# Homework 6

Matthew Bradley, Ayanna Fisher, Ben Howell, Hayley Zorkic

4/11/2022

```
library(ISLR)
library(MASS)
library(tidyverse)
library(boot)
library(splines)
data(Boston)
```

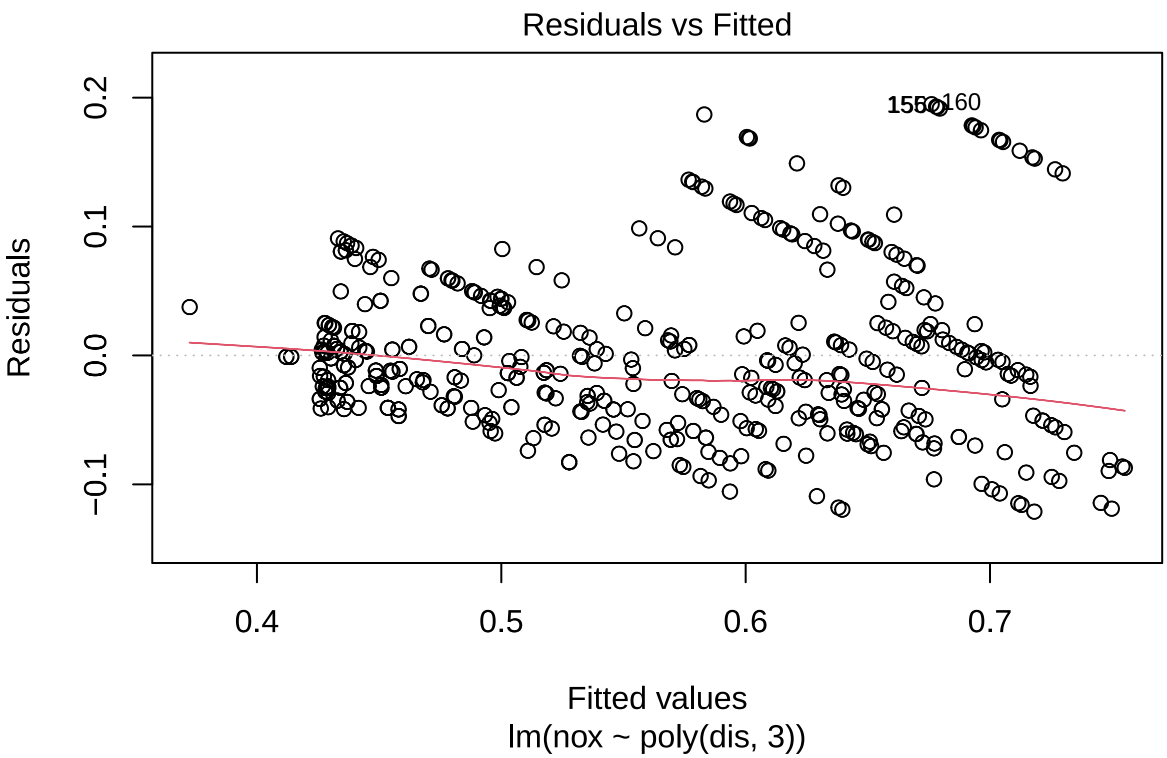
## Question 9:

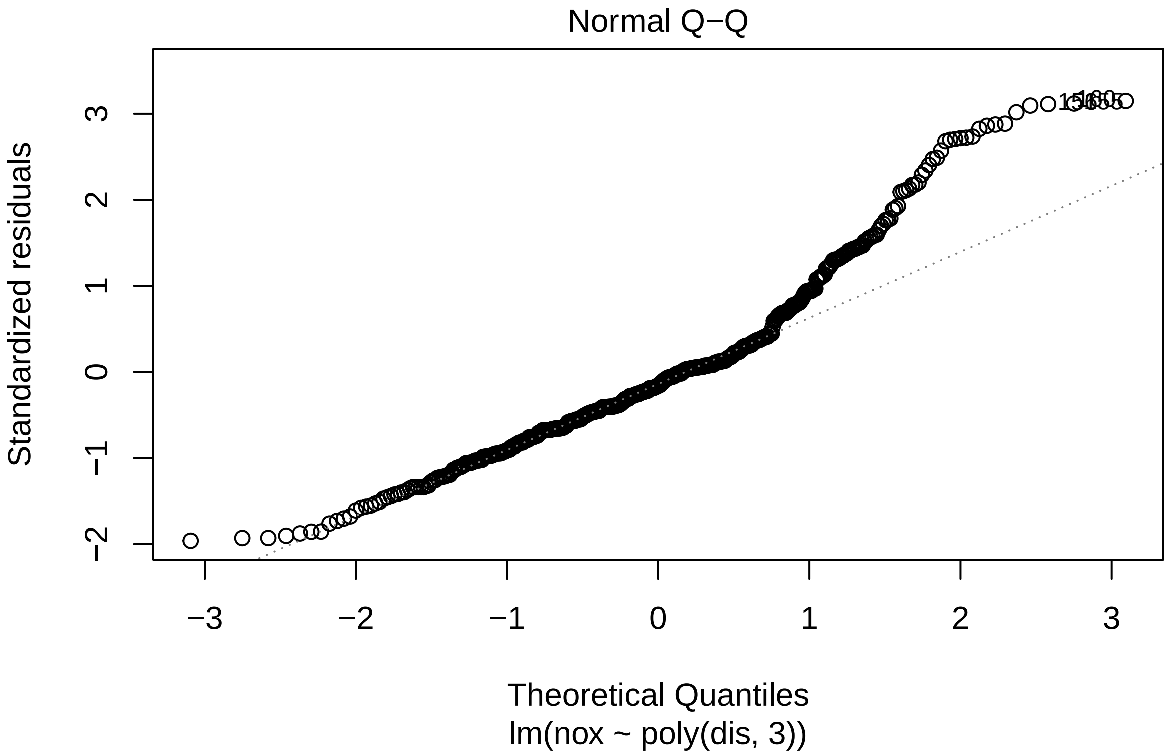
### a:

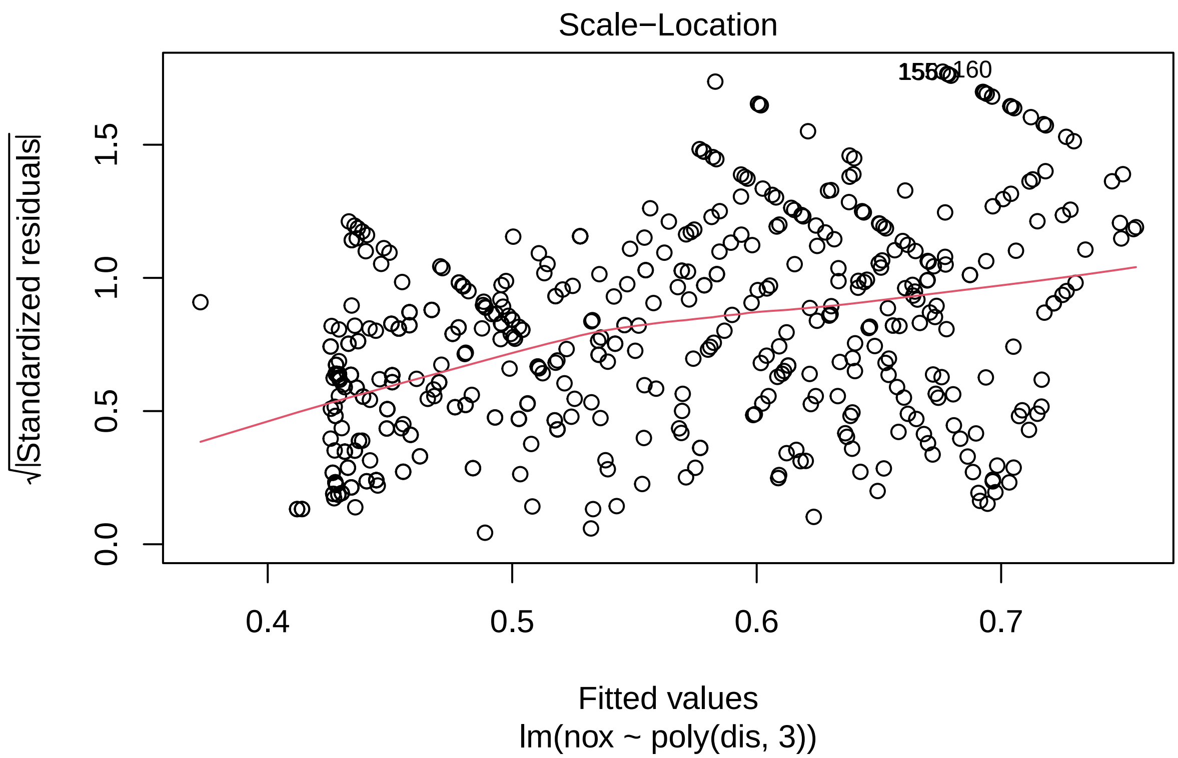
```
fit <- lm(nox ~ poly(dis , 3), data = Boston)
summary(fit)
```

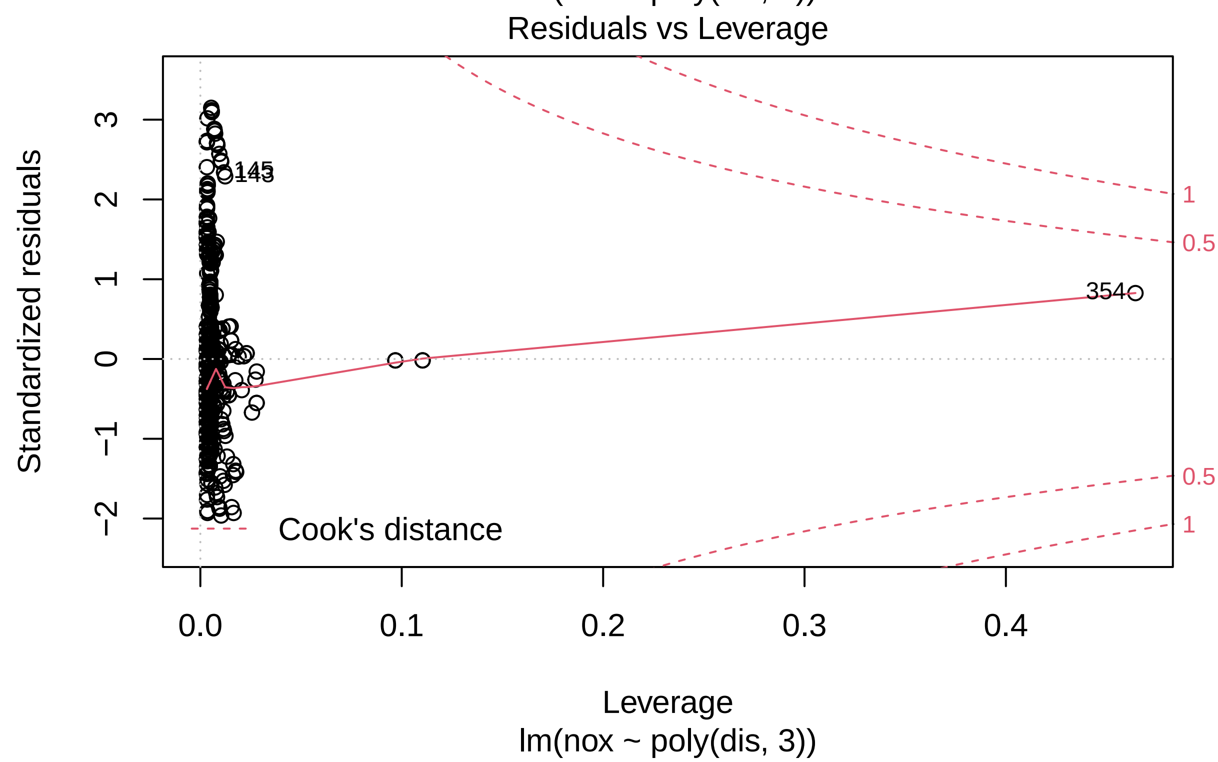
```
##
## Call:
## lm(formula = nox ~ poly(dis, 3), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.121130 -0.040619 -0.009738  0.023385  0.194904
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.554695   0.002759 201.021  < 2e-16 ***
## poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
## poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
## poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06207 on 502 degrees of freedom
## Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131
## F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
plot(fit)
```

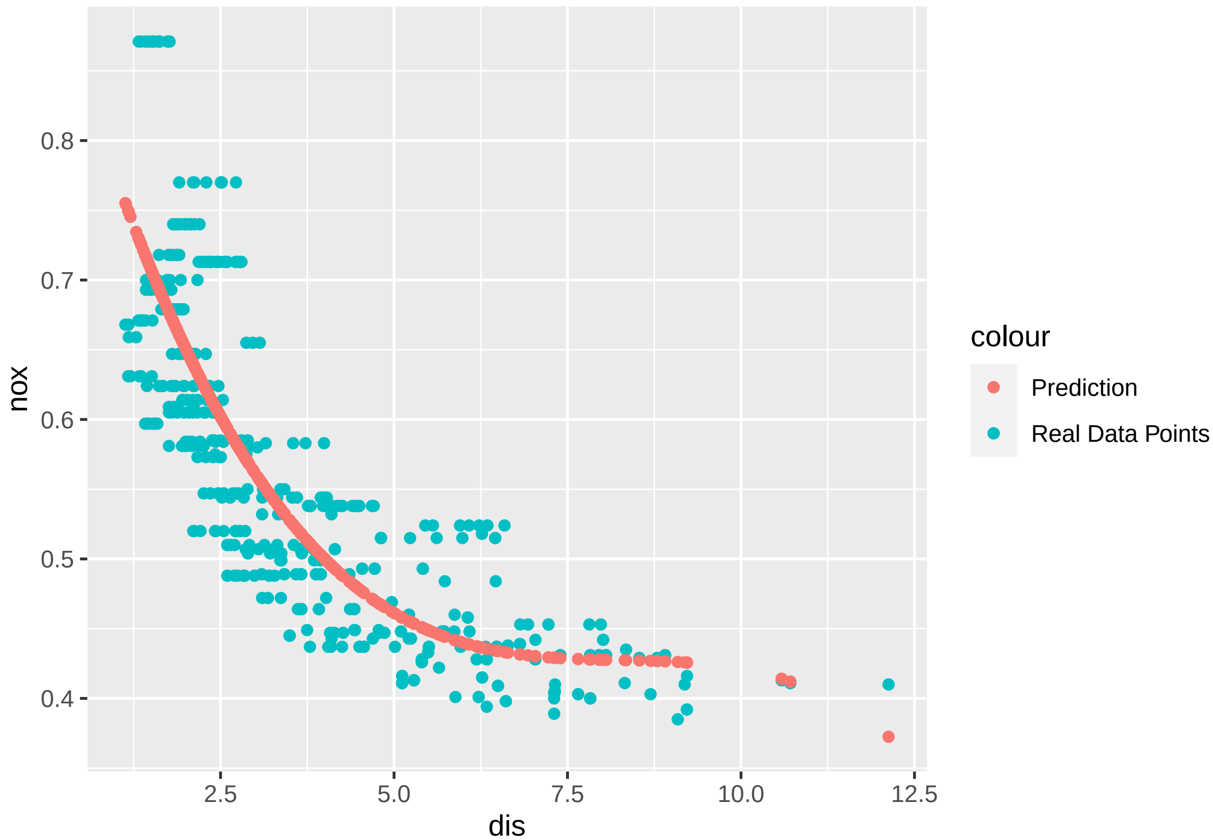

Residuals vs Fitted
Residuals
Fitted values
lm(nox ~ poly(dis, 3))
160


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(nox ~ poly(dis, 3))



