Problem set 7

Jingyuan Zhou

2/25/2017

Part 1: Sexy Joe Biden (redux)

```
blm <- lm(biden ~ age + female + educ + dem + rep, data = biden_data)
tidy(blm)
```

```
##          term     estimate std.error  statistic      p.value
## 1 (Intercept)  58.81125899 3.1244366  18.822996 2.694143e-72
## 2         age   0.04825892 0.0282474   1.708438 8.772744e-02
## 3      female   4.10323009 0.9482286   4.327258 1.592601e-05
## 4        educ  -0.34533479 0.1947796  -1.772952 7.640571e-02
## 5         dem  15.42425563 1.0680327  14.441745 8.144928e-45
## 6         rep -15.84950614 1.3113624 -12.086290 2.157309e-32
```

```
mse <- function(model, data) {
  x <- modelr:::residuals(model, data)
  mean(x ^ 2, na.rm = TRUE)
}

mse(blm, biden_data)
```

```
## [1] 395.2702
```

1.After fitting the linear regression model, the mse of the entire data set is 395.2702.

2.After fitting a linear model using only 70% of the data, the mse of the testing dataset is 399.8303, which is a little bit larger than the previous value.

```
biden_split <- resample_partition(biden_data, c(test = 0.3, train = 0.7))
tlm <-lm(biden ~ age + female + educ + dem + rep, data = biden_split$train)
mse(tlm, biden_split$test)
```

```
## [1] 399.8303
```

```
mse_variable <- function(biden_data){
  biden_split <- resample_partition(biden_data, c(test = 0.7, train = 0.3))
  biden_train <- biden_split$train %>%
    tbl_df()
  biden_test <- biden_split$test %>%
    tbl_df()

  result <- mse(tlm <-lm(biden ~ age + female + educ + dem + rep, data = biden_split$train), biden_spli

  return(result)
}

results <- unlist(rerun(100, mse_variable(biden_data)))
summary(results)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   376.3   395.3   400.1   401.4   407.1   423.9
```

3. Looking at the distribution of mean squared errors of 100 iterations, the 3rd quantile is 14 higher than the 1st quantile value. This shows that this approach is highly unstable and that validation estimates of the test MSE can be highly depending on the observations sampled into the training and test sets.

```
loocv_data <- crossv_kfold(biden_data, k = nrow(biden_data))
loocv_models <- map(loocv_data$train, ~ lm(biden ~ age + female + educ + dem + rep, data = .))
loocv_mse <- map2_dbl(loocv_models, loocv_data$test, mse)
mean(loocv_mse)
```

```
## [1] 397.9555
```

4.Using leave-one-out cross-validation (LOOCV) approach, we get a mean value that's close to 401.7, the average of MSEs of 100 iterations.

```
cv10_data <- crossv_kfold(biden_data, k = 10)
cv10_models <- map(cv10_data$train, ~ lm(biden ~ age + female + educ + dem + rep, data = .))
cv10_mse <- map2_dbl(cv10_models, cv10_data$test, mse)
mean(cv10_mse)
```

```
## [1] 398.0729
```

5.Using 10-fold cross validation, the mean mse we get is 398.1127, which is extremely close to the value that we get using leave-one-out cross-valiation approach.

```
cv_mse <- c()
for (i in 1:100){
  cv10_data <- crossv_kfold(biden_data, k = 10)
  cv10_models <- map(cv10_data$train, ~ lm(biden ~ age + female + educ + dem + rep, data = .))
  cv10_mse <- map2_dbl(cv10_models, cv10_data$test, mse)
  cv_mse[[i]] <- mean(cv10_mse)
}
mean(cv_mse)
```

```
## [1] 397.9661
```

6.Repeating the 10-fold cross-validation approach 100 times using 100 different splits of the observations into 10-folds, the mean mse we get is 398.0694, which is extremely similar to our results from 10-fold cross validation. Thus, in practice, we can safely depend on 10-fold cross validation to get the highest efficiency.

```
# bootstrapped estimates of the parameter estimates and standard errors
biden_boot <- biden_data%>%
  modelr::bootstrap(1000) %>%
  mutate(model = map(strap, ~  lm(biden ~ age + female + educ + dem + rep, data = .)),
         coef = map(model, tidy))

biden_boot %>%
  unnest(coef) %>%
  group_by(term) %>%
  summarize(est.boot = mean(estimate),
            se.boot = sd(estimate, na.rm = TRUE))
```

```
## # A tibble: 6 × 3
##          term     est.boot    se.boot
##         <chr>        <dbl>      <dbl>
## 1 (Intercept)  58.91337251 2.97814255
## 2         age   0.04770968 0.02883481
## 3         dem  15.43020645 1.10724812
## 4        educ  -0.34950530 0.19214401
## 5      female   4.08800549 0.94879605
```

```
## 6         rep -15.87431840 1.44433208
```

```
tidy(blm)
```

```
##          term     estimate std.error  statistic      p.value
## 1 (Intercept)  58.81125899 3.1244366  18.822996 2.694143e-72
## 2         age   0.04825892 0.0282474   1.708438 8.772744e-02
## 3      female   4.10323009 0.9482286   4.327258 1.592601e-05
## 4        educ  -0.34533479 0.1947796  -1.772952 7.640571e-02
## 5         dem  15.42425563 1.0680327  14.441745 8.144928e-45
## 6         rep -15.84950614 1.3113624 -12.086290 2.157309e-32
```

Bootstrapped estimate of intercept is 58.69711076 with sd of 3.07088573, original model estimate of intercept is 58.81125899 with sd of 3.1244366.

Bootstrapped estimate of age is 0.04754621 with sd of 0.02929158, original model estimate of age is 0.04825892 with sd of 0.0282474.

Bootstrapped estimate of dem is 15.43735011 with sd of 1.08848988, original model estimate of dem is 15.42425563 with sd of 1.0680327.

Bootstrapped estimate of educ is -0.33391564 with sd of 0.19947285, original model estimate of educ is -0.34533479 with sd of 0.1947796.

Bootstrapped estimate of female is 4.08901065 with sd of 0.94314140, original model estimate of female is 4.10323009 with sd of 0.9482286.

Bootstrapped estimate of rep is -15.85370969 with sd of 1.42368299, original model estimate of rep is -15.84950614 with sd of 1.3113624.

By comparing values, we can see that both two approaches get very similar estimates. Original model generally has smaller standard deviations for these estimates than the bootstrapped estimates. The reason might be that the true relationship between biden scores and the parameters is indeed linear, and we do not make any assumptions of the distribution with this bootstrap approach.

Part 2: College (bivariate)

```
c_data <- read.csv(file="College.csv",head=TRUE)
glm <-  lm(Outstate~ ., data = c_data)
tidy(glm)
```

```
##           term      estimate    std.error  statistic      p.value
## 1  (Intercept) -1.587267e+03 766.03018305 -2.0720689 3.859619e-02
## 2   PrivateYes  2.263757e+03 247.99111993  9.1283788 6.176363e-19
## 3         Apps -3.034481e-01   0.06733909 -4.5062696 7.638013e-06
## 4       Accept  8.123743e-01   0.12924565  6.2855058 5.507887e-10
## 5       Enroll -5.492393e-01   0.35414380 -1.5508934 1.213441e-01
## 6    Top10perc  2.834130e+01  10.97760762  2.5817373 1.001682e-02
## 7    Top25perc -3.779314e+00   8.47480689 -0.4459469 6.557628e-01
## 8  F.Undergrad -9.566599e-02   0.06152438 -1.5549281 1.203800e-01
## 9  P.Undergrad  1.166082e-02   0.06049460  0.1927580 8.472000e-01
## 10  Room.Board  8.816138e-01   0.08557925 10.3017238 2.205686e-23
## 11       Books -4.592264e-01   0.44785918 -1.0253813 3.055100e-01
## 12    Personal -2.294487e-01   0.11829884 -1.9395681 5.280239e-02
## 13         PhD  1.124167e+01   8.72953279  1.2877751 1.982168e-01
## 14    Terminal  2.467266e+01   9.53843663  2.5866568 9.876200e-03
## 15   S.F.Ratio -4.643932e+01  24.41406299 -1.9021543 5.752927e-02
```

```
## 16 perc.alumni  4.179887e+01   7.56097306  5.5282397 4.450461e-08
## 17      Expend  1.989838e-01   0.02269250  8.7687001 1.176232e-17
## 18   Grad.Rate  2.400159e+01   5.50649138  4.3587813 1.488086e-05
```

```
#the three parameters that I choose are : Apps, Room.Board,Accept
lm1 <- lm(Outstate~ Room.Board, data = c_data)
summary(lm1)
```

```
##
## Call:
## lm(formula = Outstate ~ Room.Board, data = c_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8781.0 -2070.6  -350.8  1877.4 11877.4
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.44525  447.76786  -0.039    0.969
## Room.Board    2.40001    0.09965  24.084   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3044 on 775 degrees of freedom
## Multiple R-squared:  0.4281, Adjusted R-squared:  0.4273
## F-statistic:   580 on 1 and 775 DF,  p-value: < 2.2e-16
```

```
lm2 <- lm(Outstate~ Apps, data = c_data)
summary(lm2)
```

```
##
## Call:
## lm(formula = Outstate ~ Apps, data = c_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8328.2 -3175.8  -402.2  2525.8 11388.8
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.028e+04  1.826e+02  56.328   <2e-16 ***
## Apps        5.214e-02  3.729e-02   1.398    0.162
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4021 on 775 degrees of freedom
## Multiple R-squared:  0.002516,   Adjusted R-squared:  0.001229
## F-statistic: 1.955 on 1 and 775 DF,  p-value: 0.1625
```

