p8106_hw2_qz2266

Qing Zhou

2023-03-10

Data preparation

In this exercise, we build nonlinear models using the "College" data. The dataset contains statistics for 565 US Colleges from a previous issue of US News and World Report. The response variable is the out-of-state tuition (Outstate).

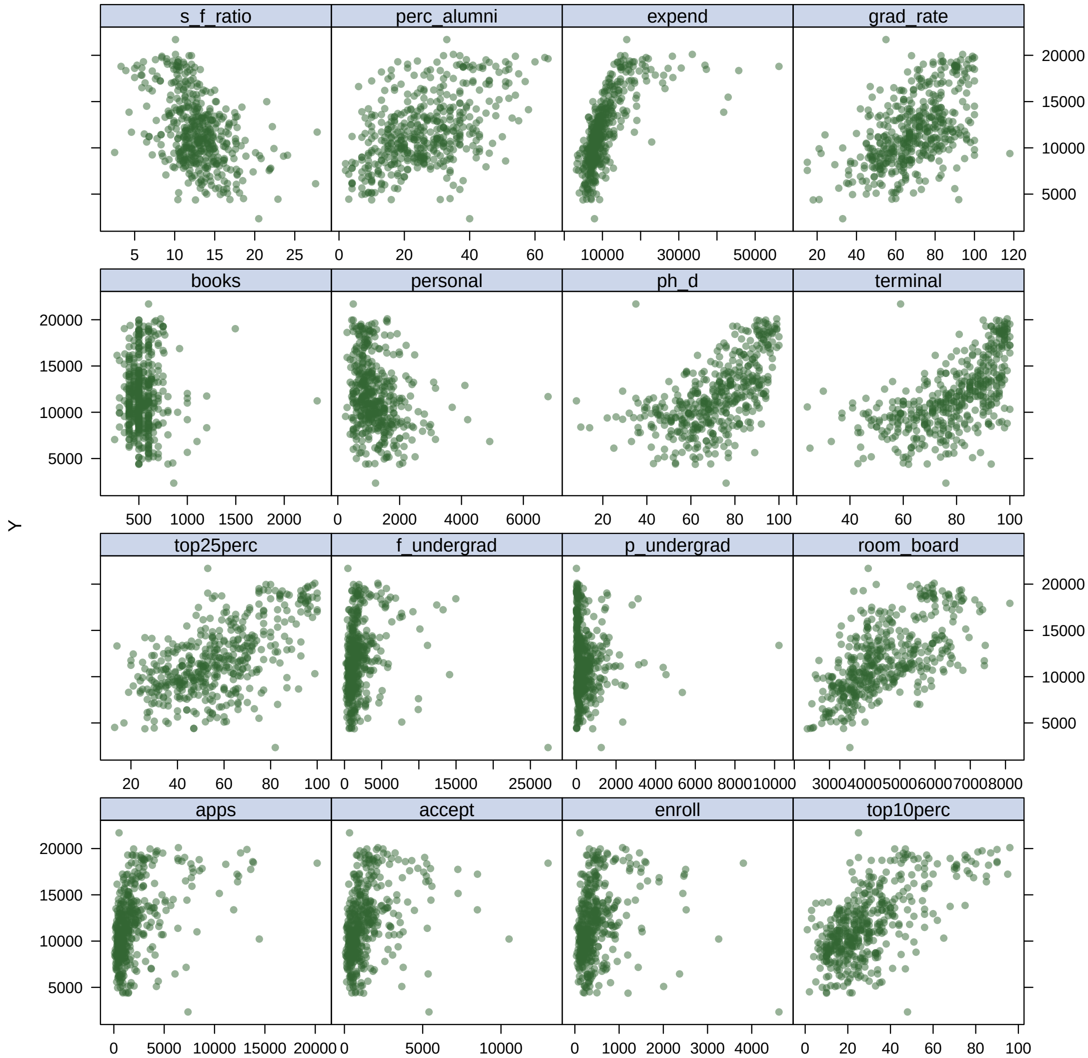
```
college_df = read_csv('./data/College.csv') %>%
  janitor::clean_names() %>%
  na.omit() %>%
  relocate("outstate", .after = "grad_rate") %>%
  select(-college)
```

We then partition the dataset into two parts: training data (80%) and test data (20%)

```
set.seed(1)
trainRows <- createDataPartition(y = college_df$outstate, p = 0.8, list = FALSE)

# Training data
train_df = college_df[trainRows, ]
x_train = model.matrix(outstate~.,train_df)[, -1]
y_train = train_df$outstate
# Testing data
test_df = college_df[-trainRows, ]
x_test = model.matrix(outstate~.,test_df)[, -1]
y_test = test_df$outstate
```

Exploratory data analysis

```
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)

# feature plot
featurePlot(x_train, y_train, plot = "scatter", labels = c("","Y"), type = c("p"), layout = c(4, 4))
```



- Based on the feature plot, we can see there might be linear trends between the outcome and predictors `perc_alumni`, `grad_rate`, `ph_d`, `terminal`, `top25perc`, `room_board`, and `top10perc`.

a). Smoothing spline models

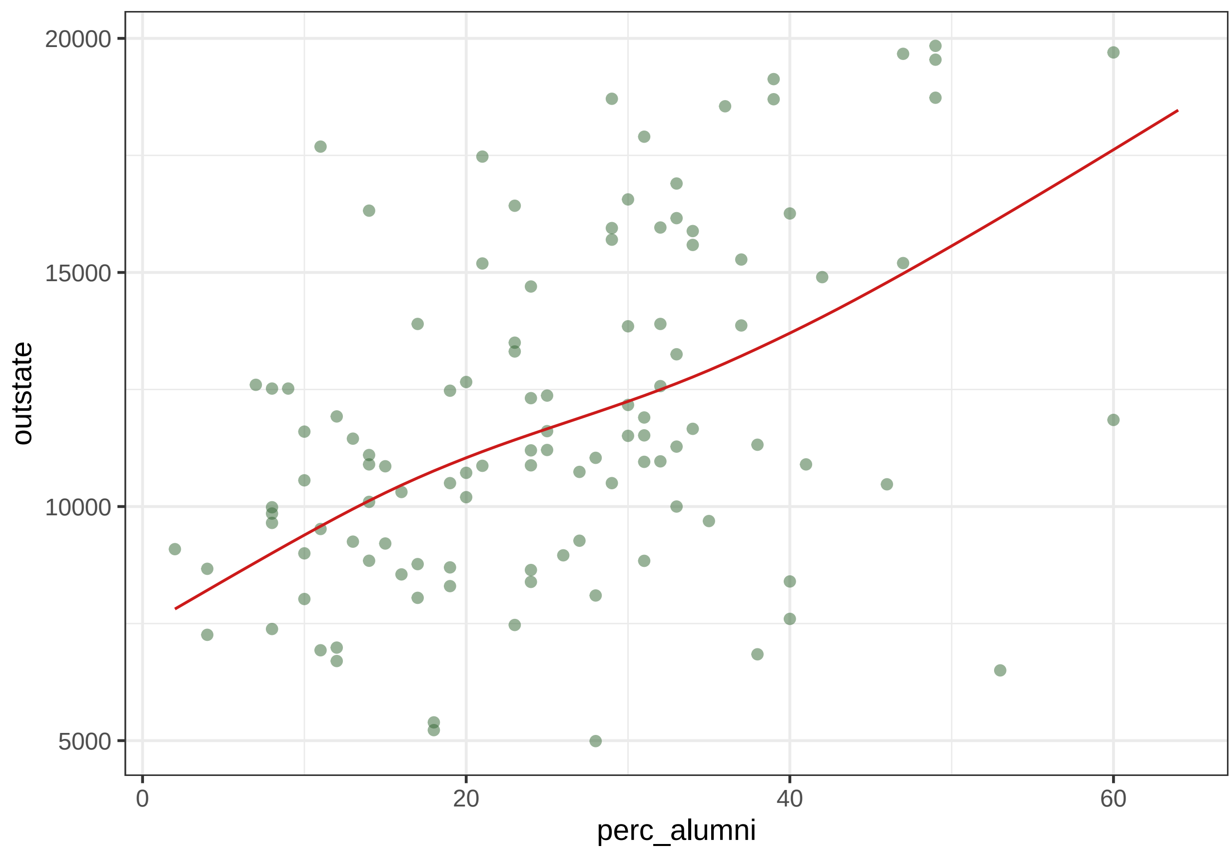
Fit smoothing spline models using perc.alumni as the only predictor of Outstate for a range of degrees of freedom.

1). The degree of freedom is obtained by generalized cross-validation:

```r
set.seed(1)

# fit model
fit.ss = smooth.spline(train_df$perc_alumni, train_df$outstate)
# optimal degrees of freedom based on cross-validation
fit.ss$df
```

```
## [1] 3.779231
```

```r
#plot the resulting fits
palumnilims <- range(train_df$perc_alumni)
palumni.grid <- seq(from = palumnilims[1],to = palumnilims[2])

pred.ss = predict(fit.ss, x = palumni.grid)
pred.ss.df = data.frame(pred = pred.ss$y, perc_alumni = palumni.grid)

p = ggplot(data = test_df, aes(x = perc_alumni, y = outstate)) +
geom_point(color = rgb(.2, .4, .2, .5))

p + geom_line(aes(x = perc_alumni, y = pred), data = pred.ss.df, color = rgb(.8, .1, .1, 1)) + theme_bw
```

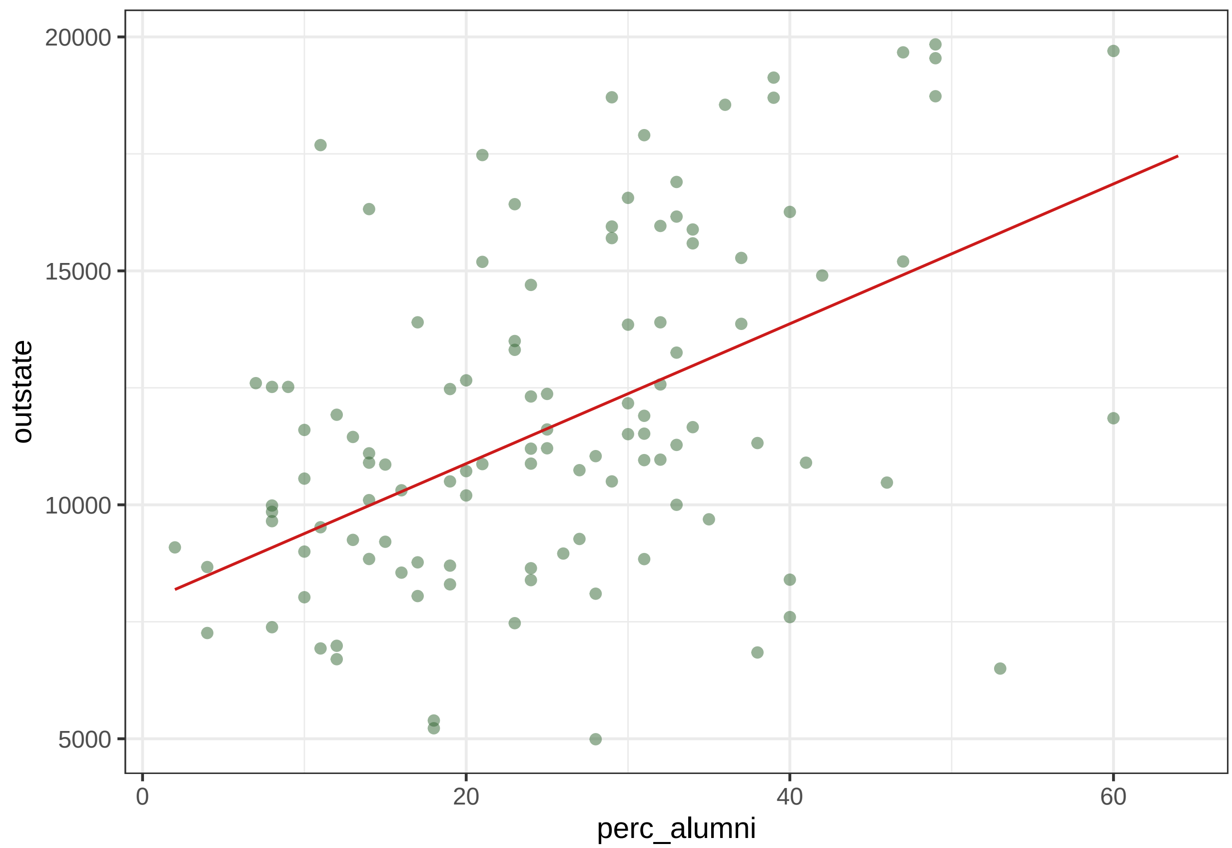


2). The degree of freedom is obtained by my choice:

```r
set.seed(1)

# 2 degree of freedom
fit.ss = smooth.spline(train_df$perc_alumni, train_df$outstate, df = 2)
fit.ss$df
```

```
## [1] 2.000232
```