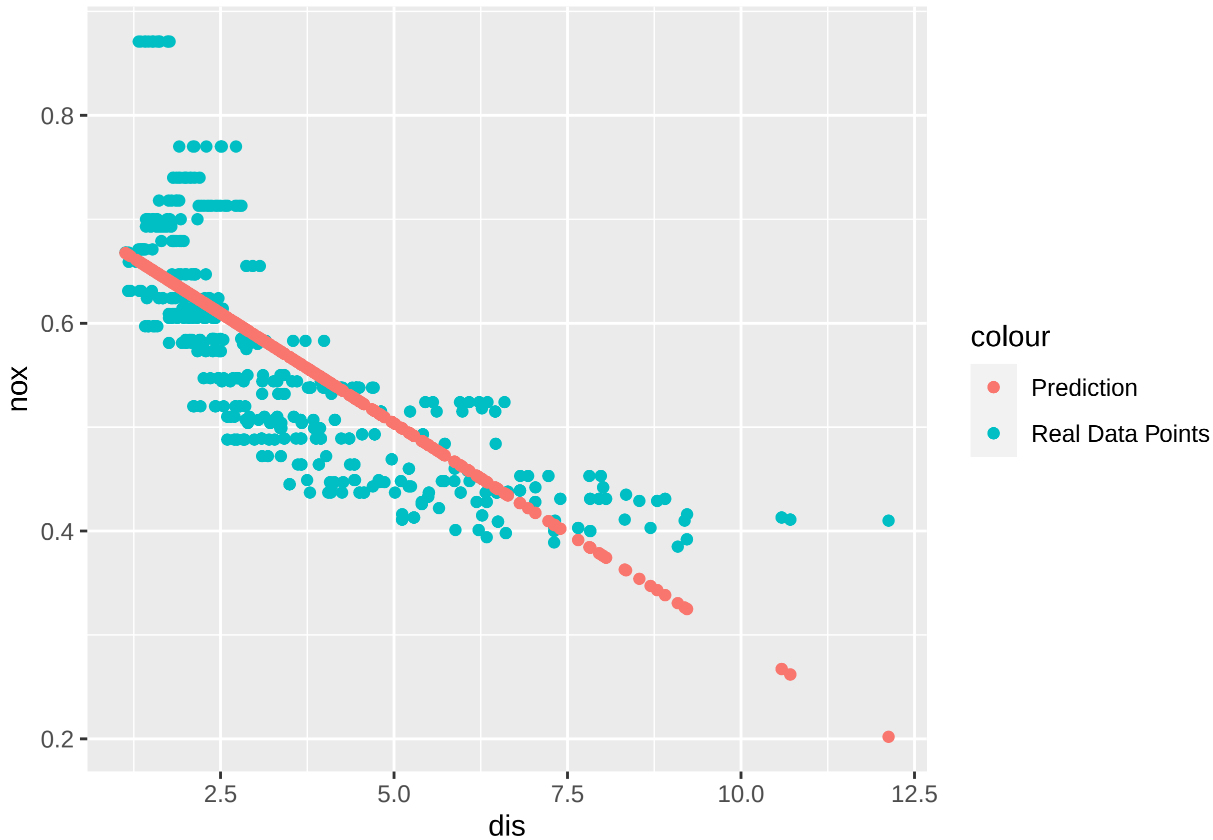



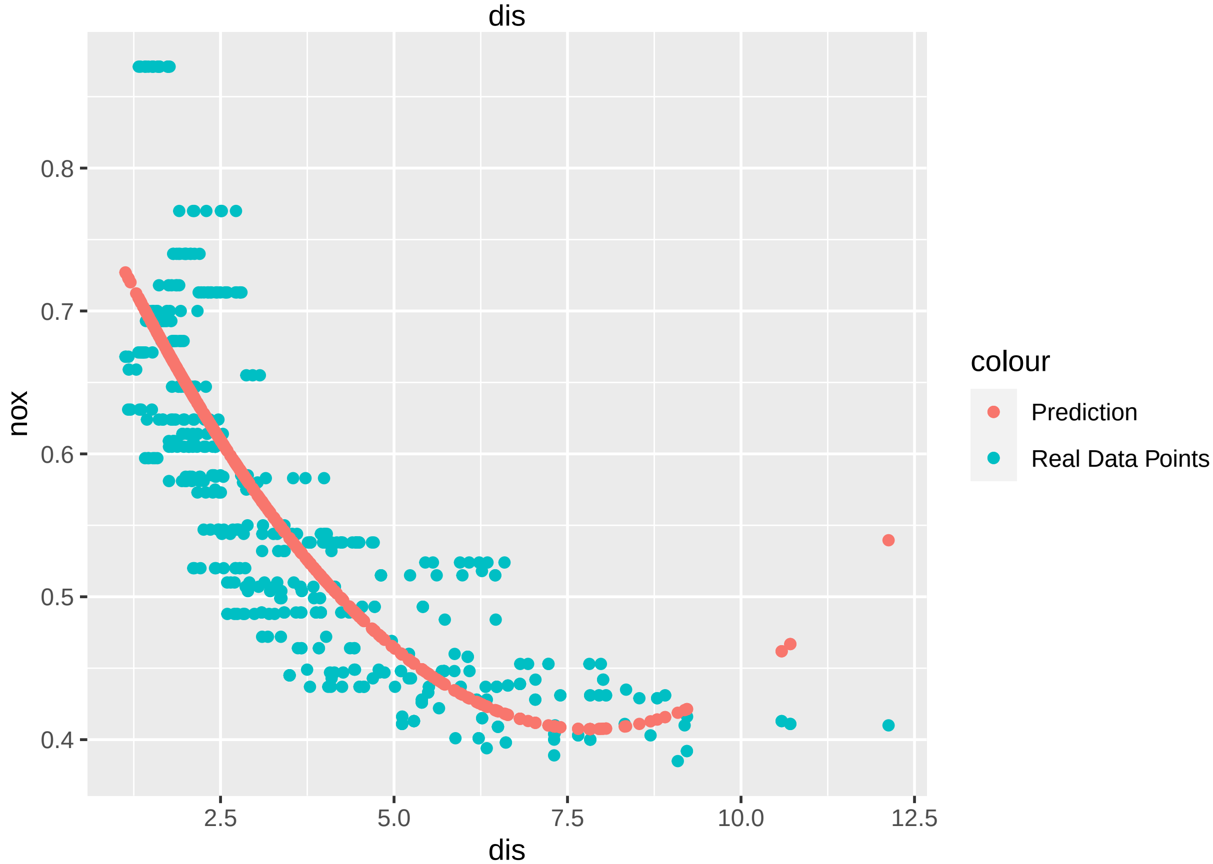

```r
pred = predict(fit, Boston)
ggplot(data = Boston, aes(y = nox, x = dis))+
  geom_point(aes(color = "Real Data Points"))+
  geom_point(aes(x = dis, y = pred, color = "Prediction"))+
  scale_fill_manual(name = "", values = c("Real Data Points" = "red"))+
  scale_fill_manual(name = "", values = c("Prediction" = "Blue"))
```