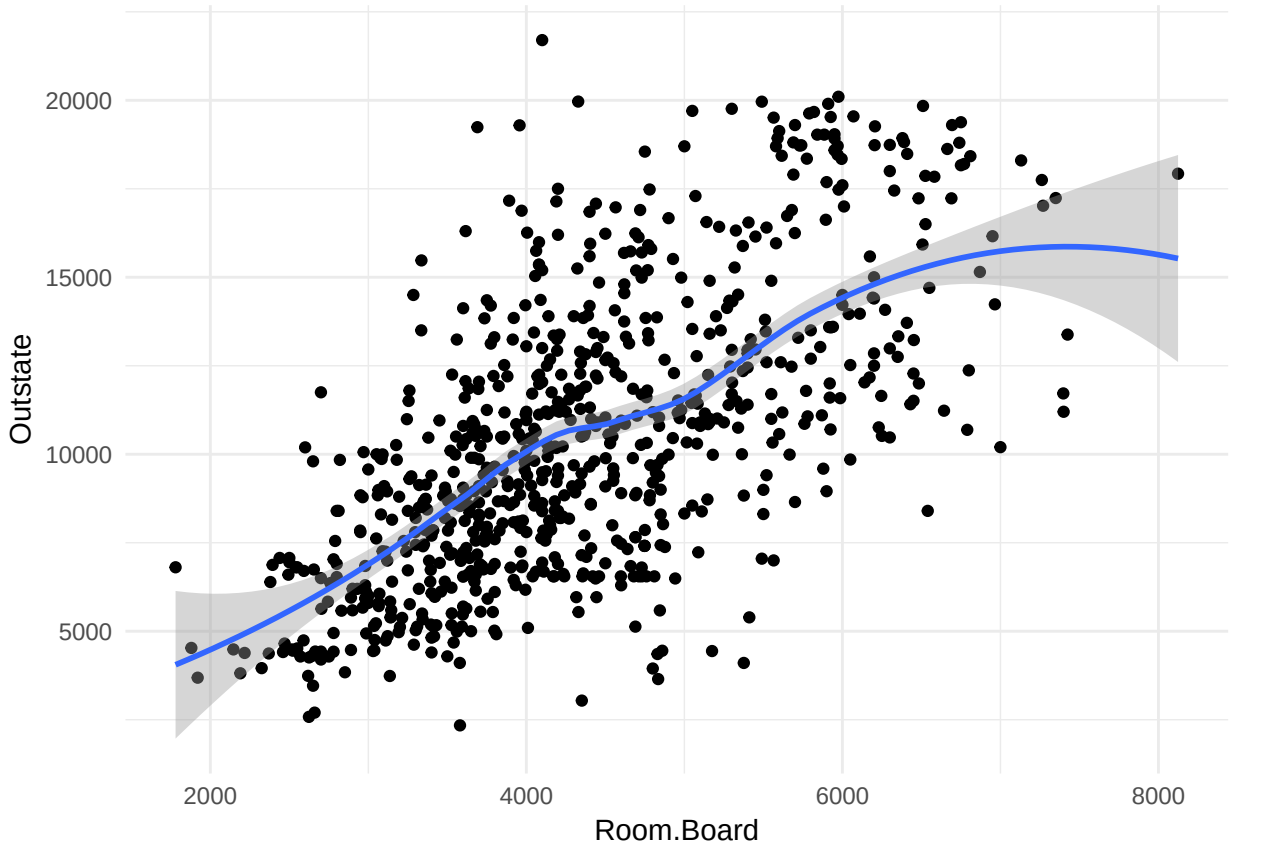
```
lm3 <- lm(Outstate~ Accept, data = c_data)
summary(lm3)
```

```
##
## Call:
## lm(formula = Outstate ~ Accept, data = c_data)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7957.7 -3080.4  -512.2  2425.3 11187.8
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.053e+04  1.871e+02  56.264   <2e-16 ***
## Accept      -4.227e-02  5.894e-02  -0.717    0.473
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4024 on 775 degrees of freedom
## Multiple R-squared:  0.0006633,  Adjusted R-squared:  -0.0006262
## F-statistic: 0.5144 on 1 and 775 DF,  p-value: 0.4735
```

After fitting the three linear models, we get that the model with Room.Board has R^2 as 0.4281; Apps has R^2 as 0.002516; Accept has R^2 as 0.0006633. This shows that Room.Board is most likely to have a linear relationship, so we can start from that.

```
ggplot(c_data, aes(Room.Board, Outstate)) +
  geom_point() +
  geom_smooth()
```




```
  labs(title = 'Room.Board against Outstate')
```

