Chapter 7 Moving Beyond Linearity

Jishen Yin

2020/5/12

```
knitr::opts_chunk$set(echo = TRUE)
library(ISLR)
library(MASS)
library(splines)
library(tidyverse)
```

Problem 6

In this exercise, you will further analyze the `Wage` data set considered throughout this chapter.

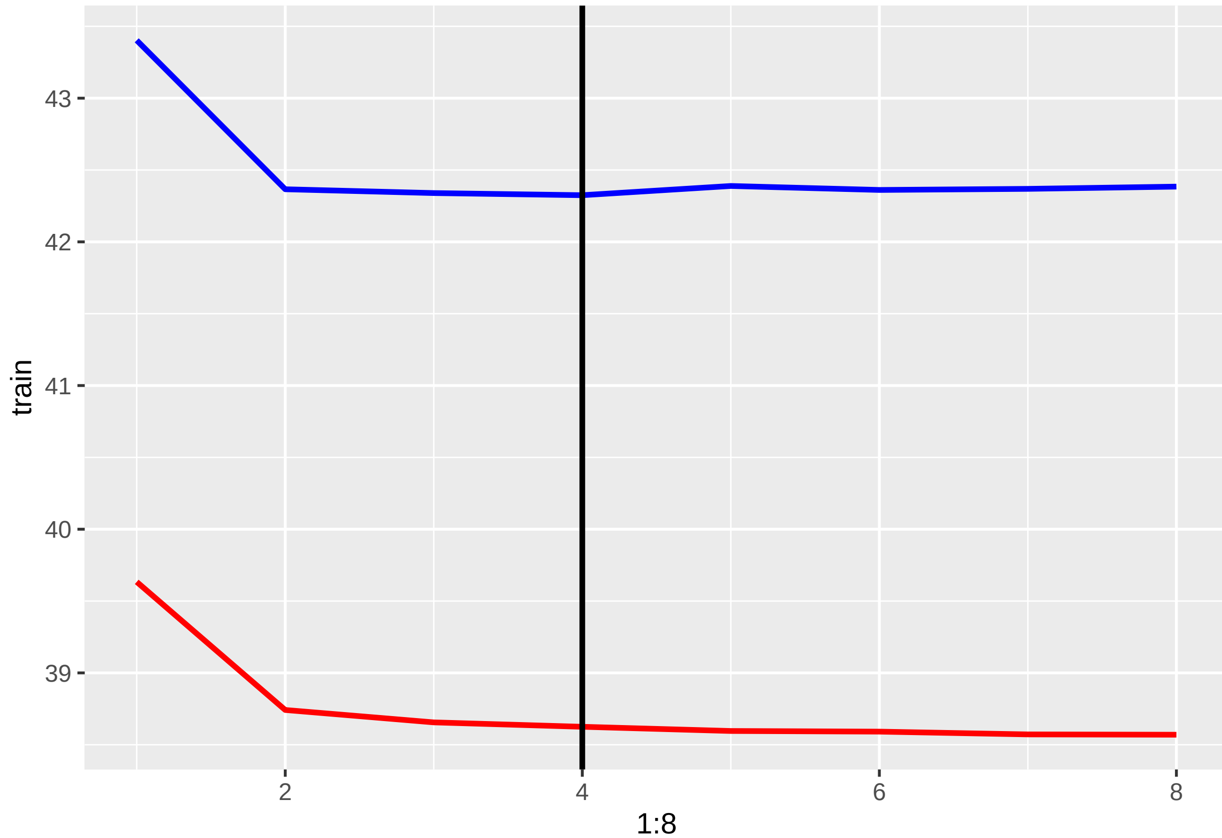
```
data(Wage)
```

(a) Perform polynomial regression to predict `wage` using `age`. Use cross-validation to select the optimal degree d for the polynomial. What degree was chosen, and how does this compare to the results of hypothesis testing using ANOVA? Make a plot of the resulting polynomial fit to the data.

```
set.seed(1)
train <- sample(1:3000, 2000)
Wage_train <- Wage[train,]
Wage_val <- Wage[-train,]
```

```
RMSE <- function(y_pred, y){
  return(sqrt(mean((y_pred-y)^2)))
}
```

```
cv <- lapply(1:8, function(x){
  model <- lm(wage ~ poly(age, x), data = Wage_train)
  train_rmse <- RMSE(predict(model, Wage_train), Wage_train$wage)
  val_rmse <- RMSE(predict(model, Wage_val), Wage_val$wage)
  return(data.frame(train = train_rmse, val = val_rmse))
})

ggplot(data = do.call(rbind, cv)) +
  geom_line(aes(x = 1:8, y = train), color = "red", size = 1) +
  geom_line(aes(x = 1:8, y = val), color = "blue", size = 1) +
  geom_vline(aes(xintercept = 4), size = 1)
```




```r
model1 <- lm(wage ~ poly(age, 1), data = Wage_train)
model2 <- lm(wage ~ poly(age, 2), data = Wage_train)
model3 <- lm(wage ~ poly(age, 3), data = Wage_train)
model4 <- lm(wage ~ poly(age, 4), data = Wage_train)
model5 <- lm(wage ~ poly(age, 5), data = Wage_train)
model6 <- lm(wage ~ poly(age, 6), data = Wage_train)
model7 <- lm(wage ~ poly(age, 7), data = Wage_train)
model8 <- lm(wage ~ poly(age, 8), data = Wage_train)
anova(model1, model2, model3, model4, model5, model6, model7, model8)
```

