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
```

```r
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)

```r
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 with smallest p-values are: Private, Room.Board,Accept
lm1 <- lm(Outstate~ Room.Board, data = c_data)
tidy(lm1)
```

```
##          term   estimate    std.error   statistic      p.value
## 1 (Intercept) -17.445254 447.76785808 -0.03896049 9.689319e-01
## 2  Room.Board   2.400012   0.09965361 24.08354107 4.135091e-96
```

```
lm2 <- lm(Outstate~ Private, data = c_data)
tidy(lm2)
```

```
##          term estimate std.error statistic       p.value
## 1 (Intercept) 6813.410  230.4223  29.56924 3.098874e-129
## 2  PrivateYes 4988.283  270.2158  18.46037  2.400798e-63
```

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

```
##          term      estimate    std.error  statistic       p.value
## 1 (Intercept)  1.052601e+04 187.08247753 56.2639869 6.714539e-276
## 2      Accept -4.227097e-02   0.05893773 -0.7172141  4.734581e-01
```

Part 3: College (GAM)

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

1.Split the data into a training set and a test set.

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

```
##          term      estimate    std.error statistic      p.value
## 1 (Intercept) -4874.5426486 874.32169219 -5.575228 6.988453e-08
## 2  PrivateYes  2547.8478627 395.58881343  6.440647 7.018206e-10
## 3  Room.Board     1.0603228   0.15615644  6.790132 9.687439e-11
## 4         PhD    38.5041462  10.55231669  3.648881 3.268033e-04
## 5 perc.alumni    44.1276447  14.97627189  2.946504 3.548949e-03
## 6      Expend     0.1508409   0.03043943  4.955445 1.410736e-06
## 7   Grad.Rate    53.9106802  10.43492908  5.166368 5.221519e-07
```

```
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_line(aes(y = pred), data = grid, color = "red", size = 1)
#labs(title = 'Plot of Biden score against age with Least Squares Regression Line',x = 'Age',y = 'Biden
```

2.Estimate an OLS model on the training data, using out-of-state tuition (Outstate) as the response variable and the other six variables as the predictors. Interpret the results and explain your findings, using appropriate techniques (tables, graphs, statistical tests, etc.).