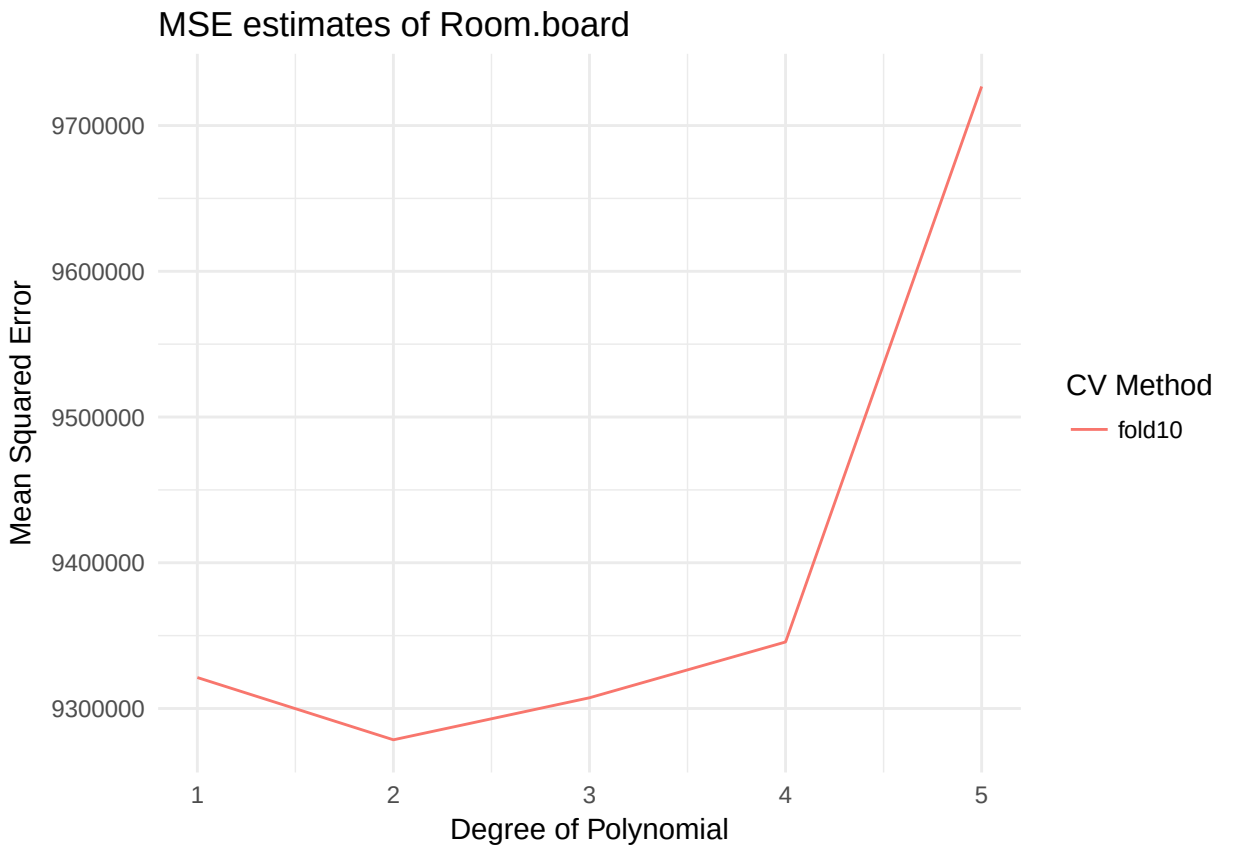
```
## $title
## [1] "Room.Board against Outstate"
##
## attr(,"class")
```

```
## [1] "labels"
```

```
set.seed(1234)
cv10_data <- crossv_kfold(c_data, k = 10)

cv_error_fold10 <- vector("numeric", 5)
terms <- 1:5

for(i in terms){
  cv10_models <- map(cv10_data$train, ~ lm(Outstate ~ poly(Room.Board, i), data = .))
  cv10_mse <- map2_dbl(cv10_models, cv10_data$test, mse)
  cv_error_fold10[[i]] <- mean(cv10_mse)
}

cv_error_fold10
```

```
## [1] 9321262 9278530 9307359 9345650 9726796
```

```
data_frame(terms = terms,
           fold10 = cv_error_fold10) %>%
  gather(method, MSE, fold10) %>%
  ggplot(aes(terms, MSE, color = method)) +
  geom_line() +
  labs(title = "MSE estimates of Room.board",
       x = "Degree of Polynomial",
       y = "Mean Squared Error",
       color = "CV Method")
```



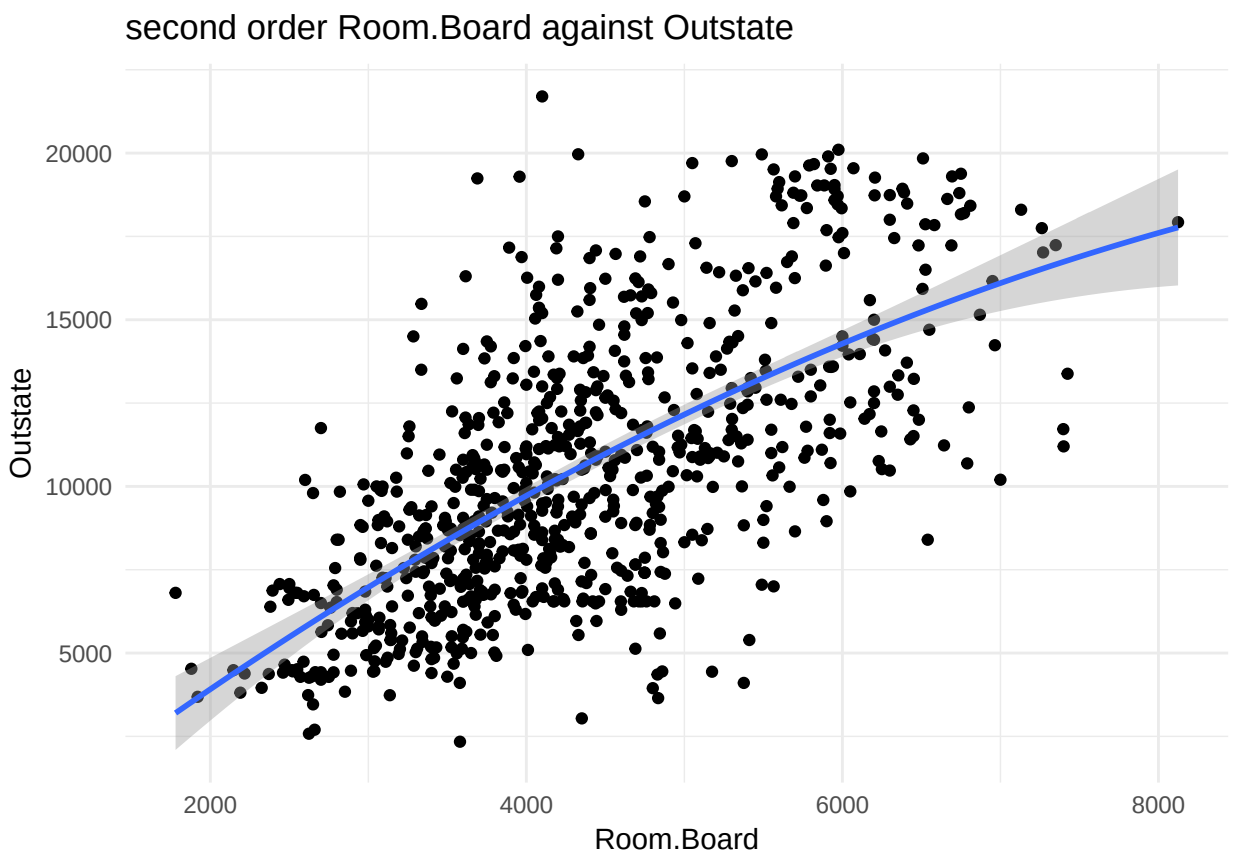
Plotting Room.Board shows that there is indeed a linear relationship. We use cross-validation methods

to adjustify this finding. From the plot, we see that 2-degree has the lowest mse, so we fit a 2-degree polynomial for this variable.

```
rb_2 <- lm(Outstate~ poly(Room.Board, 2), data = c_data)
summary(rb_2)
```

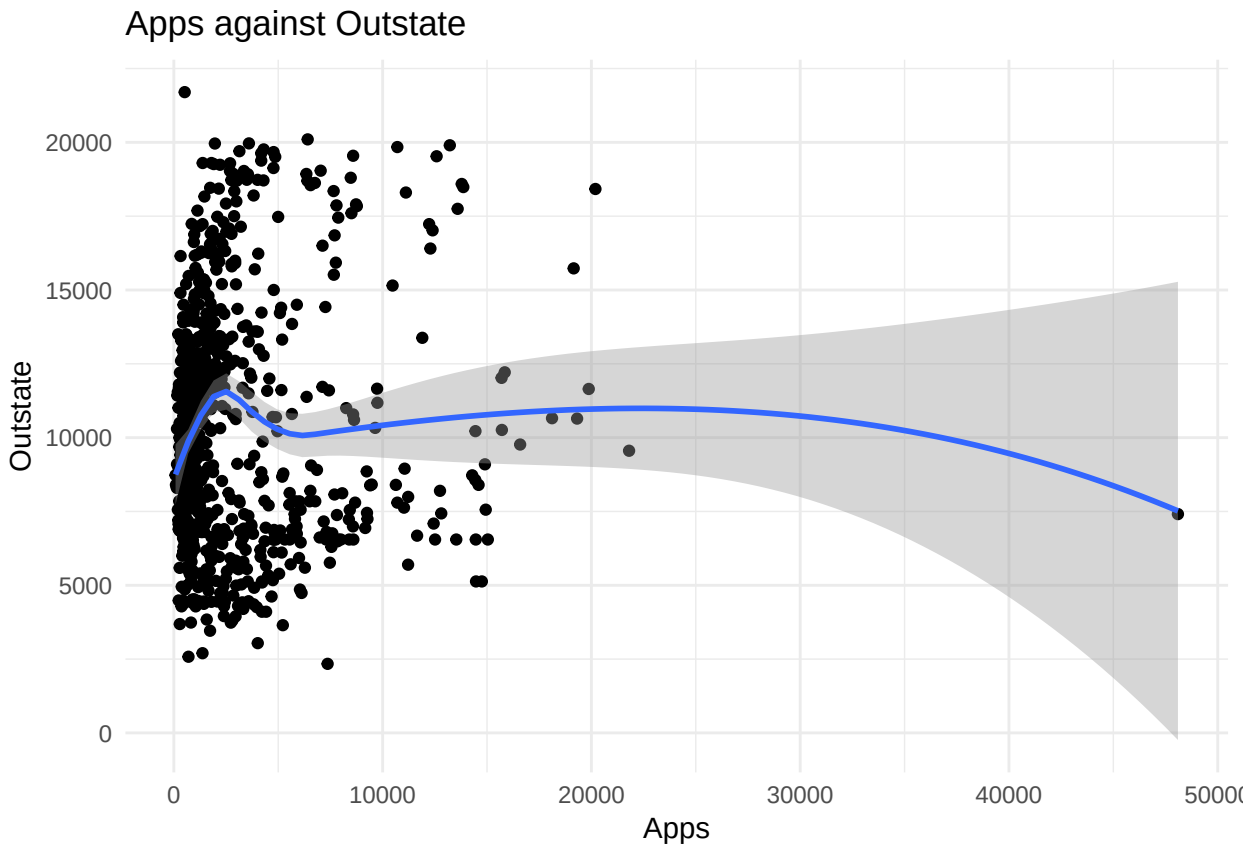
```
##
## Call:
## lm(formula = Outstate ~ poly(Room.Board, 2), data = c_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8889.1 -2057.8  -318.4  1896.0 11722.0
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)              10441        109  95.817   <2e-16 ***
## poly(Room.Board, 2)1     73321       3037  24.140   <2e-16 ***
## poly(Room.Board, 2)2     -6535       3037  -2.151   0.0317 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3037 on 774 degrees of freedom
## Multiple R-squared:  0.4315, Adjusted R-squared:   0.43
## F-statistic: 293.7 on 2 and 774 DF,  p-value: < 2.2e-16
```

```
#plot
ggplot(c_data, aes(x = Room.Board, y = Outstate)) +
  geom_point() +
  geom_smooth(method = 'lm', formula = y ~ poly(x, 2)) +
  labs(title = 'second order Room.Board against Outstate',x = 'Room.Board',y = 'Outstate')
```