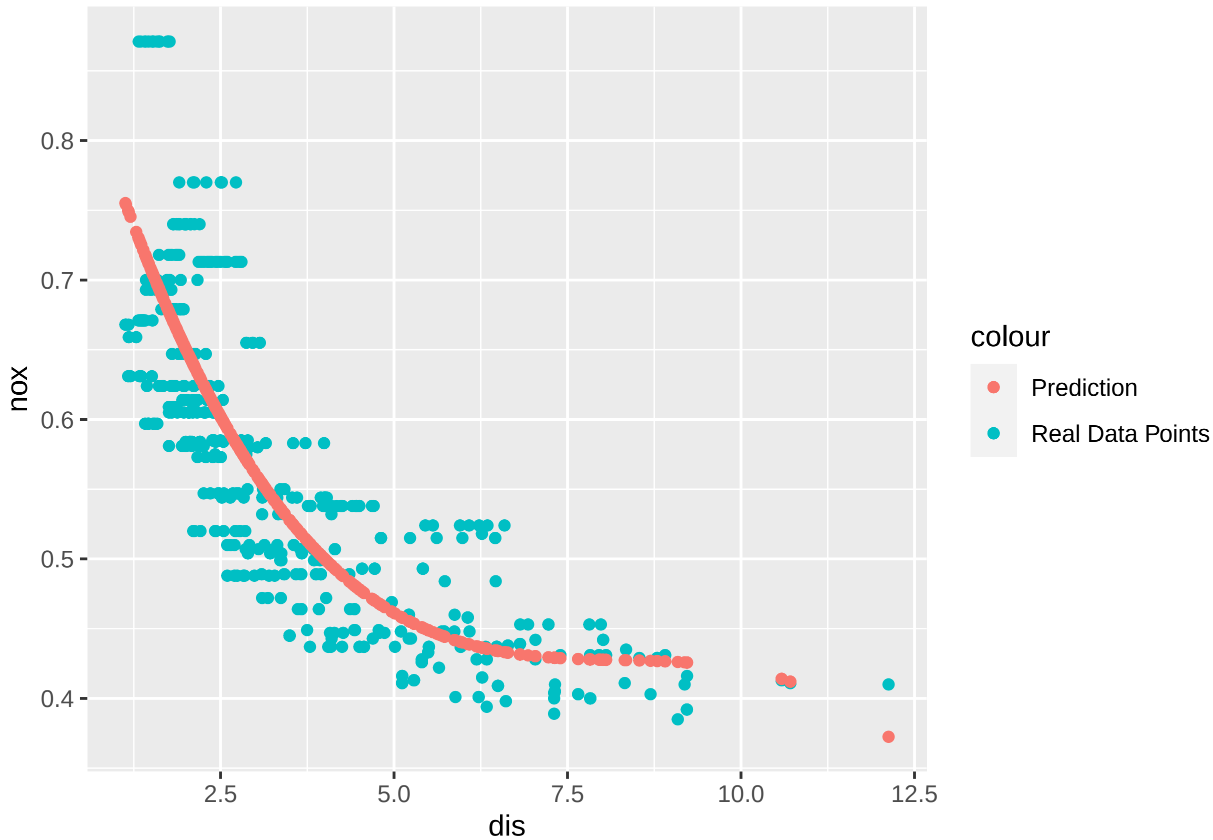


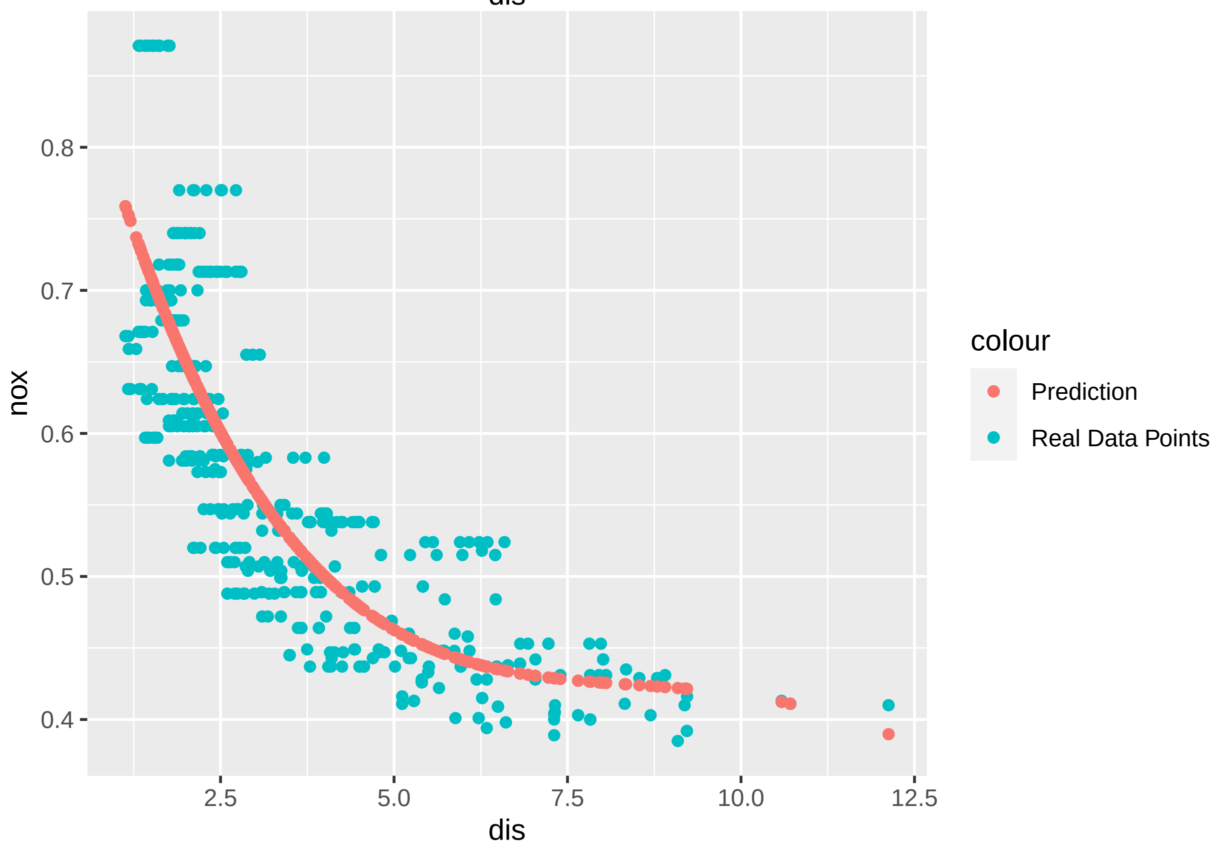

**b:**

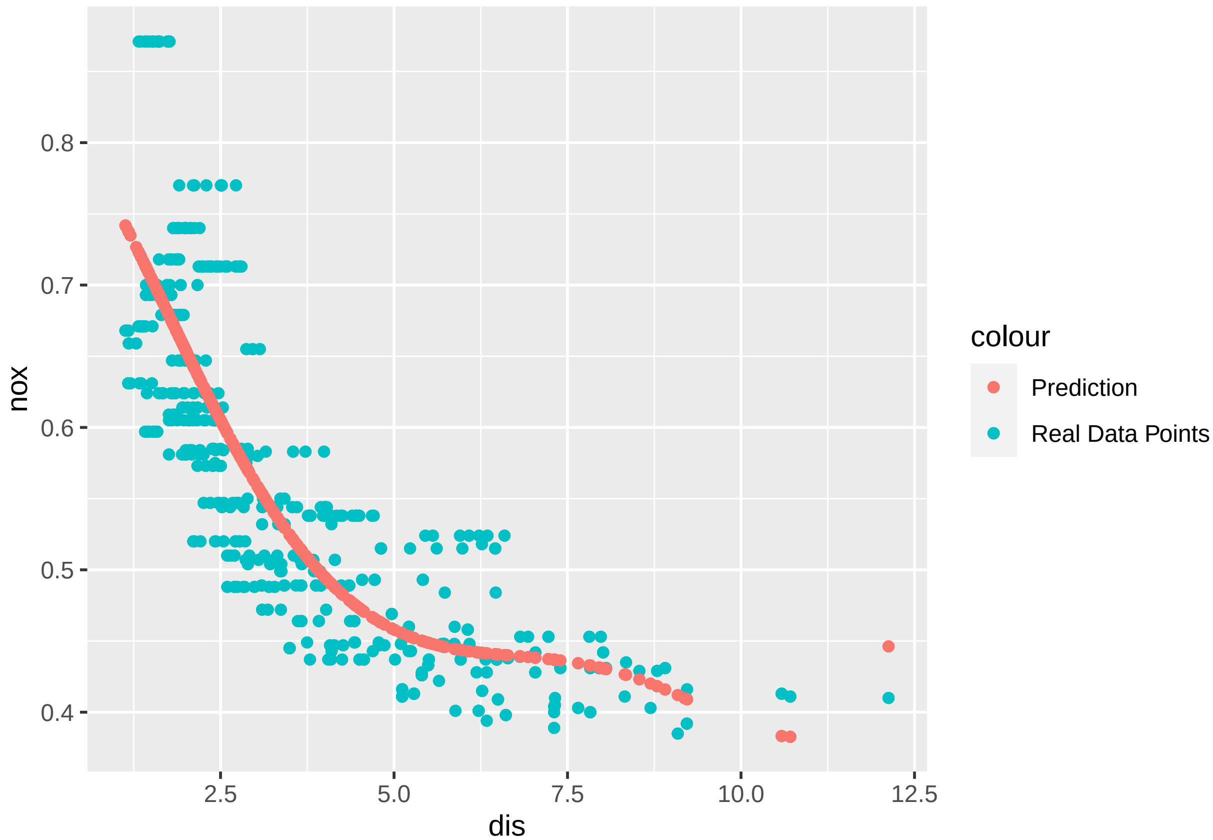
```
errors = rep(0,10)
preds = list()
for (i in 1:10){
  fit <- lm(nox ~ poly(dis , i), data = Boston)
  pred = predict(fit, Boston)
  preds[[length(preds) + 1]] = pred
  error = sum((pred - Boston$nox)^2)
  errors[i] = error

}


for (i in 1:10){
  currentPlot <- ggplot(data = Boston, aes(y = nox, x = dis))+
  geom_point(aes(col = "Real Data Points"))+
  geom_point(aes(x = dis, y = as.numeric(preds[[i]]), col = "Prediction"))+
  scale_fill_manual(name = "", values = c("Real Data Points" = "red"))+
  scale_fill_manual(name = "", values = c("Prediction" = "Blue"))
  print(currentPlot)
}
```

