HW2 R Markdown

Santanu Mukherjee

03/13/2022

R Markdown

Chapter 7 - Book - An Introduction to Statistical Learning - 7.1 - Exercises page 297:

Q7.1

It was mentioned in the chapter that a cubic regression spline with one knot at ξ can be obtained using a basis of the form x, x^2, x^3, $(x-\xi)^3$+, where $(x-\xi)^3_+$ = $(x-\xi)^3$ if x > ξ and equals 0 otherwise.

We will now show that a function of the form

$f(x)$ = β_0 + $\beta_1 x$ + $\beta_2 x^2$ + $\beta_3 x^3$ + $\beta_4 (x-\xi)^3_+$

is indeed a cubic regression spline, regardless of the values of β_0,β_1,β_2,β_3,β_4.

Q7.1a

Find a cubic polynomial

$f_1(x)$ = a_1 + $b_1 x$ + $c_1 x^2$ + $d_1 x^3$

such that $f(x)$ = $f_1(x)$ for all $x \leq \xi$. Express a_1, b_1, c_1, d_1 in terms of β_0,β_1,β_2,β_3,β_4.

Answer 7.1 (a)

As we are given that $(x-\xi)^3_+$ = $(x-\xi)^3$ if x > ξ and equals 0 otherwise.

So, in this question for $x \leq \xi$, we have $f(x)$ = β_0 + $\beta_1 x$ + $\beta_2 x^2$ + $\beta_3 x^3$, because $(x-\xi)^3_+ = 0$

Also in this question it is given that $f(x)$ = $f_1(x)$

so if we equate the corresponding coefficients of $f(x)\ and\ f_1(x)$

we get a_1= β_0, b_1= β_1, c_1= β_2 and d_1= β_3.

Q7.1b

Find a cubic polynomial

$f_2(x)$ = a_2 + $b_2 x$ + $c_2 x^2$ + $d_2 x^3$

such that $f(x)$ = $f_2(x)$ for all $x > \xi$. Express a_2, b_2, c_2, d_2 in terms of β_0,β_1,β_2,β_3,β_4.

We have now established that $f(x)$ is a piece wise polynomial.

Answer 7.1 (b)

For $x > \xi$, we have

$f(x)$ = β_0 + $\beta_1 x$ + $\beta_2 x^2$ + $\beta_3 x^3$ + $\beta_4 (x - \xi)^3_+$, which means

$f(x)$ = $(\beta_0 - \beta_4 \xi^3)$ + $(\beta_1 + 3\xi^2 \beta_4)x$ + $(\beta_2 - 3\beta_4 \xi)x^2$ + $(\beta_3 + \beta_4)x^3$.

Given that $f_2(x)$ = a_2 + $b_2 x$ + $c_2 x^2$ + $d_2 x^3$

Also in this question it is given that $f(x)$ = $f_2(x)$

so if we equate the corresponding coefficients of $f(x)\ and\ f_2(x)$

we get a_2= $\beta_0 - \beta_4 \xi^3$, b_2= $\beta_1 + 3\xi^2 \beta_4$, c_2= $\beta_2 - 3\beta_4 \xi$ and d_2= $\beta_3 + \beta_4$.

Q7.1c

Show that $f_1(\xi)$ = $f_2(\xi)$. That is, $f(x)$ is continuous at ξ.

Answer 7.1 (c)

So, from the above information we can say that

$f_1(\xi)$ = β_0 + $\beta_1 \xi$ + $\beta_2 \xi^2$ + $\beta_3 \xi^3$

and

$f_2(\xi)$ = $(\beta_0 - \beta_4 \xi^3)$ + $(\beta_1 + 3\xi^2 \beta_4)\xi$ + $(\beta_2 - 3\beta_4 \xi)\xi^2$ + $(\beta_3 + \beta_4)\xi^3$, which in turn gives

$f_2(\xi)$ = β_0 + $\beta_1 \xi$ + $\beta_2 \xi^2$ + $\beta_3 \xi^3$

So, from the above we can see that $f_1(\xi)$ = $f_2(\xi)$, which means $f(x)$ is continuous at ξ.

Q7.1d

Show that $f_1^{/}(\xi)$ = $f_2^{/}(\xi)$. That is, $f^{/}(x)$ is continuous at ξ.

Answer 7.1 (d)

So, from the above, after taking the first derivative , we get

$f_1^{/}(\xi)$ = β_1 + $2\beta_2 \xi$ + $3\beta_3 \xi^2$

and

$f_2^{/}(\xi)$ = β_1 + $3\xi^2 \beta_4$ + $2(\beta_2 - 3\beta_4 \xi)\xi$ + $3(\beta_3 + \beta_4)\xi^2$, which in turn gives

$f_2^{/}(\xi)$ = β_1 + $2\beta_2 \xi$ + $3\beta_3 \xi^2$

So, from the above we can see that $f_1^{/}(\xi)$ = $f_2^{/}(\xi)$, which means $f^{/}(x)$ is continuous at ξ.

Q7.1e

Show that $f_1^{//}(\xi)$ = $f_2^{//}(\xi)$. That is, $f^{//}(x)$ is continuous at ξ.

Answer 7.1 (e)

So, from the above, after taking the second derivative , we get

$f_1^{//}(\xi)$ = $2\beta_2$ + $6\beta_3\xi$

and

$f_2^{//}(\xi)$ = $2(\beta_2 - 3\beta_4\xi)$ + $6(\beta_3 + \beta_4)\xi$, which in turn gives

$f_2^{//}(\xi)$ = $2\beta_2$ + $6\beta_3\xi$

So, from the above we can see that $f_1^{//}(\xi)$ = $f_2^{//}(\xi)$, which means $f^{//}(x)$ is continuous at ξ.

Therefore, $f(x)$ is truly a cubic spline.

Q7.4

Suppose we fit a curve with basis functions

$b_1(X)$ = $I(0 \leq X \leq 2)$ - $(X-1)I(1 \leq X \leq 2)$,
$b_2(X)$ = $(X-3)I(3 \leq X \leq 4)$ + $I(4 < X \leq 5)$.

