p8106 hw2

Yijing Tao yt2785

2022-03-02

```r
library(tidyverse)
library(readxl)
library(ISLR)
library(glmnet)
library(caret)
library(corrplot)
library(plotmo)
library(mgcv)
library(earth)
```

```r
college_df_nores = read_csv("./College.csv") %>%
  data.frame() %>%
  na.omit() %>%
  select(-Outstate)
```

```
## Rows: 565 Columns: 18

## -- Column specification --------------------------------------------
------------
## Delimiter: ","
## chr  (1): College
## dbl (17): Apps, Accept, Enroll, Top10perc, Top25perc, F.Undergrad, P.
Undergr...

##
## i Use `spec()` to retrieve the full column specification for this da
ta.
## i Specify the column types or set `show_col_types = FALSE` to quiet
this message.
```

```r
college_df_res = read_csv("./College.csv") %>%
  data.frame() %>%
  na.omit() %>%
  select(Outstate)
```

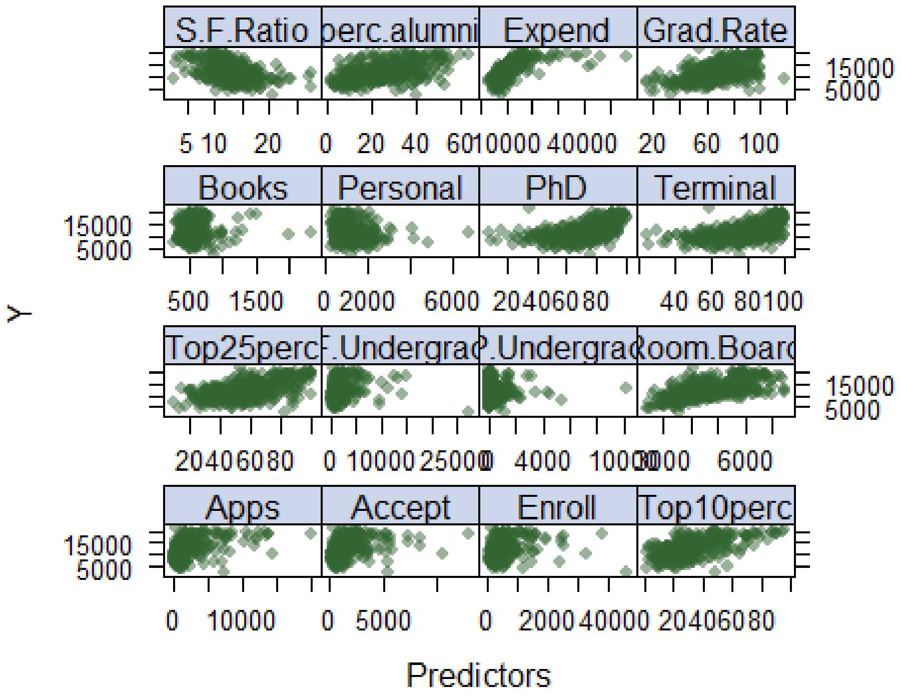
```
## Rows: 565 Columns: 18

## -- Column specification --------------------------------------------
------------
## Delimiter: ","
## chr  (1): College
## dbl (17): Apps, Accept, Enroll, Top10perc, Top25perc, F.Undergrad, P.
Undergr...
```

```
##
## i Use `spec()` to retrieve the full column specification for this da
ta.
## i Specify the column types or set `show_col_types = FALSE` to quiet
this message.
```

```
college_df = cbind(college_df_nores, college_df_res) %>%
  data.frame()

college_df2 <- model.matrix(Outstate ~ ., college_df)[ ,-1]

set.seed(2022)
trainRows <- createDataPartition(college_df$Outstate, p = .8, list = F)

# matrix of predictors (glmnet uses input matrix)
x1 <- college_df2[trainRows,]
# vector of response
y1 <- college_df$Outstate[trainRows]
train <- college_df[trainRows,]
# matrix of predictors (glmnet uses input matrix)
x2 <- college_df2[-trainRows,]
# vector of response
y2 <- college_df$Outstate[-trainRows]
test <- college_df[-trainRows,]

ctrl1 <- trainControl(method = "cv", number = 10)
```