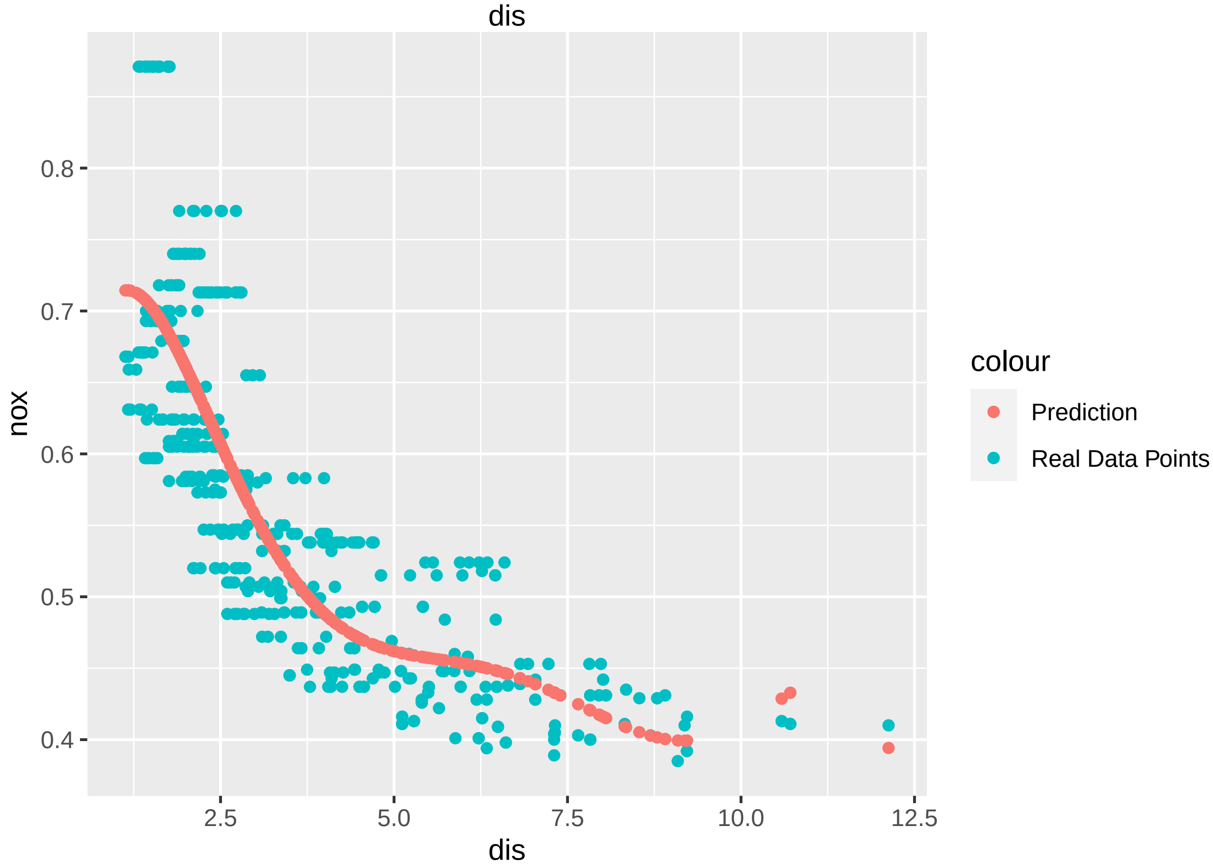

0.8
0.6
0.4
0.2
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points

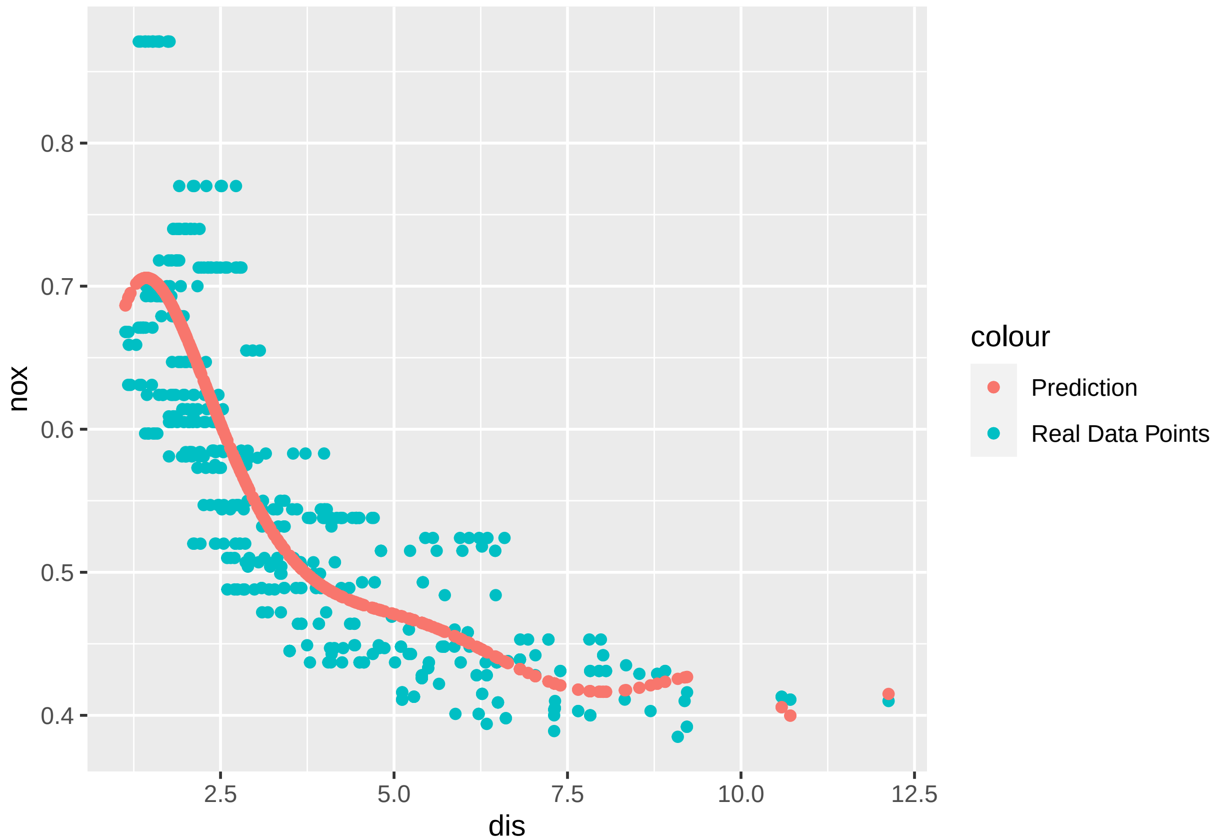

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points

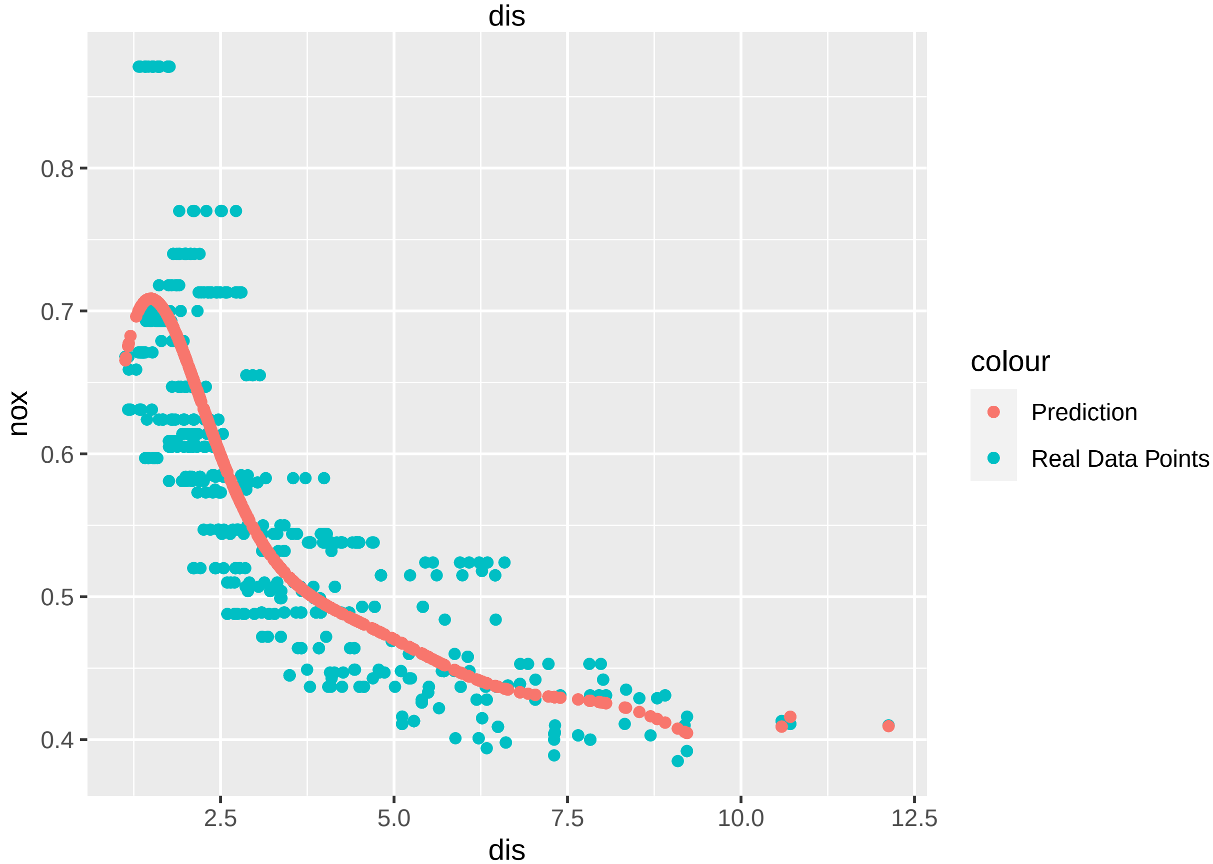

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points

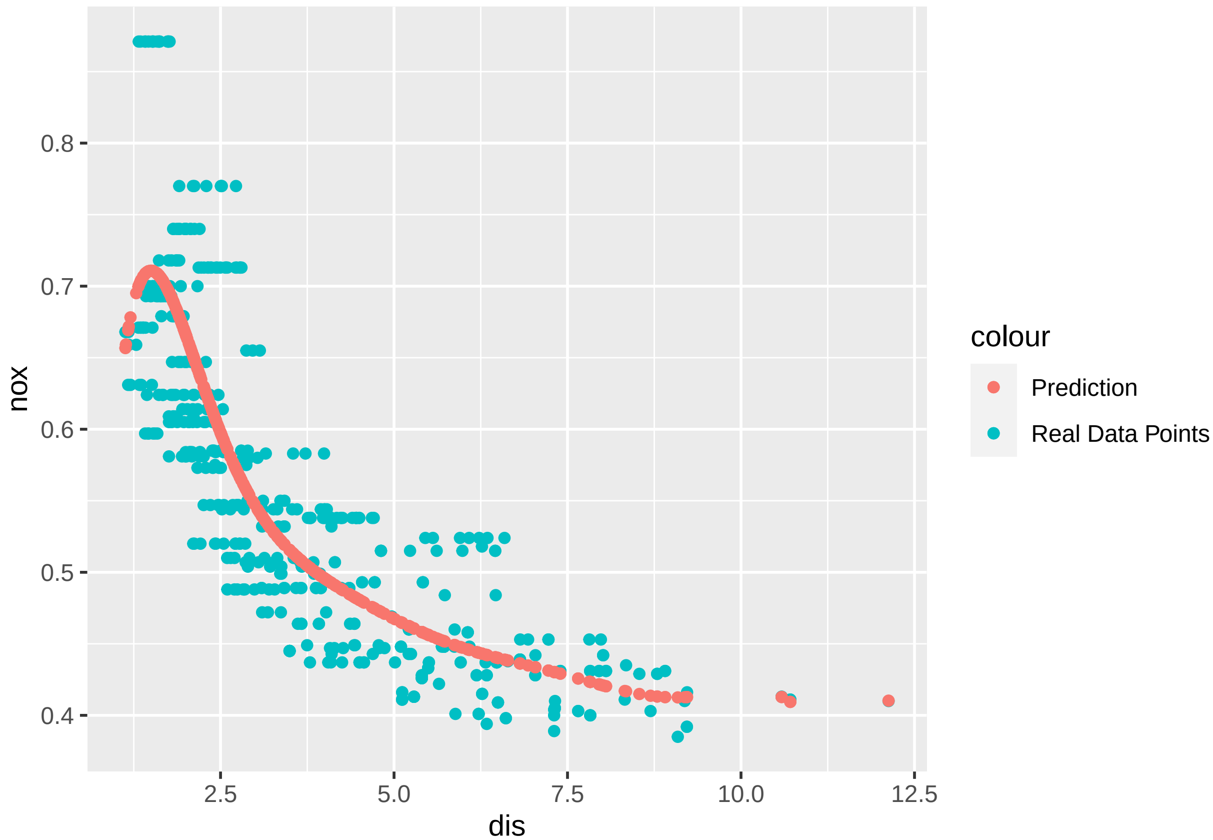

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points

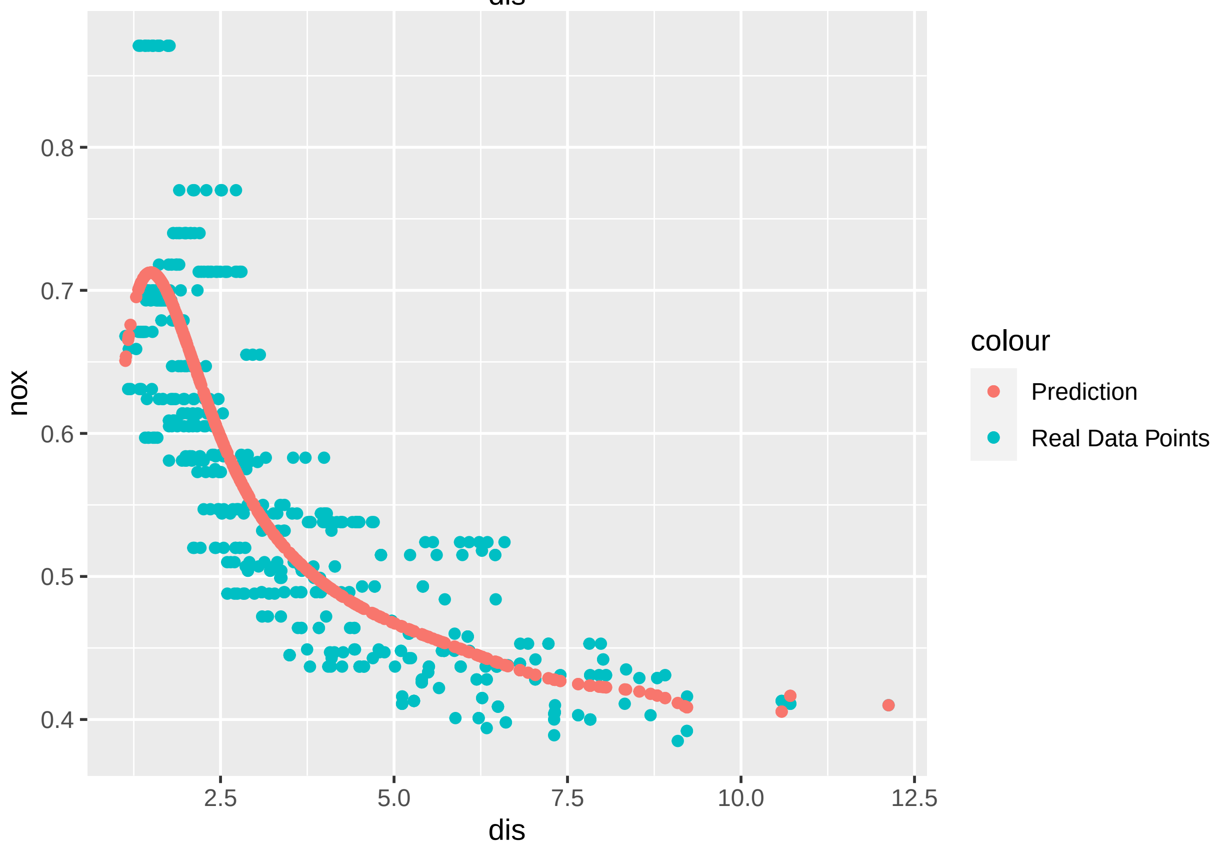

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points


0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points


0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points


0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points

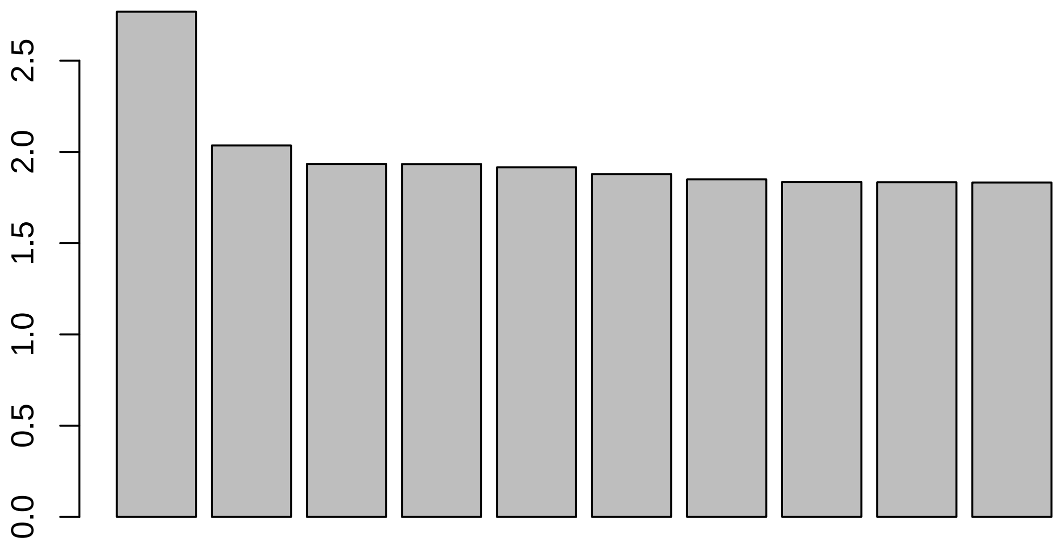

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points


0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
Prediction
Real Data Points

```r
printErrors = c("Sum of squares = ", errors)
print(printErrors)
```

```
##  [1] "Sum of squares = " "2.76856285896928"  "2.03526186893526"
##  [4] "1.93410670717907"  "1.93298132729859"  "1.9152899610843"
##  [7] "1.87825729850816"  "1.84948361458298"  "1.83562968906769"
## [10] "1.83333080449159"  "1.83217112393134"
```

```r
barplot(errors)
```

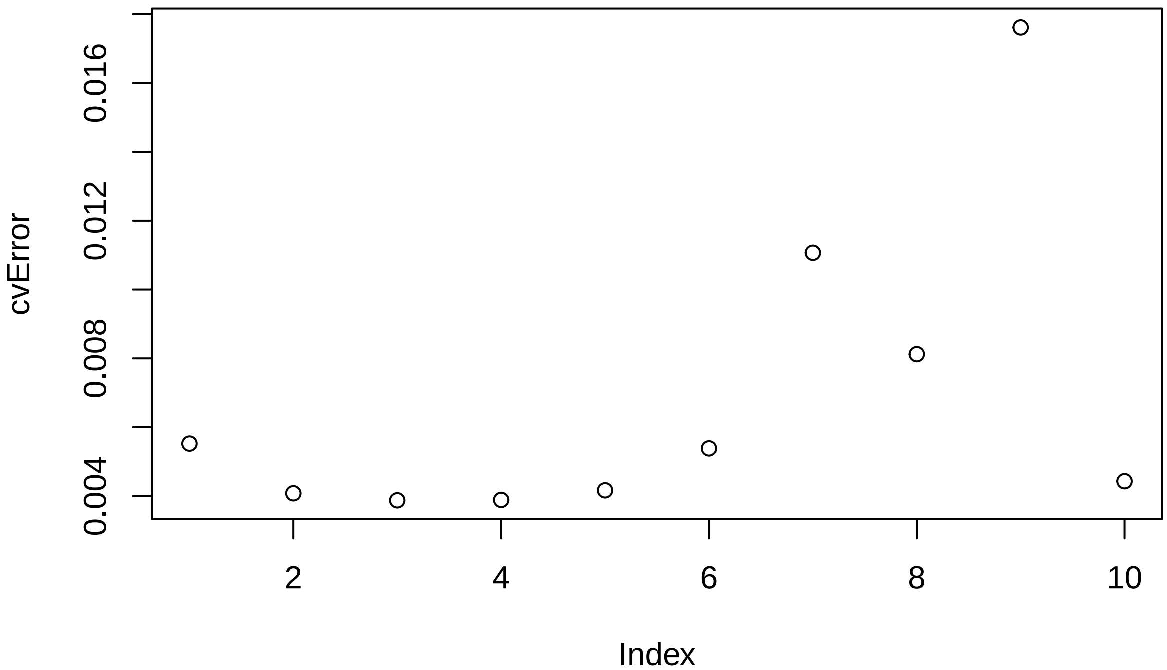



**c:**

```r
# K fold cross validation

cvError <- rep(0,10)
for (i in 1:10){
  glmFit <- glm(nox~poly(dis,i), data = Boston)
  cvError[i] = cv.glm(Boston, glmFit)$delta[1]
}

cvError
```

```
##  [1] 0.005523868 0.004079449 0.003874762 0.003887521 0.004164865 0.005384278
##  [7] 0.011068782 0.008121397 0.017616356 0.004430276
```

```r
plot(cvError)
```




My results from cross validation show that the error for a polynomial regression is lowest when the degree is 4, and highest when it is 7. Thus, I should use 4 as the polynomial for the model.

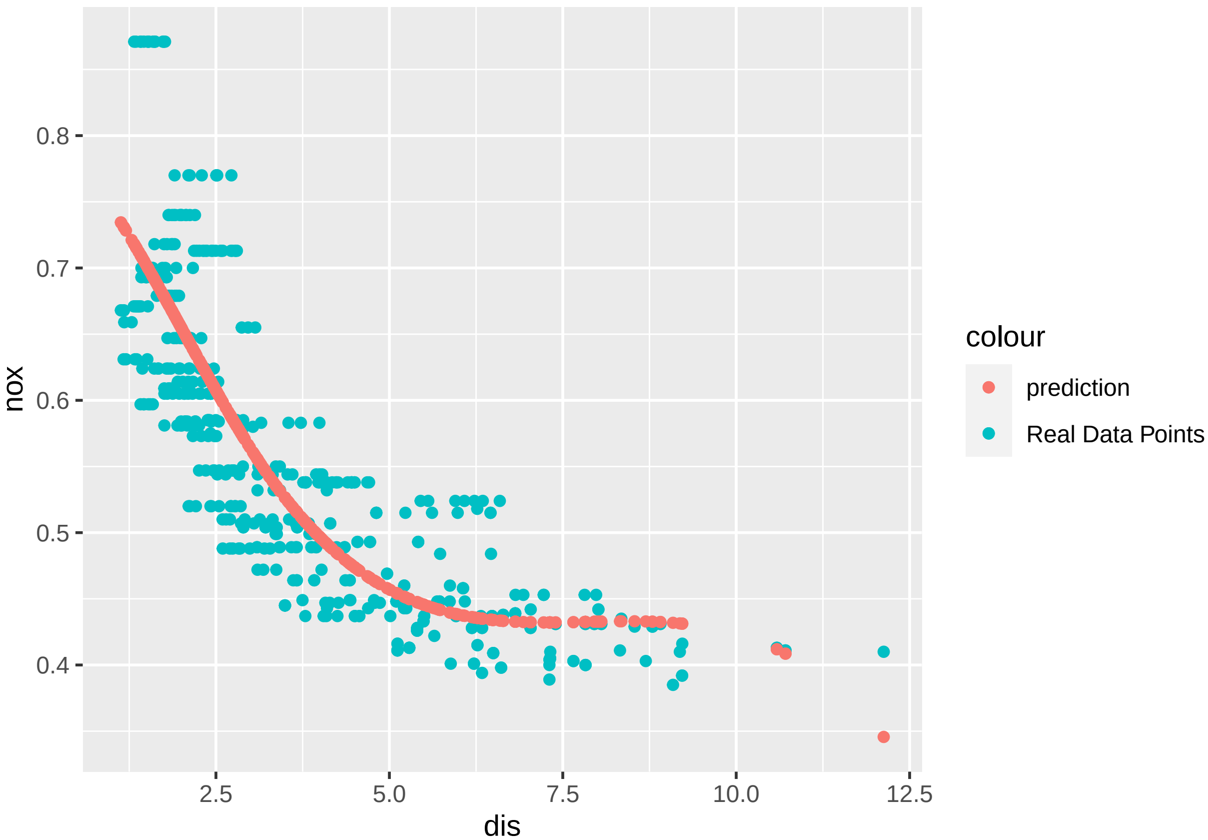
### d:

```r
fit <- lm(nox ~ bs(dis , df = 4), data = Boston)
summary(fit)
```

```
##
## Call:
## lm(formula = nox ~ bs(dis, df = 4), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.124622 -0.039259 -0.008514  0.020850  0.193891
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.73447    0.01460  50.306  < 2e-16 ***
## bs(dis, df = 4)1 -0.05810    0.02186  -2.658  0.00812 **
## bs(dis, df = 4)2 -0.46356    0.02366 -19.596  < 2e-16 ***
## bs(dis, df = 4)3 -0.19979    0.04311  -4.634 4.58e-06 ***
## bs(dis, df = 4)4 -0.38881    0.04551  -8.544  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06195 on 501 degrees of freedom
## Multiple R-squared:  0.7164, Adjusted R-squared:  0.7142
## F-statistic: 316.5 on 4 and 501 DF,  p-value: < 2.2e-16
```