R^2 is 0.4317, which is slightly higher than previous linear model. We obtained a desired bivariate model for Room.Board.

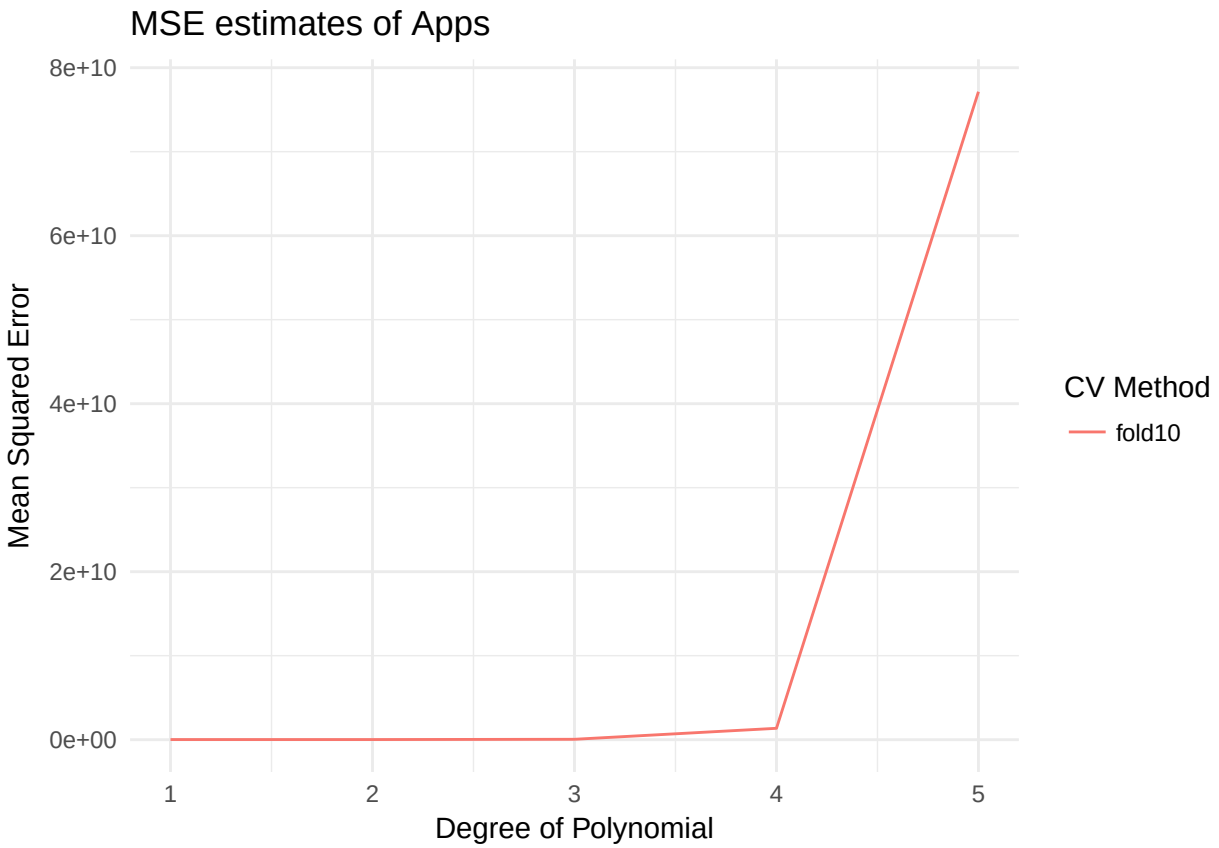
```
ggplot(c_data, aes(Apps, Outstate)) +
  geom_point() +
  geom_smooth()+
  labs(title = 'Apps against Outstate')
```




```
set.seed(1234)
cv10_data <- crossv_kfold(c_data, k = 10)

cv_error_fold10 <- vector("numeric", 5)
terms <- 1:5

for(i in terms){
  cv10_models <- map(cv10_data$train, ~ lm(Outstate ~ poly(Apps, i), data = .))
  cv10_mse <- map2_dbl(cv10_models, cv10_data$test, mse)
  cv_error_fold10[[i]] <- mean(cv10_mse)
}

cv_error_fold10
```