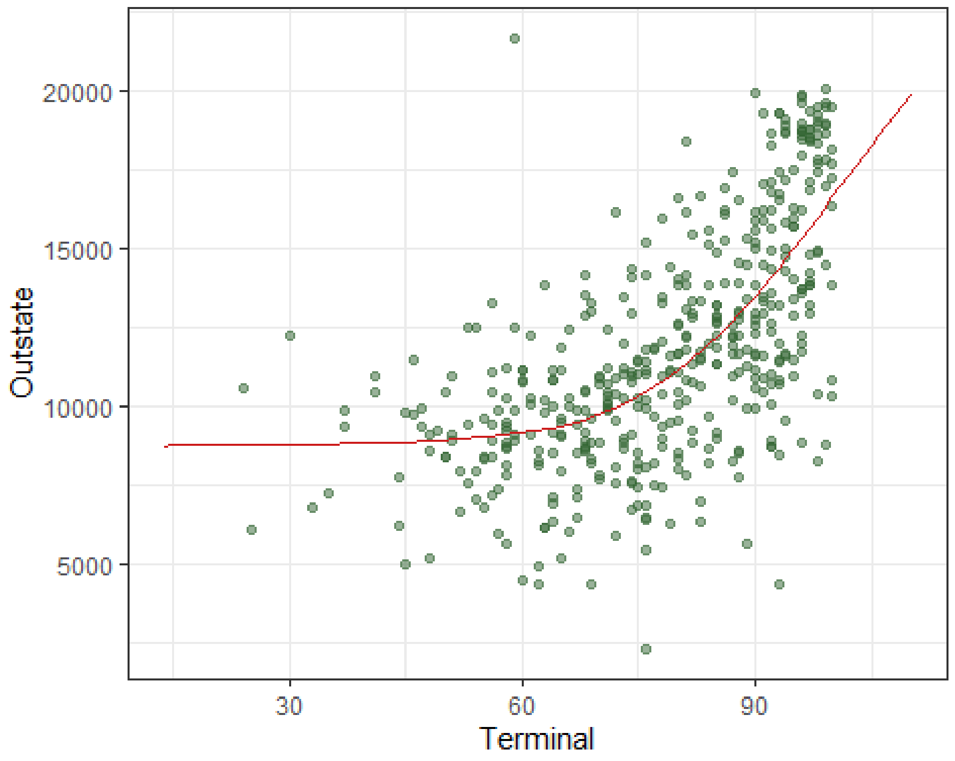
(a) Perform exploratory data analysis using the training data (e.g., scatter plots of response vs. predictors).

```
theme1 <- trellis.par.get()
theme1$plot.symbol$col <- rgb(.2, .4, .2, .5)
theme1$plot.symbol$pch <- 16
theme1$plot.line$col <- rgb(.8, .1, .1, 1)
theme1$plot.line$lwd <- 2
theme1$strip.background$col <- rgb(.0, .2, .6, .2)
trellis.par.set(theme1)

featurePlot(x = train[,2:17],
            y = train[,18],
            plot = "scatter",
            span = .5,
            labels = c("Predictors","Y"),
            type = c("p"))
```



(b) Fit smoothing spline models using Terminal as the only predictor of Outstate for a range of degrees of freedom, as well as the degree of freedom obtained by generalized cross-validation, and plot the resulting fits. Describe the results obtained.