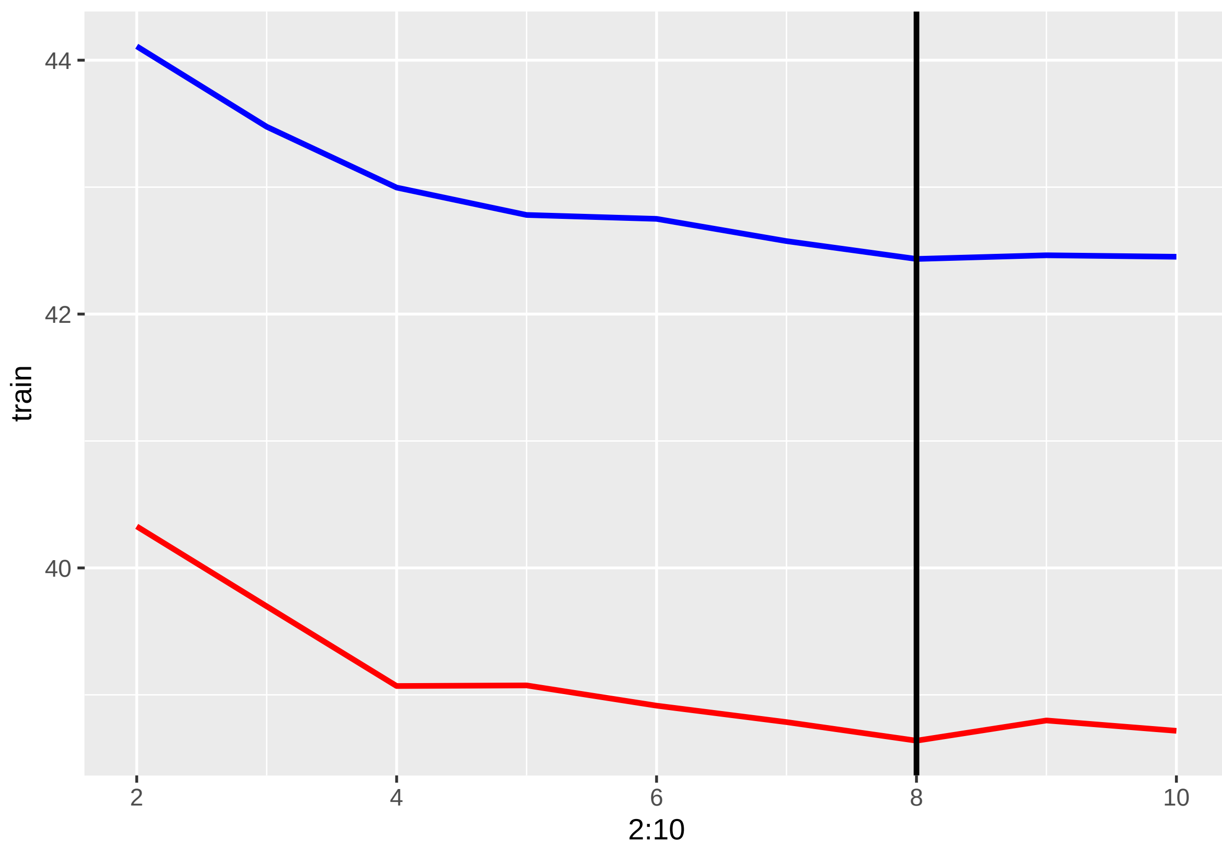
```
## Analysis of Variance Table
##
## Model 1: wage ~ poly(age, 1)
## Model 2: wage ~ poly(age, 2)
## Model 3: wage ~ poly(age, 3)
## Model 4: wage ~ poly(age, 4)
## Model 5: wage ~ poly(age, 5)
## Model 6: wage ~ poly(age, 6)
## Model 7: wage ~ poly(age, 7)
## Model 8: wage ~ poly(age, 8)
##   Res.Df     RSS Df Sum of Sq       F    Pr(>F)
## 1   1998 3141646
## 2   1997 3001882  1    139764 93.5286 < 2.2e-16 ***
## 3   1996 2988505  1     13377  8.9518  0.002806 **
## 4   1995 2983787  1      4718  3.1573  0.075741 .
```

```
