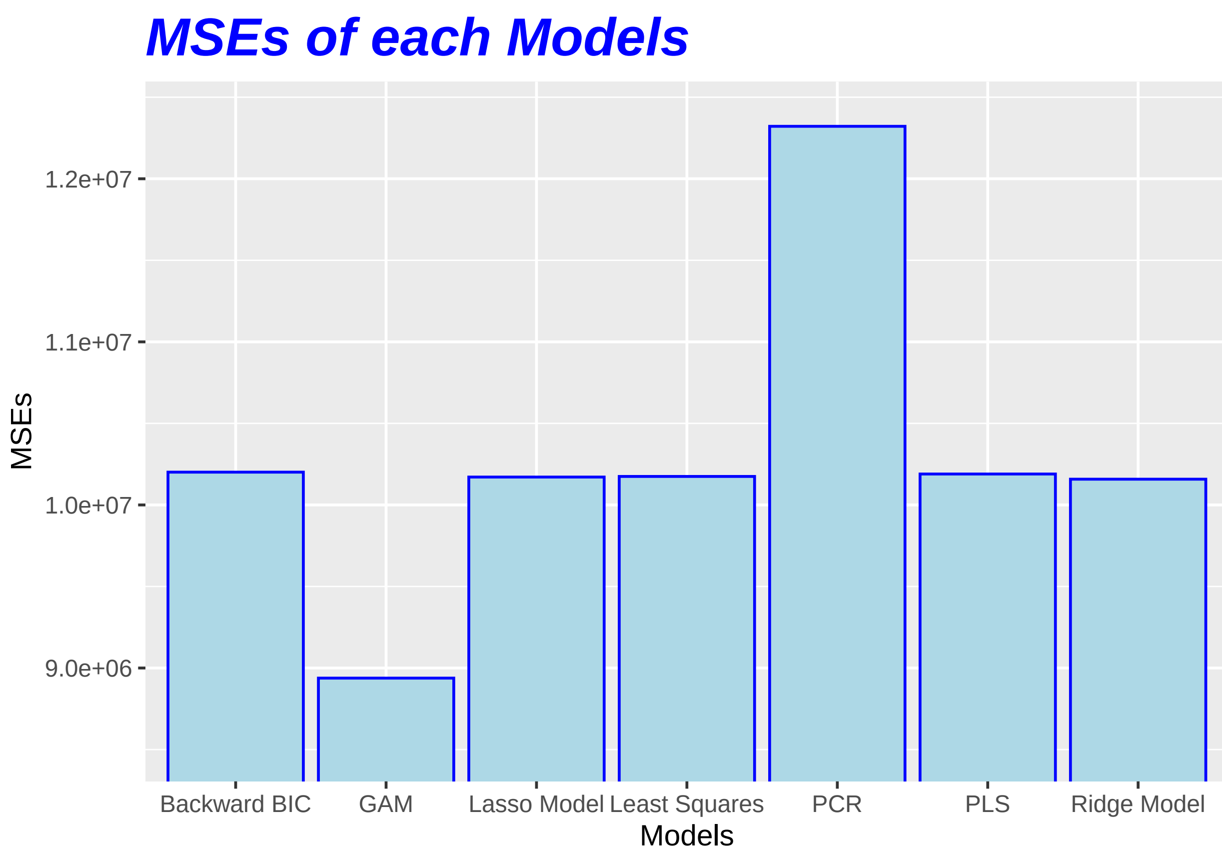
##(d) For which variables, if any, is there evidence of a non-linear relationship with the response?

The variables with a non-linear relationship with the response are Apps, Top25perc, Terminal, S.F.Ratio, Grad.Rate

Q4) Comment on the results obtained. How accurately can we predict the amount of money expend by college students? Is there much difference among the testing MSEs resulting from these seven approaches?