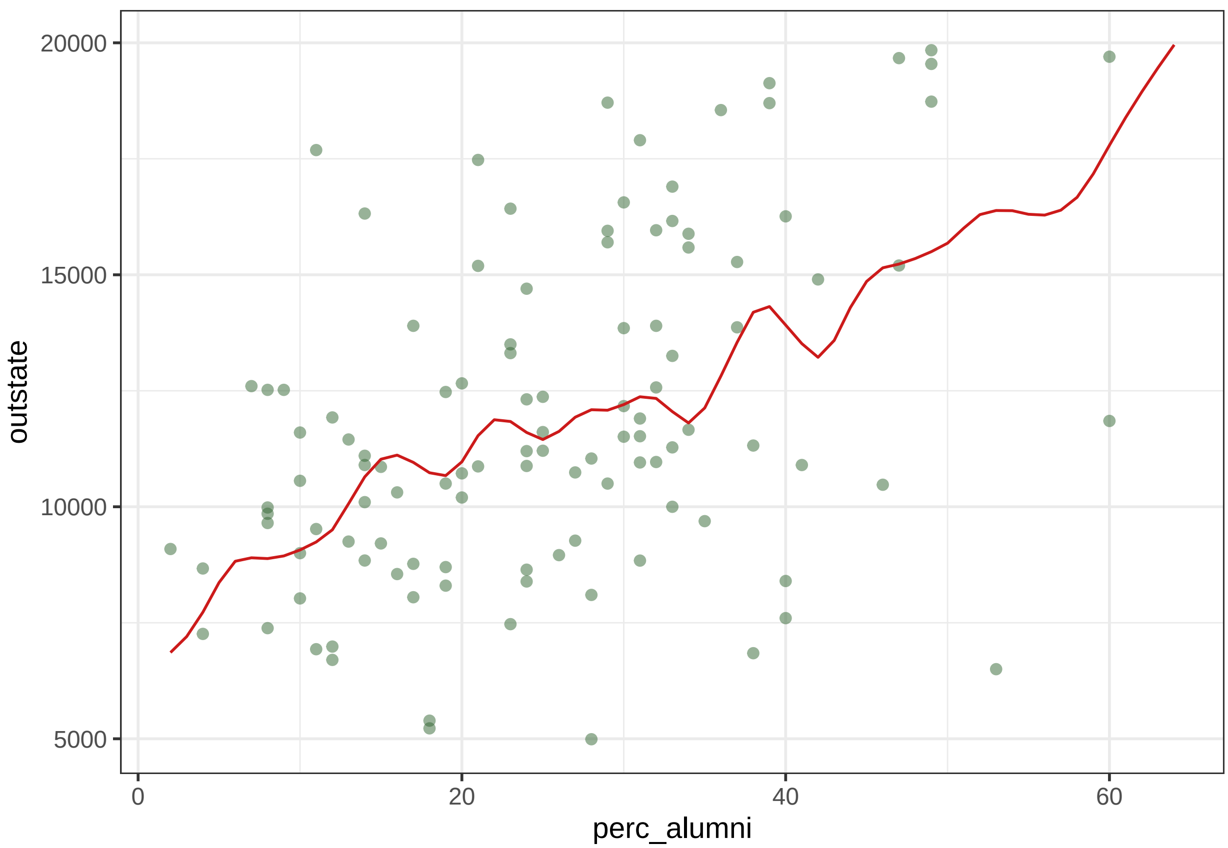
```r
pred.ss = predict(fit.ss, x = palumni.grid)
pred.ss.df = data.frame(pred = pred.ss$y, perc_alumni = palumni.grid)

p = ggplot(data = test_df, aes(x = perc_alumni, y = outstate)) +
geom_point(color = rgb(.2, .4, .2, .5))

p + geom_line(aes(x = perc_alumni, y = pred), data = pred.ss.df, color = rgb(.8, .1, .1, 1)) + theme_bw
```




```r
# 20 degrees of freedom
fit.ss = smooth.spline(train_df$perc_alumni, train_df$outstate, df = 20)
fit.ss$df
```

```
## [1] 20.00271
```

```r
pred.ss = predict(fit.ss, x = palumni.grid)
pred.ss.df = data.frame(pred = pred.ss$y, perc_alumni = palumni.grid)

p = ggplot(data = test_df, aes(x = perc_alumni, y = outstate)) +
geom_point(color = rgb(.2, .4, .2, .5))

p + geom_line(aes(x = perc_alumni, y = pred), data = pred.ss.df, color = rgb(.8, .1, .1, 1)) + theme_bw
```



- When setting df = 2 and df = 20, we can see that df = 2 shows a more linear trend, while df = 10 shows a more wiggly fitted line. Therefore, larger df values make the line much more wiggly, suggesting potential overfitting issue, while smaller degrees of freedom make the fitted line more linear and smooth, but less flexible and thus might underfit the data points.
- The degree of freedom obtained by generalized cross validation is 3.779231, and the fitted curve has more details than the line of df = 2 and is more linear and smooth than the line of df = 10. By comparing these three models, we conclude that the optimized model fits the data best, with a positive relationship between `perc_alumni` and `Outstate`.

b). Generalized additive models

```
set.seed(1)

# 10-fold cross-validation
ctrl1 = trainControl(method = "cv", number = 10)

# fit a gam model using all predictors
gam_fit_all = train(x_train, y_train,
                    method = "gam",
                    trControl = ctrl1)

gam_fit_all$bestTune
```

```
##   select method
## 1  FALSE GCV.Cp
```

```r
gam_fit_all$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ s(perc_alumni) + s(terminal) + s(books) + s(grad_rate) +
##     s(ph_d) + s(top10perc) + s(top25perc) + s(s_f_ratio) + s(personal) +
##     s(p_undergrad) + s(room_board) + s(enroll) + s(accept) +
##     s(f_undergrad) + s(apps) + s(expend)
##
## Estimated degrees of freedom:
## 6.05 1.00 2.17 3.56 1.81 1.00 1.00
## 3.69 1.00 1.00 2.47 1.00 4.19 5.51
## 4.45 6.87  total = 47.75
##
## GCV score: 2824207
```

```r
# fit GAM alternatively
gam_fit_alt <- train(x_train, y_train,
                 method = "gam",
                 tuneGrid = data.frame(method = "GCV.Cp", select = c(TRUE)),
                 trControl = ctrl1)

gam_fit_alt$bestTune
```

```
##   select method
## 1   TRUE GCV.Cp
```

```r
gam_fit_alt$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ s(perc_alumni) + s(terminal) + s(books) + s(grad_rate) +
##     s(ph_d) + s(top10perc) + s(top25perc) + s(s_f_ratio) + s(personal) +
##     s(p_undergrad) + s(room_board) + s(enroll) + s(accept) +
##     s(f_undergrad) + s(apps) + s(expend)
##
## Estimated degrees of freedom:
## 6.077 0.198 1.095 1.437 0.000 0.832 0.000
## 3.853 0.638 0.796 3.770 1.000 4.625 5.917
## 4.603 5.930  total = 41.77
##
## GCV score: 2766492
```

1. The GAM model for all predictors .outcome ~ s(perc_alumni) + s(terminal) + s(books) + s(grad_rate) + s(ph_d) + s(top10perc) + s(top25perc) + s(s_f_ratio) + s(personal) + s(p_undergrad) + s(room_board) + s(enroll) + s(accept) + s(f_undergrad) + s(apps) + s(expend)

Estimated degrees of freedom: 6.05 1.00 2.17 3.56 1.81 1.00 1.00 3.69 1.00 1.00 2.47 1.00 4.19 5.51 4.45 6.87 total = 47.75

GCV score: 2824207

2.The GAM model for the selection specification: .outcome ~ s(perc_alumni) + s(terminal) + s(books) + s(grad_rate) + s(ph_d) + s(top10perc) + s(top25perc) + s(s_f_ratio) + s(personal) + s(p_undergrad) + s(room_board) + s(enroll) + s(accept) + s(f_undergrad) + s(apps) + s(expend)

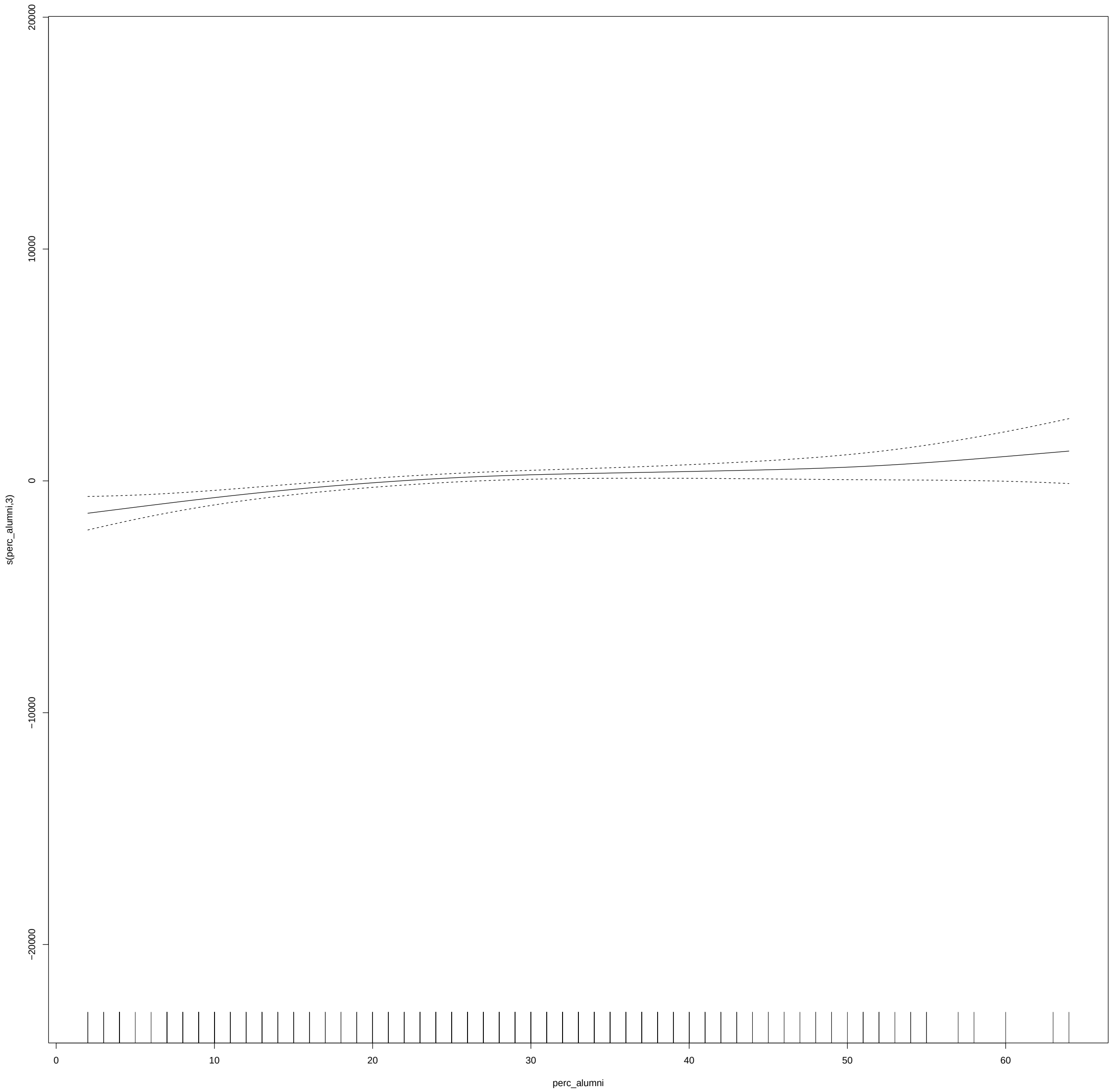
Estimated degrees of freedom: 6.077 0.198 1.095 1.437 0.000 0.832 0.000 3.853 0.638 0.796 3.770 1.000 4.625 5.917 4.603 5.930 total = 41.77