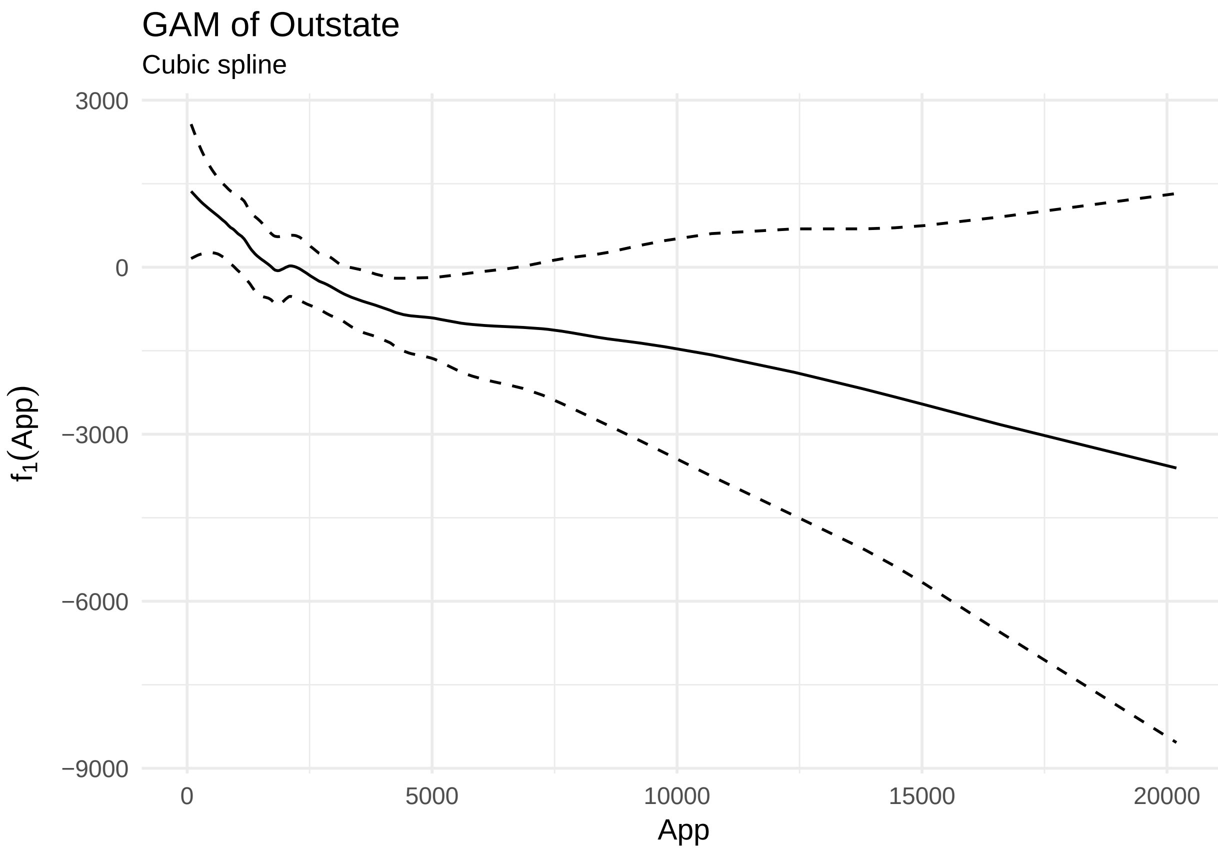
```
library(gam)
c_gam <- gam(Outstate ~ Private + lo(Apps) + lo(Accept) + lo(Enroll) + lo(Top10perc) + lo(Top25perc) +
tidy(c_gam)
```

```
##               term       df      sumsq     meansq   statistic      p.value
## 1          Private   1.0000 1091596199 1091596199 244.4632101 2.339587e-36
## 2         lo(Apps)   1.0000  582892416  582892416 130.5388855 1.601952e-23
## 3       lo(Accept)   1.0000   10770933   10770933   2.4121527 1.220025e-01
## 4       lo(Enroll)   1.0000  295893474  295893474  66.2654089 4.410008e-14
## 5    lo(Top10perc)   1.0000  394110715  394110715  88.2611817 1.475796e-17
## 6    lo(Top25perc)   1.0000    3697183    3697183   0.8279848 3.639684e-01
## 7  lo(F.Undergrad)   1.0000   21220846   21220846   4.7524131 3.044334e-02
## 8  lo(P.Undergrad)   1.0000    3138190    3138190   0.7027984 4.028615e-01
## 9   lo(Room.Board)   1.0000  323386742  323386742  72.4225324 4.383932e-15
## 10       Residuals 196.8998  879212303    4465278          NA           NA
```