The MSEs for the seven approaches are as follows 1. Least Squares full model using lm 10174615 2. Ridge model with the best lambda 10157872 3. Lasso model with the best lambda 10170911 4. PCR model 12321549 5. PLS model 10189596 6. Stepwise backward regression using BIC 10200996 7. GAM 8938132

```r
temp <- c(10174615,10157872,10170911,12321549,10189596,10200996,8938132)

temprow <- c("Least Squares", "Ridge Model", "Lasso Model", "PCR", "PLS", "Backward BIC", "GAM")
tempdf <- data.frame("Models" <- temprow, "MSEs" <- temp)


library(ggplot2)
ggplot(data = tempdf, aes(Models, MSEs)) + geom_bar(stat= "identity", color = "blue", fill = "lightblue
```



In the table above, we are able to see the various MSEs by the seven models that we have generated. We can see that the GAM model has the lowest MSE, the PCR model has the highest MSE, and the other models have fairly similar MSE values.

Q5) You may have seen the "betterbirths2000" data in Stats 10. It consists of a random sample of 2000 births in North Carolina that are collected in order to track health issues in new born babies. These data are saved in a file better2000births.csv on bruinlearn Week 8.

```r
temp <- read.csv("/Users/takaooba/Downloads/better2000births.csv")
births <- na.omit(temp)
```

```
births$Marital <- factor(births$Marital)
births$Racemom <- factor(births$Racemom)
births$Racedad <- factor(births$Racedad)
births$Hispmom <- factor(births$Hispmom)
births$Hispdad <- factor(births$Hispdad)
births$Habit <- factor(births$Habit)
births$MomPriorCond <- factor(births$MomPriorCond)
births$BirthDef <- factor(births$BirthDef)
births$DelivComp <- factor(births$DelivComp)
births$BirthComp <- factor(births$BirthComp)
head(births)
```

```
##   Gender Premie weight Apgar1 Fage Mage Feduc Meduc TotPreg Visits   Marital
## 1   Male     No    124      8   31   25    13    14       1     13   Married
## 2 Female     No    177      8   36   26     9    12       2     11 Unmarried
## 3   Male     No    107      3   30   16    12     8       2     10 Unmarried
## 4 Female     No    144      6   33   37    12    14       2     12 Unmarried
## 5   Male     No    117      9   36   33    10    16       2     19   Married
## 6 Female     No     98      4   31   29    14    16       3     20   Married
##   Racemom Racedad Hispmom Hispdad Gained     Habit MomPriorCond BirthDef
## 1   White   White NotHisp NotHisp     40 NonSmoker         None     None
## 2   White   White Mexican Mexican     20 NonSmoker         None     None
## 3   White Unknown Mexican Unknown     70 NonSmoker At Least One     None
## 4   White   White NotHisp NotHisp     50 NonSmoker         None     None
## 5   White   Black NotHisp NotHisp     40 NonSmoker At Least One     None
## 6   White   White NotHisp NotHisp     21 NonSmoker         None     None
##      DelivComp BirthComp
## 1 At Least One      None
## 2 At Least One      None
## 3 At Least One      None
## 4 At Least One      None
## 5         None      None
## 6         None      None
```

a) Split your data into Training and Testing. You should have 1000 observations in your training data after omitting the missing values in your data. Use the set.seed "1128" to do the split. Use a tree (not pruned) to predict whether a baby will be born prematurely or normal. What is the testing misclassification error?

```
dim(births)
```

```
## [1] 1998   21
```

```
set.seed(1128)
test.i <- sample(1:nrow(births), 1000, replace = F)

births.test <- births[-test.i,]
births.train <- births[test.i,]
```