## 5   1994 2979263  1      4524  3.0275  0.082017 .
## 6   1993 2978573  1       690  0.4614  0.497041
## 7   1992 2975584  1      2989  2.0003  0.157423
## 8   1991 2975248  1       335  0.2245  0.635676
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

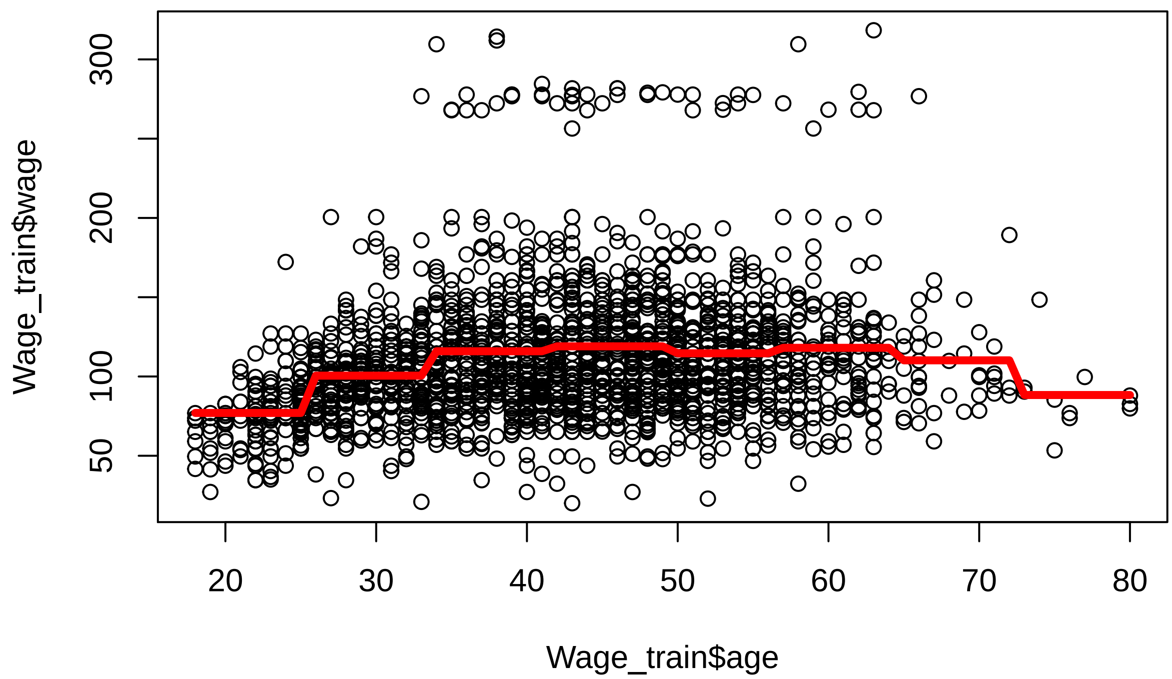
Model 2 and 3 are significantly better than the model 1 and 2 separately while model 4 and 5 are not. Cross Validation also shows that model 3 and 4 demonstrate the smallest test error.

(b) Fit a step function to predict `wage` using `age`, and perform cross-validation to choose the optimal number of cuts. Make a plot of the fit obtained.