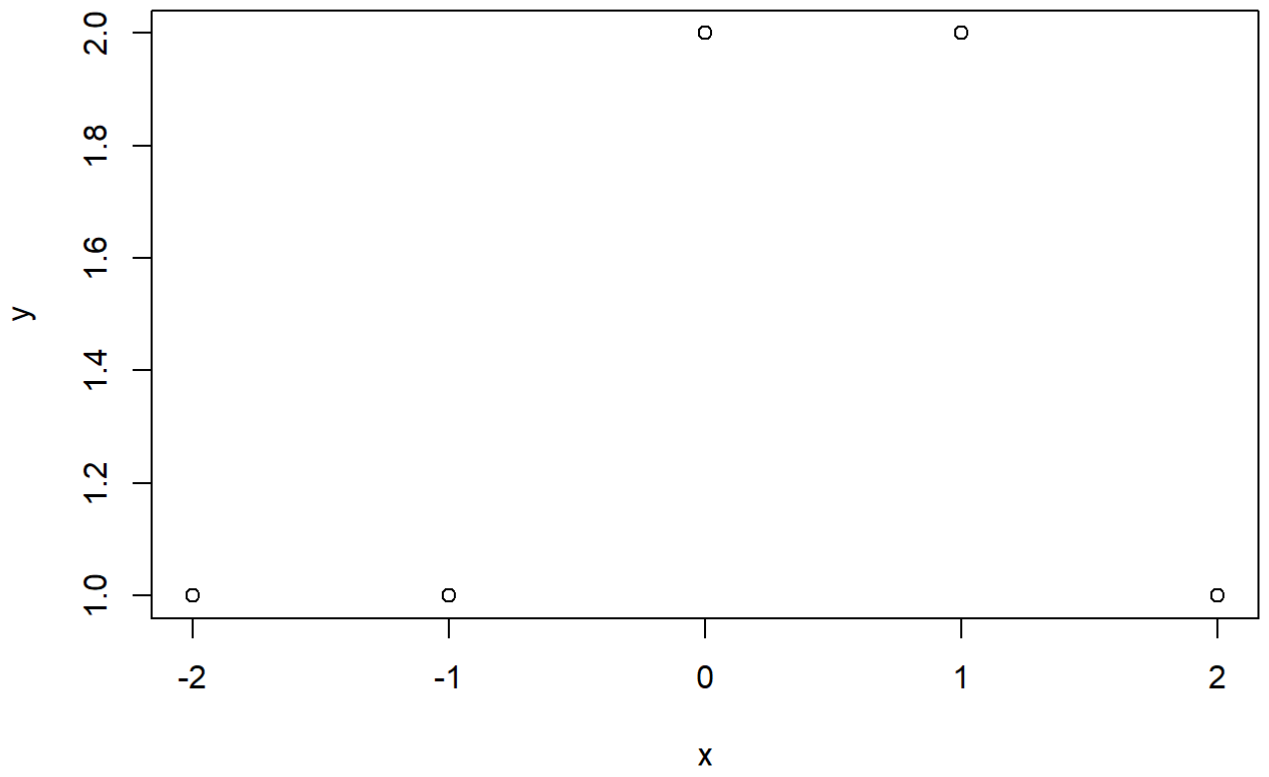
We fit the linear regression model

Y = β_0 + $\beta_1 b_1(x)$ + $\beta_2 b_2(x)$ + ϵ,

and obtain coefficient estimates $\hat{\beta}_0$ = 1, $\hat{\beta}_1$ = 1, $\hat{\beta}_2$ = 3. Sketch the estimated curve between $X = -2$ and $X = 2$. Note the intercepts, slopes, and other relevant information.

Answer 7.4

```
x = -2:2
y = c(1 + 0 + 0, # x = -2
      1 + 0 + 0, # x = -1
      1 + 1 + 0, # x = 0
      1 + (1-0) + 0, # x = 1
      1 + (1-1) + 0 # x =2
      )
plot(x,y)
```



The curve is constant between -2 and 0 :$y = 1$, constant between 0 and 1 : $y = 2$, and linear between 1 and 2 : $y = 3 - x$.

Q7.6

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

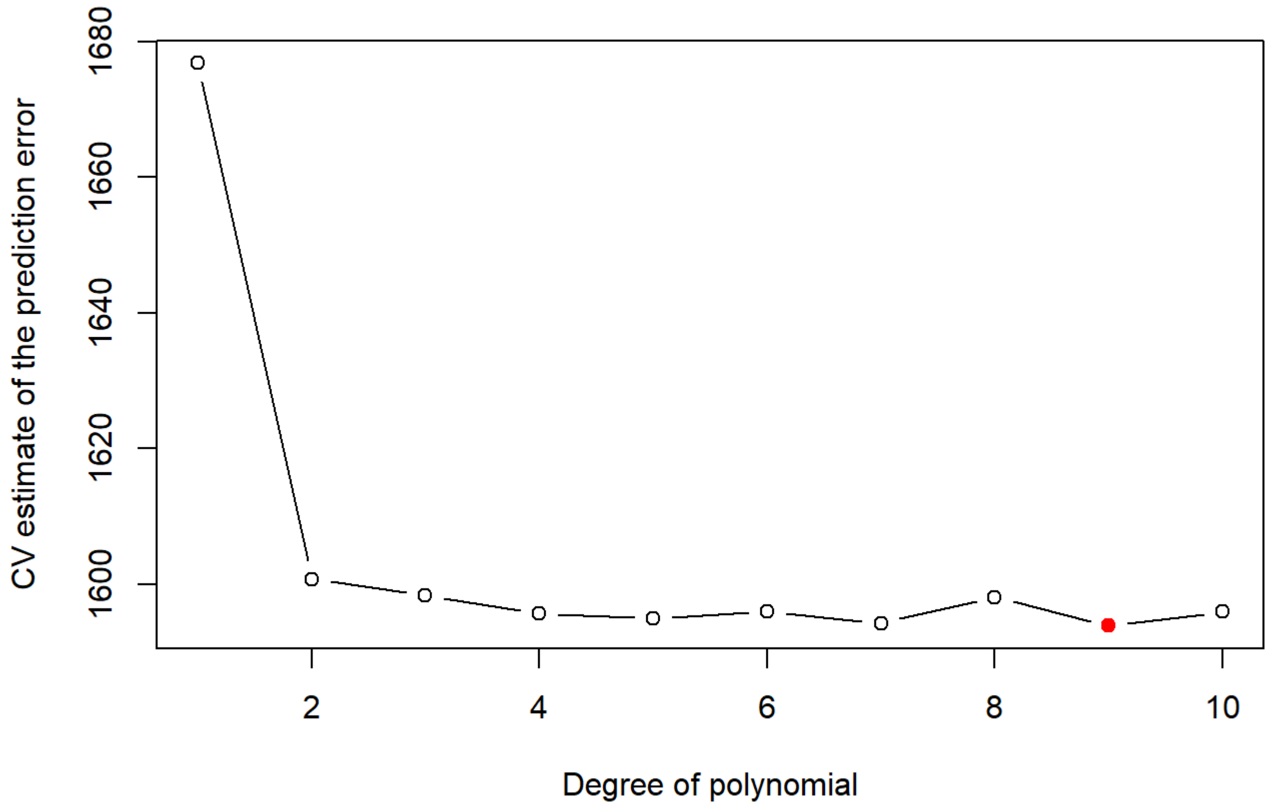
Q7.6a

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.

Answer 7.6 (a)

So, here I will perform K-fold cross-validation with $K = 10$.

```
set.seed(1)
cv.error <- rep(NA, 10)
for (i in 1:10) {
    fit <- glm(wage ~ poly(age, i), data = Wage)
    cv.error[i] <- cv.glm(Wage, fit, K = 10)$delta[1]
}
d1 <- which.min(cv.error)
x <- 1:10
plot(x, cv.error, type = "b", xlab = "Degree of polynomial", ylab = "CV estimate of the predicti
on error",
     col=ifelse(x==d1, "red", "black"),  pch=ifelse(x==d1, 19, 1))
```