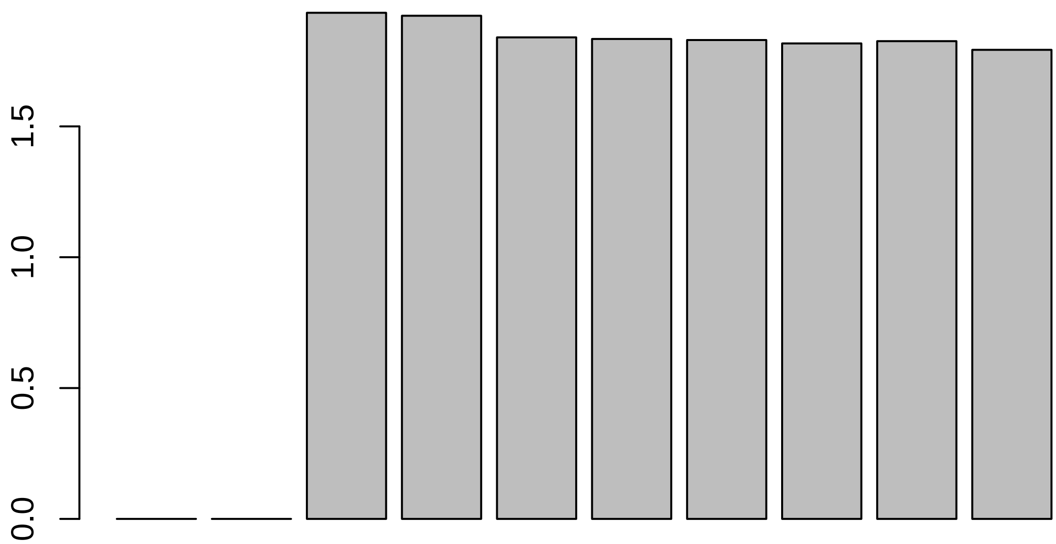
I chose the knots to be at uniform quantiles of the data (using the df option) because the book said this is a common practice when working with splines.

```r
pred = predict(fit, Boston)

ggplot(data = Boston, aes(y = nox, x = dis))+
  geom_point(aes(col = "Real Data Points"))+
  geom_point(aes(x = dis, y = pred, col = "prediction"))+
  scale_fill_manual(name = "", values = c("Real Data Points" = "red"))+
  scale_fill_manual(name = "", values = c("Prediction" = "Blue"))
```



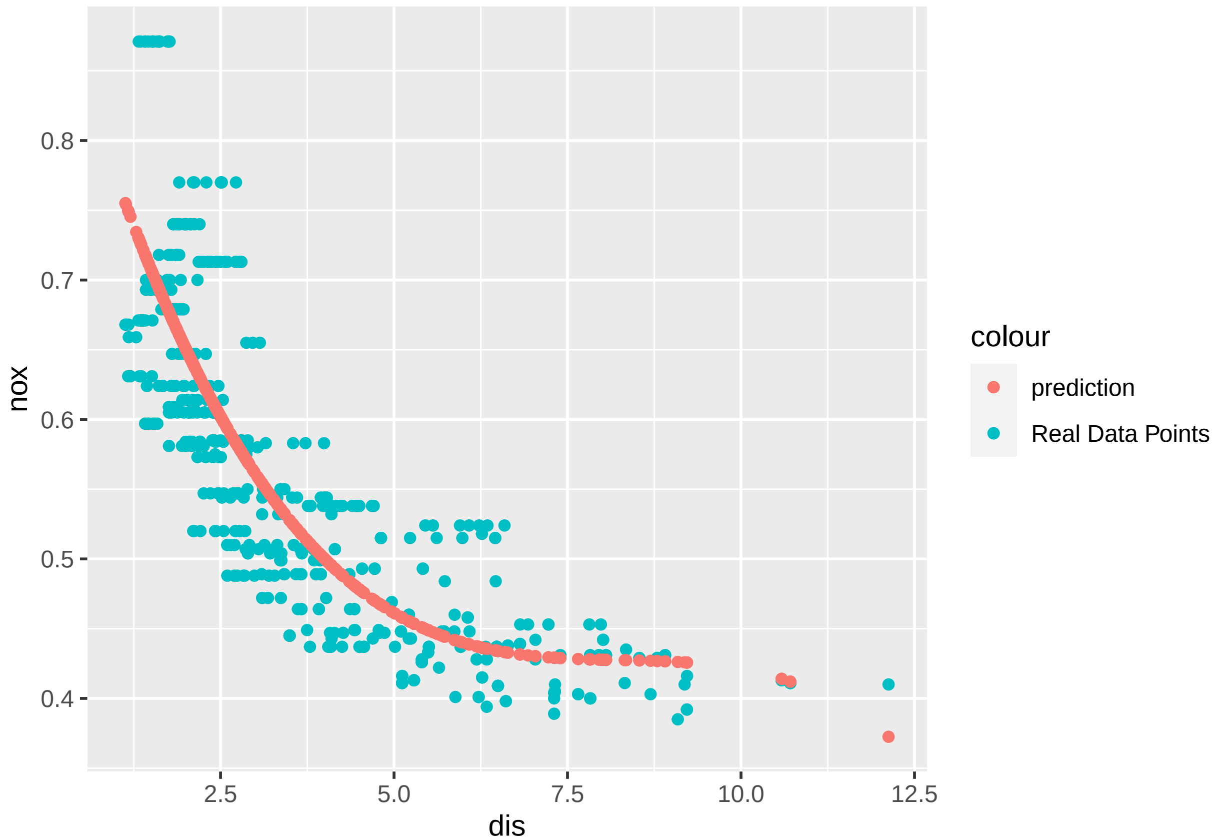

**e:**

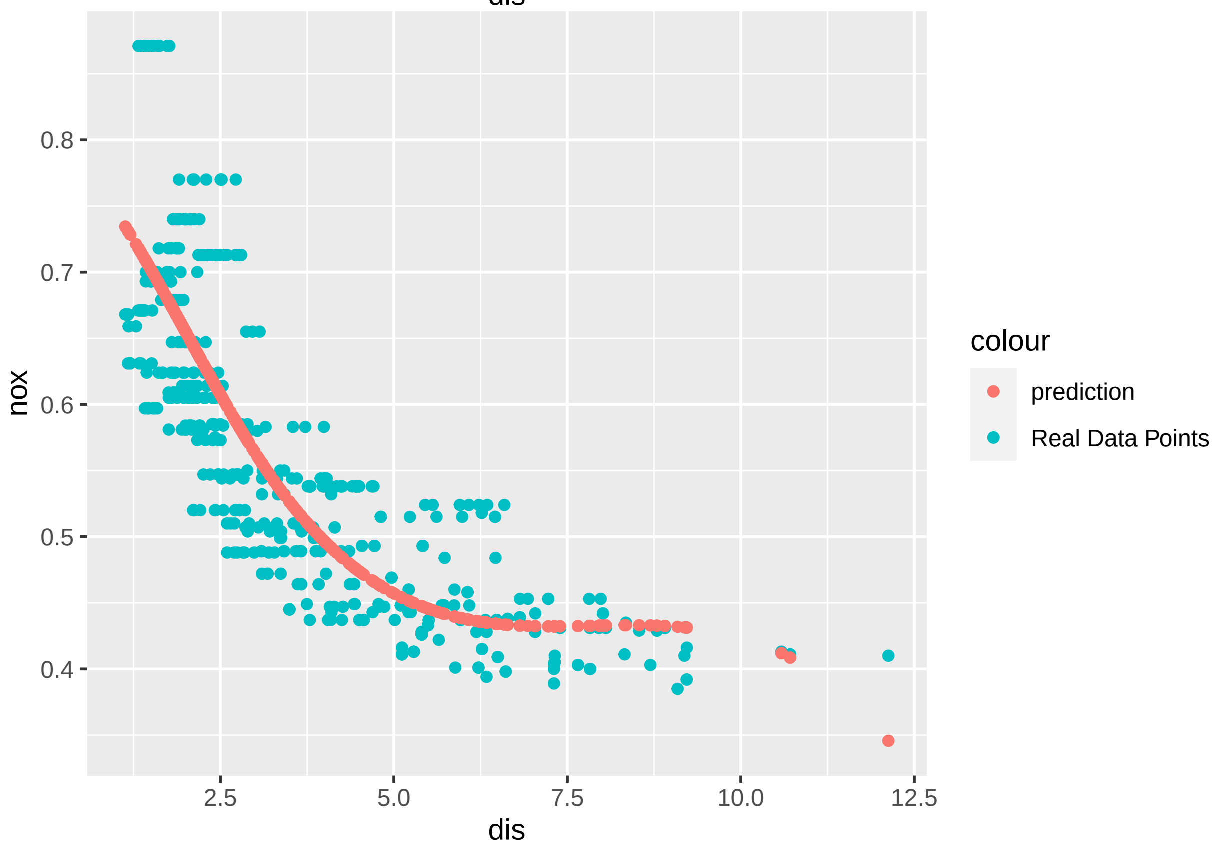
```r
preds = list()
rssList = c(rep(0,8))

for (i in 3:10){
  fit <- lm(nox ~ bs(dis , df = i), data = Boston)
  pred = predict(fit, Boston)
  preds[[length(preds)+1]] = pred
  rss = sum((pred - Boston$nox)^2)
  rssList[i] = rss
}

barplot(as.numeric(rssList))
```

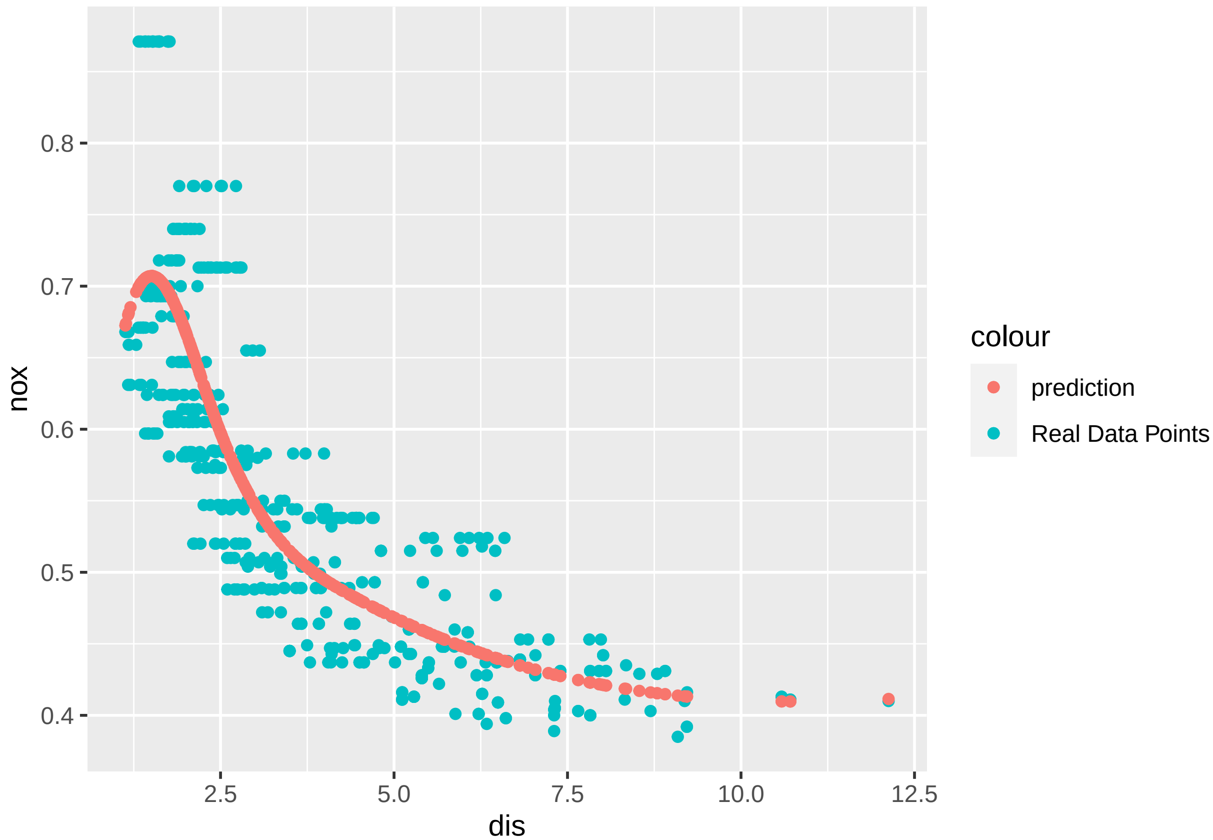



We see that as the degree of our polynomial increases, the RSS only decreases slightly, indicating that we should veer on the side of a simpler model with a lower polynomial, because these models still capture most of the variance in the data that higher degree models capture. Note that I started the polynomial at 3 because this is the minimum degree of freedom for the model I could use (which is why the barplot has values of 0 for two bars).

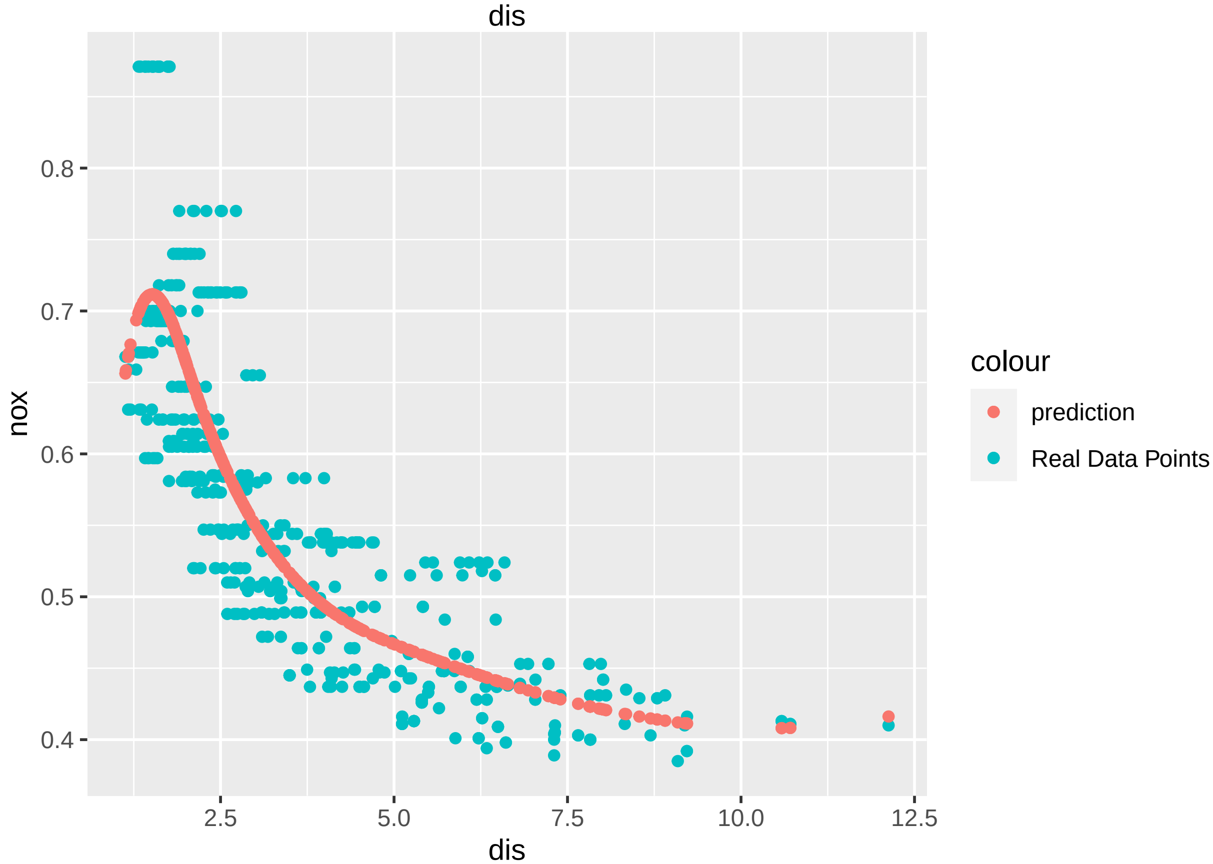
```
for (i in 1:length(preds)){
  currentPlot <- ggplot(data = Boston, aes(y = nox, x = dis))+
  geom_point(aes(col = "Real Data Points"))+
  geom_point(aes(x = dis, y = as.numeric(preds[[i]]), col = "prediction"))+
  scale_fill_manual(name = "", values = c("Real Data Points" = "red"))+
  scale_fill_manual(name = "", values = c("Prediction" = "Blue"))
  plot(currentPlot)
}
```