Using a tree

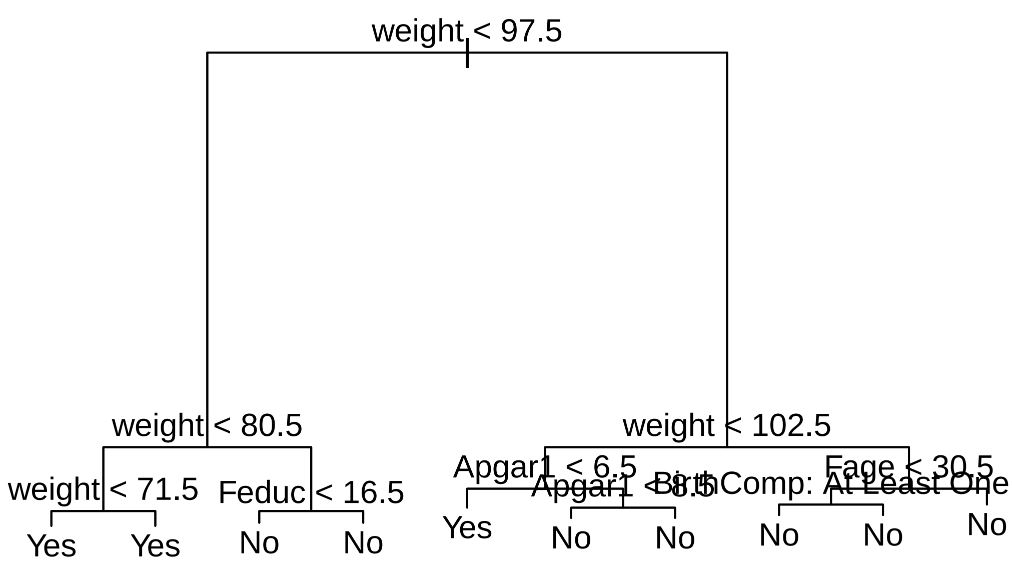
```r
# install.packages("tree")
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.1.2
```

```r
births.model <- tree(formula = factor(Premie) ~ ., data = births.train)
```

```
## Warning in tree(formula = factor(Premie) ~ ., data = births.train): NAs
## introduced by coercion
```

```r
plot(births.model)
text(births.model, pretty =0)
```




```r
births.y <- births.test$Premie

preds <- predict(births.model, newdata = births.test, type = "class")
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```r
conf.matrx <- table(preds, factor(births.test$Premie))

conf.matrx
```

```
##
## preds  No Yes
##   No  895  49
##   Yes  13  41
```

```r
(conf.matrx[2,1] + conf.matrx[1,2])/sum(conf.matrx)
```

```
## [1] 0.06212425
```

b) Use cross-validation to determine if the tree can be improved through pruning. If so, prune the tree to the appropriate size and provide a plot.

```r
cv.train <- cv.tree(births.model, FUN = prune.misclass)
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

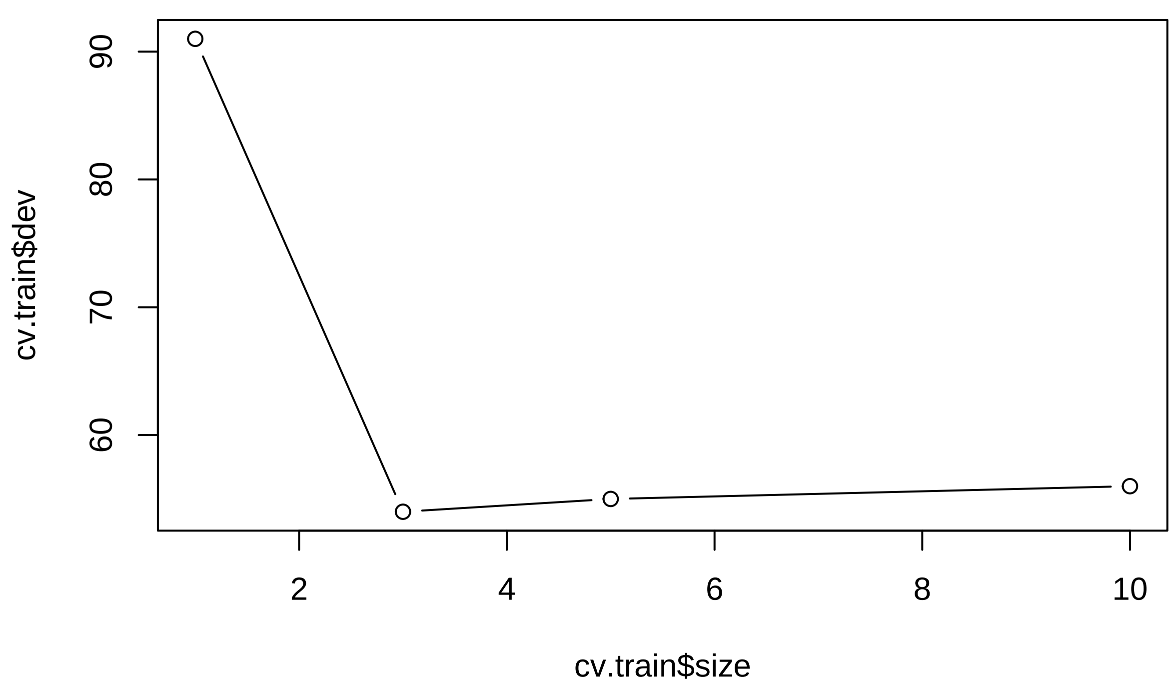
```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
names(cv.train)
```