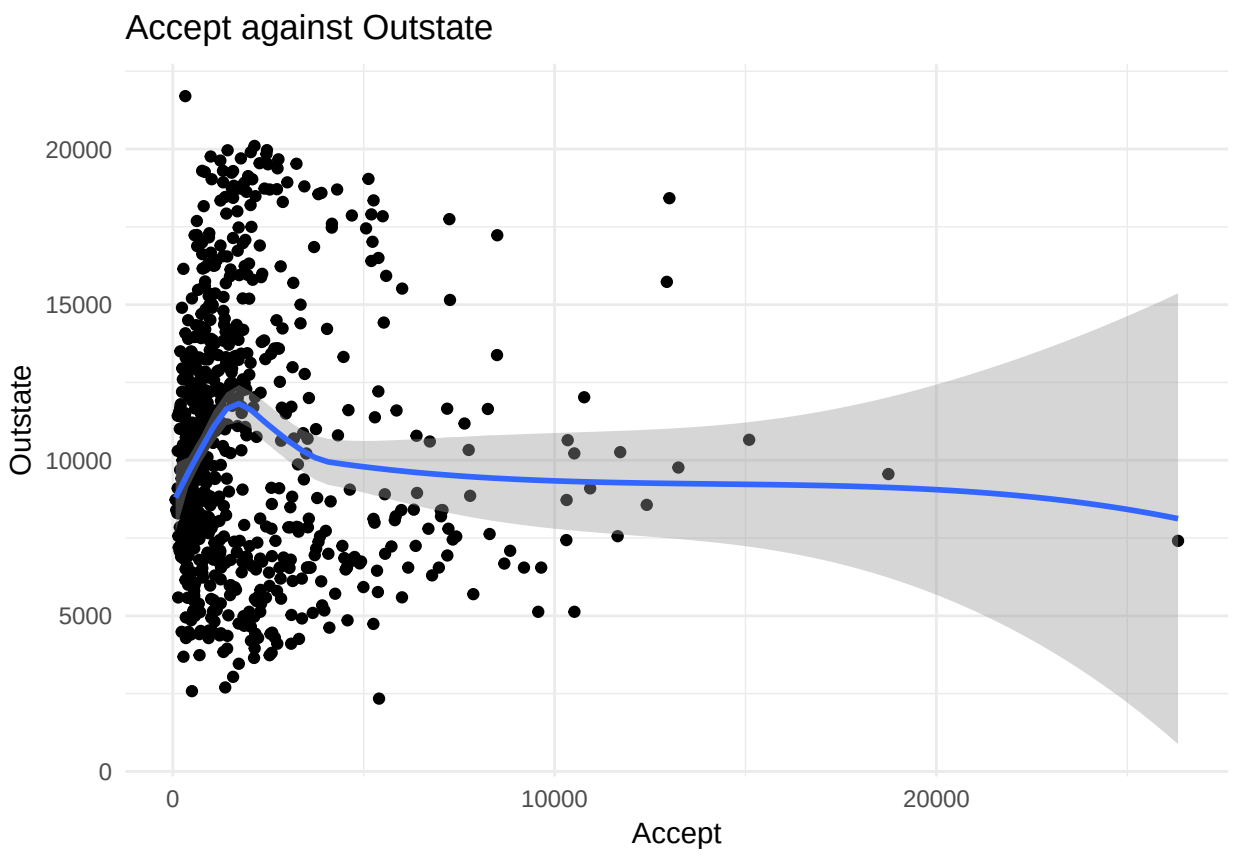
```
## [1]    16254452    16356751    49679343  1356483150 77141304615
```

```
data_frame(terms = terms,
           fold10 = cv_error_fold10) %>%
  gather(method, MSE, fold10) %>%
  ggplot(aes(terms, MSE, color = method)) +
  geom_line() +
  labs(title = "MSE estimates of Apps",
       x = "Degree of Polynomial",
       y = "Mean Squared Error",
       color = "CV Method")
```



The plot shows that increase degrees of polynomial does not lead to a smaller mse. It seems that even thought the R^2 of the linear model is not very satisfying, it is the best we can do for a bivariate model with variable Apps.

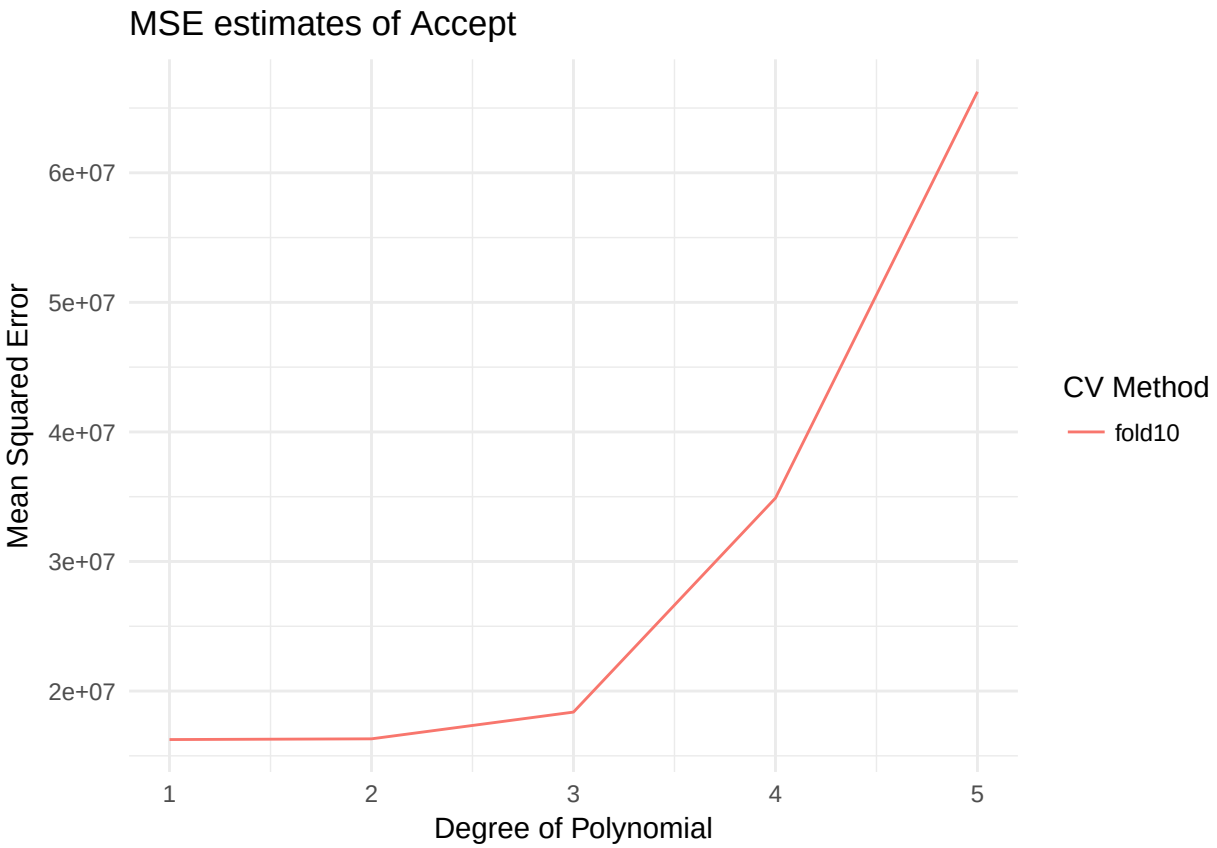
```
ggplot(c_data, aes(Accept, Outstate)) +
  geom_point() +
  geom_smooth()+
  labs(title = 'Accept against Outstate')
```




```
set.seed(1234)
cv10_data <- crossv_kfold(c_data, k = 10)

cv_error_fold10 <- vector("numeric", 5)
terms <- 1:5

for(i in terms){
  cv10_models <- map(cv10_data$train, ~ lm(Outstate ~ poly(Accept, i), data = .))
  cv10_mse <- map2_dbl(cv10_models, cv10_data$test, mse)
  cv_error_fold10[[i]] <- mean(cv10_mse)
}

cv_error_fold10
```

```
## [1] 16255985 16311222 18375920 34894757 66257750
```

```
data_frame(terms = terms,
           fold10 = cv_error_fold10) %>%
  gather(method, MSE, fold10) %>%
  ggplot(aes(terms, MSE, color = method)) +
  geom_line() +
  labs(title = "MSE estimates of Accept",
       x = "Degree of Polynomial",
       y = "Mean Squared Error",
       color = "CV Method")
```



Similar to the situation of Apps, increasing the degrees of polynomial does not seem to give lower values for mse. Thus, the best bivariate model we can have with Apps is linear even though the R^2 value is very low.

Part 3: College (GAM)