0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points

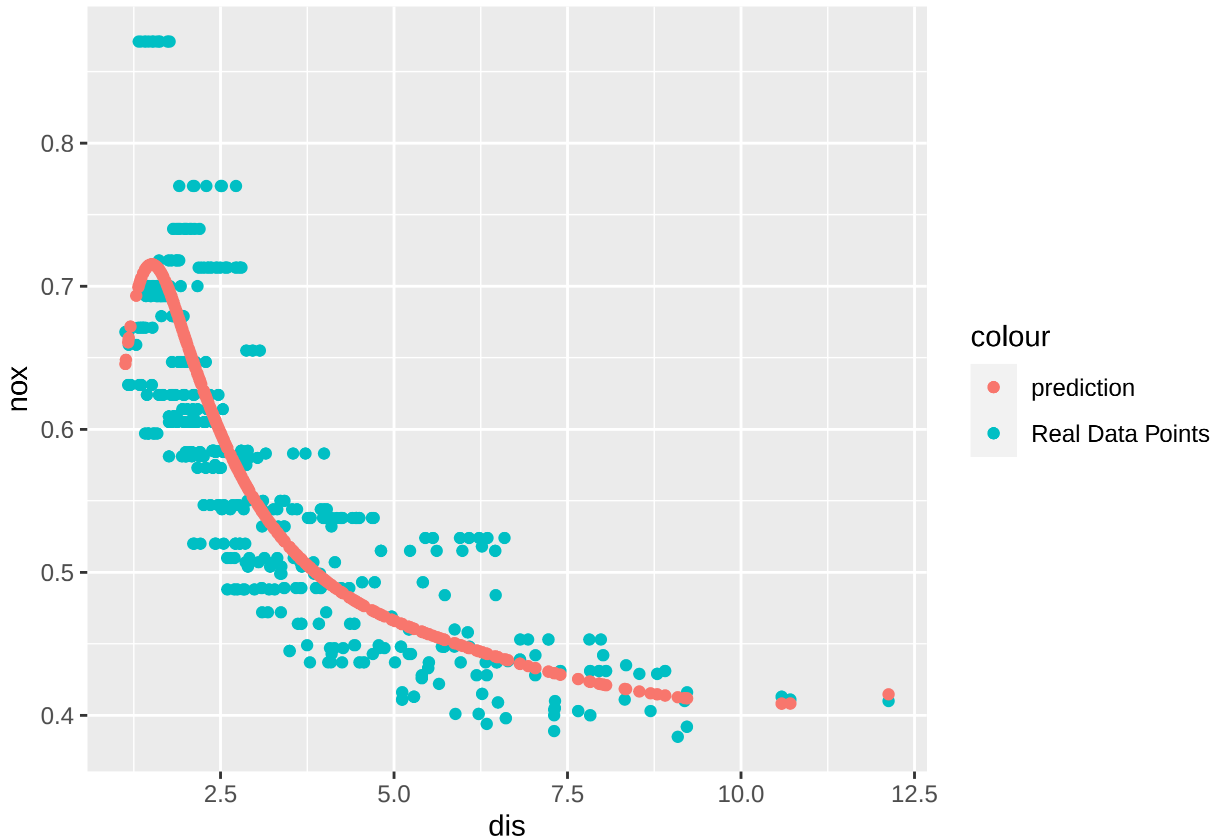

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points



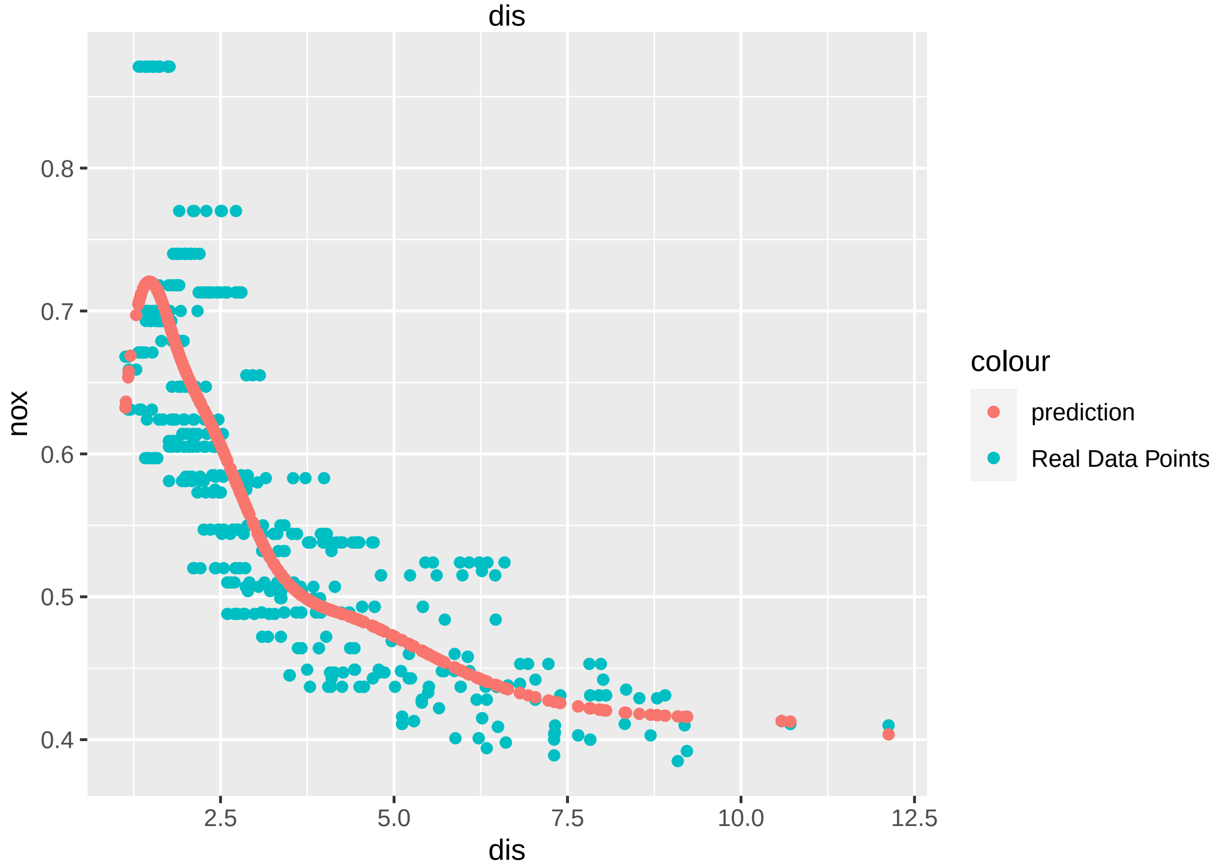
0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points

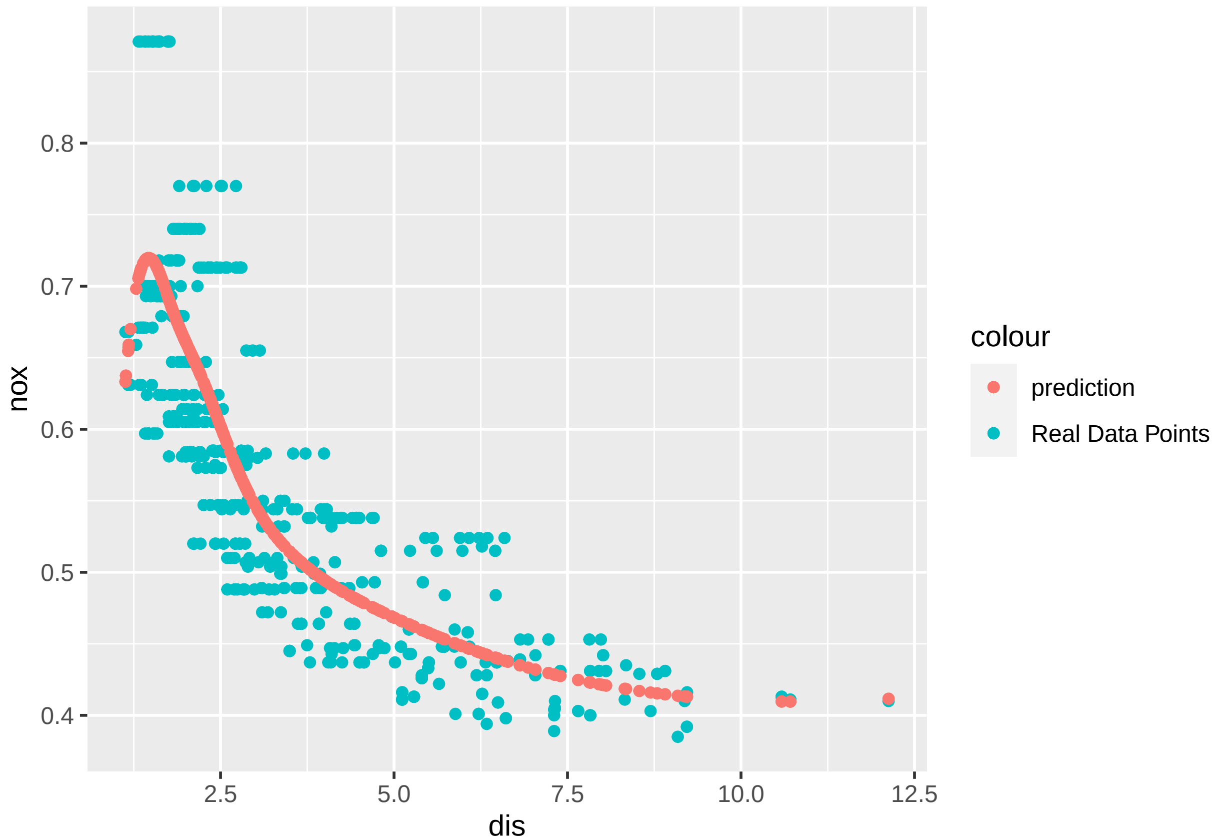
0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points

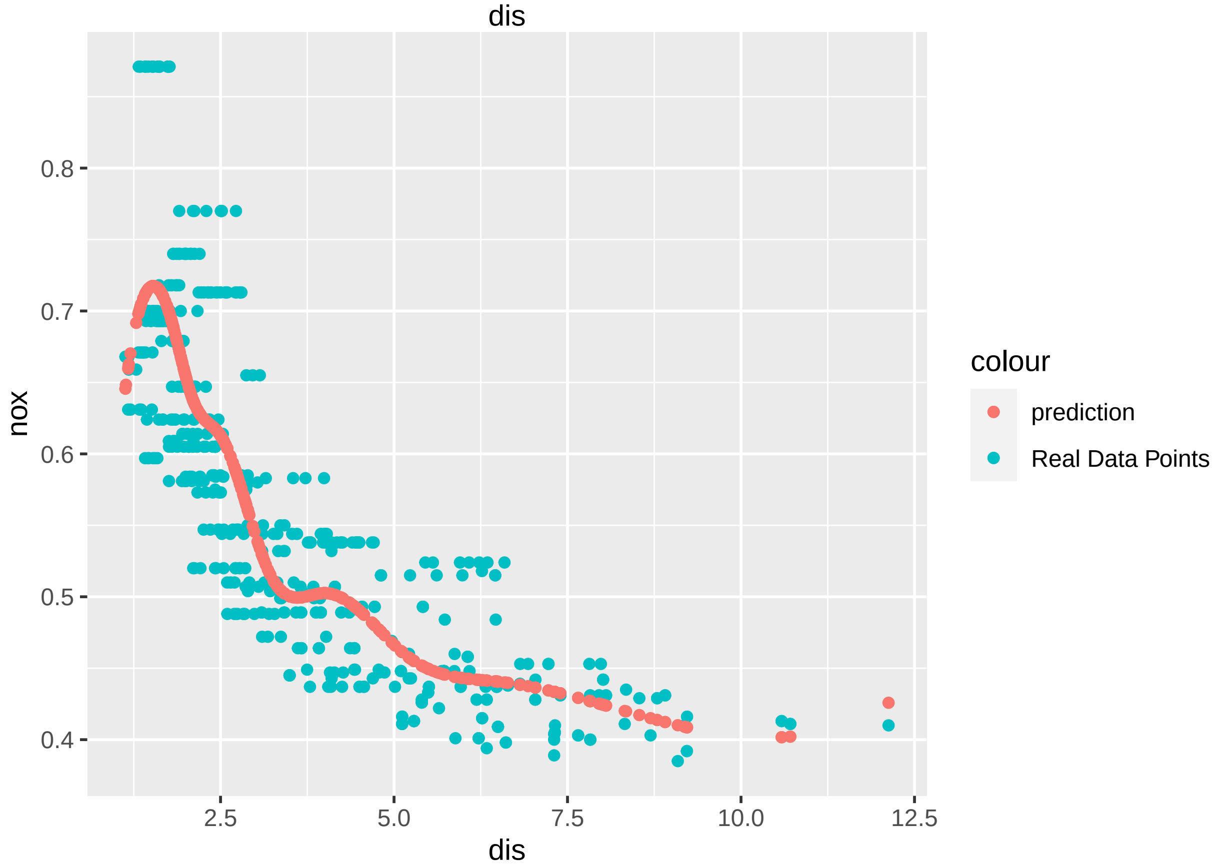


0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points


0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points

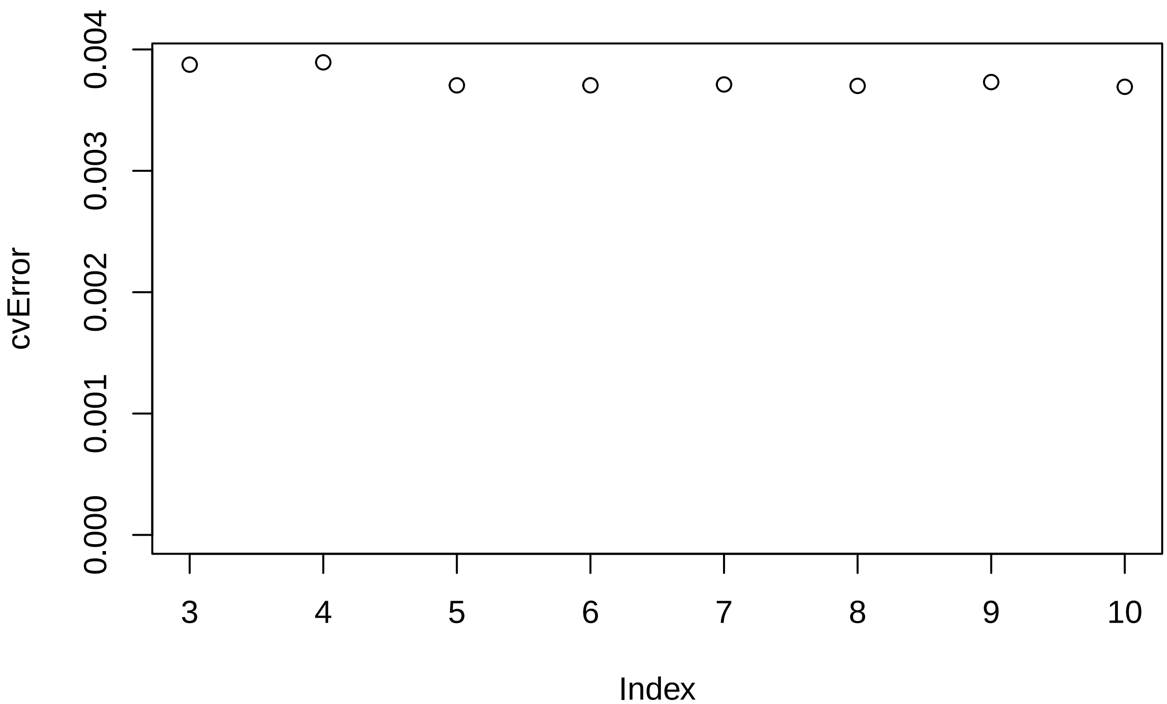

0.8
0.7
0.6
0.5
0.4
nox
2.5
5.0
7.5
10.0
12.5
dis
colour
prediction
Real Data Points