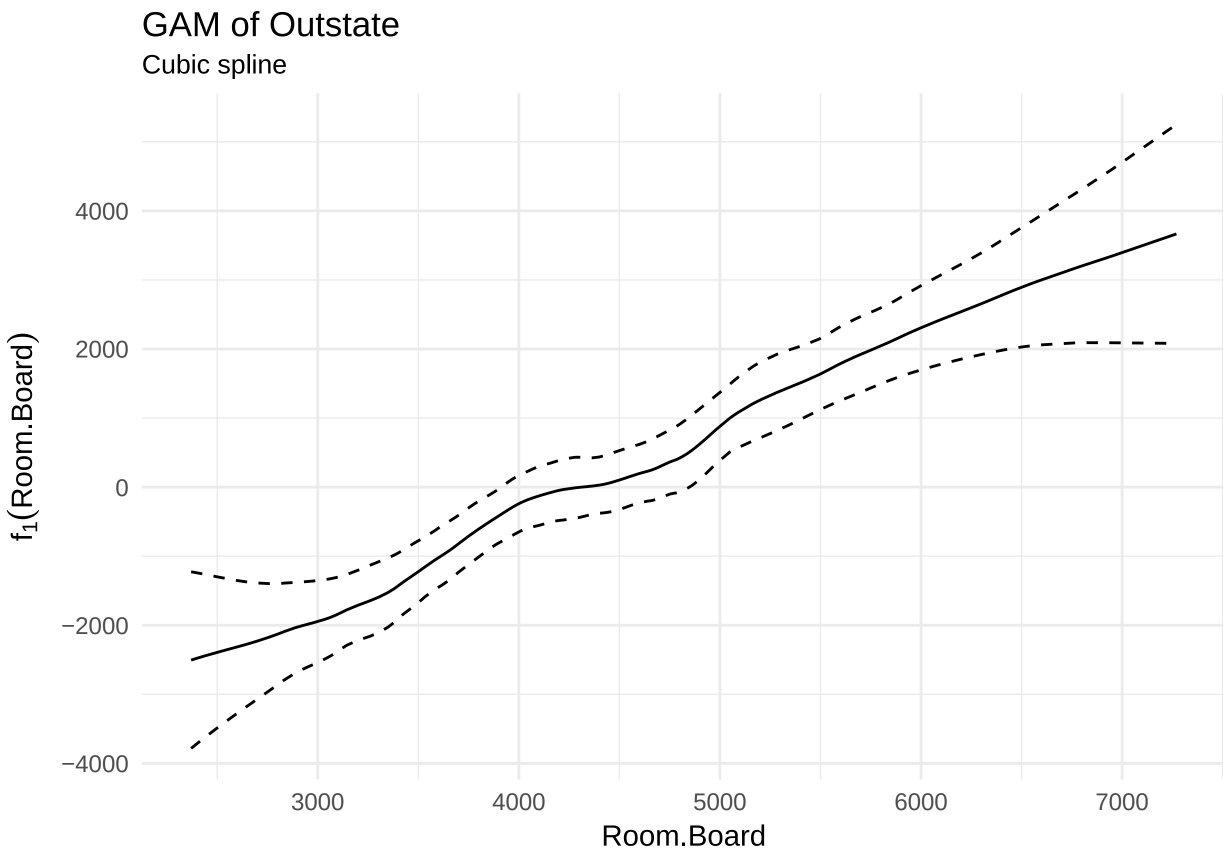
```
#top three statistically significant variables are lo(Apps), lo(Room.Board), lo(Top10perc)
# get graphs of each term
 c_gam_terms <- preplot(c_gam, se = TRUE, rug = FALSE)
```

```
## lo(Apps)
```