```r
cv <- lapply(2:10, function(x){
  model <- lm(wage ~ cut(age, x), data = Wage_train)
  train_rmse <- RMSE(predict(model, Wage_train), Wage_train$wage)
  val_rmse <- RMSE(predict(model, Wage_val), Wage_val$wage)
  return(data.frame(train = train_rmse, val = val_rmse))
})

ggplot(data = do.call(rbind, cv)) +
  geom_line(aes(x = 2:10, y = train), color = "red", size = 1) +
  geom_line(aes(x = 2:10, y = val), color = "blue", size = 1) +
  geom_vline(aes(xintercept = 8), size = 1)
```



```
bs <- lm(wage~cut(age, 8), data = Wage_train)
agelims <- range(Wage_train$age)
age.grid <- seq(from = agelims[1], to = agelims[2])
pred <- predict(bs, list(age = age.grid))

plot(Wage_train$age, Wage_train$wage)
lines(age.grid, pred, col = "red", lwd = 4)
```



Problem 9

This question uses the variables `dis` (the weighted mean of distances to five Boston employment centers) and `nox` (nitrogen oxides concentration in parts per 10 million) from the `Boston` data. We will treat `dis` as the predictor and `nox` as the response.

```
data(Boston)
```

(a) Use the `poly()` function to fit a cubic polynomial regression to predict `nox` using `dis`. Report the regression output, and plot the resulting data and polynomial fits.

```
poly3 <- lm(nox ~ poly(dis, 3), data = Boston)

RMSE(predict(poly3, Boston), Boston$nox)
```

```
## [1] 0.06182512
```

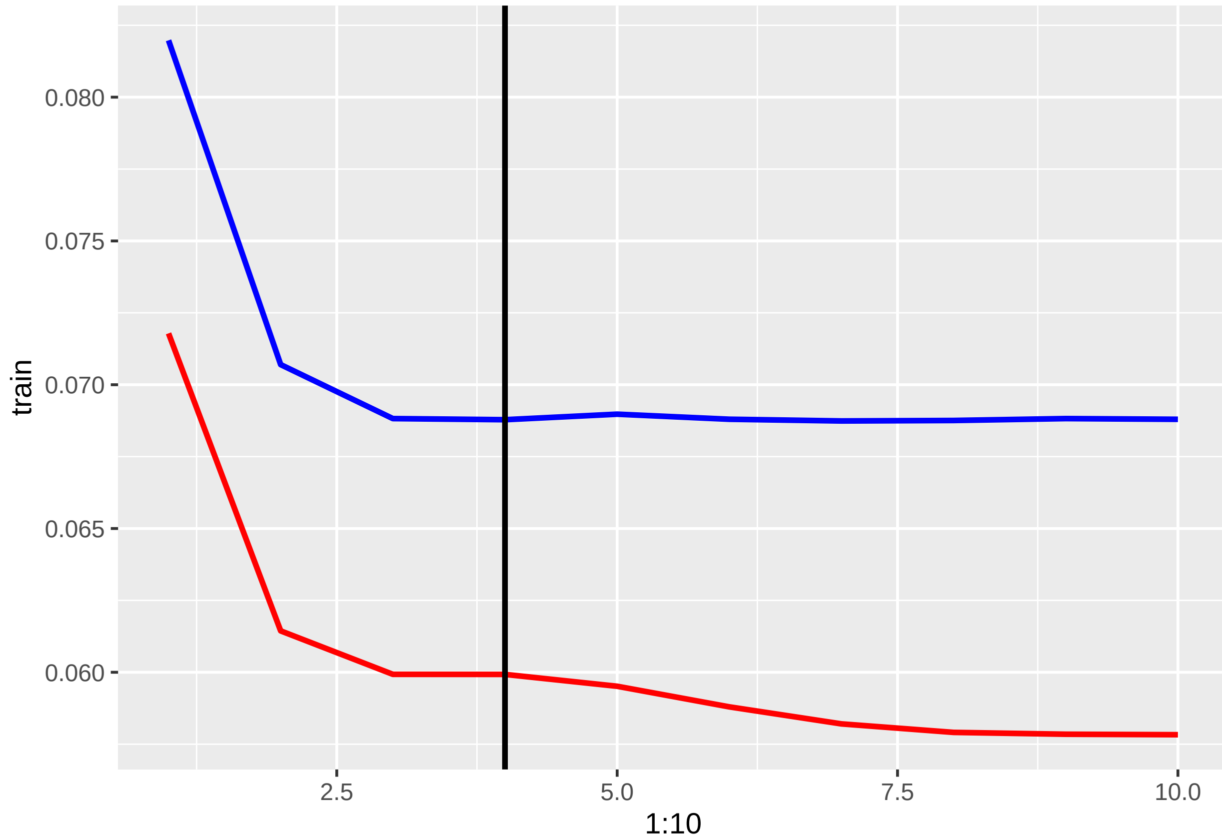
(b) Plot the polynomial fits for a range of different polynomial degrees (say, from 1 to 10), and report the associated residual sum of squares.