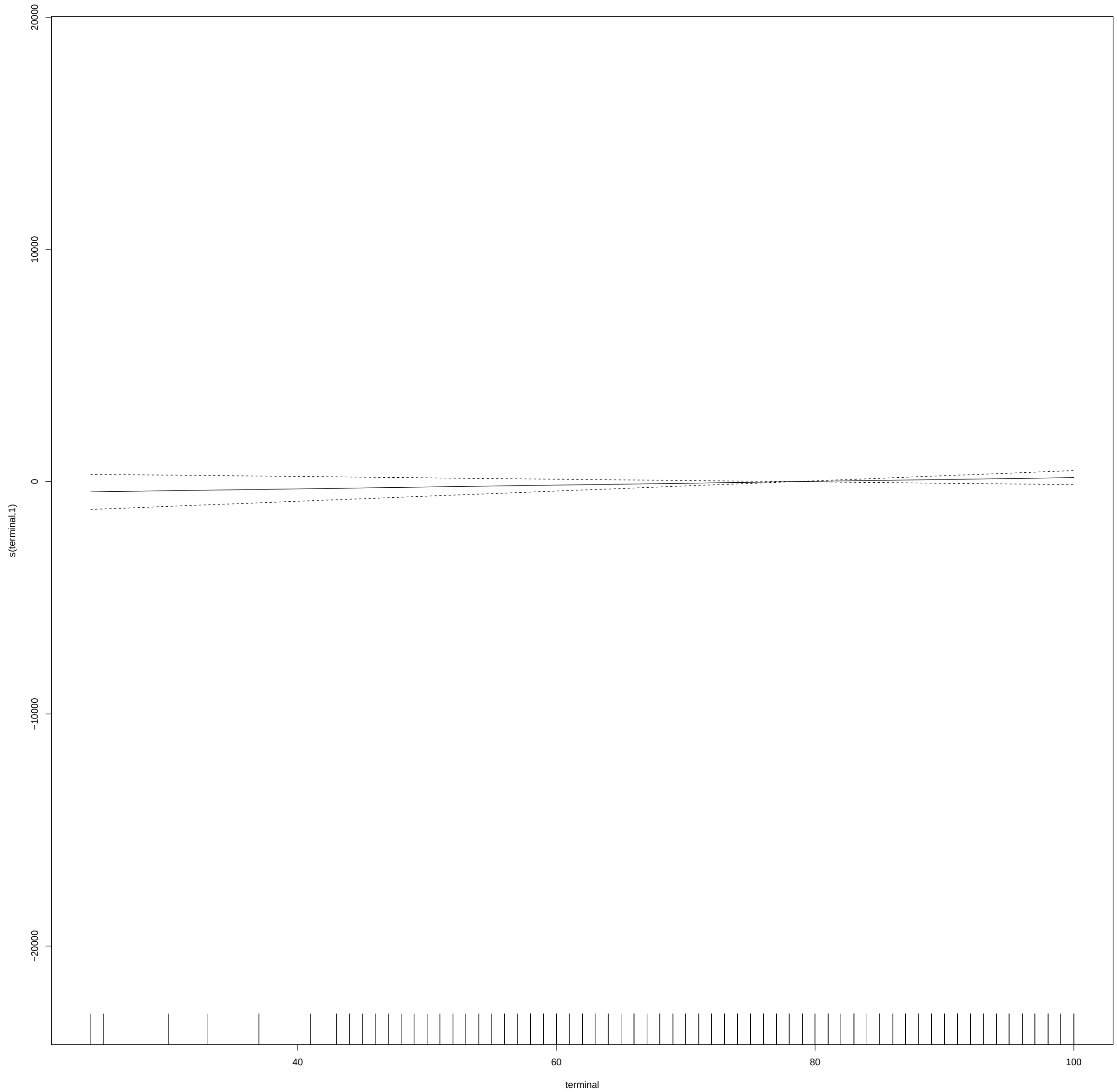
GCV score: 2766492

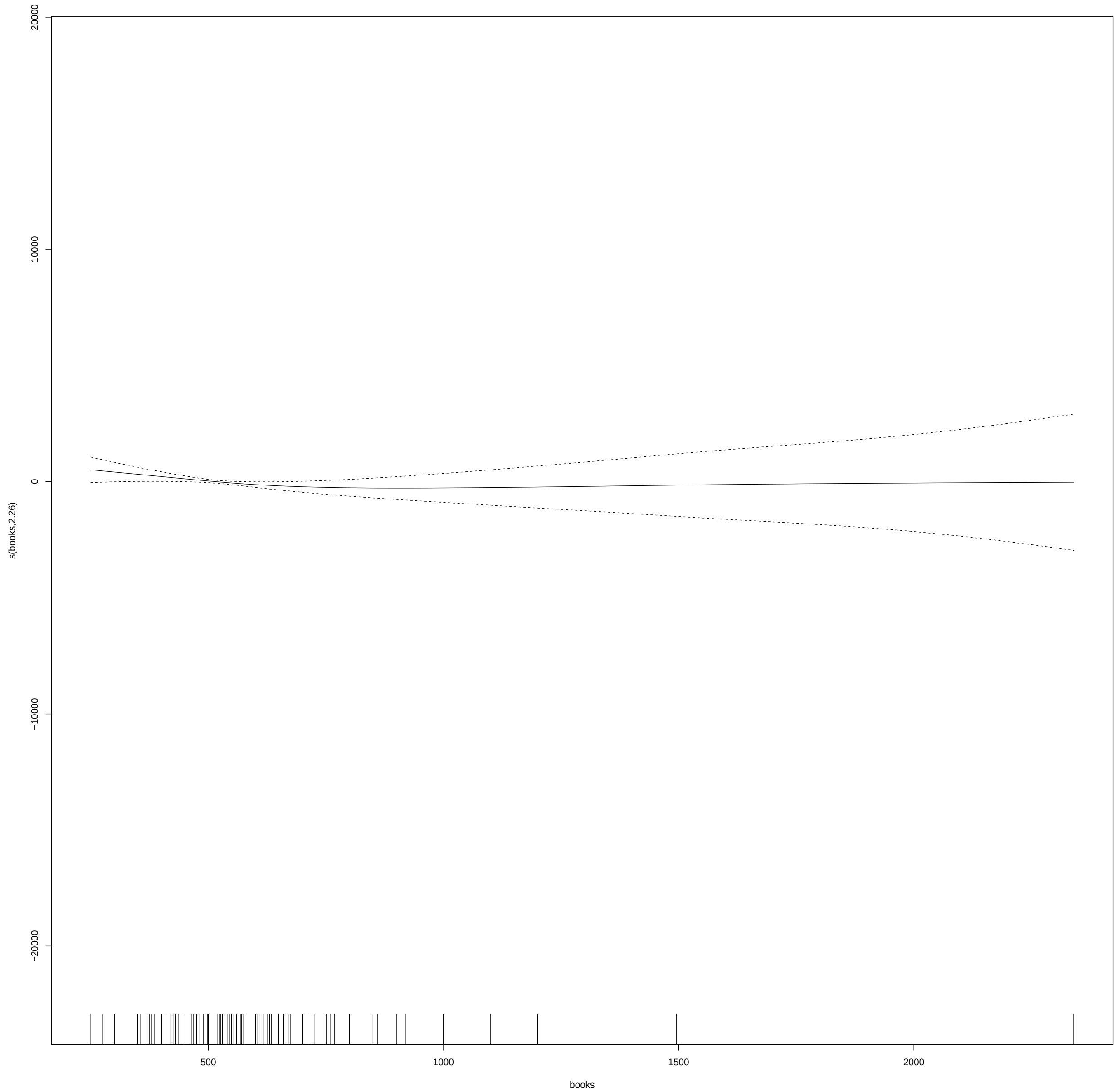
- From the result above, we remove `PhD`and `Top25perc` since their df = 0. Therefore, my final GAM model has 14 predictors. It doesn't have all 16 predictors. The model is as follows:
- Outstate ~ s(perc.alumni) + s(Terminal) + s(Books) + s(Grad.Rate) + s(Top10perc) + s(S.F.Ratio) + s(Personal) + s(P.Undergrad) + s(Room.Board) + s(Enroll) + s(Accept) + s(F.Undergrad) + s(Apps) + s(Expend)