### f:

```
cvError <- rep(0,7)
for (i in 3:10){
  fit <- glm(nox ~ bs(dis , df = i), data = Boston)
  cvError[i] = cv.glm(Boston, fit)$delta[1]
}

cvError
```

```
##  [1] 0.000000000 0.000000000 0.003874762 0.003893623 0.003704252 0.003704711
##  [7] 0.003711441 0.003699853 0.003731180 0.003692067
```

```
plot(cvError, xlim = c(3,10))
```



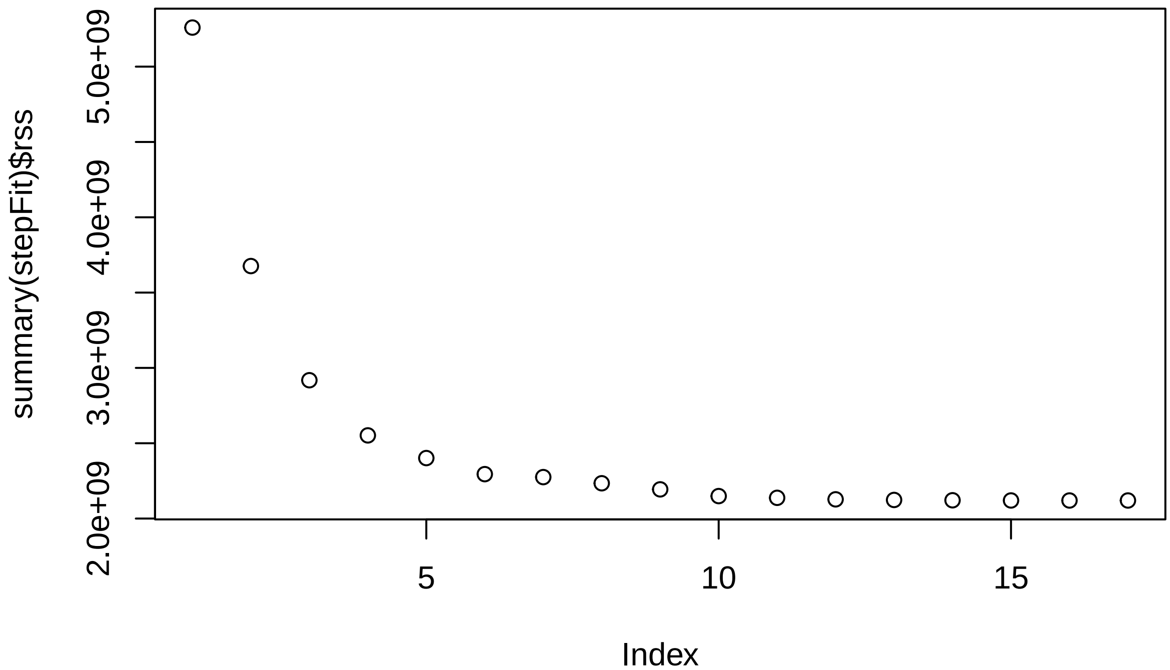

The best degree of freedom appears to be 5 according to cv error, but it is very close to the error of lower polynomial values, so it may be better to keep the model simpler with a polynomial of 3, rather than complicate it only to reduce the error a marginal amount.

## Question 10:

### a:

```
library(leaps)
library(gam)
data(College)

sample_size <- floor(0.75 * nrow(College))
train_index <- sample(seq_len(nrow(College)), size = sample_size)
College_train <- College[train_index,]
College_test <- College[-train_index,]
```

```r
stepFit <- regsubsets(Outstate ~ ., data = College_train, method = "forward", nvmax = 17)
plot(summary(stepFit)$rss)
```



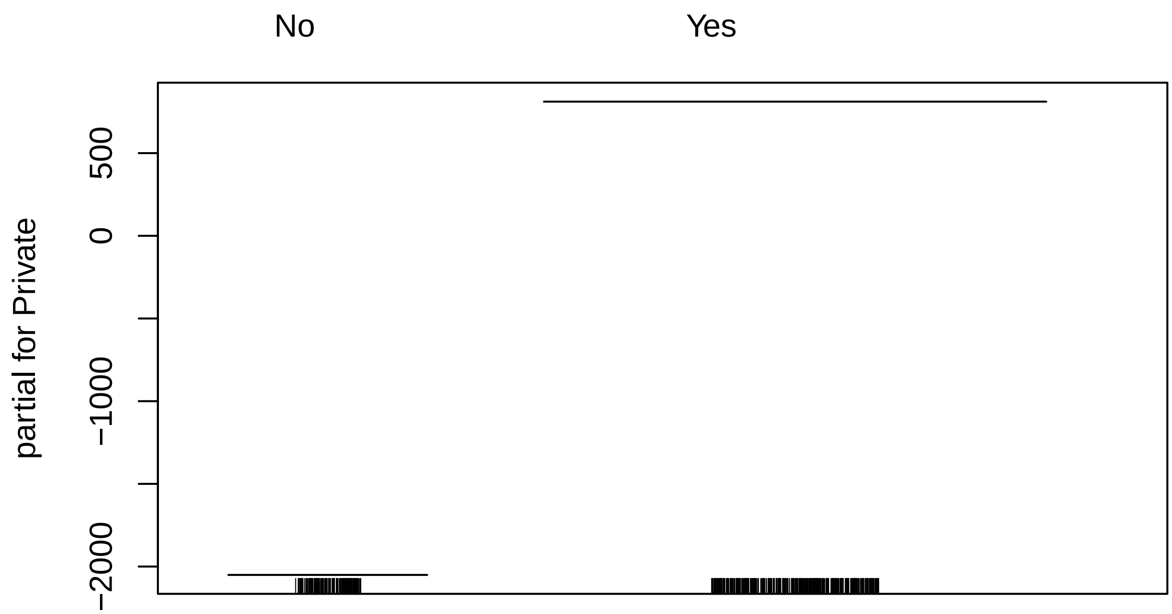

```r
summary(stepFit)
```

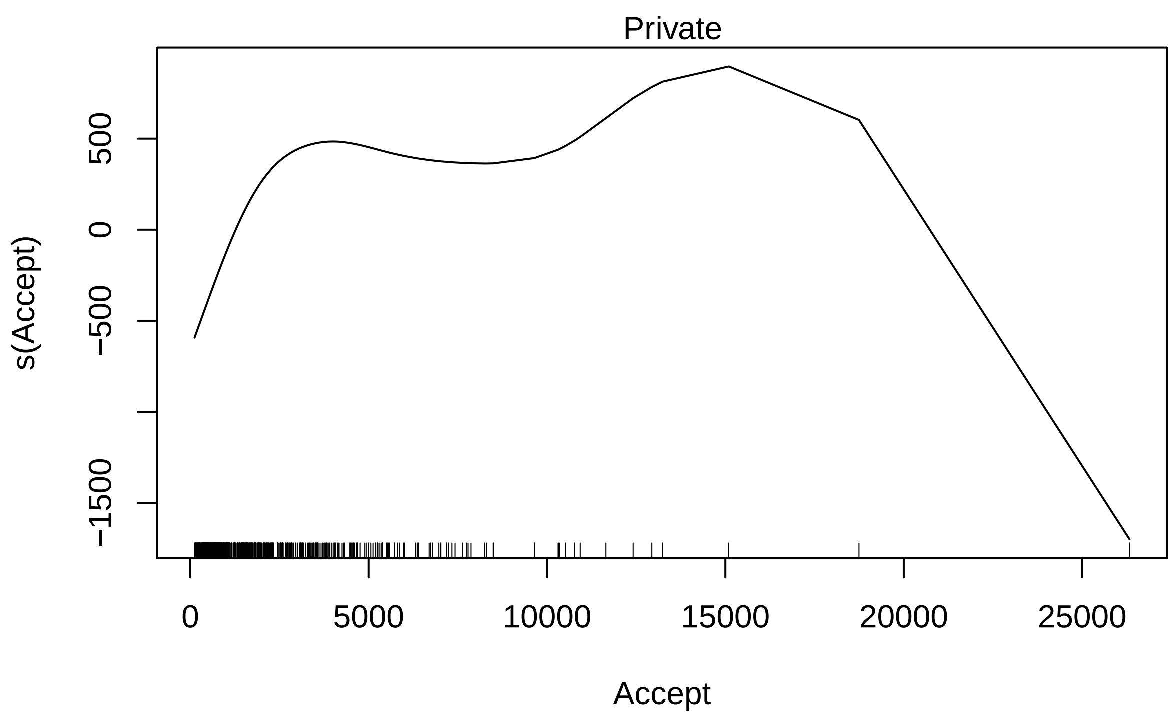
```
## Subset selection object
## Call: regsubsets.formula(Outstate ~ ., data = College_train, method = "forward",
##     nvmax = 17)
## 17 Variables  (and intercept)
##             Forced in Forced out
## PrivateYes      FALSE      FALSE
## Apps            FALSE      FALSE
## Accept          FALSE      FALSE
## Enroll          FALSE      FALSE
## Top10perc       FALSE      FALSE
## Top25perc       FALSE      FALSE
## F.Undergrad     FALSE      FALSE
## P.Undergrad     FALSE      FALSE
## 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
## Expend          FALSE      FALSE
## Grad.Rate       FALSE      FALSE
## 1 subsets of each size up to 17
## Selection Algorithm: forward
##           PrivateYes Apps Accept Enroll Top10perc Top25perc F.Undergrad
## 1  ( 1 )  " "        " "  " "    " "    " "       " "       " "
## 2  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 3  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
```

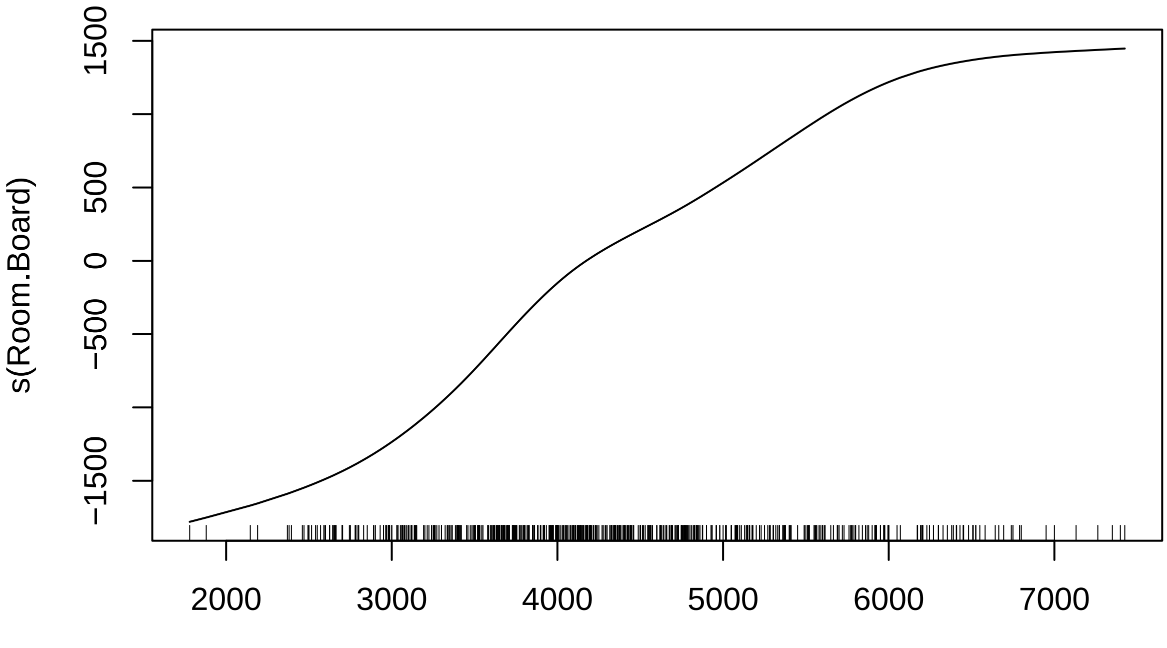
```
## 4  ( 1 )  "*"        " "  " "    " "    " "       " "       " "        
## 5  ( 1 )  "*"        " "  " "    " "    " "       " "       " "        
## 6  ( 1 )  "*"        " "  " "    " "    " "       " "       " "        
## 7  ( 1 )  "*"        " "  "*"    " "    " "       " "       " "        
## 8  ( 1 )  "*"        "*"  "*"    " "    " "       " "       " "        
## 9  ( 1 )  "*"        "*"  "*"    " "    "*"       " "       " "        
## 10  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "        
## 11  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "        
## 12  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "        
## 13  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "        
## 14  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "        
## 15  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       " "        
## 16  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"        
## 17  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"        
##           P.Undergrad 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 )  " "         "*"        " "   " "      "*" " "      " "      
## 9  ( 1 )  " "         "*"        " "   " "      "*" " "      " "      
## 10  ( 1 ) " "         "*"        " "   " "      "*" " "      " "      
## 11  ( 1 ) " "         "*"        " "   "*"      "*" " "      " "      
## 12  ( 1 ) " "         "*"        " "   "*"      "*" "*"      " "      
## 13  ( 1 ) " "         "*"        " "   "*"      "*" "*"      "*"      
## 14  ( 1 ) " "         "*"        "*"   "*"      "*" "*"      "*"      
## 15  ( 1 ) " "         "*"        "*"   "*"      "*" "*"      "*"      
## 16  ( 1 ) " "         "*"        "*"   "*"      "*" "*"      "*"      
## 17  ( 1 ) "*"         "*"        "*"   "*"      "*" "*"      "*"      
##           perc.alumni Expend Grad.Rate
## 1  ( 1 )  " "         "*"    " "      
## 2  ( 1 )  " "         "*"    " "      
## 3  ( 1 )  " "         "*"    " "      
## 4  ( 1 )  "*"         "*"    " "      
## 5  ( 1 )  "*"         "*"    " "      
## 6  ( 1 )  "*"         "*"    "*"      
## 7  ( 1 )  "*"         "*"    "*"      
## 8  ( 1 )  "*"         "*"    "*"      
## 9  ( 1 )  "*"         "*"    "*"      
## 10  ( 1 ) "*"         "*"    "*"      
## 11  ( 1 ) "*"         "*"    "*"      
## 12  ( 1 ) "*"         "*"    "*"      
## 13  ( 1 ) "*"         "*"    "*"      
## 14  ( 1 ) "*"         "*"    "*"      
## 15  ( 1 ) "*"         "*"    "*"      
## 16  ( 1 ) "*"         "*"    "*"      
## 17  ( 1 ) "*"         "*"    "*"      
```