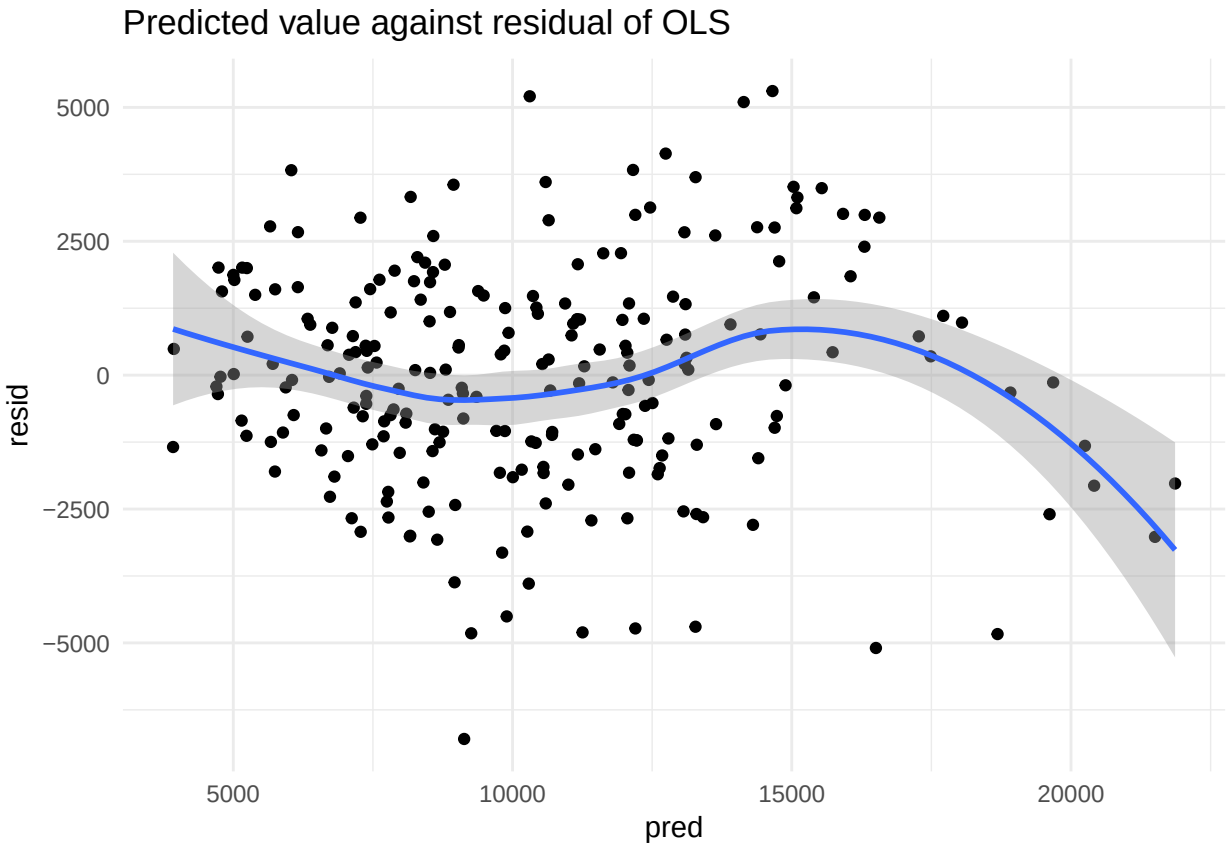
```
c_split <- resample_partition(c_data, c(test = 0.7, train = 0.3))
```

```
ols <- lm(Outstate~ Private + Room.Board + PhD + perc.alumni + Expend + Grad.Rate, data = c_split$train)
summary(ols)
```

```
##
## Call:
## lm(formula = Outstate ~ Private + Room.Board + PhD + perc.alumni +
##     Expend + Grad.Rate, data = c_split$train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6793.8 -1262.1    38.1  1354.2  5305.2
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.875e+03  8.743e+02  -5.575 6.99e-08 ***
## PrivateYes   2.548e+03  3.956e+02   6.441 7.02e-10 ***
## Room.Board   1.060e+00  1.562e-01   6.790 9.69e-11 ***
## PhD          3.850e+01  1.055e+01   3.649 0.000327 ***
## perc.alumni  4.413e+01  1.498e+01   2.947 0.003549 **
```

```
## Expend       1.508e-01  3.044e-02   4.955 1.41e-06 ***
## Grad.Rate    5.391e+01  1.043e+01   5.166 5.22e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2082 on 227 degrees of freedom
## Multiple R-squared:  0.7547, Adjusted R-squared:  0.7482
## F-statistic: 116.4 on 6 and 227 DF,  p-value: < 2.2e-16
```

```r
train <- as.data.frame(c_split$train)

grid <- train %>%
  add_predictions(ols)%>%
  add_residuals(ols)

#plot
ggplot(grid, aes(x = pred, y = resid)) +
  geom_point() +
  geom_smooth() +
  labs(title = 'Predicted value against residual of OLS',x = 'pred',y = 'resid')
```

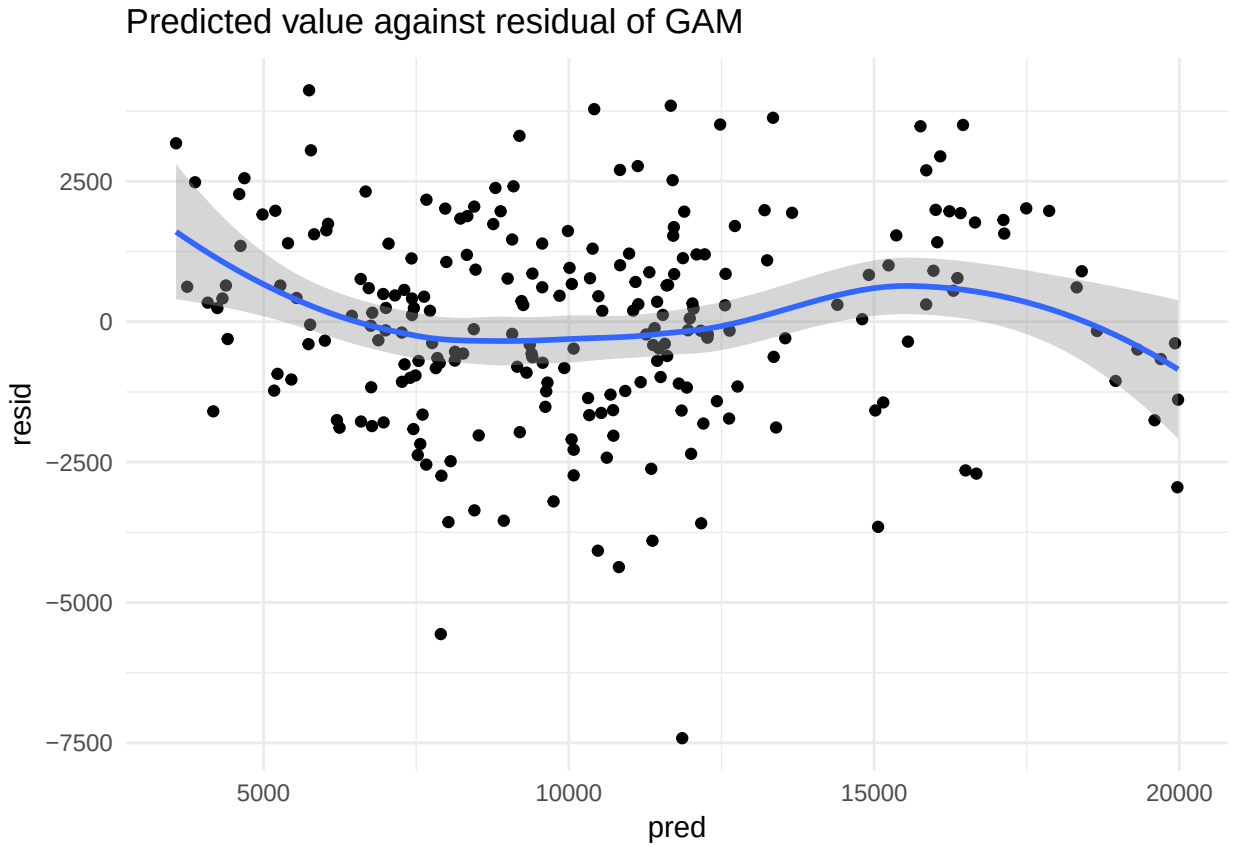


2.The OLS model using the other six variables as predictors has a R^2 value of 0.7665. This shows that it explains 76.7% of the data. Besided, all six variable are statistically significant at 95% confidence interval because their pvalues are all less than 0.025. Visualizing the model fit through the prediction against residual plot, we can see that this model fit is reasonable but not satisfying.

```r
library(gam)
c_gam <- gam(Outstate ~ Private +lo(perc.alumni) + lo(PhD) + lo(Expend) + lo(Grad.Rate) + lo(Room.Board
tidy(c_gam)
```

```
##              term       df     sumsq    meansq statistic      p.value
## 1         Private   1.0000 997059898 997059898 285.29613 3.919504e-41
## 2 lo(perc.alumni)   1.0000 492434933 492434933 140.90405 2.943106e-25
## 3         lo(PhD)   1.0000 398144684 398144684 113.92409 1.444809e-21
## 4      lo(Expend)   1.0000 569545885 569545885 162.96838 4.546086e-28
## 5   lo(Grad.Rate)   1.0000 136760252 136760252  39.13222 2.151629e-09
## 6  lo(Room.Board)   1.0000  83939142  83939142  24.01813 1.893087e-06
## 7       Residuals 212.2389 741737556   3494824        NA           NA
```