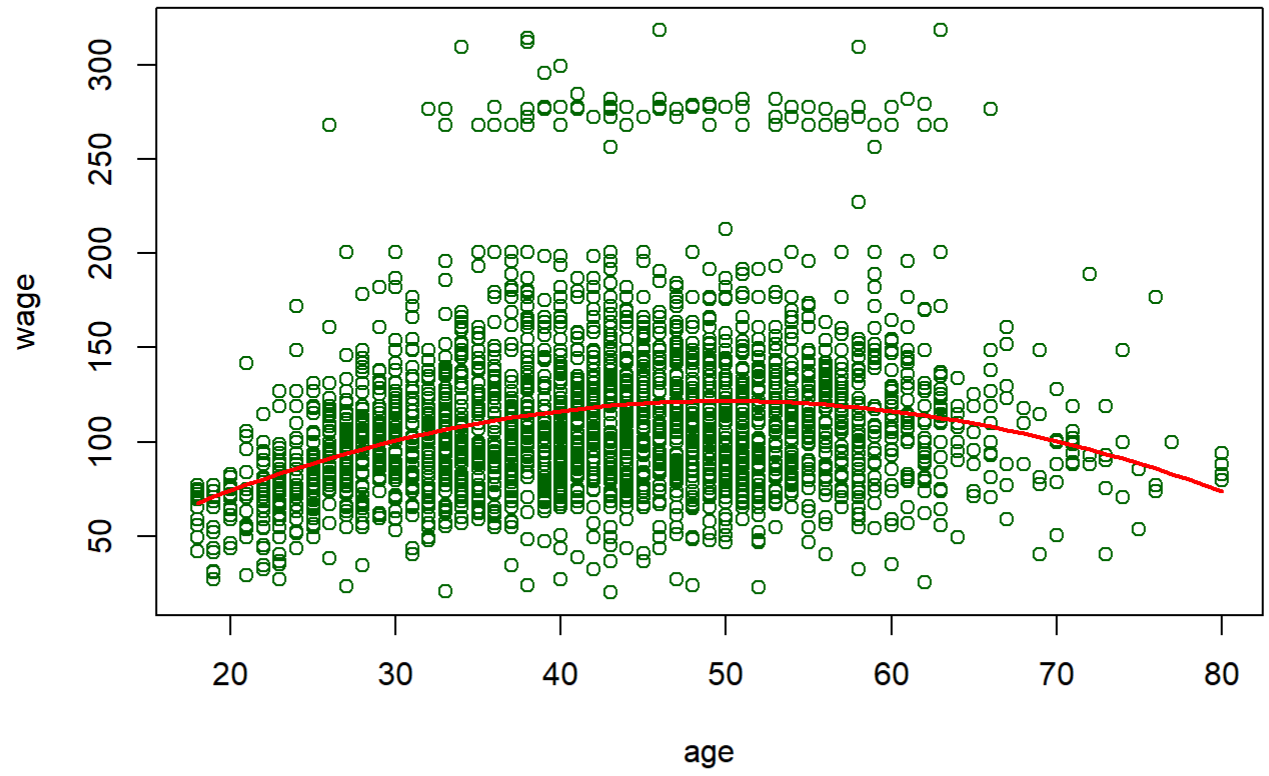
Here, We see that $d = 9$ is the optimal degree for the polynomial. We now use $ANOVA$ to test the null hypothesis that a model $M1$ is sufficient to explain the data against the alternative hypothesis that a more complex $M2$ is required

```
fit761 <- lm(wage ~ age, data = Wage)
fit762 <- lm(wage ~ poly(age, 2), data = Wage)
fit763 <- lm(wage ~ poly(age, 3), data = Wage)
fit764 <- lm(wage ~ poly(age, 4), data = Wage)
fit765 <- lm(wage ~ poly(age, 5), data = Wage)

anova(fit761, fit762, fit763, fit764, fit765)
```

```
## Analysis of Variance Table
##
## Model 1: wage ~ age
## Model 2: wage ~ poly(age, 2)
## Model 3: wage ~ poly(age, 3)
## Model 4: wage ~ poly(age, 4)
## Model 5: wage ~ poly(age, 5)
##   Res.Df     RSS Df Sum of Sq        F    Pr(>F)
## 1   2998 5022216
## 2   2997 4793430  1    228786 143.5931 < 2.2e-16 ***
## 3   2996 4777674  1     15756   9.8888  0.001679 **
## 4   2995 4771604  1      6070   3.8098  0.051046 .
## 5   2994 4770322  1      1283   0.8050  0.369682
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By looking at the p-value, we see that the square or cubic polynomial appears to be a reasonable fit for the data, but other than these , any other lower or higher models are not justified. Below is the corresponding graph.

```
plot(wage ~ age, data = Wage, col = "darkgreen")
agelims <- range(Wage$age)
age.grid <- seq(from = agelims[1], to = agelims[2])
fit <- lm(wage ~ poly(age, 2), data = Wage)
preds <- predict(fit, newdata = list(age = age.grid))
lines(age.grid, preds, col = "red", lwd = 2)
```



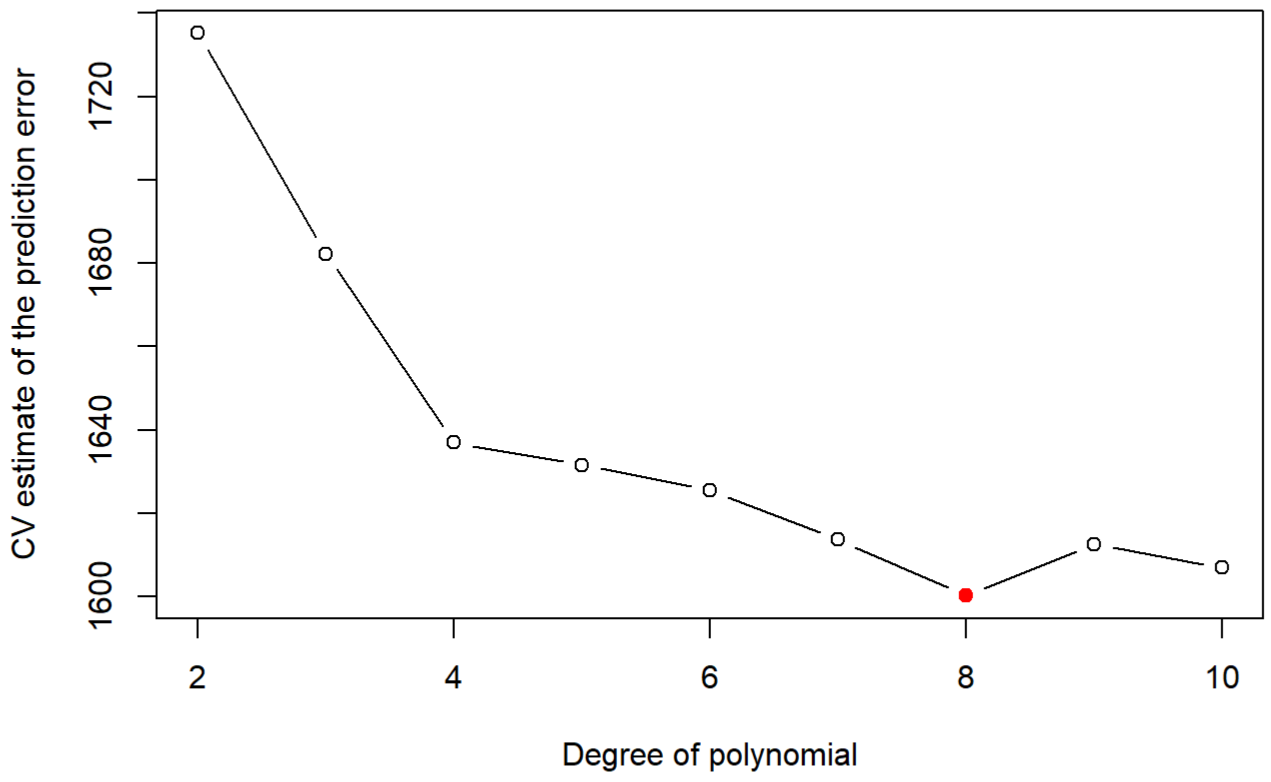
Q7.6b

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.

Answer 7.6 (b)

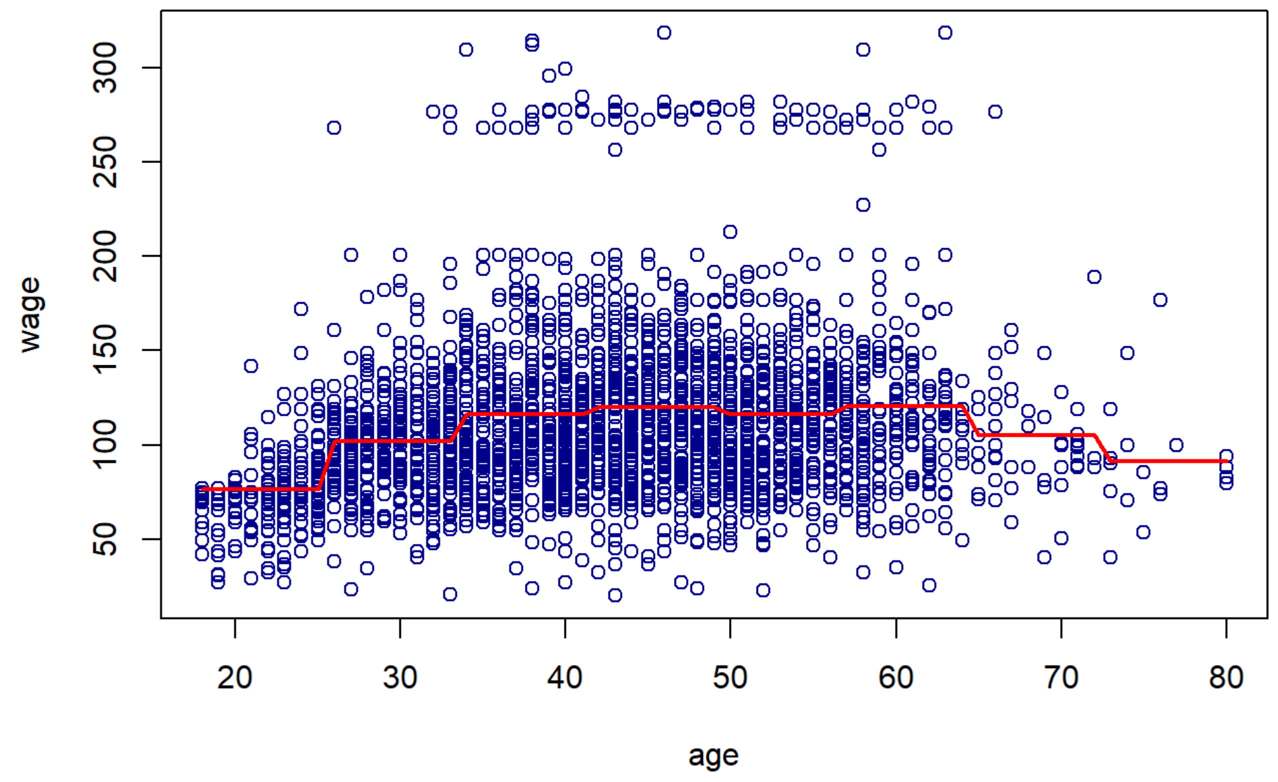
Similar to (a) above, here I will perform K-fold cross-validation with $K = 10$.