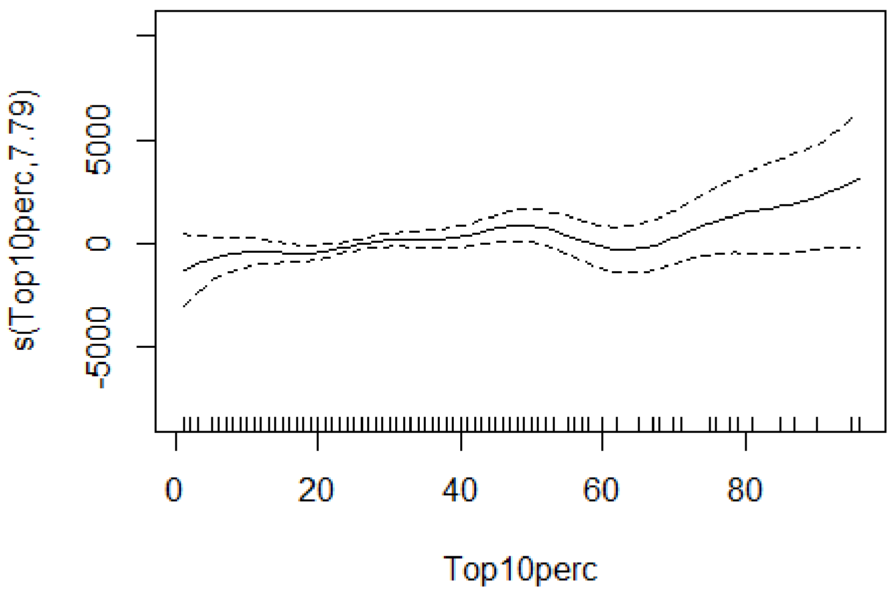
```
fit.ss <- smooth.spline(train$Terminal, train$Outstate, cv = TRUE)

## Warning in smooth.spline(train$Terminal, train$Outstate, cv = TRUE):
 cross-
## validation with non-unique 'x' values seems doubtful

fit.ss$df

## [1] 4.458339

Terminal.grid <- seq(from = 14, to = 110, by = 1)
pred.ss <- predict(fit.ss,
                   x = Terminal.grid)

pred.ss.df <- data.frame(pred = pred.ss$y,
                         Terminal = Terminal.grid)

p <- ggplot(data = train, aes(x = Terminal, y = Outstate)) +
     geom_point(color = rgb(.2, .4, .2, .5))
p +
geom_line(aes(x = Terminal, y = pred), data = pred.ss.df,
          color = rgb(.8, .1, .1, 1)) + theme_bw()
```