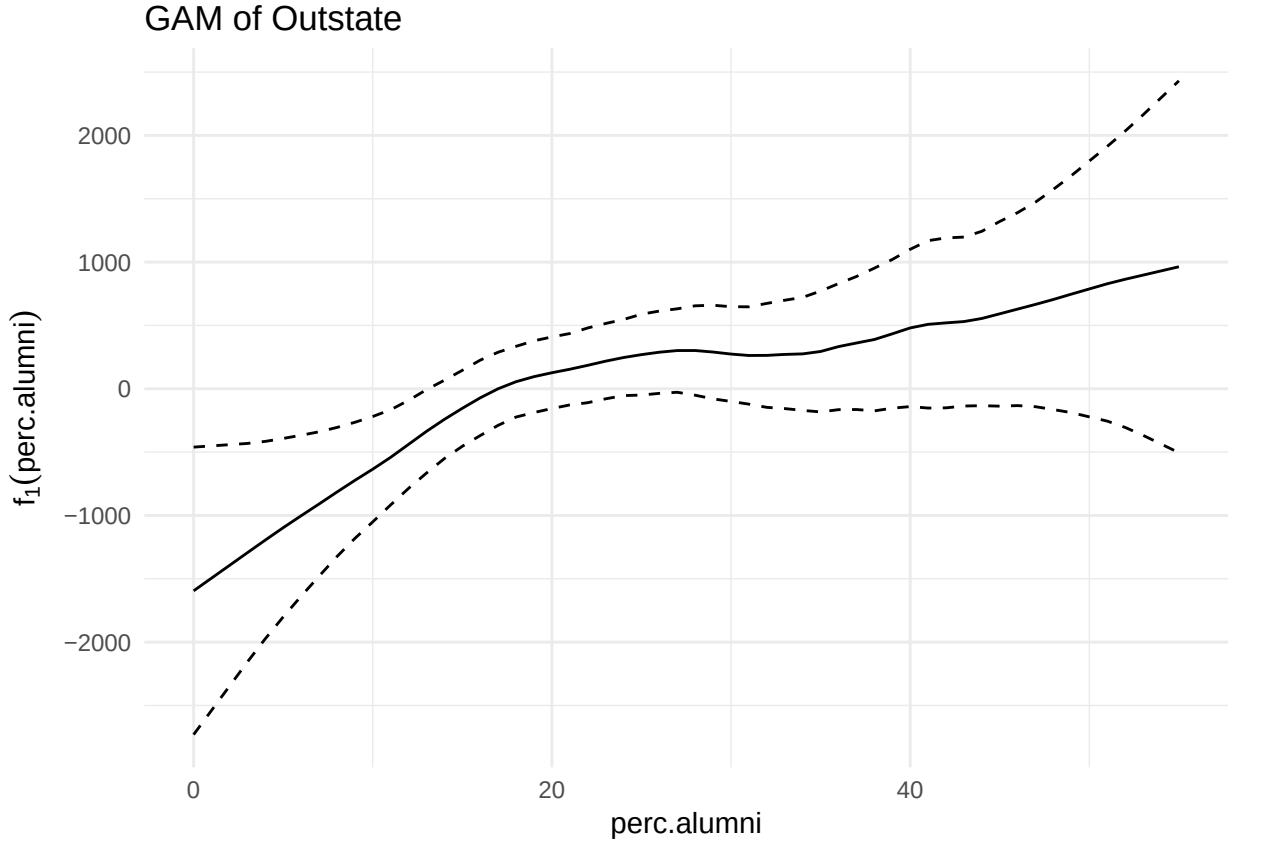
```
grid_gam <- train %>%
  add_predictions(c_gam)%>%
  add_residuals(c_gam)

#plot
ggplot(grid_gam, aes(x = pred, y = resid)) +
  geom_point() +
  geom_smooth()+
  labs(title = 'Predicted value against residual of GAM',x = 'pred',y = 'resid')
```



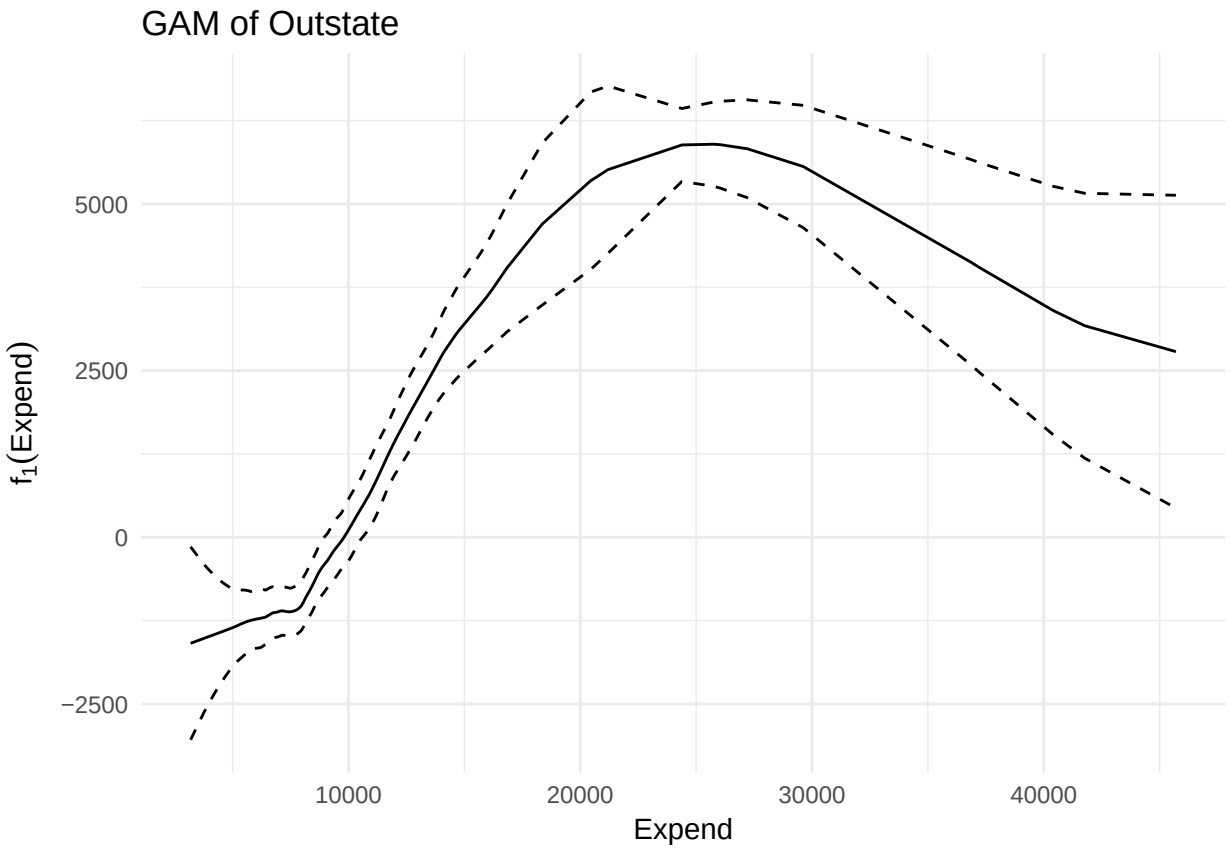
3.With this GAM model, all variables have statistically significant p-values. Again, we visualize the model by plotting the predicted value against residual. We can see that the error is more stable through different predictions, so it's a better model than OLS.

```
#top three statistically significant quantitative variables are lo(perc.alumni), lo(Expend), lo(PhD)
# get graphs of each term
 c_gam_terms <- preplot(c_gam, se = TRUE, rug = FALSE)

## lo(Apps)
data_frame(x = c_gam_terms$`lo(perc.alumni)`$x,
```