```
## [1] "size"   "dev"    "k"      "method"
```

```
# Find best value using the plot function
plot(cv.train$dev ~ cv.train$size, type = "b")
```



```
pruned.fit <- prune.misclass(births.model, best = 3)
plot(pruned.fit)
text(pruned.fit, pretty = TRUE)
```

We will see if pruning makes the misclassification rate better. Recall that without pruning, we had a misclassification rate of 0.06212425

```
summary(pruned.fit)
```

```
##
## Classification tree:
## snip.tree(tree = births.model, nodes = c(4L, 5L, 3L))
## Variables actually used in tree construction:
## [1] "weight"
## Number of terminal nodes:  3
## Residual mean deviance:  0.322 = 321.1 / 997
## Misclassification error rate: 0.054 = 54 / 1000
```

```
# births.train
```

Now, we see that the misclassification rate is 0.056 which is lower than the misclassification rate without pruning.

c) Interpret your pruned tree (or your tree in (a) if you did not need to prune). In particular, does it tell us whether smoking is a potential cause of premature births? What factors are associated with premature births?

Basing on the pruned tree, we have that the factors that are associated with premature births is Weights. The pruned plot have performed fairly well with a low misclassification rate. Intuitively, smoking should be a potential cause of premature births, but in this decision tree, we were not able to conclude that smoking is directly associated with premature births. However, according to CDC.gov (https://www.cdc.gov/tobacco/campaign/tips/diseases/pregnancy.html#:~:text=Smoking%20slows%20your%20baby's%20growth,babies%20often%20have% we have that smoking slows the baby's growth before birth, or more specifically, the weight of the baby. The factor that is most associated with premature births is Weights.

d) What is the testing misclassification error rate of your pruned tree? Keep in mind that approximately 9% of all births are premature. This means that if a doctor simply predict "not premature" ALWAYS, he or she will have only a 9% misclassification error. Did you do better based on your tree models?

```
preds <- predict(pruned.fit, newdata = births.test, type = "class")
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```
conf.matrx <- table(preds, factor(births.test$Premie))

conf.matrx
```

```
##
## preds  No Yes
##   No  901  50
##   Yes   7  40
```

```
(conf.matrx[2,1] + conf.matrx[1,2])/sum(conf.matrx)
```

```
## [1] 0.05711423
```

The misclassification rate for testing is 0.05711423. Since the misclassification rate is 5.71%, which is less than the 9% misclassification error, we have that this tree models is better than simply predicting "not premature" for all the babies.