```
crossv <- rep(NA, 10)
for (i in 2:10) {
    Wage$age.cut <- cut(Wage$age, i)
    fit <- glm(wage ~ age.cut, data = Wage)
    crossv[i] <- cv.glm(Wage, fit, K = 10)$delta[1]
}
d2 <- which.min(crossv)
x <- 2:10
plot(x, crossv[-1], type = "b", xlab = "Degree of polynomial", ylab = "CV estimate of the predic
tion error",
     col=ifelse(x==d2, "red", "black"),  pch=ifelse(x==d2, 19, 1))
```



We see here that the error is minimum for $d2 = 8$ cuts. Now, we fit the entire data with a step function using $d2 = 8$ cuts and plot it.

```
plot(wage ~ age, data = Wage, col = "darkblue")
agelims <- range(Wage$age)
age.grid <- seq(from = agelims[1], to = agelims[2])
fit <- glm(wage ~ cut(age, d2), data = Wage)
preds <- predict(fit, data.frame(age = age.grid))
lines(age.grid, preds, col = "red", lwd = 2)
```



Q7.7

The **Wage** data set contains a number of other features not explored in this chapter, such as marital status (**maritl**), job class (**jobclass**), and others. Explore the relationships between some of these other predictors and **wage**, and use non-linear fitting techniques in order to fit flexible models to the data. Create plots of the results obtained, and write a summary of your findings.

Answer 7.7

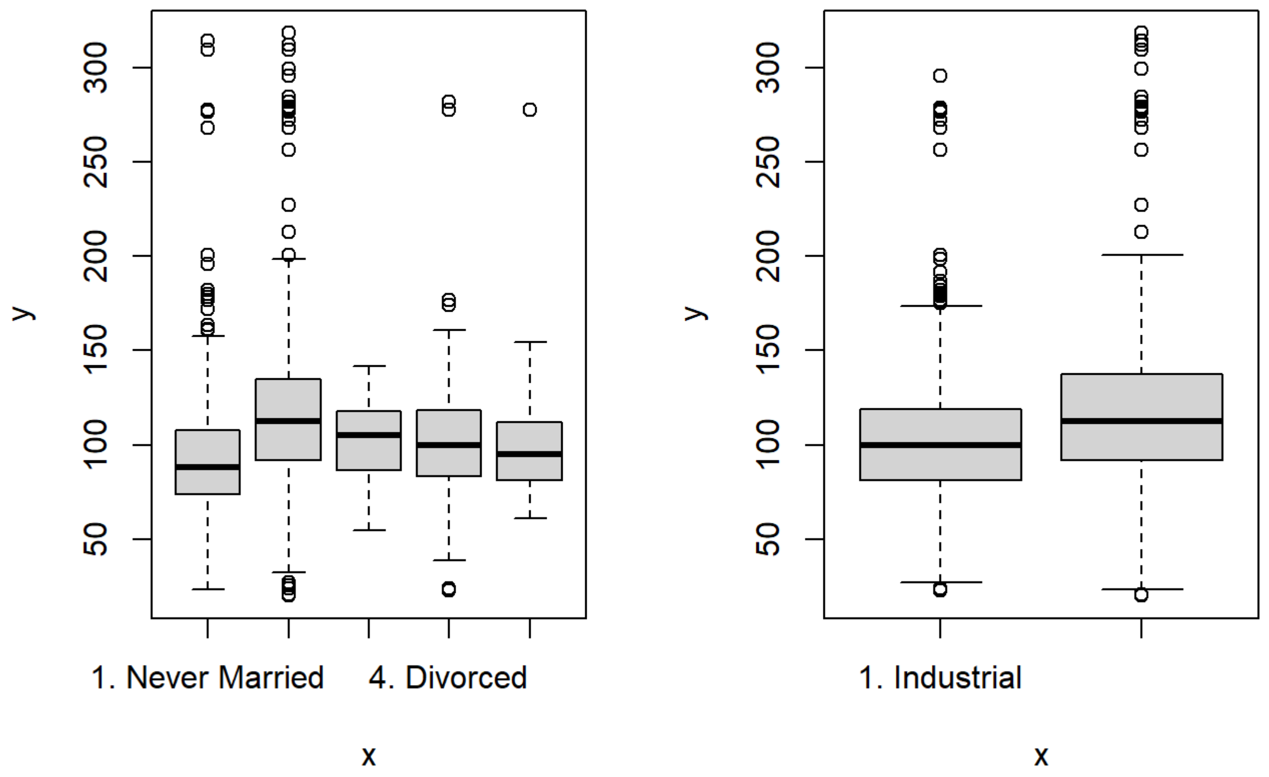
```
set.seed(1)
summary(Wage$maritl)
```

```
## 1. Never Married       2. Married       3. Widowed       4. Divorced
##               648             2074               19               204
##     5. Separated
##               55
```

```
summary(Wage$jobclass)
```

```
##  1. Industrial 2. Information
##           1544           1456
```

```
par(mfrow = c(1, 2))
plot(Wage$maritl, Wage$wage)
plot(Wage$jobclass, Wage$wage)
```



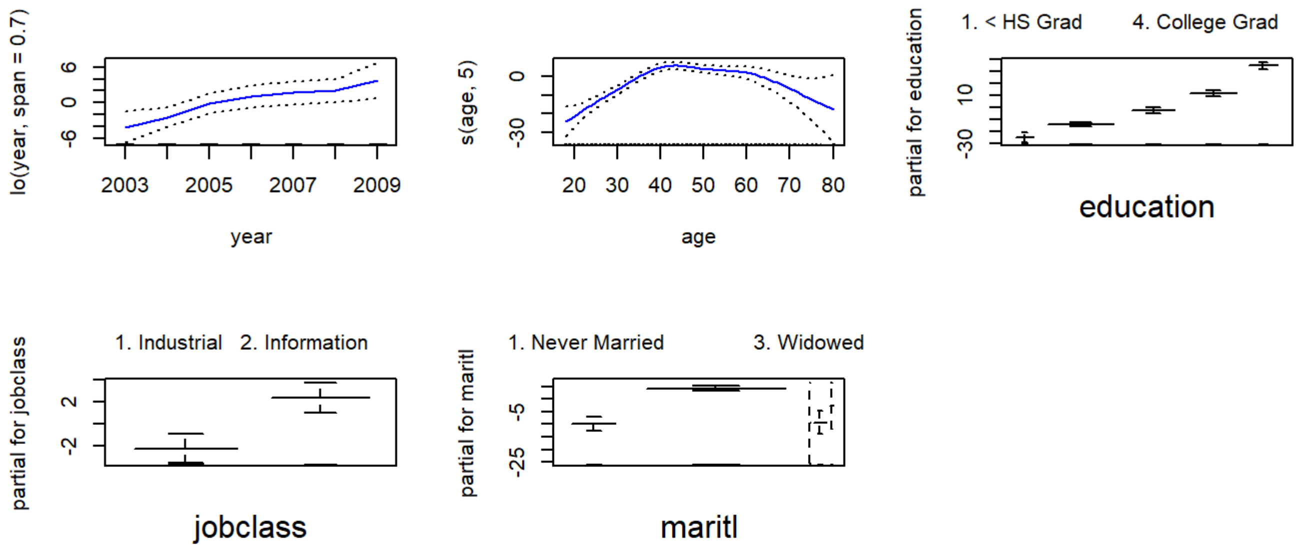
So, from the plots we conclude that a married couple earns more money on average, and also informational jobs earns more on average. We will now use **GAM** to predict *“wage”* using natural spline (ns) functions of *“year”*, *“age”*, *“education”*, *“jobclass”* and *“maritl”*.