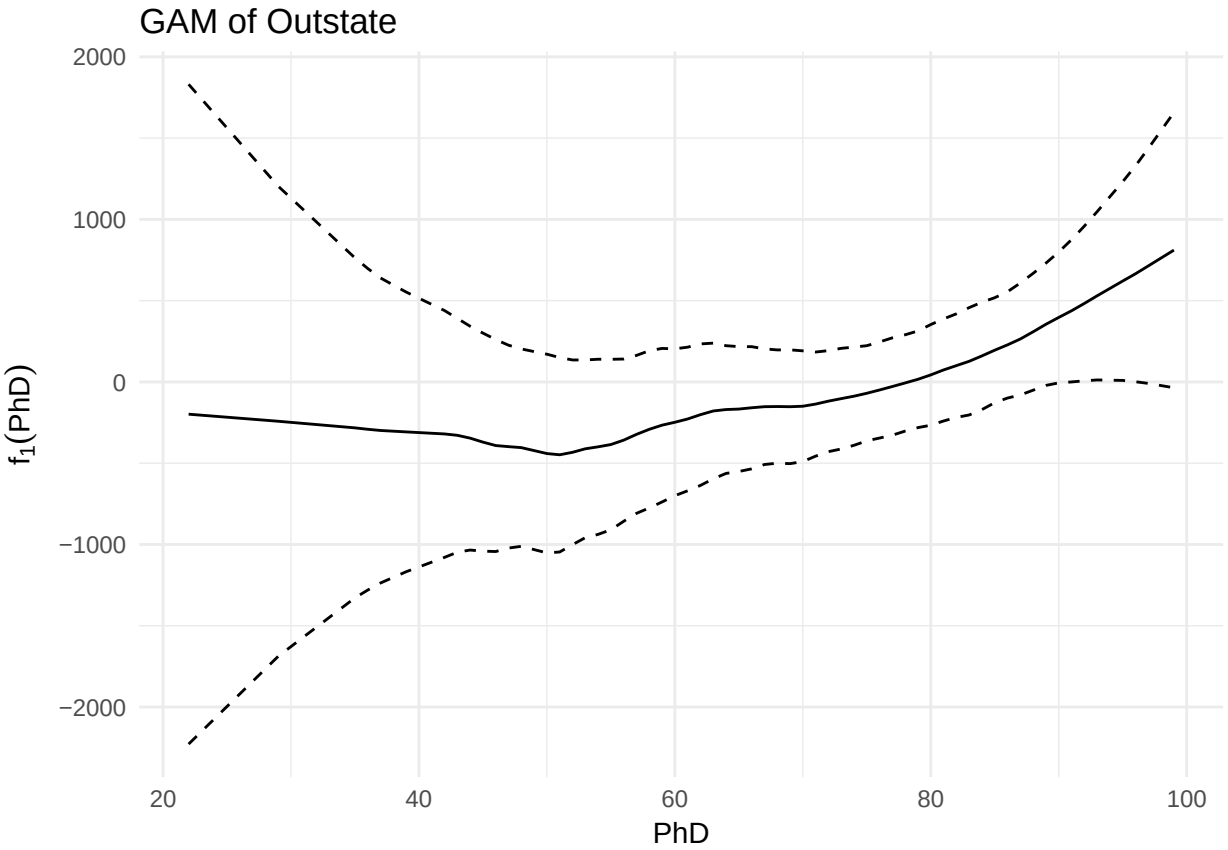
```
          y = c_gam_terms$`lo(perc.alumni)`$y,
          se.fit =  c_gam_terms$`lo(perc.alumni)`$se.y) %>%
  mutate(y_low = y - 1.96 * se.fit,
         y_high = y + 1.96 * se.fit) %>%
  ggplot(aes(x, y)) +
  geom_line() +
  geom_line(aes(y = y_low), linetype = 2) +
  geom_line(aes(y = y_high), linetype = 2) +
  labs(title = "GAM of Outstate",
       x = "perc.alumni",
       y = expression(f[1](perc.alumni)))
```




```
## lo(Room.Board)
data_frame(x = c_gam_terms$`lo(Expend)`$x,
           y = c_gam_terms$`lo(Expend)`$y,
           se.fit =  c_gam_terms$`lo(Expend)`$se.y) %>%
  mutate(y_low = y - 1.96 * se.fit,
         y_high = y + 1.96 * se.fit) %>%
  ggplot(aes(x, y)) +
  geom_line() +
  geom_line(aes(y = y_low), linetype = 2) +
  geom_line(aes(y = y_high), linetype = 2) +
  labs(title = "GAM of Outstate",
       x = "Expend",
       y = expression(f[1](Expend)))
```




```
## lo(Top10perc)
data_frame(x = c_gam_terms$`lo(PhD)`$x,
           y = c_gam_terms$`lo(PhD)`$y,
           se.fit =  c_gam_terms$`lo(PhD)`$se.y) %>%
  mutate(y_low = y - 1.96 * se.fit,
         y_high = y + 1.96 * se.fit) %>%
  ggplot(aes(x, y)) +
  geom_line() +
  geom_line(aes(y = y_low), linetype = 2) +
  geom_line(aes(y = y_high), linetype = 2) +
  labs(title = "GAM of Outstate",
       x = 'PhD',
       y = expression(f[1](PhD)))
```



Top three statistically significant quantitative variables are lo(perc.alumni), lo(Expend), lo(PhD), so we graphed each of these three terms. As perc.alumni increases, the outstate value contiuously increase. As Expend increases, outstate firstly increases and then decreases as Expend reaches around 3000. The parabola shape suggests that there is a quadratic relationship between Expend and Outstate. As PhD increases, outstate continuously increases as well. It seems to be a linear relationship.

```
mse(ols, c_split$test)
```

```
## [1] 4312245
```

```
mse(c_gam, c_split$test)
```

```
## [1] 19312847
```

4.MSE value of GAM model on the test set has a slighter smaller value than the OLS model. This shows that the GAM model indeed is a better model than OLS.

```
summary(c_gam)
```

```
##
## Call: gam(formula = Outstate ~ Private + lo(perc.alumni) + lo(PhD) +
##     lo(Expend) + lo(Grad.Rate) + lo(Room.Board), data = train)
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -7416.8 -1082.4   119.5  1196.0  4120.7
##
## (Dispersion Parameter for gaussian family taken to be 3494824)
##
##     Null Deviance: 4009571567 on 233 degrees of freedom
## Residual Deviance: 741737556 on 212.2389 degrees of freedom
```

```
## AIC: 4212.375
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                     Df    Sum Sq   Mean Sq F value    Pr(>F)
## Private           1.00 997059898 997059898 285.296 < 2.2e-16 ***
## lo(perc.alumni)   1.00 492434933 492434933 140.904 < 2.2e-16 ***
## lo(PhD)           1.00 398144684 398144684 113.924 < 2.2e-16 ***
## lo(Expend)        1.00 569545885 569545885 162.968 < 2.2e-16 ***
## lo(Grad.Rate)     1.00 136760252 136760252  39.132 2.152e-09 ***
## lo(Room.Board)    1.00  83939142  83939142  24.018 1.893e-06 ***
## Residuals       212.24 741737556   3494824
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                 Npar Df  Npar F     Pr(F)
## (Intercept)
## Private
## lo(perc.alumni)     2.3  1.4829   0.22710
## lo(PhD)             2.6  0.6129   0.58649
## lo(Expend)          4.1 13.0389 1.103e-09 ***
## lo(Grad.Rate)       2.9  2.4325   0.06858 .
## lo(Room.Board)      2.8  3.2393   0.02536 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
gam_li_e <-gam(Outstate ~ Private +lo(perc.alumni) + lo(PhD) + Expend + lo(Grad.Rate) + lo(Room.Board)

anova(gam_li_e, c_gam)
```

```
## Analysis of Deviance Table
##
## Model 1: Outstate ~ Private + lo(perc.alumni) + lo(PhD) + Expend + lo(Grad.Rate) +
##     lo(Room.Board)
## Model 2: Outstate ~ Private + lo(perc.alumni) + lo(PhD) + lo(Expend) +
##     lo(Grad.Rate) + lo(Room.Board)
##   Resid. Df Resid. Dev    Df  Deviance Pr(>Chi)
## 1    216.35  900061420
## 2    212.24  741737556 4.107 158323864 3.98e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.Looking at the ANOVA test, only lo(Expend) has a statistically significant result from the Nonparametric Effect. This might be due to the fact that the relationship between it and outstate is nonlinear as we have found out in the last part. Thus, we performed an ANOVA test between the original model and a GAM with linear Expend. In the test, the original model is more significant. This shows that Expend is indeed nonlinear. All the other variables should be linear.