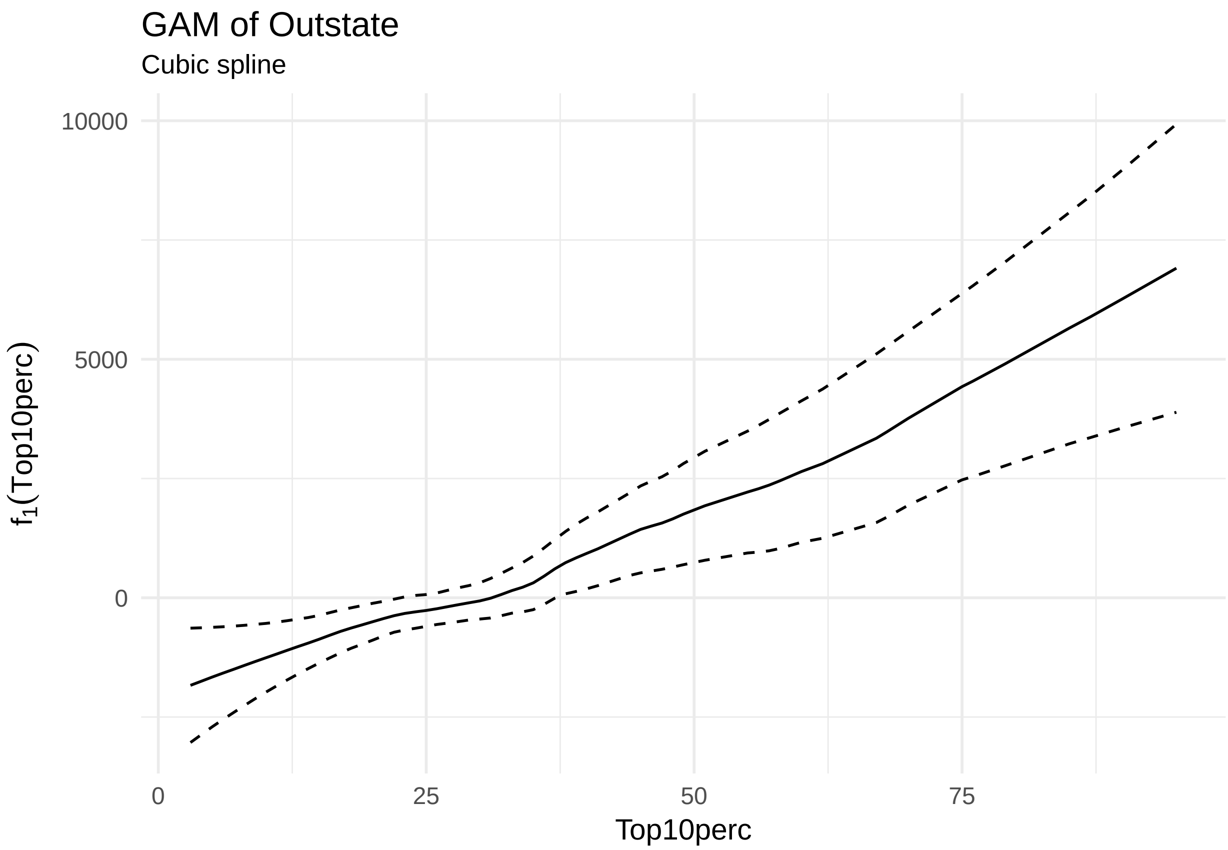
```
data_frame(x = c_gam_terms$`lo(Apps)`$x,
           y = c_gam_terms$`lo(Apps)`$y,
           se.fit =  c_gam_terms$`lo(Apps)`$se.y) %>%
  mutate(y_low = y - 1.96 * se.fit,
         y_high = y + 1.96 * se.fit) %>%
  ggplot(aes(x, y)) +
  geom_line() +
  geom_line(aes(y = y_low), linetype = 2) +
 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




```
## lo(Room.Board)
data_frame(x = c_gam_terms$`lo(Room.Board)`$x,
           y = c_gam_terms$`lo(Room.Board)`$y,
           se.fit =  c_gam_terms$`lo(Room.Board)`$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",
       subtitle = "Cubic spline",
       x = "Room.Board",
       y = expression(f[1](Room.Board)))
```




```
## lo(Top10perc)
data_frame(x = c_gam_terms$`lo(Top10perc)`$x,
           y = c_gam_terms$`lo(Top10perc)`$y,
           se.fit =  c_gam_terms$`lo(Top10perc)`$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",
       subtitle = "Cubic spline",
       x = 'Top10perc',
       y = expression(f[1](Top10perc)))
```



3.Estimate a GAM on the training data, using out-of-state tuition (Outstate) as the response variable and the other six variables as the predictors. You can select any non-linear method (or linear) presented in the readings or in-class to fit each variable. Plot the results, and explain your findings. Interpret the results and explain your findings, using appropriate techniques (tables, graphs, statistical tests, etc.).

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

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

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

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

4.Use the test set to evaluate the model fit of the estimated OLS and GAM models, and explain the results obtained.

5. For which variables, if any, is there evidence of a non-linear relationship with the response?