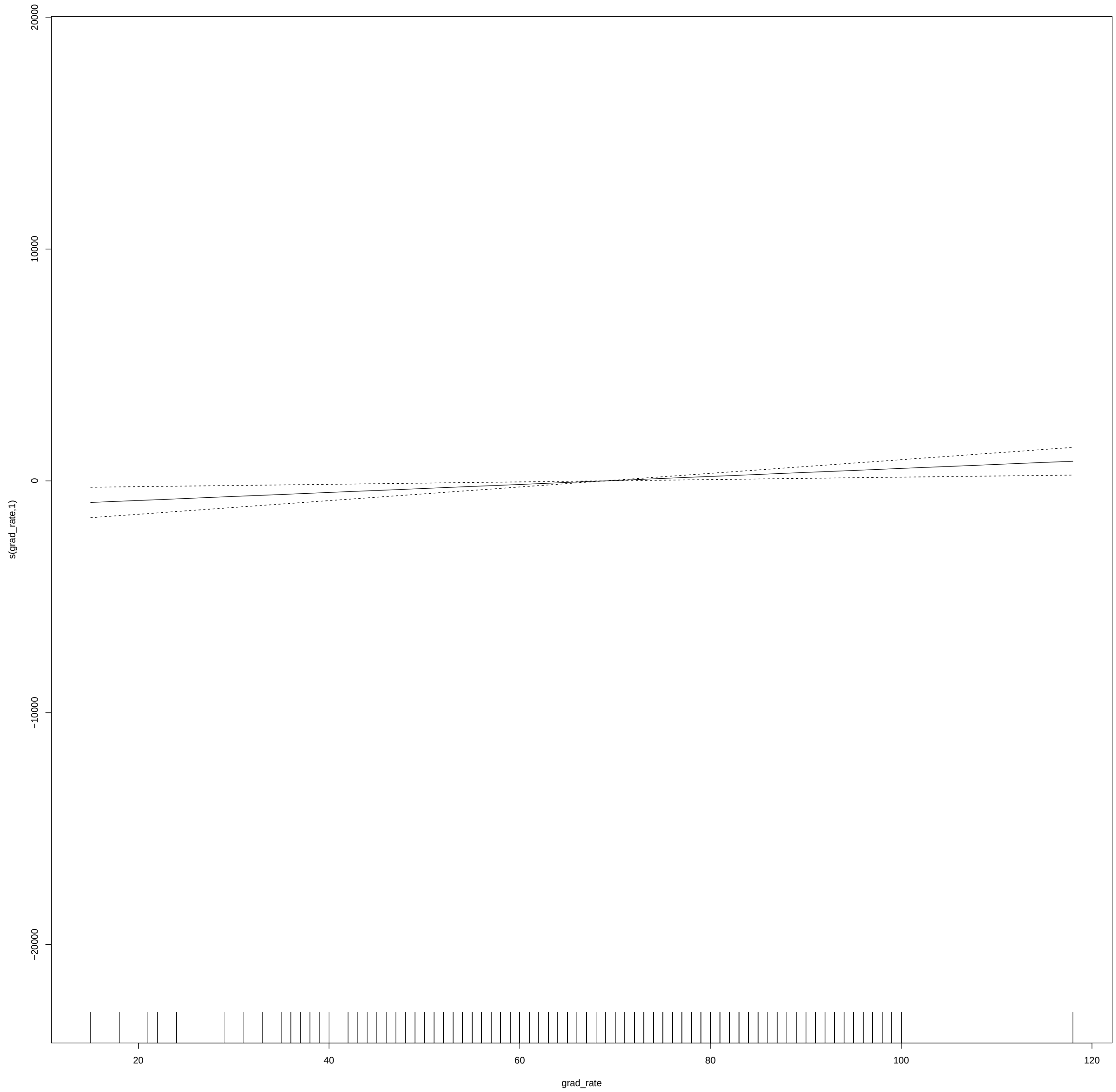
Plot of final model

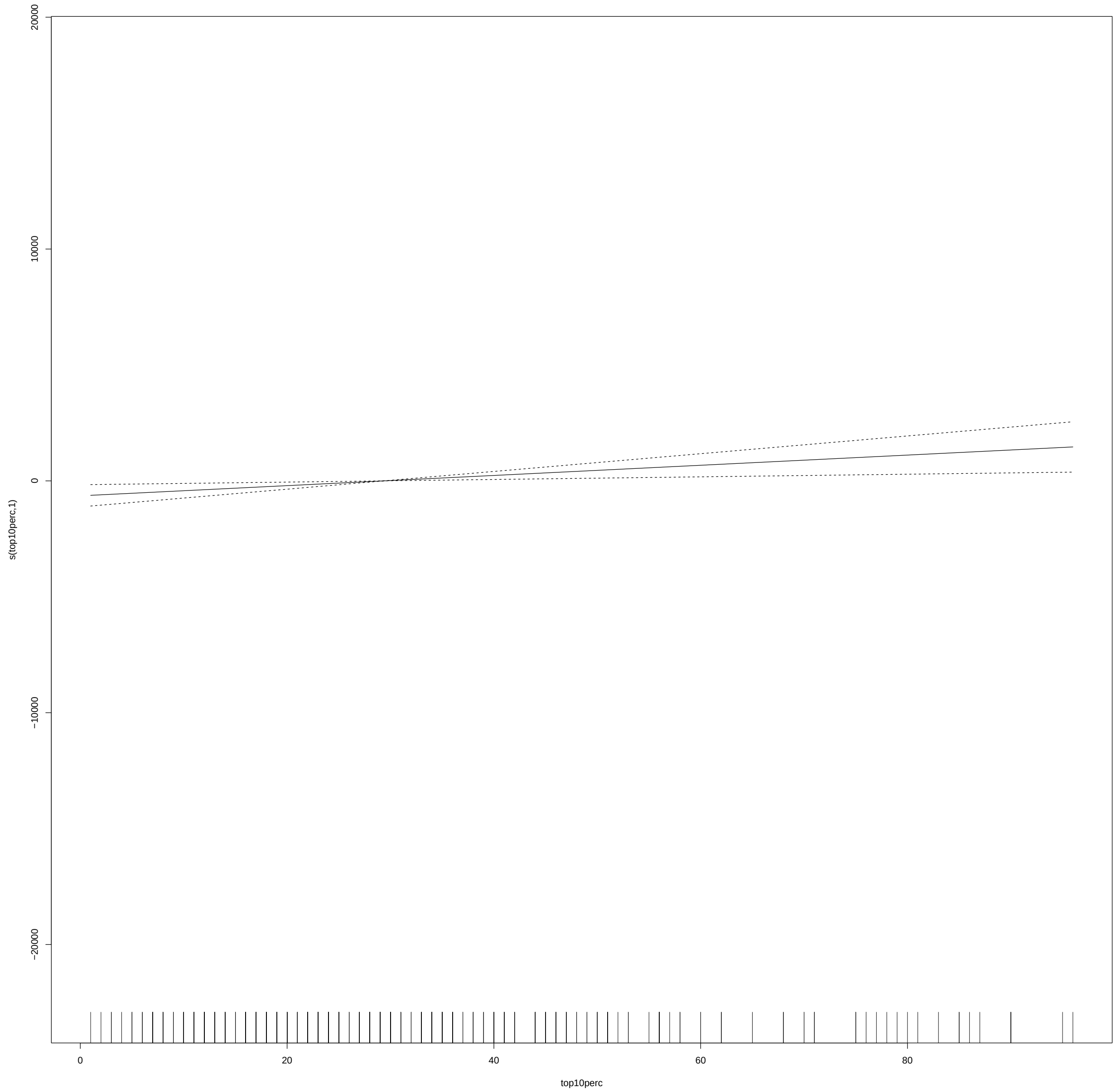
```r
gam_final <- gam(outstate ~ s(perc_alumni) + s(terminal) + s(books) +
                s(grad_rate)+ s(top10perc) + s(s_f_ratio) + s(personal) +
                s(p_undergrad) + s(room_board) + s(enroll) + s(accept) +
                s(f_undergrad) + s(apps) + s(expend),
              data = train_df)

plot(gam_final)
```