```
library(gam)

fit770 <- gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education, data = Wage)
fit771 <- gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass, data = Wage)
fit772 <- gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education + maritl, data = Wage)
fit773 <- gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass + maritl, data = Wa
ge)
anova(fit770, fit771, fit772, fit773)
```

```
## Analysis of Deviance Table
##
## Model 1: wage ~ lo(year, span = 0.7) + s(age, 5) + education
## Model 2: wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass
## Model 3: wage ~ lo(year, span = 0.7) + s(age, 5) + education + maritl
## Model 4: wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass +
##     maritl
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1    2987.1    3691855
## 2    2986.1    3679689  1    12166 0.0014637 **
## 3    2983.1    3597526  3    82163  9.53e-15 ***
## 4    2982.1    3583675  1    13852 0.0006862 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the above table, we may conclude that the model “fit773” is significantly better.

```
par(mfrow = c(3, 3))
plot(fit773, se = T, col = "blue")
```



Q7.9

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.

Q7.9a

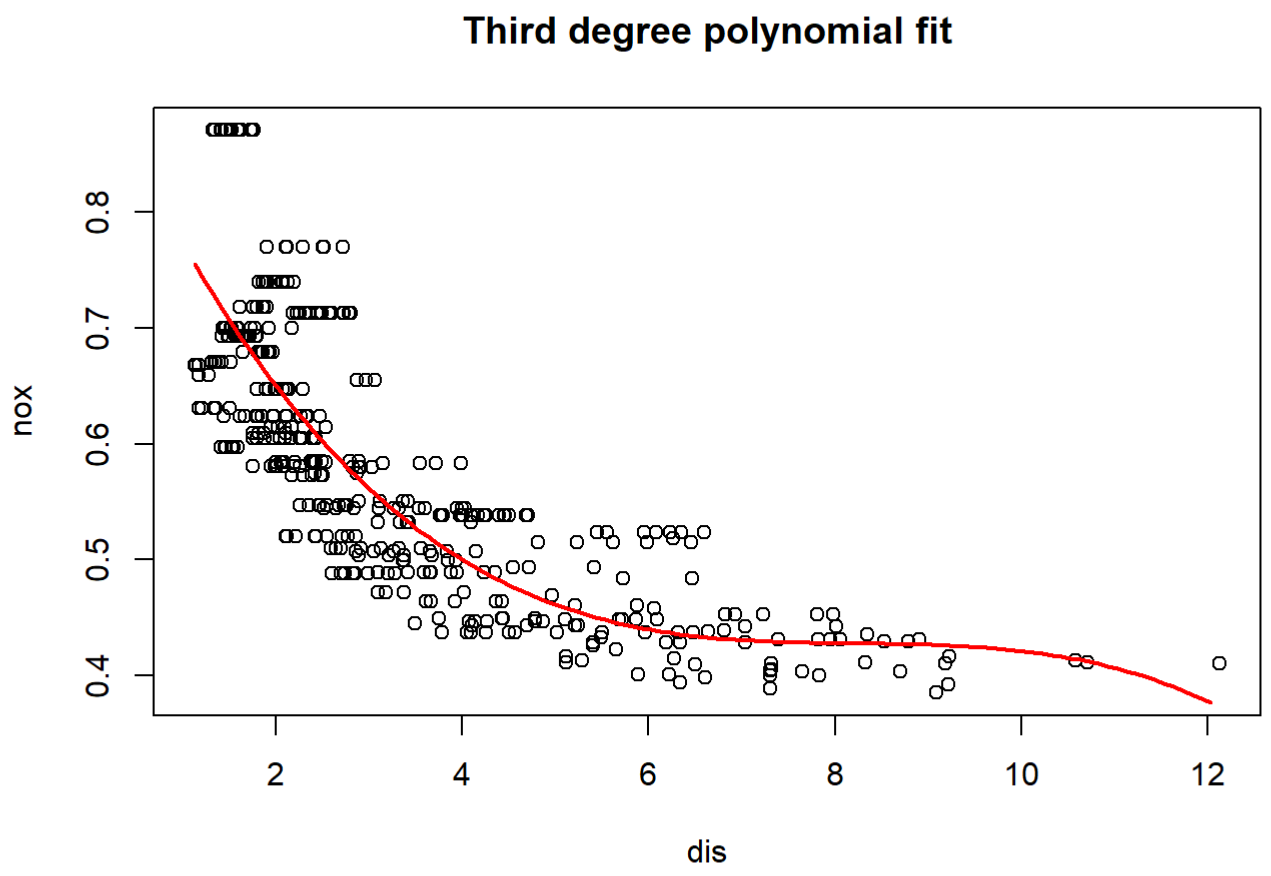
Use the **poly()** function to fit a cubic polynomial regression to predict **nox** using **dis**. Report the regression output, and plot the resulting data and polynomial fits.

Answer 7.9 (a)

```
set.seed(1)

fit791 <- lm(nox ~ poly(dis, 3), data = Boston)
summary(fit791)
```

```
## 
## 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
```

```
dislims <- range(Boston$dis)
dis.grid <- seq(from = dislims[1], to = dislims[2], by = 0.1)

preds <- predict(fit791, list(dis = dis.grid))

plot(nox ~ dis, data = Boston, main = "Third degree polynomial fit")

lines(dis.grid, preds, col = "red", lwd = 2)
```



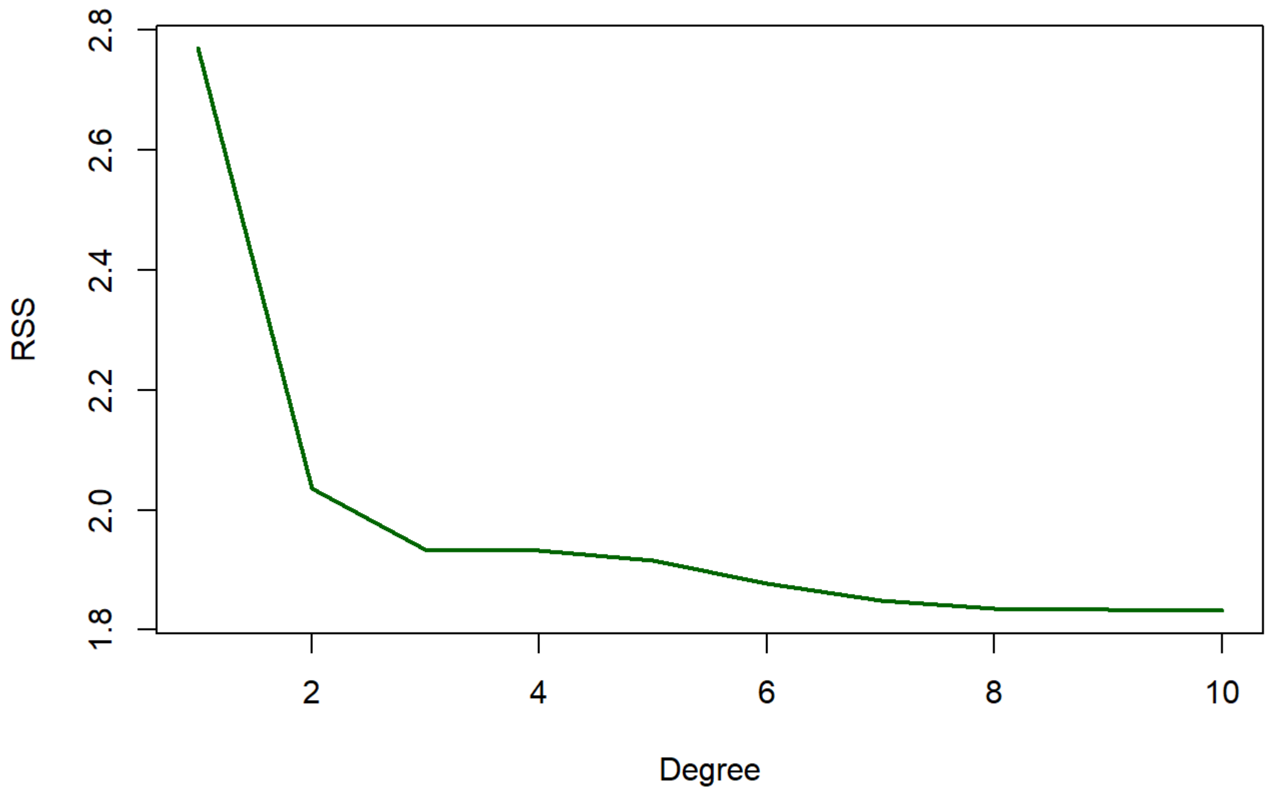
It can be concluded that all polynomial terms are significant.