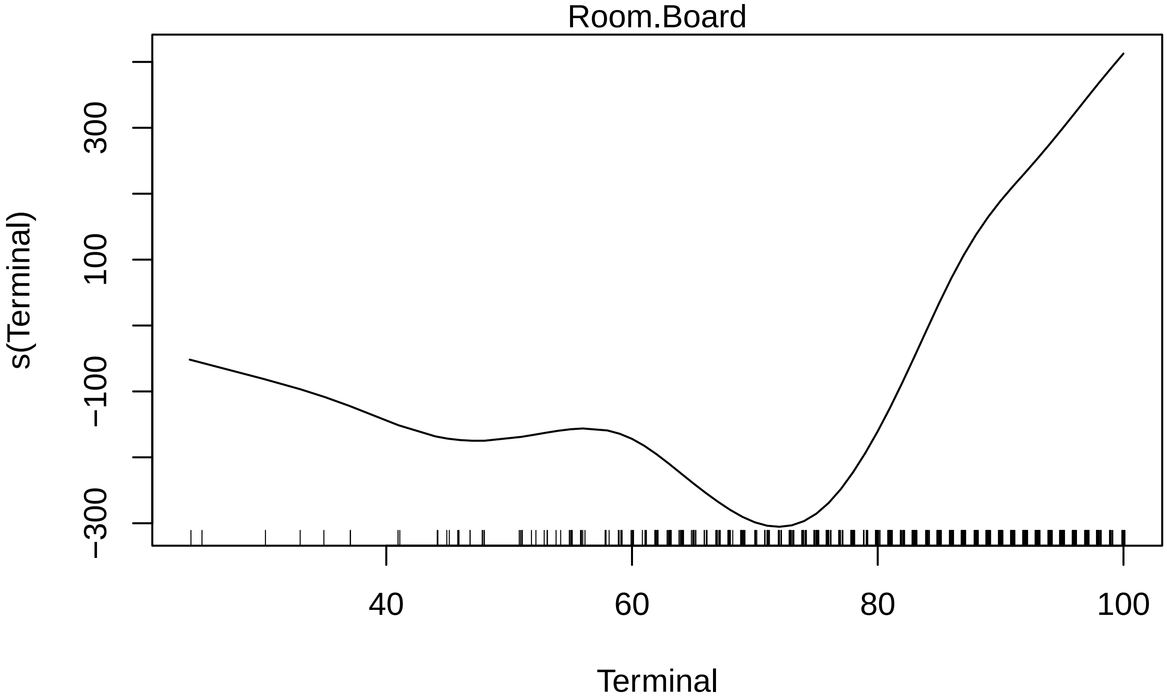
**b:**

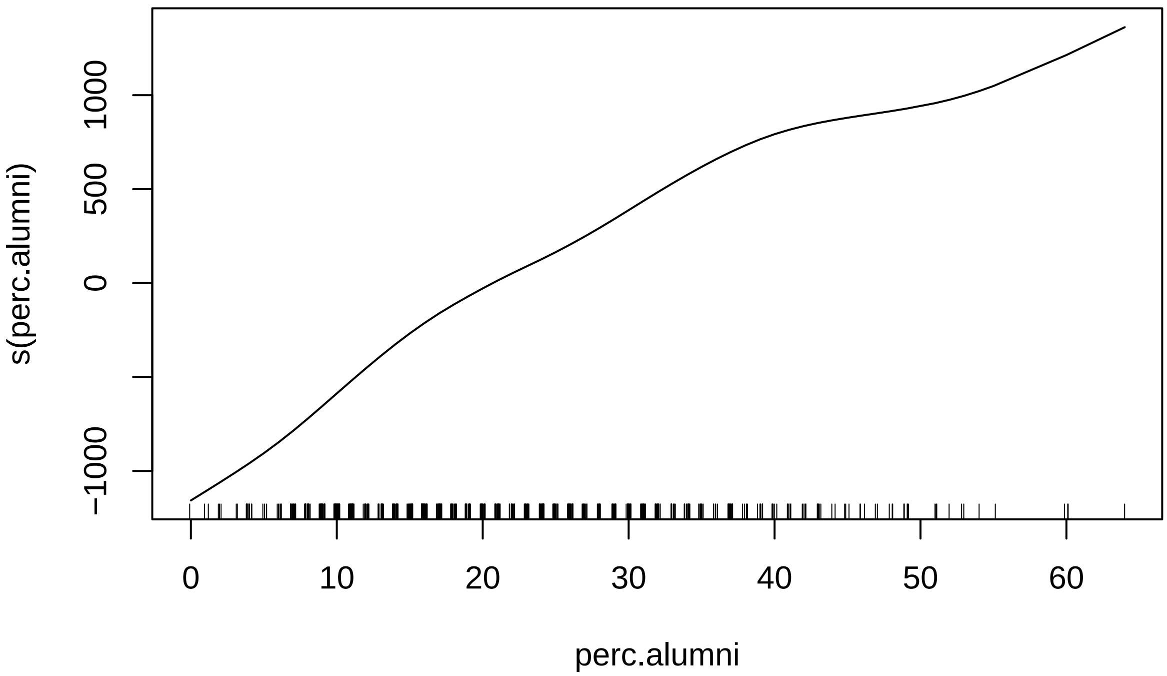
```r
gamModel <- gam(Outstate ~ Private + s(Accept) + s(Room.Board) + s(Terminal) +
                  s(perc.alumni) + s(Expend) + s(Grad.Rate), data = College_train)

plot(gamModel)
```



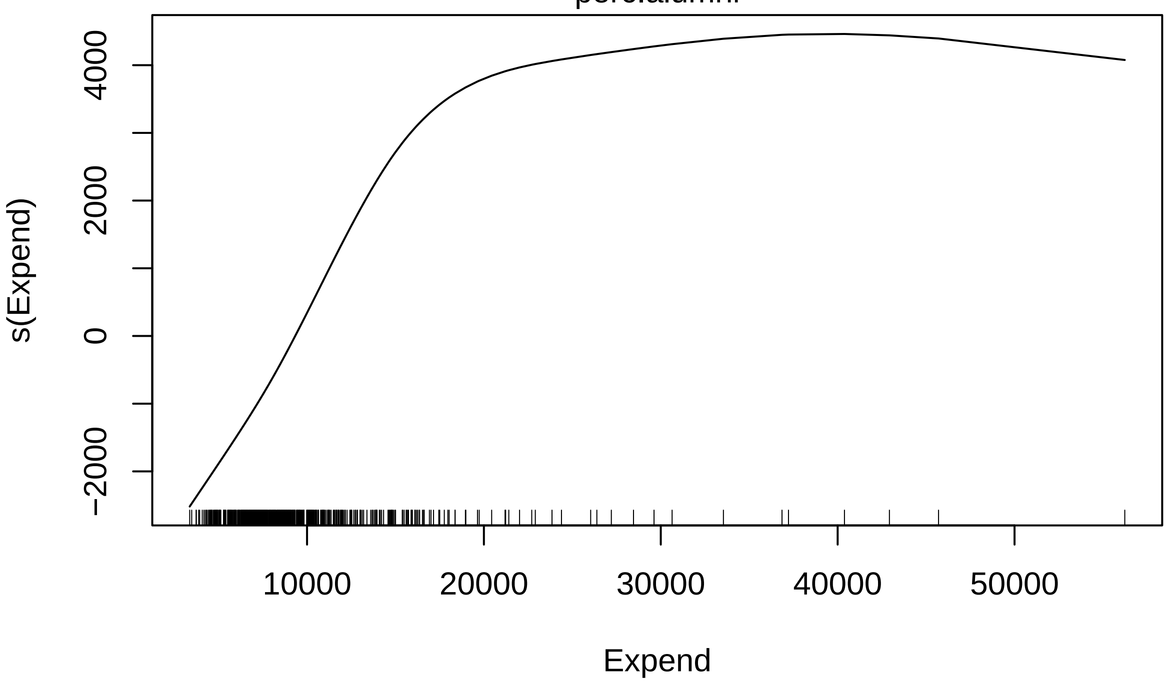




s(Room.Board)
1500
500
0
−500
−1500
2000
3000
4000
5000
6000
7000
Room.Board

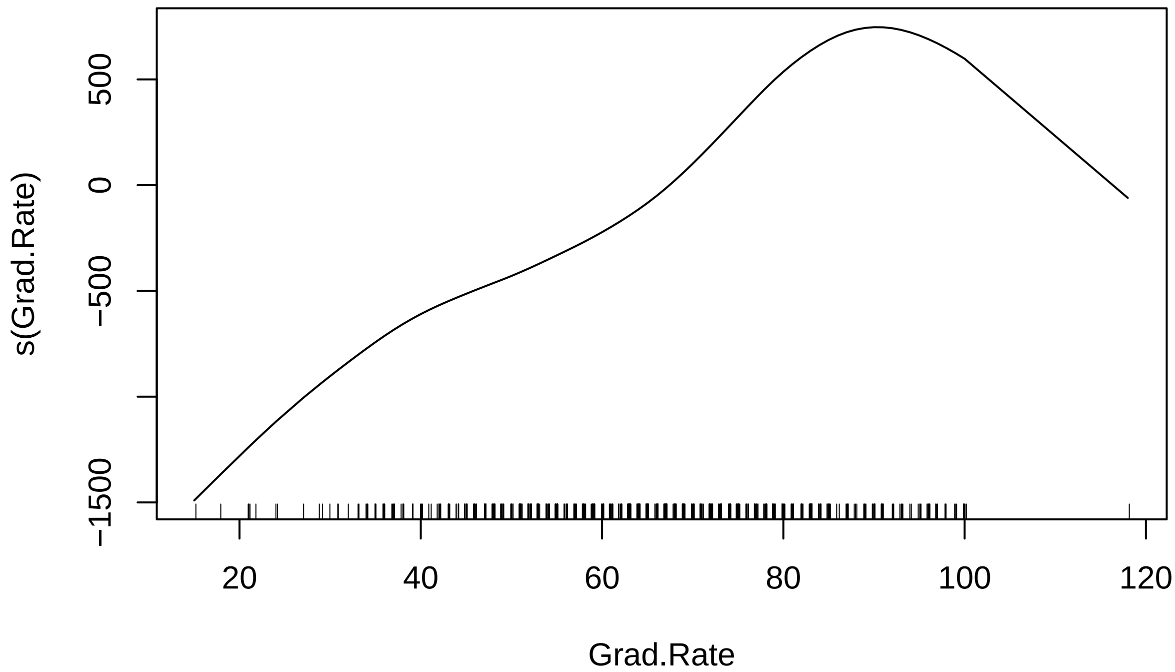

s(Terminal)
300
100
−100
−300
40
60
80
100
Terminal



s(perc.alumni)
−1000
0
500
1000
0
10
20
30
40
50
60
perc.alumni

s(Expend)
−2000
0
2000
4000
10000
20000
30000
40000
50000
Expend




When we plot GAM we get partial regression plots, which show us the effect of each variable in the model. All of the plots have noticeable patterns, suggesting they are adding important aspects to our model. For example, the perc.alumni partial regression plot shows there is a clear positive correlation for this variable, which tells us how it affects our predictions (a higher perc.alumni correlates with higher Outstate).

### c:

```
preds <- predict (gamModel , newdata = College_test)

RMSE = sqrt(mean(preds - College_test$Outstate)^2)
RMSE
```

```
## [1] 224.0902
```

Our root mean squared error tells us, on average, how many units the predictions are from the actual data points.

### d:

```
summary(gamModel)
```

```
##
## Call: gam(formula = Outstate ~ Private + s(Accept) + s(Room.Board) +
##     s(Terminal) + s(perc.alumni) + s(Expend) + s(Grad.Rate),
##     data = College_train)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -7107.69 -1056.42    50.56  1148.46  7751.61
##
## (Dispersion Parameter for gaussian family taken to be 3241383)
##
##     Null Deviance: 9565678129 on 581 degrees of freedom
## Residual Deviance: 1802208108 on 555.9997 degrees of freedom
```

```
## AIC: 10404.11
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                 Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private          1 2793995928 2793995928 861.976 < 2.2e-16 ***
## s(Accept)        1  608019077  608019077 187.580 < 2.2e-16 ***
## s(Room.Board)    1 1364900469 1364900469 421.086 < 2.2e-16 ***
## s(Terminal)      1  346065812  346065812 106.765 < 2.2e-16 ***
## s(perc.alumni)   1  421890089  421890089 130.157 < 2.2e-16 ***
## s(Expend)        1  733406007  733406007 226.263 < 2.2e-16 ***
## s(Grad.Rate)     1   79741242   79741242  24.601 9.383e-07 ***
## Residuals      556 1802208108    3241383
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                Npar Df Npar F     Pr(F)
## (Intercept)
## Private
## s(Accept)            3  6.526 0.0002415 ***
## s(Room.Board)        3  2.720 0.0438546 *
## s(Terminal)          3  1.798 0.1464515
## s(perc.alumni)       3  0.865 0.4587665
## s(Expend)            3 33.519 < 2.2e-16 ***
## s(Grad.Rate)         3  2.025 0.1093883
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on the Anova for Nonparametric Effects, “Accept”, “Expend”, and “Grad.Rate” all show evidence of a nonlinear effect ($p < 0.05$).