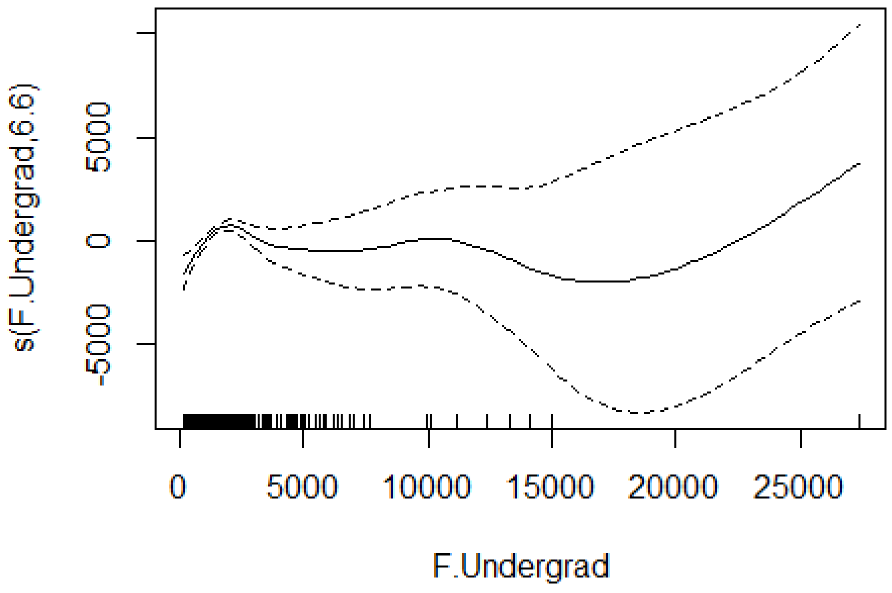


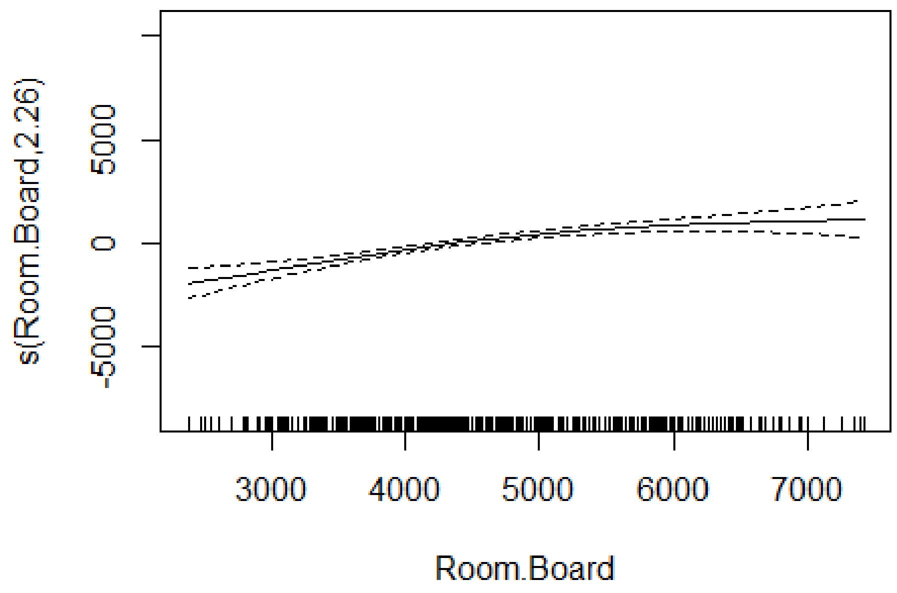
From the plot we can see that the model we fit will lead to a smooth line, and the trend of the smooth line is the same as the trend of the points of the real data set. So we can consider the model is a good fit.

(c) Fit a generalized additive model (GAM) using all the predictors. Plot the results and explain your findings. Report the test error.

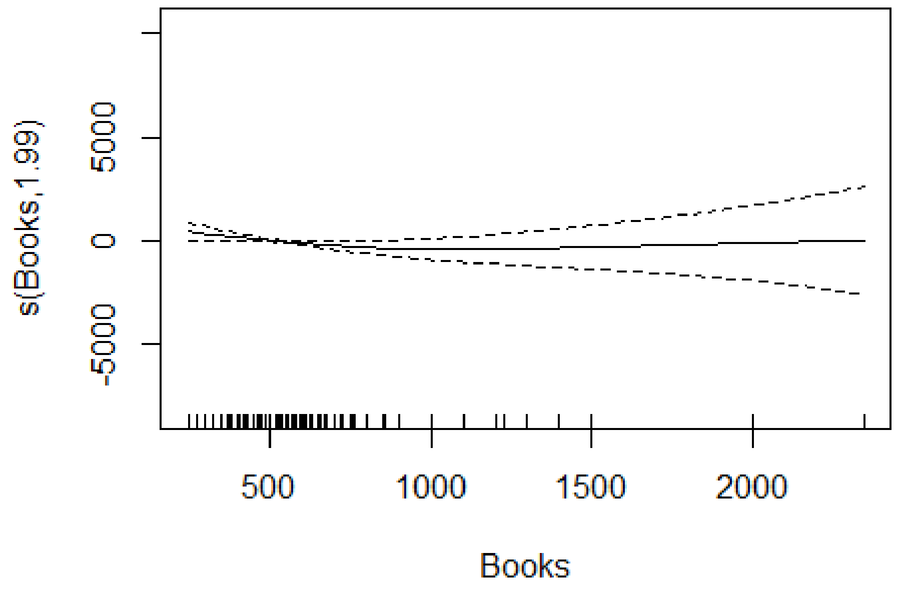
```
gam.fit <- gam(Outstate ~ Apps+Accept+Enroll+s(Top10perc)+Top25perc+s(F.
Undergrad)+P.Undergrad+s(Room.Board)+s(Books)+Personal+s(PhD)+Terminal+
s(S.F.Ratio)+perc.alumni+s(Expend)+s(Grad.Rate), data = train)

plot(gam.fit)
```