Q7.9b

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

Answer 7.9 (b)

```
rss <- rep(NA, 10)
for (i in 1:10) {
    fit792 <- lm(nox ~ poly(dis, i), data = Boston)
    rss[i] <- sum(fit792$residuals^2)
}
plot(1:10, rss, pch=19,  col = "darkgreen", xlab = "Degree", ylab = "RSS", type = "l", lwd = 2)
```



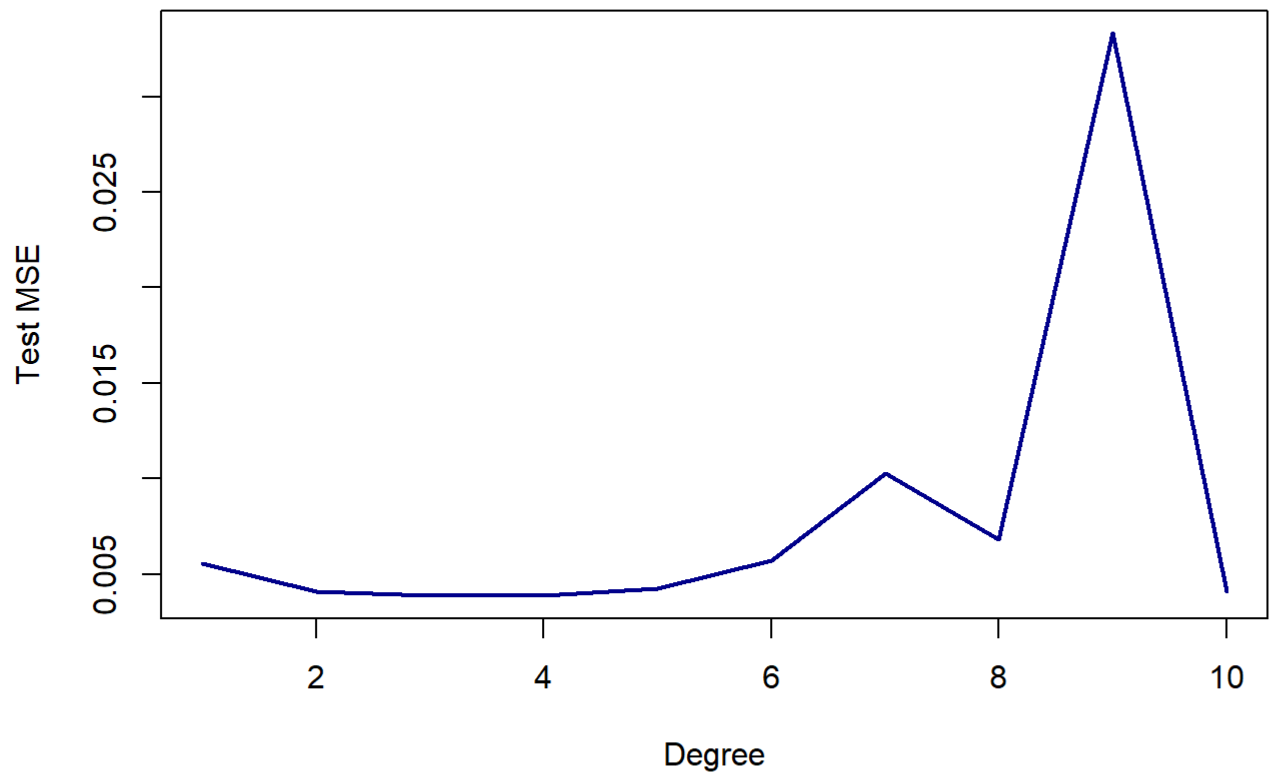
So, from the graph it seems that the model **RSS** decreases with the degree of the polynomial, and so it is minimum for a polynomial of degree 10.

Q7.9c

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

Answer 7.9 (c)

```
cvpoly <- rep(NA, 10)

for (i in 1:10) {
    fit793 <- glm(nox ~ poly(dis, i), data = Boston)
    cvpoly[i] <- cv.glm(Boston, fit793, K = 10)$delta[1]
}
plot(1:10, cvpoly, col = "darkblue", xlab = "Degree", ylab = "Test MSE", type = "l", lwd = 2)
```



It may be seen that a polynomial of degree 4 minimizes the test MSE

Q7.9d

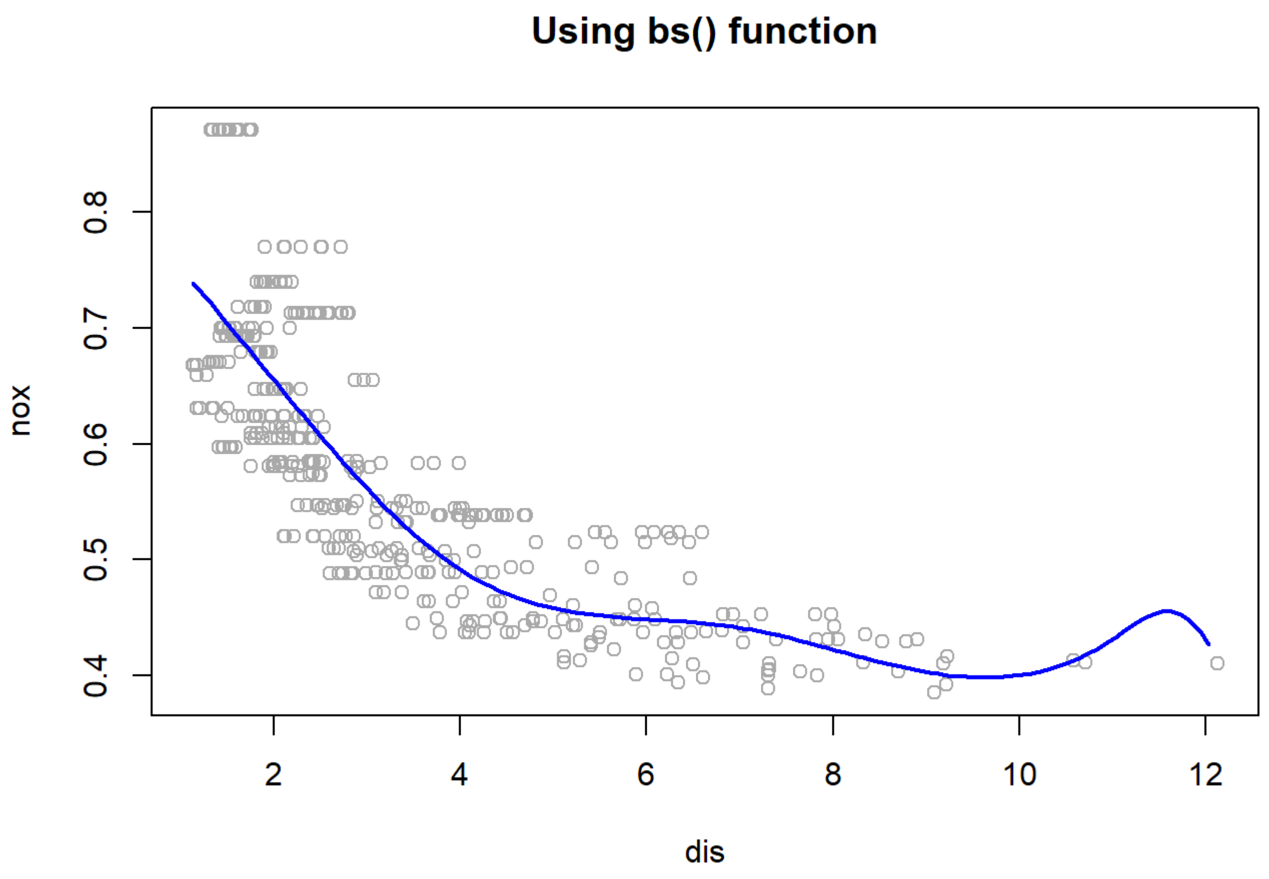
Use the **bs()** function to fit a regression spline to predict **nox** using **dis**. Report the output for the fit using four degrees of freedom. How did you choose the knots? Plot the resulting fit.