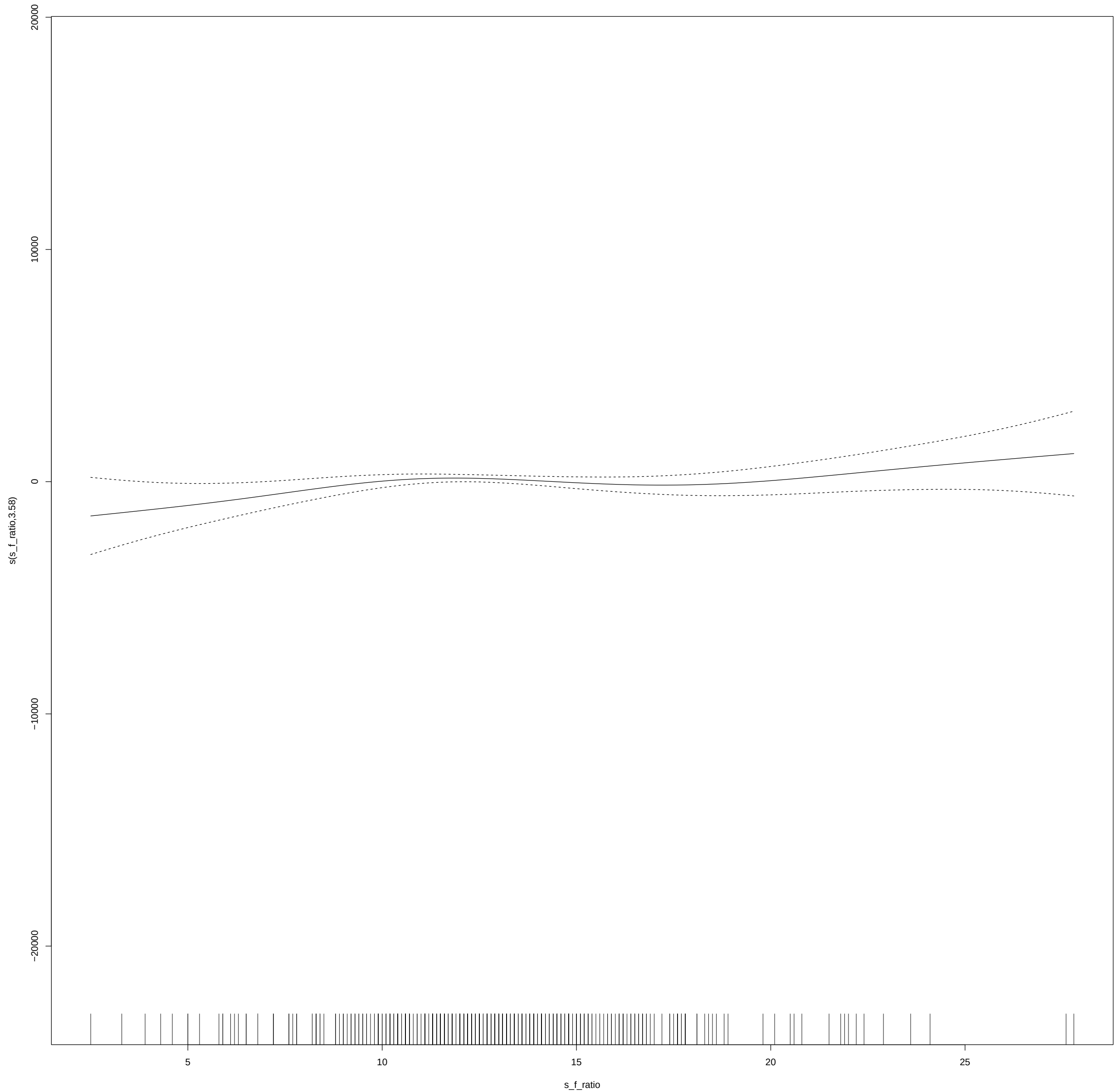

20000
10000
0
−10000
−20000
s(perc_alumni,3)
0
10
20
30
40
50
60
perc_alumni

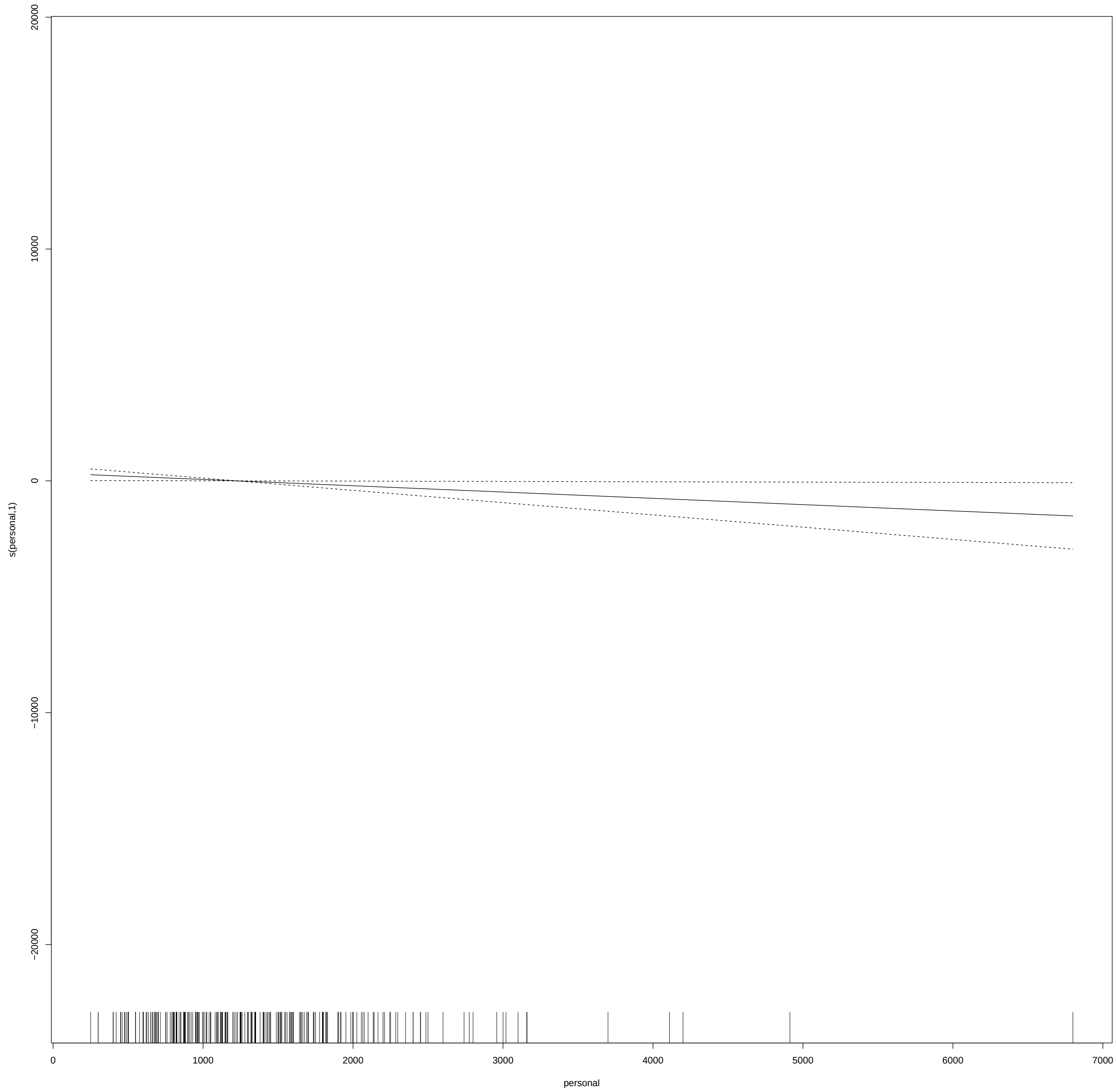

20000
10000
0
−10000
−20000
s(terminal,1)
40
60
80
100
terminal


20000
10000
0
−10000
−20000
s(books,2.26)
500
1000
1500
2000
books

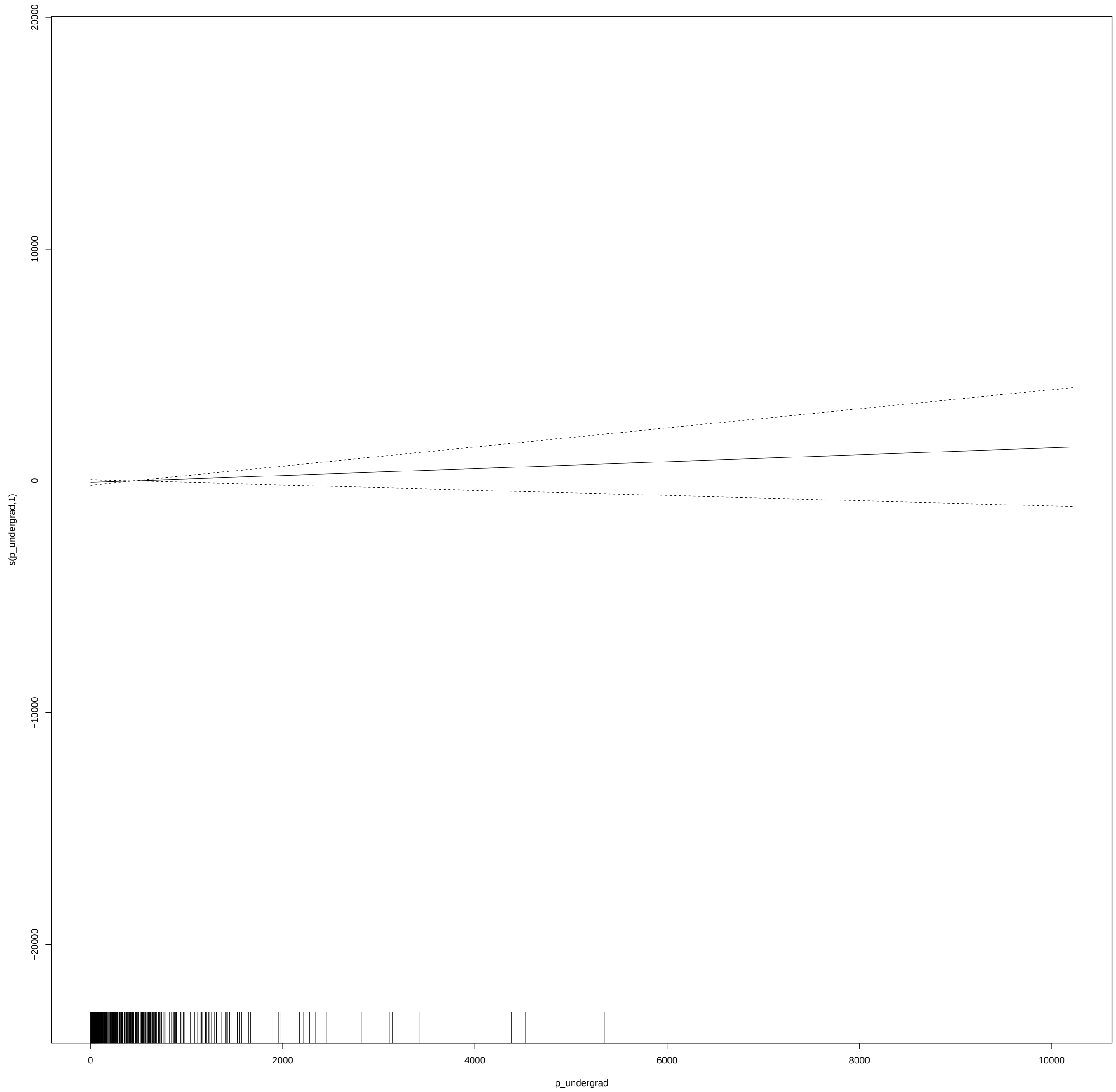

s(grad_rate,1)
20000
10000
0
−10000
−20000
20
40
60
80
100
120
grad_rate

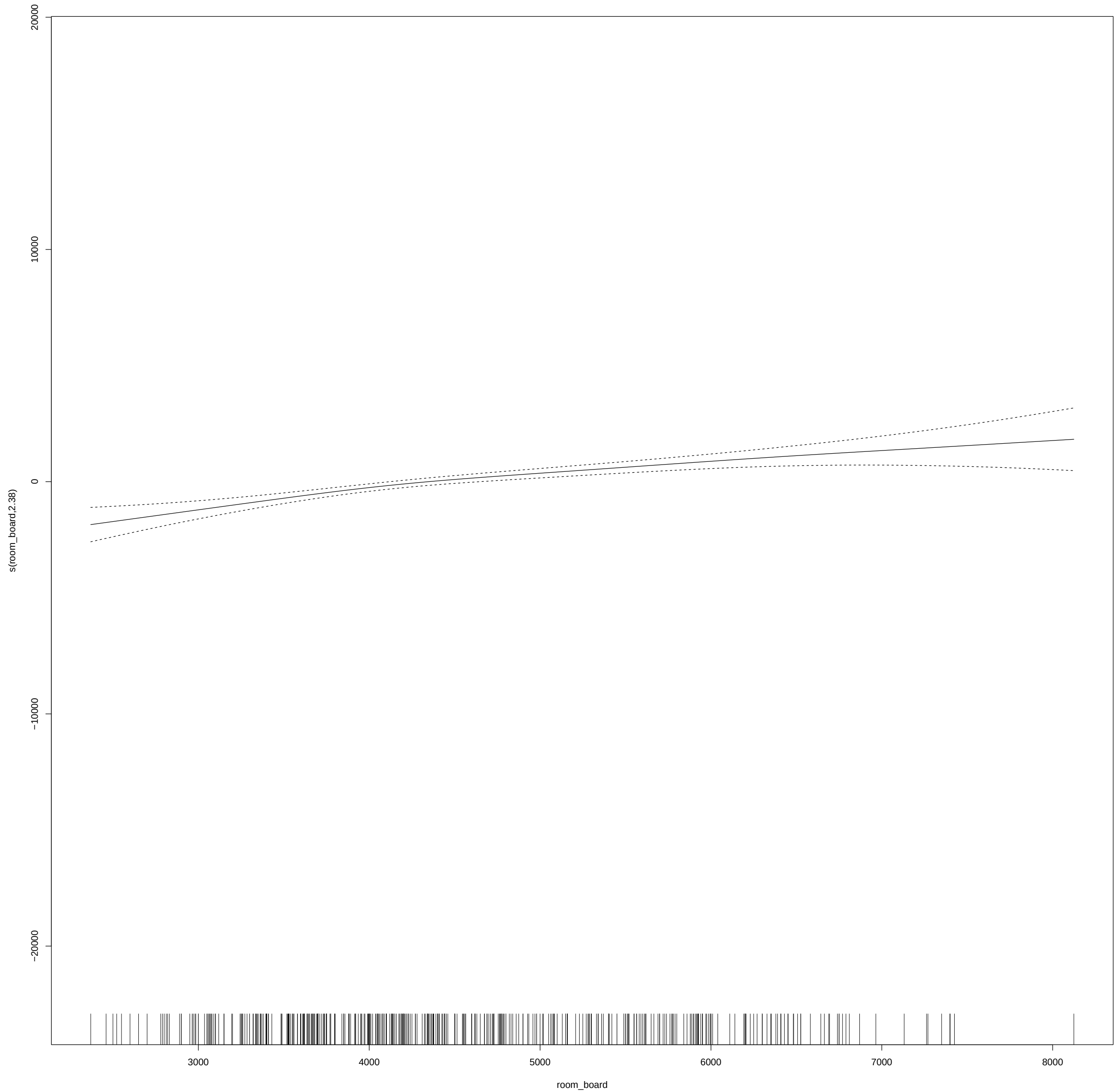