s(Top10perc,7.79)
5000
0
-5000
0
20
40
60
80
Top10perc

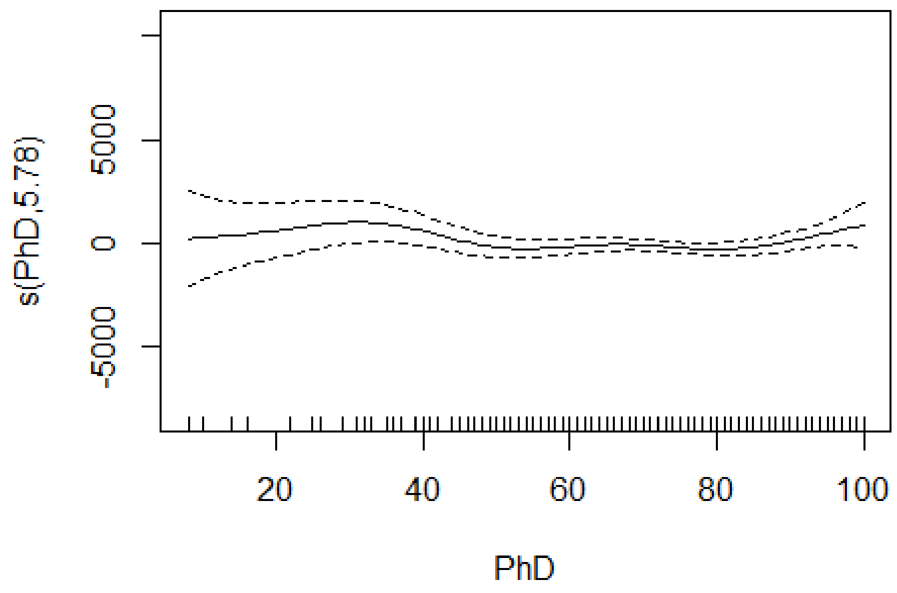

s(F.Undergrad,6.6)
5000
0
-5000
0
5000
10000
15000
20000
25000
F.Undergrad



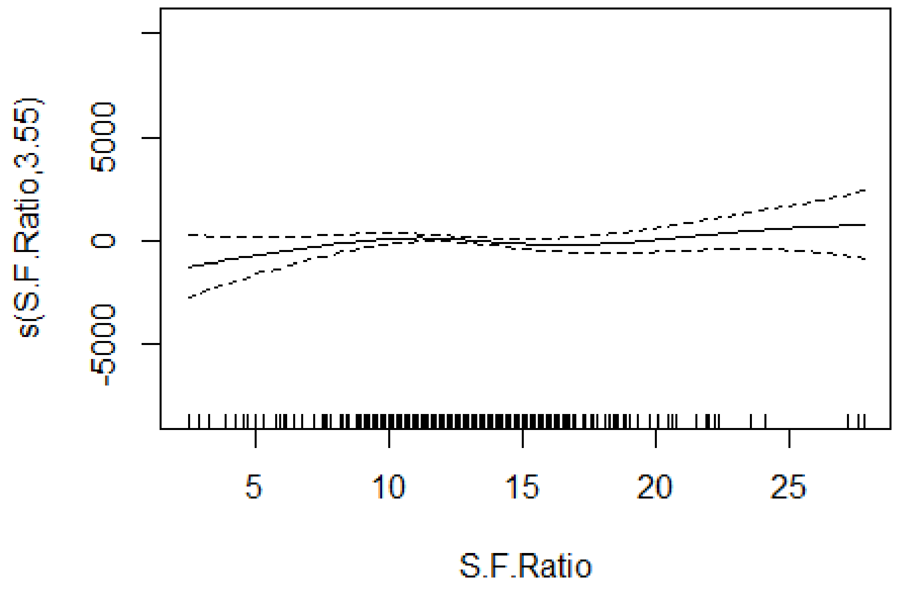
s(Room.Board,2.26)
-5000
0
5000
3000
4000
5000
6000
7000
Room.Board