Answer 7.9 (d)

```
set.seed(1)

fit794 <- lm(nox ~ bs(dis, knots= c(4,7,11)), data = Boston)
summary(fit794)
```

```
##
## Call:
## lm(formula = nox ~ bs(dis, knots = c(4, 7, 11)), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.124567 -0.040355 -0.008702  0.024740  0.192920
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    0.73926    0.01331  55.537  < 2e-16 ***
## bs(dis, knots = c(4, 7, 11))1 -0.08861    0.02504  -3.539  0.00044 ***
## bs(dis, knots = c(4, 7, 11))2 -0.31341    0.01680 -18.658  < 2e-16 ***
## bs(dis, knots = c(4, 7, 11))3 -0.26618    0.03147  -8.459 3.00e-16 ***
## bs(dis, knots = c(4, 7, 11))4 -0.39802    0.04647  -8.565  < 2e-16 ***
## bs(dis, knots = c(4, 7, 11))5 -0.25681    0.09001  -2.853  0.00451 **
## bs(dis, knots = c(4, 7, 11))6 -0.32926    0.06327  -5.204 2.85e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06185 on 499 degrees of freedom
## Multiple R-squared:  0.7185, Adjusted R-squared:  0.7151
## F-statistic: 212.3 on 6 and 499 DF,  p-value: < 2.2e-16
```

```
dislims <- range(Boston$dis)
dis.grid <- seq(from = dislims[1], to = dislims[2], by = 0.1)

preds <- predict(fit794, list(dis = dis.grid))

plot(nox ~ dis, data = Boston, main = "Using bs() function", col = "darkgrey")

lines(dis.grid, preds, col = "blue", lwd = 2)
```



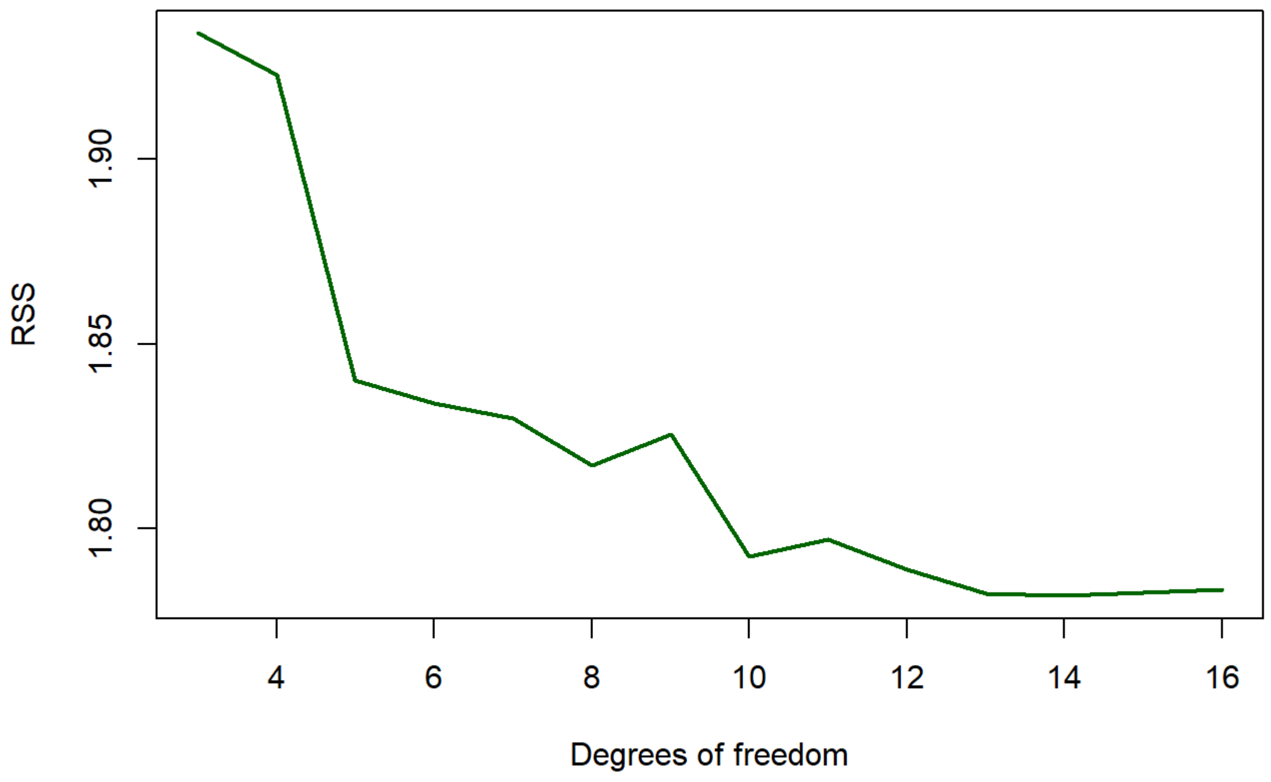
We may want to conclude that all terms in the spline fit are significant.

Q7.9e

Now fit a regression spline for a range of degrees of freedom, and plot the resulting fits and report the resulting RSS. Describe the results obtained.

Answer 7.9 (e)

```
splinerss <- rep(NA, 16)
for (i in 3:16) {
    fit795 <- lm(nox ~ bs(dis, df = i), data = Boston)
    splinerss[i] <- sum(fit795$residuals^2)
}
plot(3:16, splinerss[-c(1, 2)], xlab = "Degrees of freedom", ylab = "RSS", type = "l", col = "da
rkgreen", lwd = 2)
```



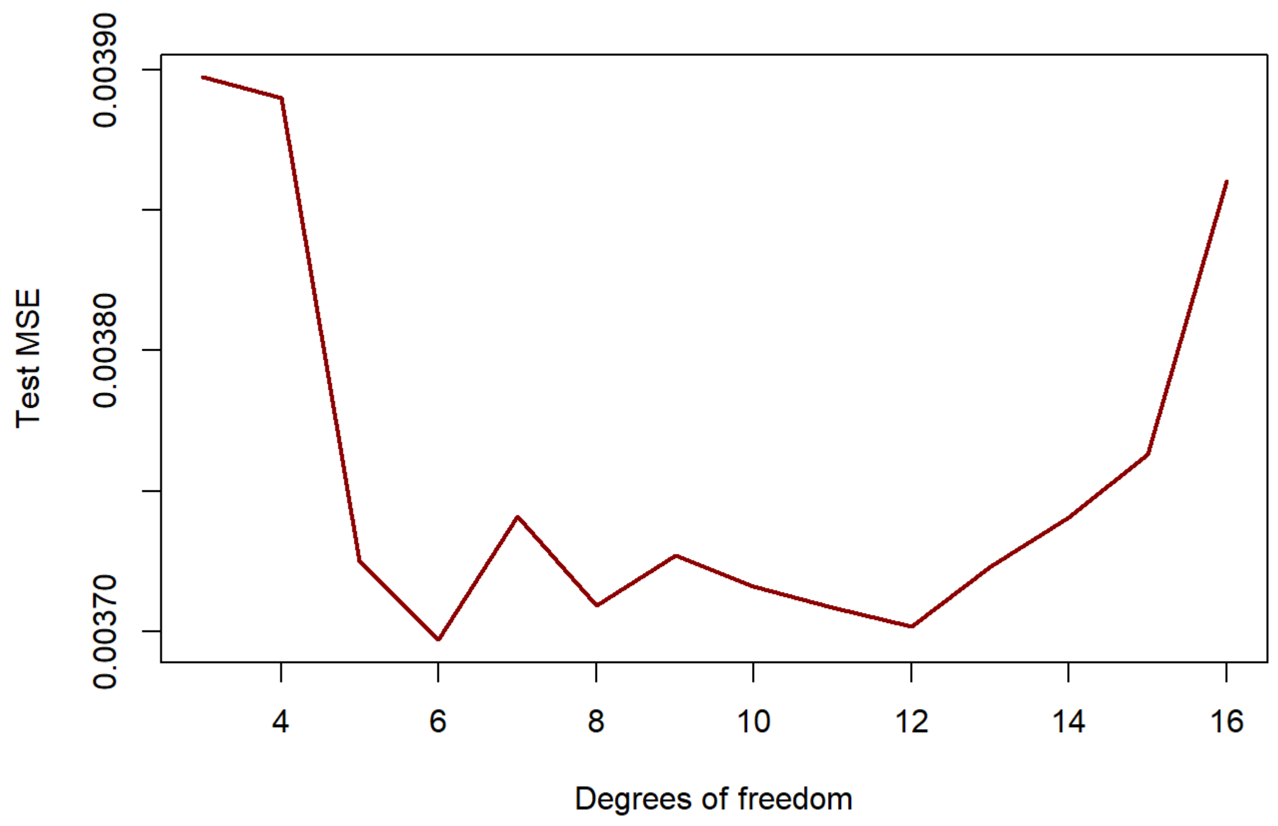
It may be seen that **Spline RSS** decreases until 14 and then slightly increases after that.