```
rm <- sapply(1:10, function(x){
  model <- lm(nox ~ poly(dis, x), data = Boston)
  return(RMSE(predict(model, Boston), Boston$nox))
})

rm
```

```
##  [1] 0.07396937 0.06342126 0.06182512 0.06180713 0.06152364 0.06092595
##  [7] 0.06045747 0.06023061 0.06019289 0.06017384
```

(c) Perform cross-validation or another approach to select the optimal degree for the polynomial, and explain your results.

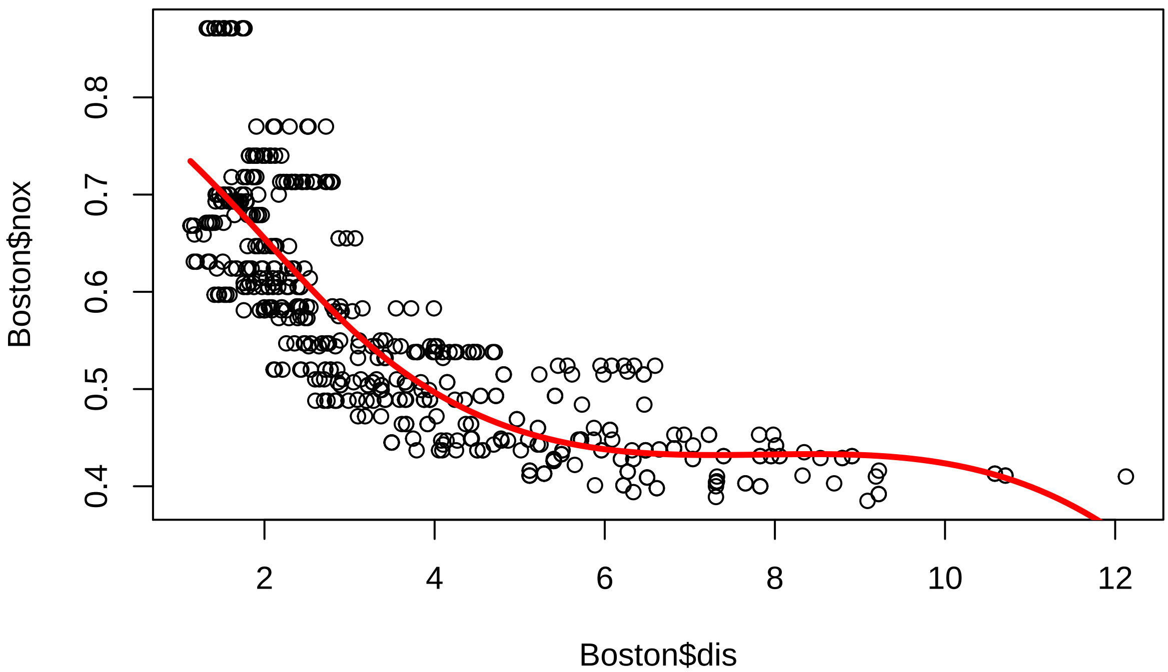
```
set.seed(4)
train <- sample(1:506, 400)
Boston_train <- Boston[train,]
Boston_val <- Boston[-train,]

cv <- lapply(1:10, function(x){
  model <- lm(nox ~ poly(dis, x), data = Boston_train)
  train_rmse <- RMSE(predict(model, Boston_train), Boston_train$nox)
  val_rmse <- RMSE(predict(model, Boston_val), Boston_val$nox)
  return(data.frame(train = train_rmse, val = val_rmse))
})