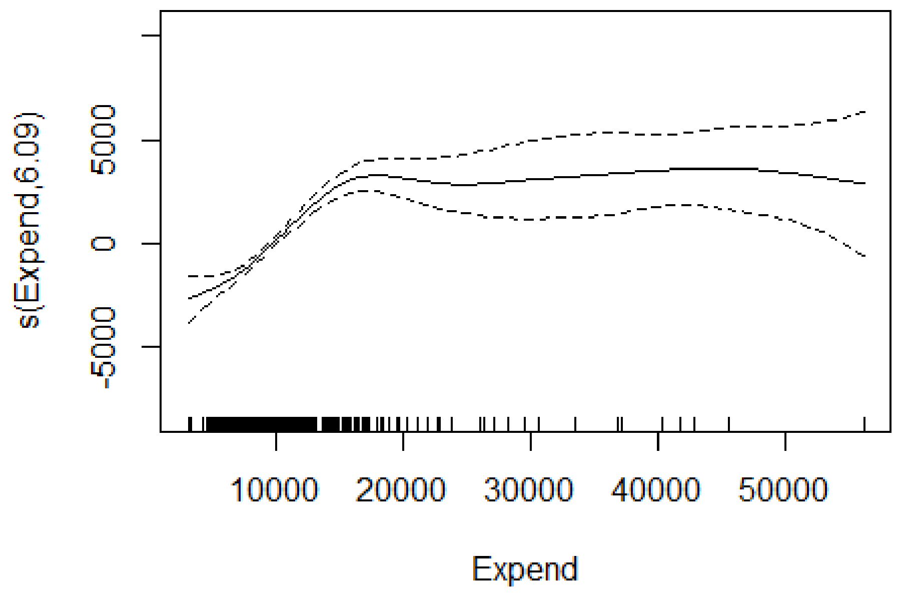
s(Books,1.99)
-5000
0
5000
500
1000
1500
2000
Books