20000
10000
0
−10000
−20000
s(top10perc,1)
0
20
40
60
80
top10perc


20000
10000
0
−10000
−20000
s(s_f_ratio,3.58)
5
10
15
20
25
s_f_ratio

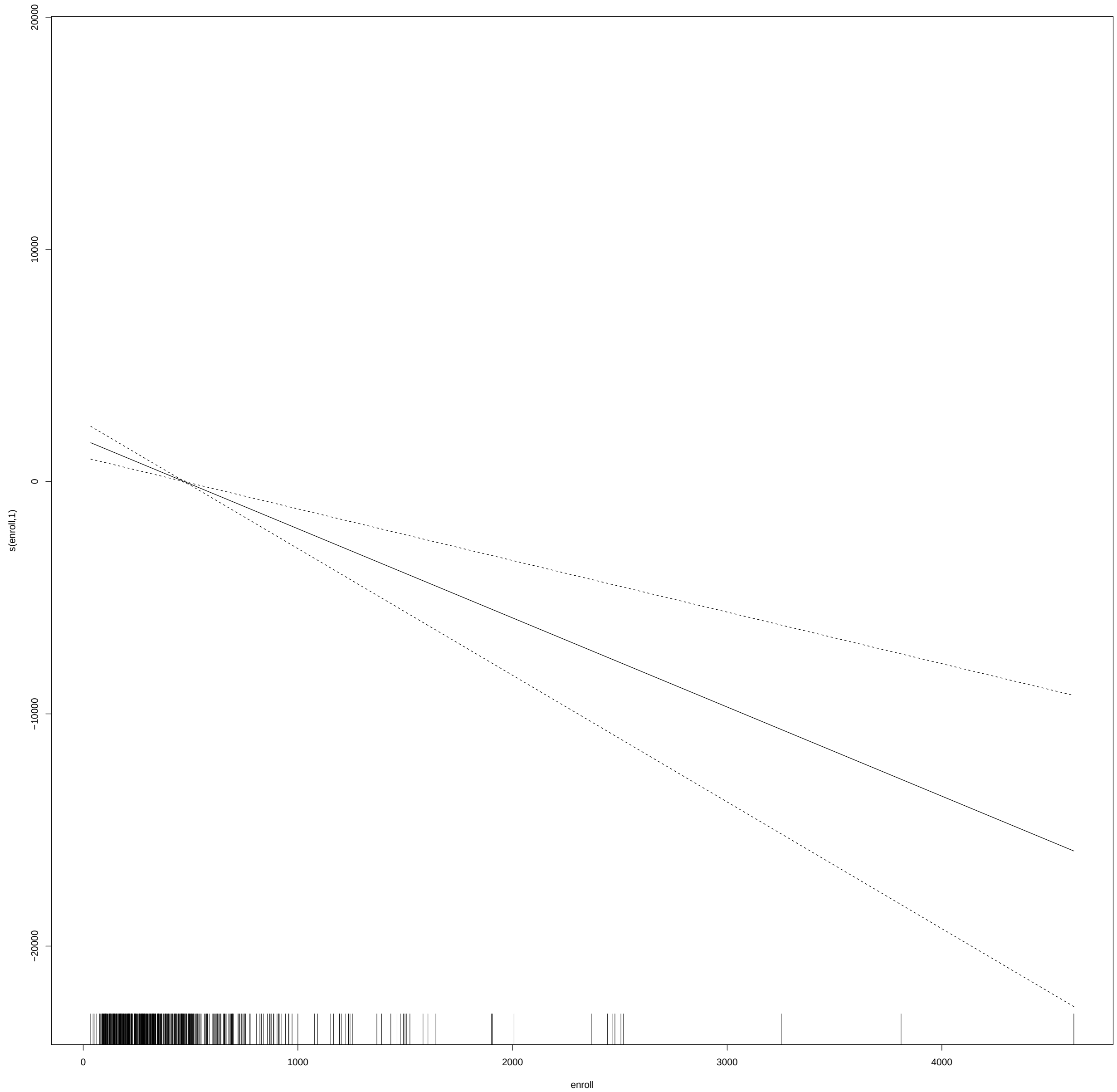


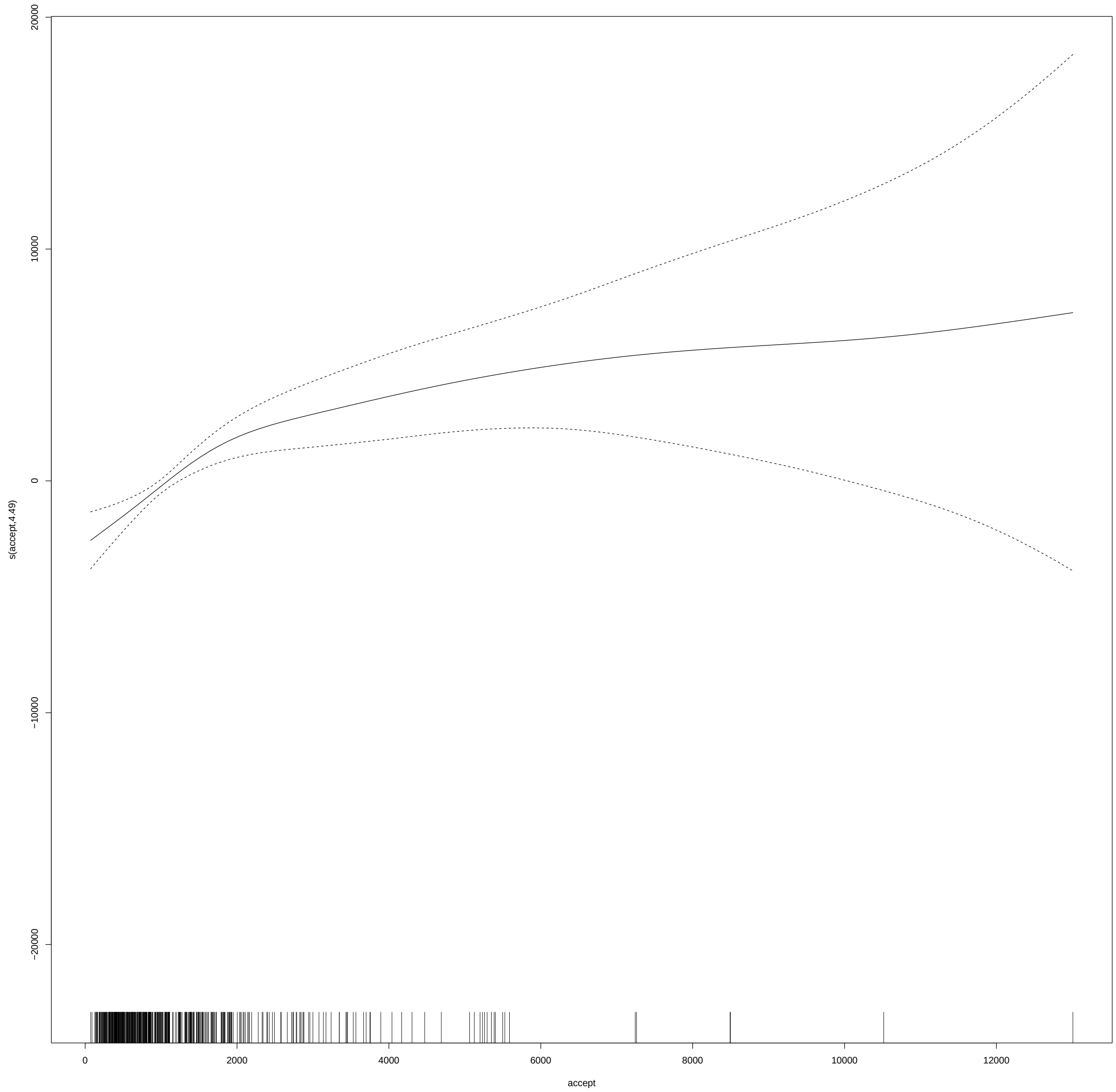
20000
10000
0
−10000
−20000
s(personal,1)
0
1000
2000
3000
4000
5000
6000
7000
personal

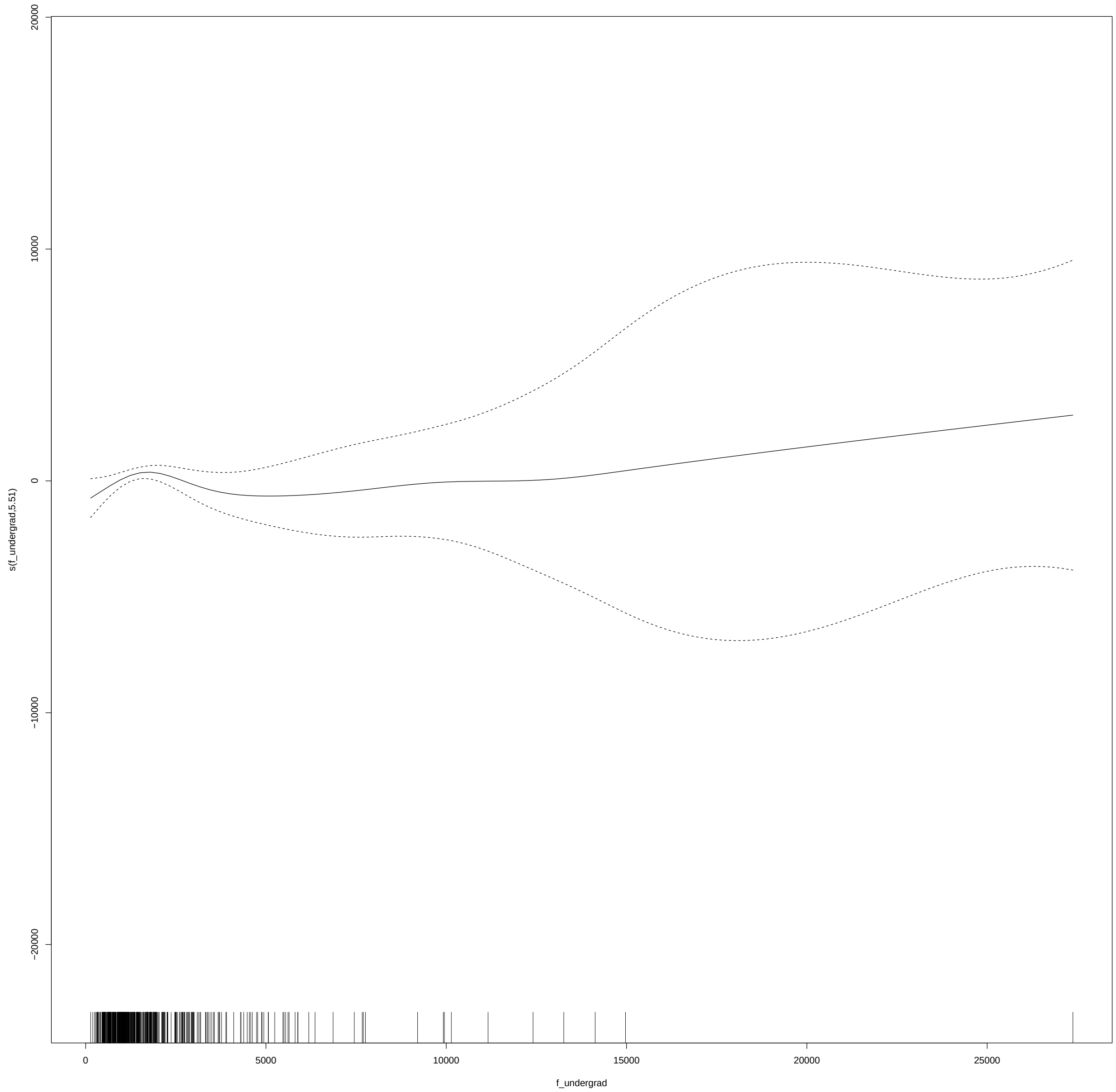

20000
10000
0
−10000
−20000
s(p_undergrad,1)
0
2000
4000
6000
8000
10000
p_undergrad