Q7.9f

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.

Answer 7.9 (f)

```
cvspline <- rep(NA, 16)
for (i in 3:16) {
    fit796 <- glm(nox ~ bs(dis, df = i), data = Boston)
    cvspline[i] <- cv.glm(Boston, fit796, K = 10)$delta[1]
}

plot(3:16, cvspline[-c(1, 2)], xlab = "Degrees of freedom", ylab = "Test MSE", type = "l", col =
"darkred", lwd = 2)
```



It may be seen that **Test MSE** is minimum for **6** degrees of freedom.

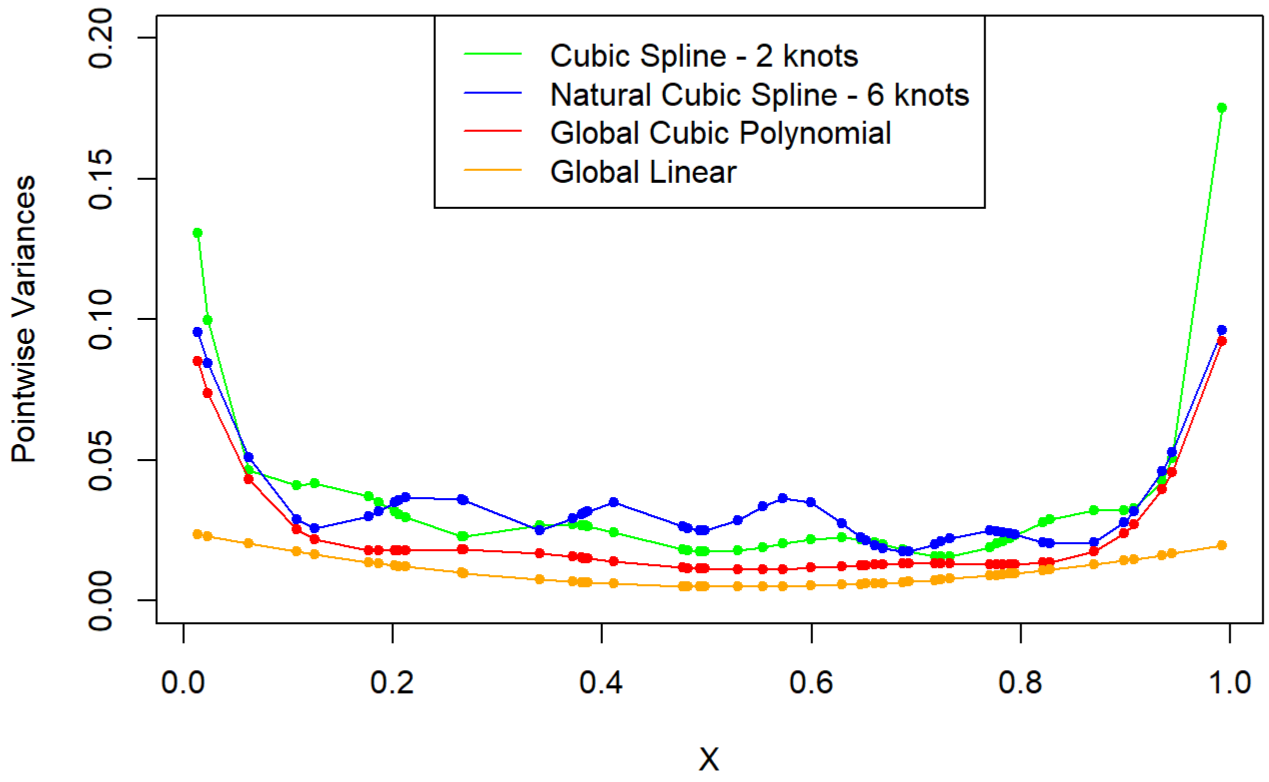
Chapter 5 - E-Book - The Elements of Statistical Learning - Chapter 5.3 - Exercises page 183:

Q 5.3

Write a program to reproduce Figure 5.3 on page 145.

Answer 5.3

```
set.seed(1)
x<-runif(50,0,1)
x<-sort(x)
Y<-x + rnorm(50,0,1)


# Linear Model

data53 <- data.frame(x,y)

#Global Linear
fit.linear = lm(y~x,data = data53)
d.linear = model.matrix(fit.linear)
fit.linear.var = d.linear %*% vcov(fit.linear) %*% t(d.linear)
var4 = diag(fit.linear.var)


#Cubic Spline - 2 knots
fit.bs2 = lm(y~bs(x,knots = c(0.33,0.66)), data = data53)
d.bs2 = model.matrix(fit.bs2)
fit.bs2.var = d.bs2 %*% vcov(fit.bs2) %*% t(d.bs2)
var1 = diag(fit.bs2.var)


#Global Cubic Polynomial of order 3
fit.cubic3 = lm(y~poly(x,3),data = data53)
d.cubic3 = model.matrix(fit.cubic3)
fit.cubic3.var = d.cubic3 %*% vcov(fit.cubic3) %*% t(d.cubic3)
var3 = diag(fit.cubic3.var)


#Natural Cubic Spline - 6 knots
knot6.spacing = (.9-.1)/(4+1)
ns.knots6 = seq.default(from = .1, to = .9, by = knot6.spacing)
fit.ns6 = lm(y~ns(x,knots = ns.knots6[-c(1,6)], Boundary.knots = c(0.1,0.9) ), data = data53)
d.ns6 = model.matrix(fit.ns6)
fit.ns6.var = d.ns6 %*% vcov(fit.ns6) %*% t(d.ns6)
var2 = diag(fit.ns6.var)


#par(mar = c(4,4,0.5,0.5))
plot(x,var1,pch=20,type='l',ylim = c(0,0.2),col="green",
     xlab = "X",ylab = "Pointwise Variances")
points(x,var1,pch = 20, col = "green")
lines(x,var2,pch=20,type='l',col="blue")
points(x,var2,pch = 20, col = "blue")
lines(x,var3,pch=20,type='l',col="red")
points(x,var3,pch = 20, col = "red")
lines(x,var4,pch=20,type='l',col="orange")
points(x,var4,pch = 20, col = "orange")
legend("top",legend=c("Cubic Spline - 2 knots","Natural Cubic Spline - 6 knots","Global Cubic Po
lynomial","Global Linear"),
       col=c("green","blue","red","orange"),lty=1)
```


Cubic Spline - 2 knots
Natural Cubic Spline - 6 knots
Global Cubic Polynomial
Global Linear
Pointwise Variances
X