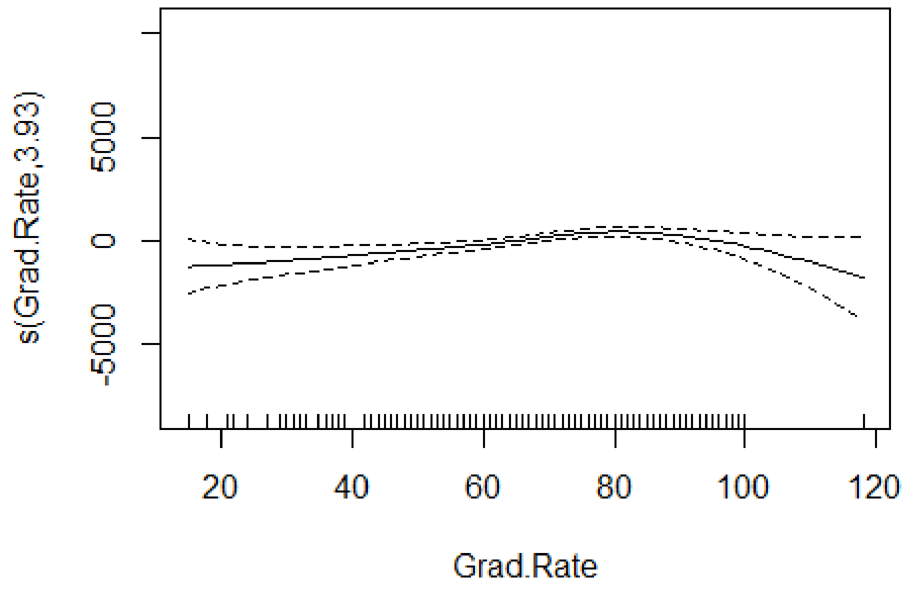


s(PhD,5.78)
5000
0
-5000
20
40
60
80
100
PhD



s(S.F.Ratio,3.55)
5000
0
-5000
5
10
15
20
25
S.F.Ratio





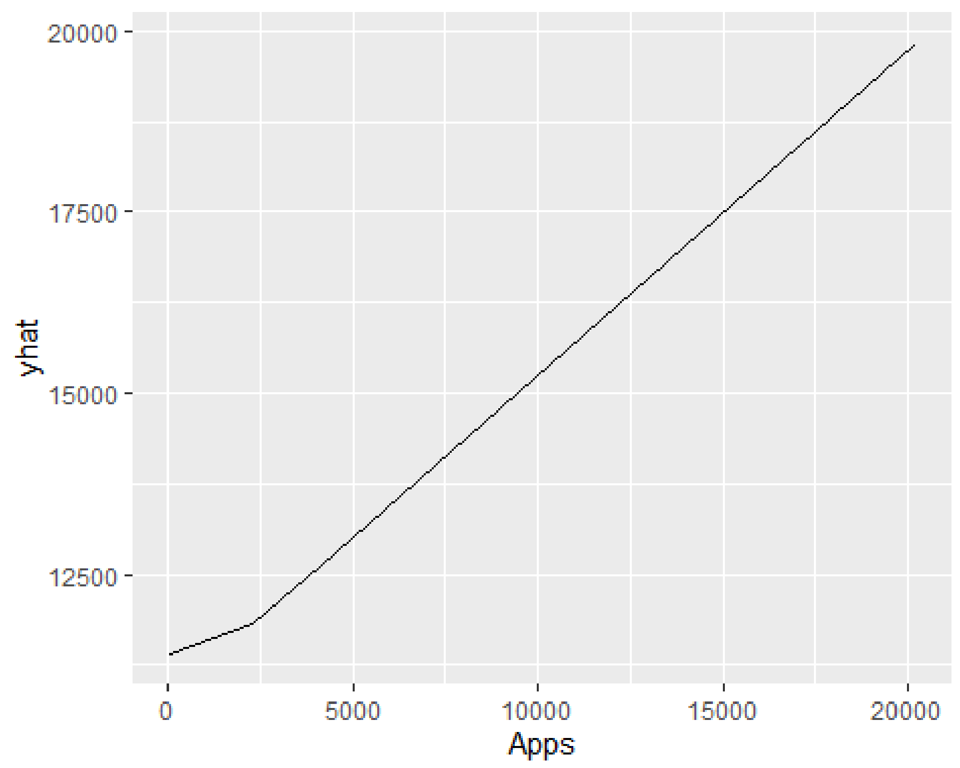
```
gam.pred <- predict(gam.fit, newdata = test[,2:17])
# test error
test_error_gam = mean((gam.pred - y2)^2)
test_error_gam
```

```
## [1] 3529399
```

After plotting the gam model, I found that only the variebles “Top10perc”, “F.Undergrad”, “Room.Board”, “Books”, “PhD”, “S.F.Ratio Expend” and “Grad.Rate” are non-linear, so we only add “s()” to these variables when building the gam model. The test error is 3.5293992^{6}.

(d) Train a multivariate adaptive regression spline (MARS) model using all the predictors. Report the final model. Present the partial dependence plot of an arbitrary predictor in your final model. Report the test error.

```
mars_grid <- expand.grid(degree = 1:3,
                         nprune = 2:15)

set.seed(2)
mars.fit <- train(x = train[,2:17],
                  y1,
                  method = "earth",
                  tuneGrid = mars_grid,
                  trControl = ctrl1)

pdp::partial(mars.fit, pred.var = c("Apps"), grid.resolution = 10) %>%
  autoplot()

## Warning: Use of `object[[1L]]` is discouraged. Use `.data[[1L]]` ins
tead.

## Warning: Use of `object[["yhat"]]` is discouraged. Use `.data[["yhat
"]]`
## instead.
```