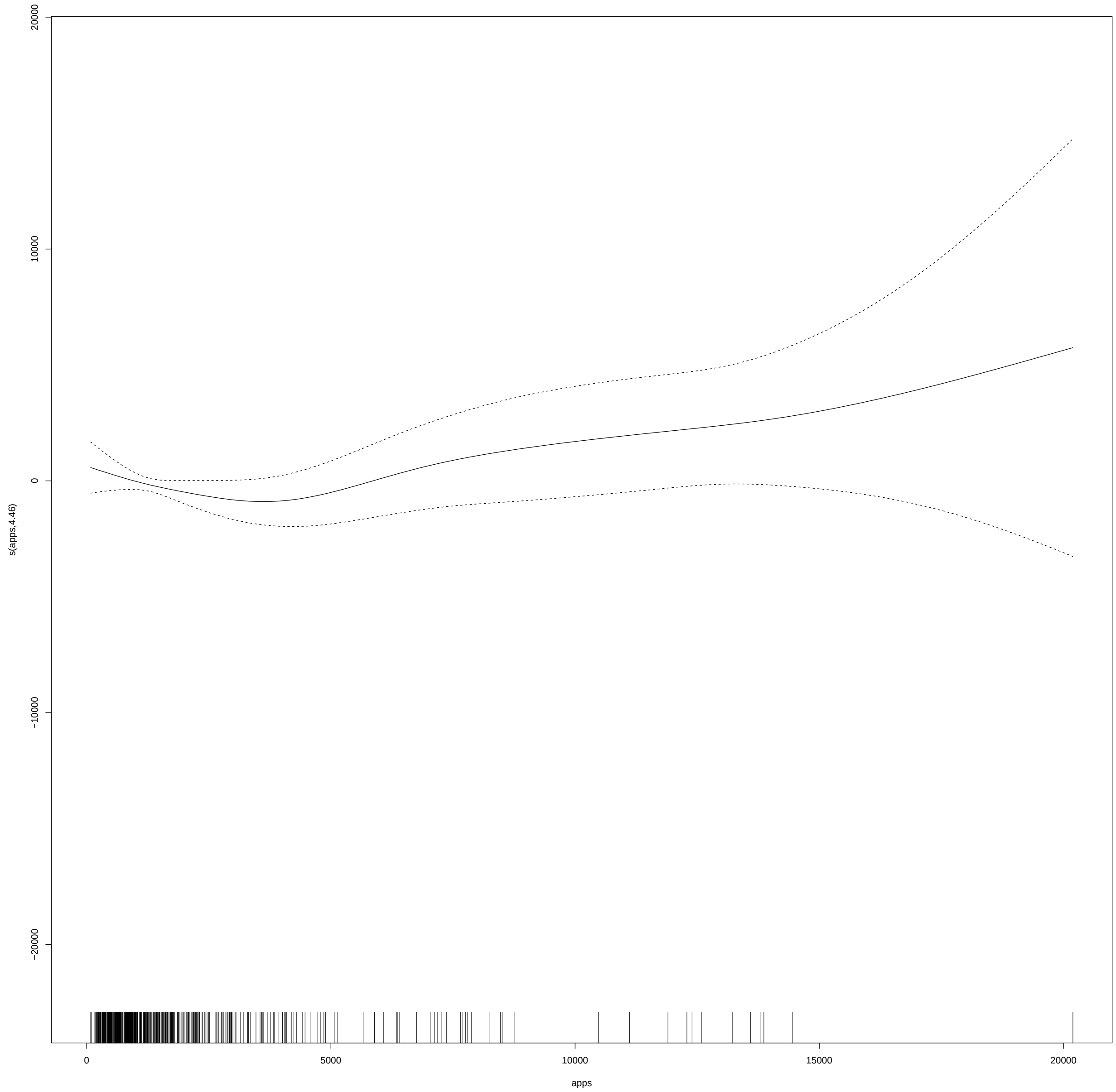


20000
10000
0
−10000
−20000
s(room_board,2.38)
3000
4000
5000
6000
7000
8000
room_board

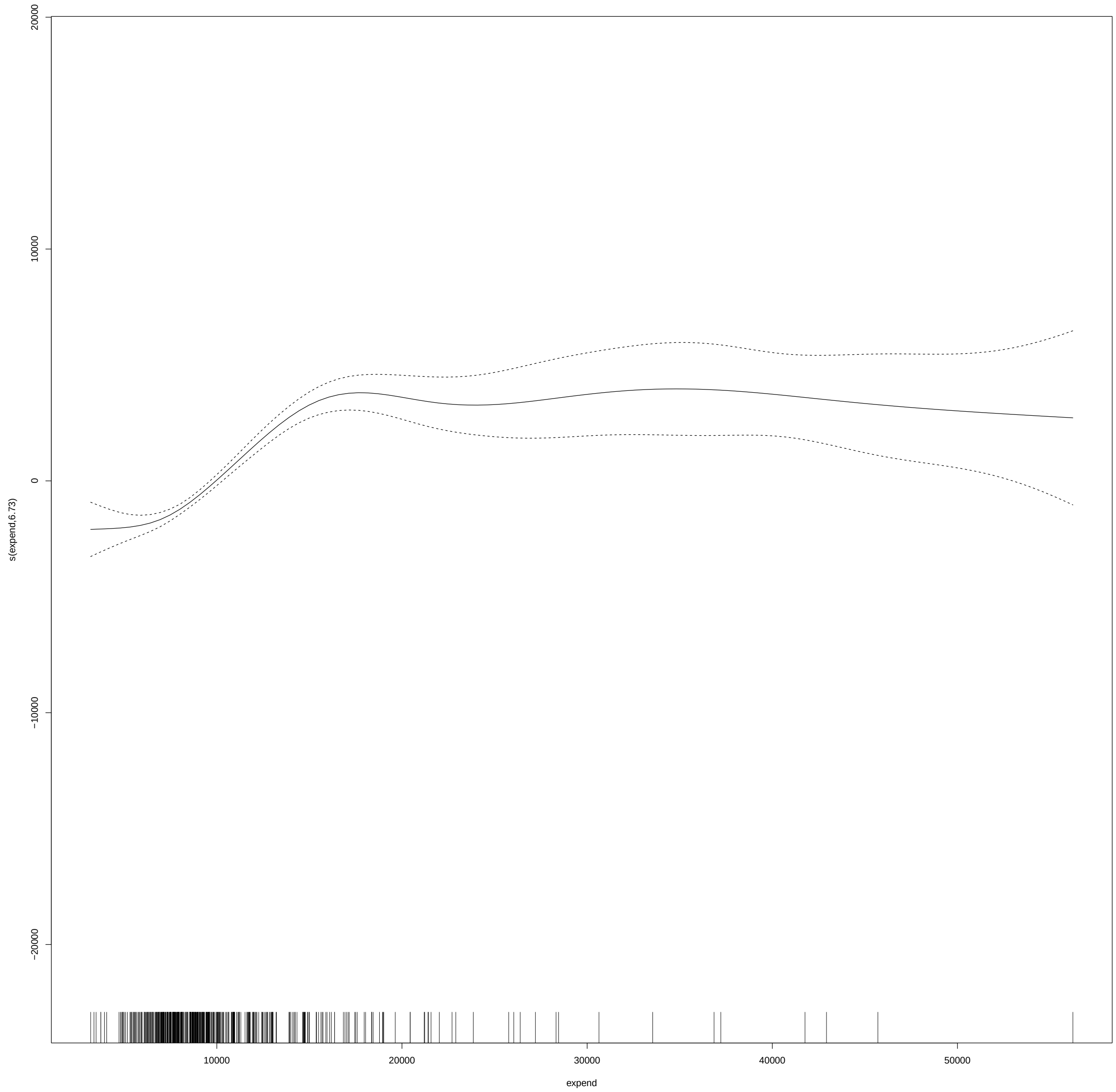

20000
10000
0
−10000
−20000
s(enroll,1)
0
1000
2000
3000
4000
enroll


s(accept,4.49)
20000
10000
0
−10000
−20000
0
2000
4000
6000
8000
10000
12000
accept


20000
10000
0
−10000
−20000
s(f_undergrad,5.51)
0
5000
10000
15000
20000
25000
f_undergrad



s(apps,4.46)
20000
10000
0
−10000
−20000
0
5000
10000
15000
20000
apps



Report test error.

```
# Calculate training MSE of GAM model 1
gam_train_MSE = mean((y_train - predict(gam_fit_all))^2)
gam_train_MSE
```

```
## [1] 2260226
```

```
gam_train_RMSE = sqrt(gam_train_MSE)
gam_train_RMSE
```

```
## [1] 1503.405
```

```r
# Calculate test MSE of GAM model 1
test_predictions = predict(gam_fit_all, x_test)
gam_test_MSE = mean((y_test - test_predictions)^2)
gam_test_MSE
```

```
## [1] 3012372
```

```r
gam_test_RMSE = sqrt(gam_test_MSE)
gam_test_RMSE
```

```
## [1] 1735.619
```

```r
# Calculate training MSE of GAM model 2
gam_train_MSE = mean((y_train - predict(gam_fit_alt))^2)
gam_train_MSE
```

```
## [1] 2279817
```

```r
gam_train_RMSE = sqrt(gam_train_MSE)
gam_train_RMSE
```

```
## [1] 1509.906
```

```r
# Calculate test MSE of GAM model 2
test_predictions = predict(gam_fit_alt, x_test)
gam_test_MSE = mean((y_test - test_predictions)^2)
gam_test_MSE
```

```
## [1] 3010842
```

```r
gam_test_RMSE = sqrt(gam_test_MSE)
gam_test_RMSE
```

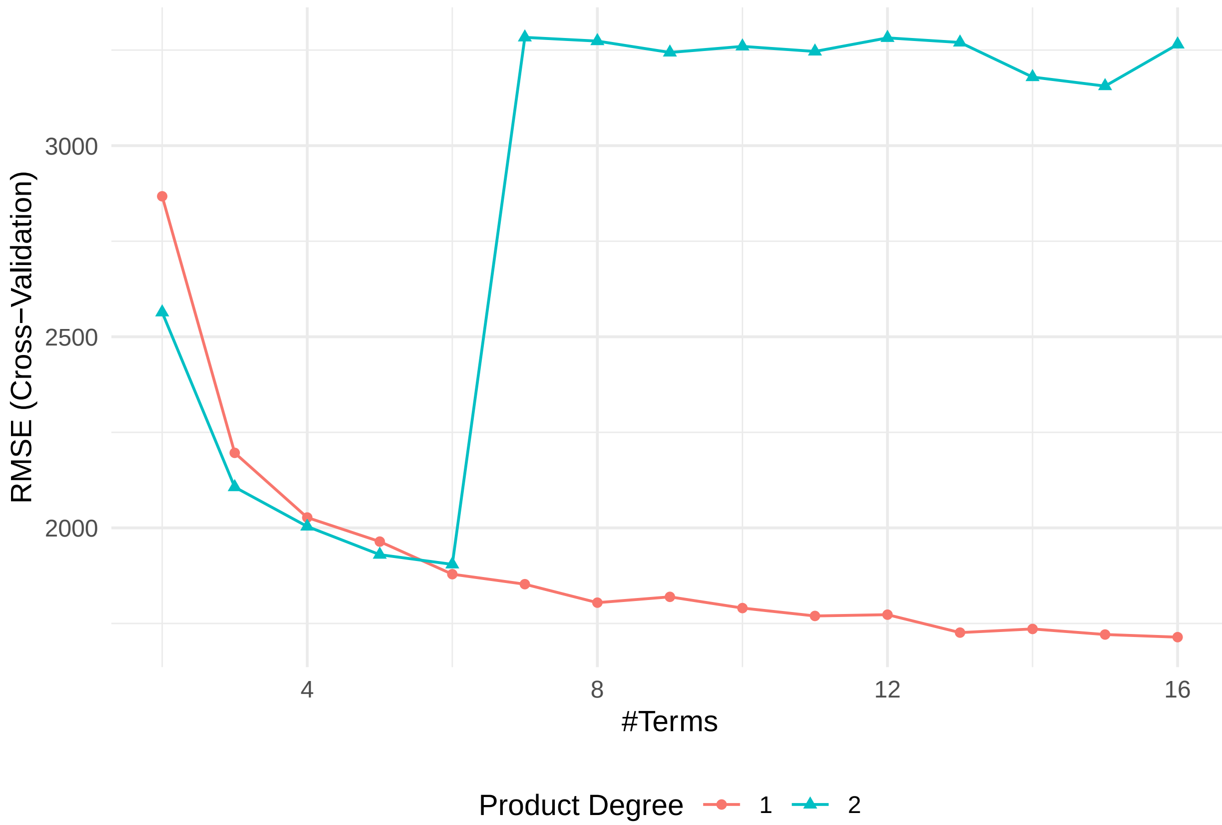
```
## [1] 1735.178
```

We may choose to fit our GAM using either MCGV or the `caret` package. Notably, the latter may result in loss of flexibility since it automatically precludes the possibility of nonlinear transformations for predictors that take fewer than 10 unique values. However, in this case, all of our predictors take more than 10 unique values, and so we do not expect loss of flexibility by using `caret`.