ggplot(data = do.call(rbind, cv)) +
  geom_line(aes(x = 1:10, y = train), color = "red", size = 1) +
  geom_line(aes(x = 1:10, y = val), color = "blue", size = 1) +
  geom_vline(aes(xintercept = 4), size = 1)
```



(d) Use the `bs()` function to fit a regression spline to predict `nox` using `dis`. Report the output for the fit using four degrees of freedom. How did you choose the knots? Plot the resulting fit.

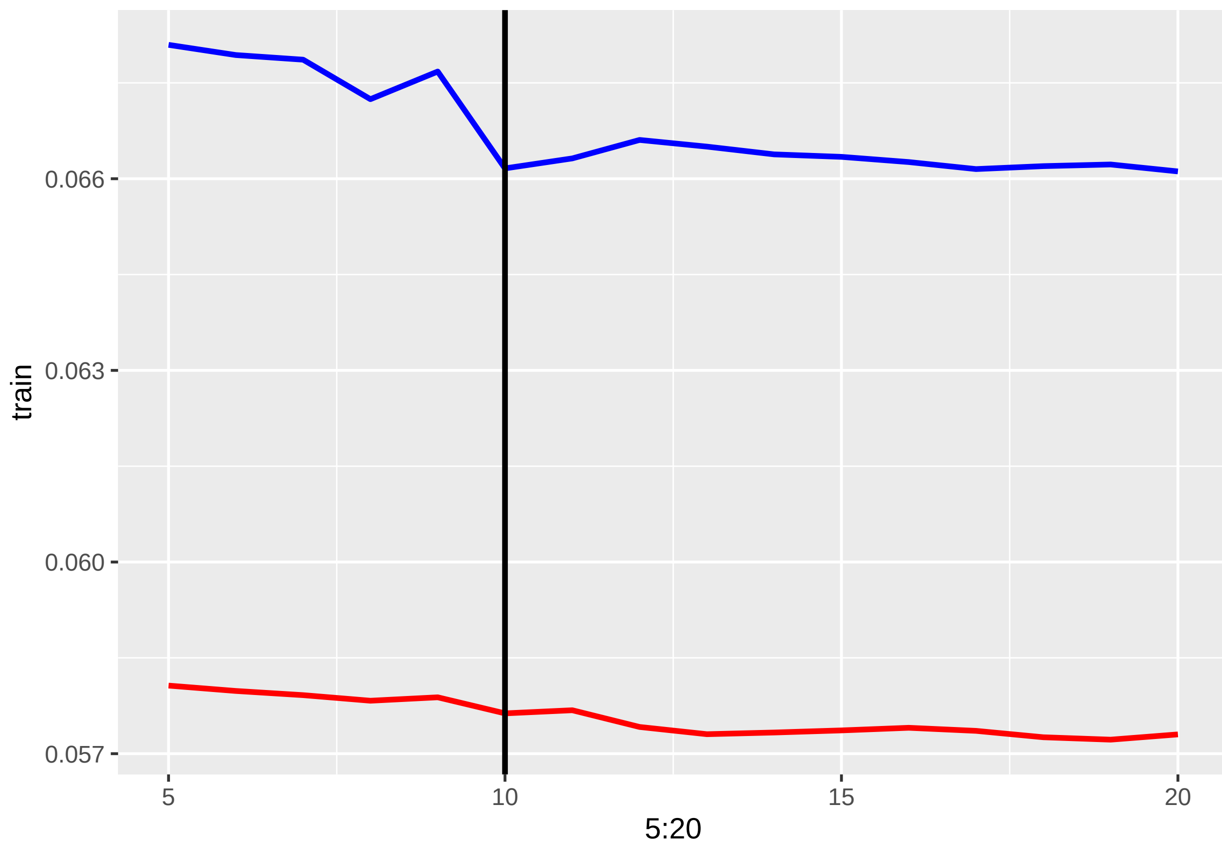
```
knots <- attr(bs(Boston$dis, df = 4), "knots")
rs <- lm(nox ~ bs(dis, knots = knots), data = Boston)
```

```
plot(Boston$dis, Boston$nox)
x.new <- seq(min(Boston$dis), max(Boston$dis), 0.1)
lines(x.new, predict(rs, list(dis = x.new)), col = "red", lwd = 3)
```



(f) Perform cross-validation or another approach in order to select the best degrees of freedom for a regression spline on this data. Describe your results.

```
cv <- lapply(5:20, function(x){
  knots <- attr(bs(Boston$dis, df = x), "knots")
  model <- lm(nox ~ bs(dis, knots = knots), data = Boston)
  train_rmse <- RMSE(predict(model, Boston_train), Boston_train$nox)
  val_rmse <- RMSE(predict(model, Boston_val), Boston_val$nox)
  return(data.frame(train = train_rmse, val = val_rmse))
})

ggplot(data = do.call(rbind, cv)) +
  geom_line(aes(x = 5:20, y = train), color = "red", size = 1) +
  geom_line(aes(x = 5:20, y = val), color = "blue", size = 1) +
  geom_vline(aes(xintercept = 10), size = 1)
```


train
0.066
0.063
0.060
0.057
5
10
15
20
5:20