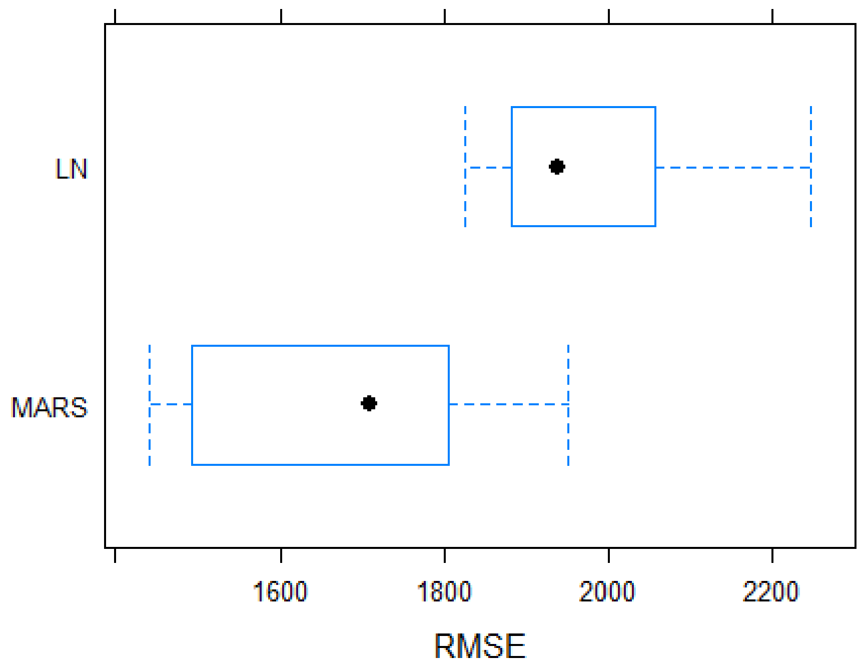


```
mars.fit$bestTune

##    nprune degree
## 11     12      1

coef(mars.fit$finalModel)

##          (Intercept)     h(16262-Expend)  h(5620-Room.Board) h(1365-F.
Undergrad)
##        15180.8099938          -0.5587913          -0.8251141
 -1.7285188
##   h(32-perc.alumni)        h(Apps-1422)       h(Enroll-911)       h
(911-Enroll)
##          -43.3839576           0.4489395          -1.9054530
  5.1918949
##     h(83-Grad.Rate)    h(1323-Personal)           h(PhD-81)      h(1
228-Accept)
##          -23.1058912           0.8852083          54.3923953
 -2.3277448

mars.pred <- predict(mars.fit, newdata = test[,2:17])
# test error
test_error_mars = mean((mars.pred - y2)^2)
test_error_mars

## [1] 3699180
```

I presented the partial dependence plot of "Apps" in my final model. The test error is 3.6991801^{6}.

(e) In this data example, do you prefer the use of MARS model over a linear model when predicting the out-of-state tuition? Why?

```
set.seed(2)
lm.fit <- train(x = train[,2:17],
                y1,
                method = "lm",
                trControl = ctrl1)

resamp <- resamples(list(LN = lm.fit,
                         MARS = mars.fit))
summary(resamp)

##
## Call:
## summary.resamples(object = resamp)
##
## Models: LN, MARS
## Number of resamples: 10
##
## MAE
##          Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## LN   1378.324 1536.575 1585.856 1580.324 1669.019 1693.438    0
## MARS 1072.662 1202.016 1356.984 1313.225 1425.515 1447.072    0
##
## RMSE
##          Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## LN   1825.570 1881.487 1937.965 1975.433 2045.672 2244.553    0
## MARS 1442.274 1532.909 1709.697 1687.899 1804.998 1951.304    0
##
## Rsquared
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA'
s
## LN   0.6389549 0.6980285 0.7102279 0.7180683 0.7484025 0.7839554
0
## MARS 0.7248618 0.7565213 0.7761406 0.7928721 0.8446916 0.8666082
0

bwplot(resamp, metric = "RMSE")
```



I prefer to use MARS model since it has a smaller RMSE