- Using all of our predictors, our model has an MSE of 2260226 (RMSE 1503.405) when we apply our model to the training data and an MSE of 3012372 (RMSE 1735.619) when we apply it to the testing data from our original partitioning.
- Alternativelly, the other model has an MSE of 2279817 (RMSE 1509.906) when we apply our model to the training data and an MSE of 3010842 (RMSE 1735.178) when we apply it to the testing data from our original partitioning.
- In summary, when using all of our predictors, we have effective degrees of freedom, which represent the complexity of the smooth function. `terminal`, `top10perc`, `top25perc`, `personal`, `p_undergrad`, and `enroll` all have df = 1, corresponding to a linear trend. Those with dfs around 2, such as `room_board`` andbooks, suggesting a quadratic relationship, whereas those with dfs around three, such as`grad_rateandaccept‘, are cubically incorporated, and so forth.

c). Train a multivariate adaptive regression spline (MARS) model using all the predictors

```
mars_grid <- expand.grid(degree = 1:2,
                         nprune = 2:16)
set.seed(1)
mars.fit <- train(x_train, y_train,
                  method = "earth",
                  tuneGrid = mars_grid,
                  trControl = ctrl1)
ggplot(mars.fit)
```




```
mars.fit$bestTune
```

```
##    nprune degree
## 15     16      1
```

Report the model

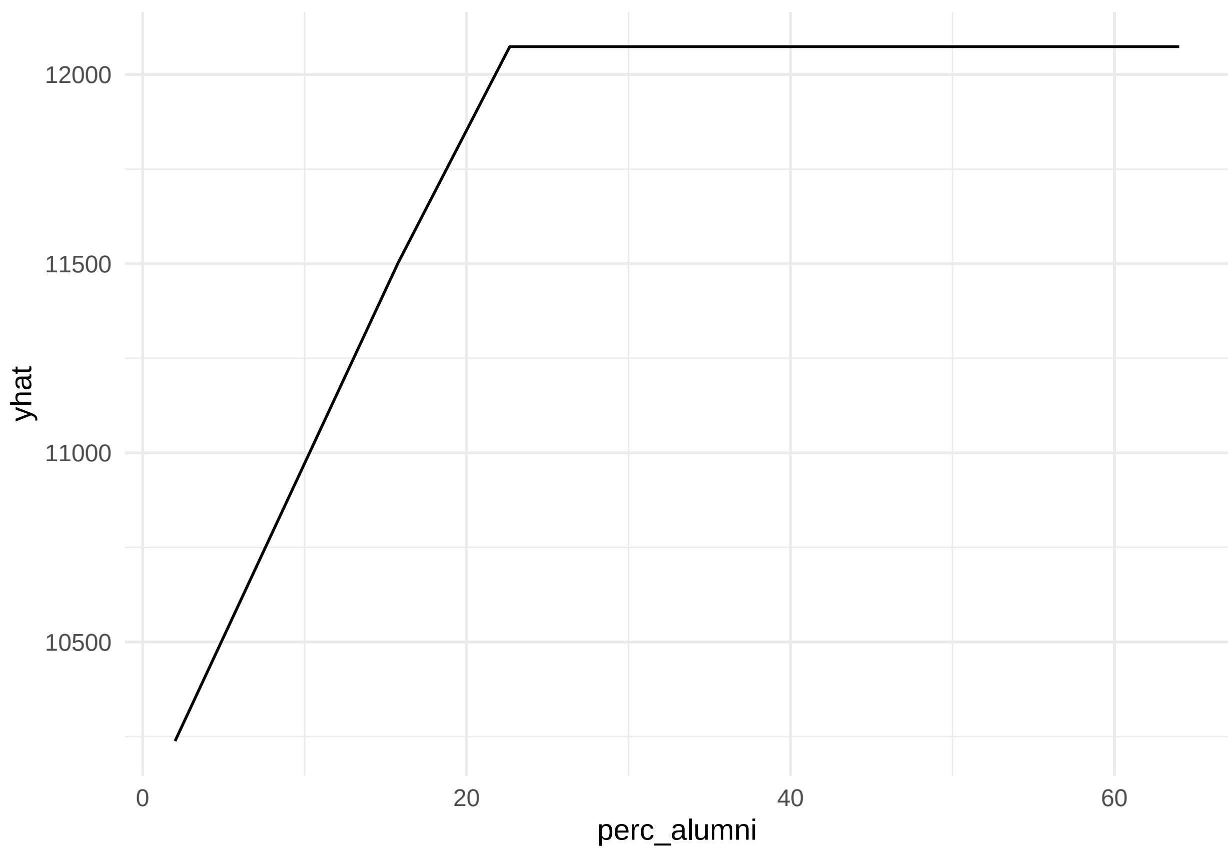
```
# The final MARS model has the following predictors, coefficients, and hinge functions:
coef(mars.fit$finalModel)
```

```
##          (Intercept)      h(expend-15886)      h(79-grad_rate)   h(room_board-4323)
```

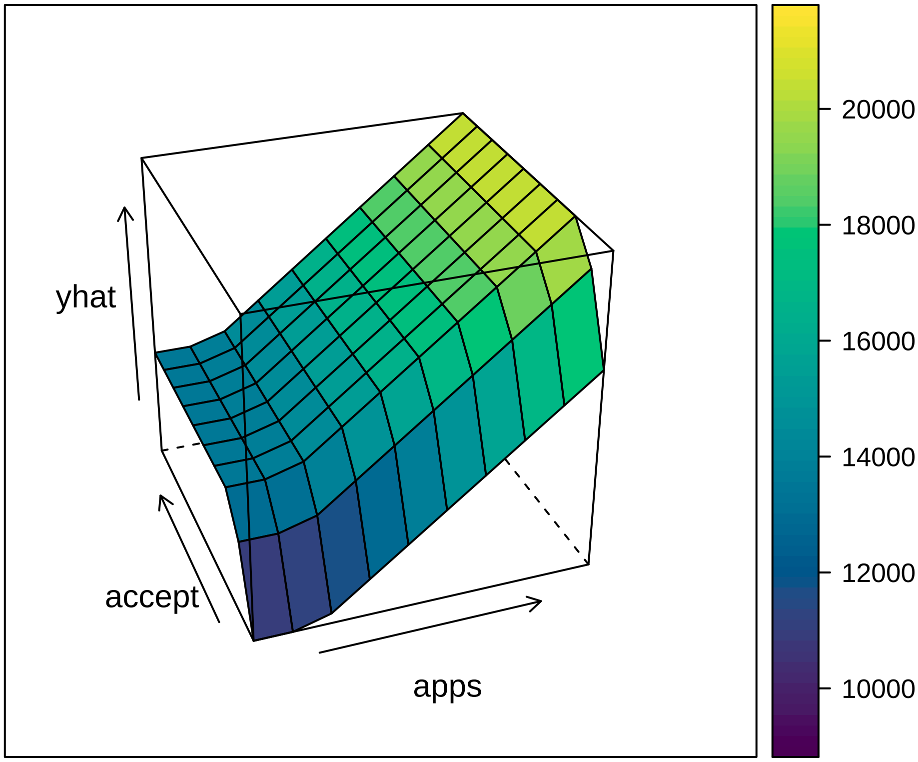
```
##        9750.9084463          -0.7366761         -27.4149388           0.3555943
##  h(4323-room_board) h(1379-f_undergrad)   h(22-perc_alumni)        h(apps-3712)
##          -1.0463218          -1.5733517         -91.7755202           0.4447256
##    h(1300-personal)      h(expend-6897)       h(enroll-911)       h(911-enroll)
##           0.8665098           0.7149307          -2.0263362           5.7508922
##      h(2109-accept)
##          -1.9904298
```

Partial dependence plot:

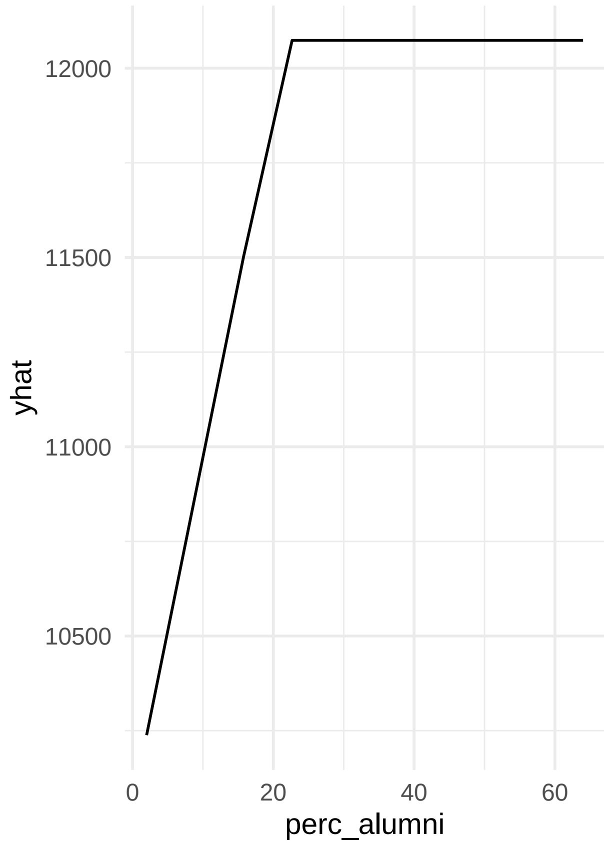
```r
# a single arbitrary predictor in the final model `perc.alumni`:
p1 = pdp::partial(mars.fit, pred.var = c("perc_alumni"), grid.resolution = 10) %>% autoplot()
p1
```

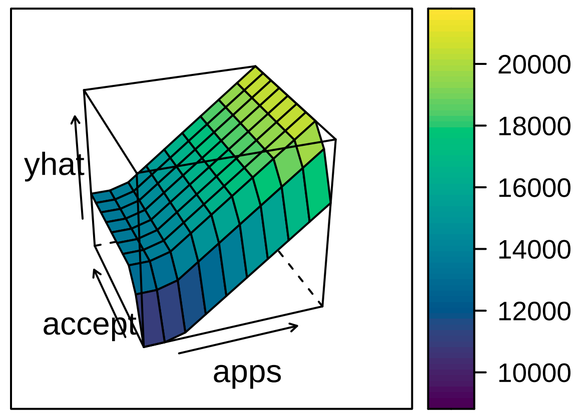



```r
# interaction partial dependence plot between arbitrary predictors in the final model `Apps` and `Accep
p2 = pdp::partial(mars.fit, pred.var =c("apps","accept"), grid.resolution = 10) %>% plotPartial(levelpl
p2
```




```
grid.arrange(p1, p2, ncol = 2)
```





- The partial dependence plot can be used to visualize and analyze interaction between the target response and a set of input features of interest. Here it shows the marginal effect `perc.alumni`, `Apps`, and `Accept` features have on the predicted outcome 'Outcome'.

Report test error

```
set.seed(1)

mars.fit_test <- train(x_test, y_test,
                       method = "earth",
                       tuneGrid = mars_grid,
                       trControl = ctrl1)

mars.fit_test$results$RMSE
```

```
##  [1] 2106.016 2106.016 1990.475 2061.542 1886.144 1863.037 1889.813 1865.985
##  [9] 1943.363 1874.553 2053.224 1787.298 1999.930 1808.253 2063.908 2057.821
## [17] 2063.030 2052.791 2119.763 2057.074 2113.995 2071.716 2107.173 2072.341
## [25] 2107.173 2128.225 2107.173 2130.499 2107.173 2142.306
```

```
## Test Error
mars_pred = predict(mars.fit, x_test)
mars_test_rmse = RMSE(mars_pred, y_test)
mars_test_rmse
```

```
## [1] 1665.72
```

- The final model using MARS is:
- f(x) = 9750.9084463 - 0.74h(expend-15886) - 27.4h(79-grad_rate) + 0.36h(room_board-4323) - 1.05 h(4323-room_board) - 1.57h(1379-f_undergrad) - 91.8h(22-perc_alumni) + 0.44h(apps-3712) + 0.87h(1300-personal) + 0.71h(expend-6897) - 2.03h(enroll-911) + 5.75h(911-enroll) - 1.99h(2109-accept)
- Test error is 1665.72.

e).

Based on the summary results and the plot, MARS model is preferred compared to linear model, since the RMSE of MARS model is much more smaller than linear model, as well as the R squared of MARS is a little bit larger than linear model, which denotes more proportion of y is explained by x.

Here we can see that the test error of MARS (RMSE = 1691.296) is smaller than linear model (RMSE = 1983.44), which further enforces our decision above.