ISLR_CH7

Sky Liu

2/28/2019

7.9.3

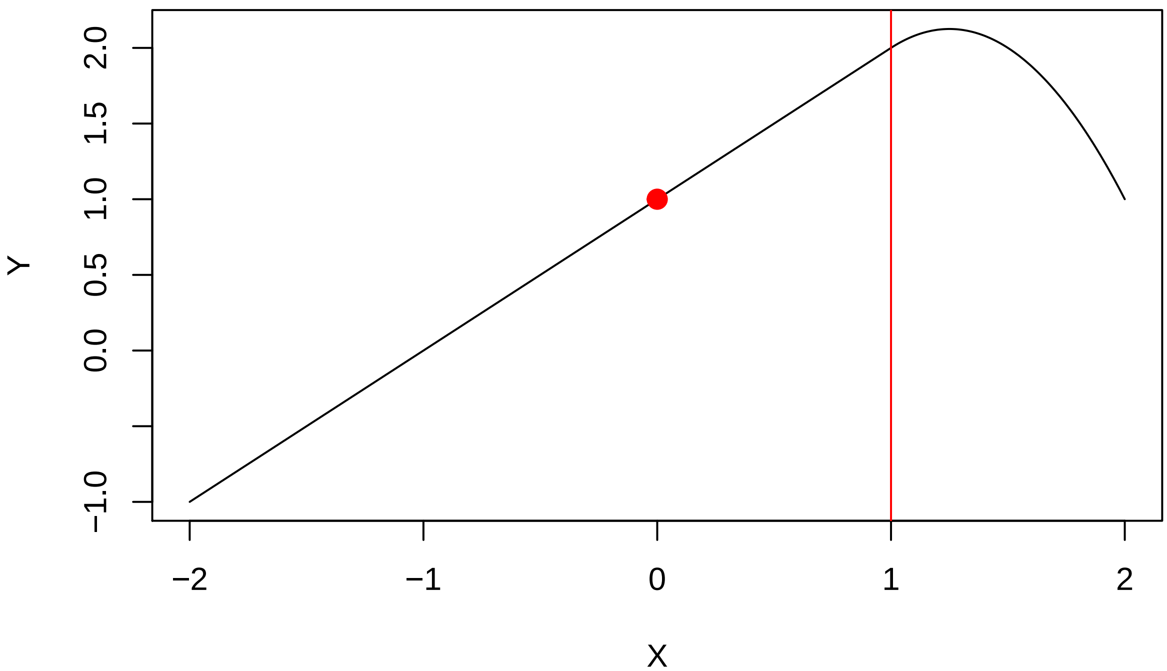
Given $\hat{\beta}_0 = 1, \hat{\beta}_1 = 1$ and $\hat{\beta}_2 = -2$, we obtain:

$$\hat{Y} = 1 + X - 2(X-1)^2 I(X \geq 1) = \begin{cases} 1 + X, X < 1 \\ -1 + 5X - 2X^2, X \geq 1 \end{cases}$$

```
X <- seq(from = -2, to = 2, length.out = 500)

Y <- 1 + X - 2 * (X - 1)^2 * (X >= 1)

plot(X, Y, type = "l");abline(v = 1, col = "red");points(0, 1,  col = "red", cex = 2, pch = 20)
```



The intercept is 1. When $X < 1$, the slope is 1, while when $X \geq 1$, the slope is $5 - 10X$.

7.9.9

part a

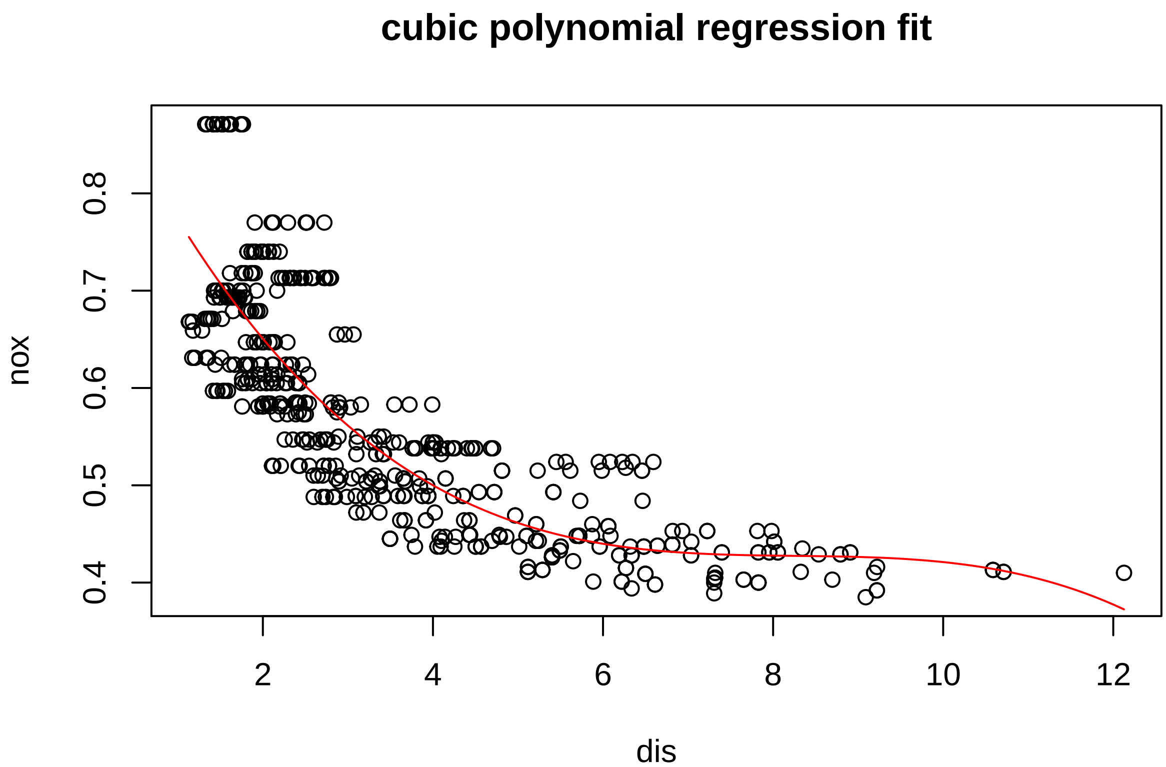
```
data("Boston")

set.seed(799)
lm1 = lm(nox ~ poly(dis, 3), data = Boston)
summary(lm1)
```

```
##
## 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
```

```r
dislims <- range(Boston$dis)
dis.grid <- seq(dislims[1], dislims[2], 0.1)
dis_range <- range(Boston$dis)
dis_samples <- seq(from = dis_range[1], to = dis_range[2], length.out = 100)
y_hat <- predict(lm1, newdata = list(dis = dis_samples))

plot(Boston$dis, Boston$nox, xlab = "dis", ylab = "nox", main = "cubic polynomial regression fit");line
```



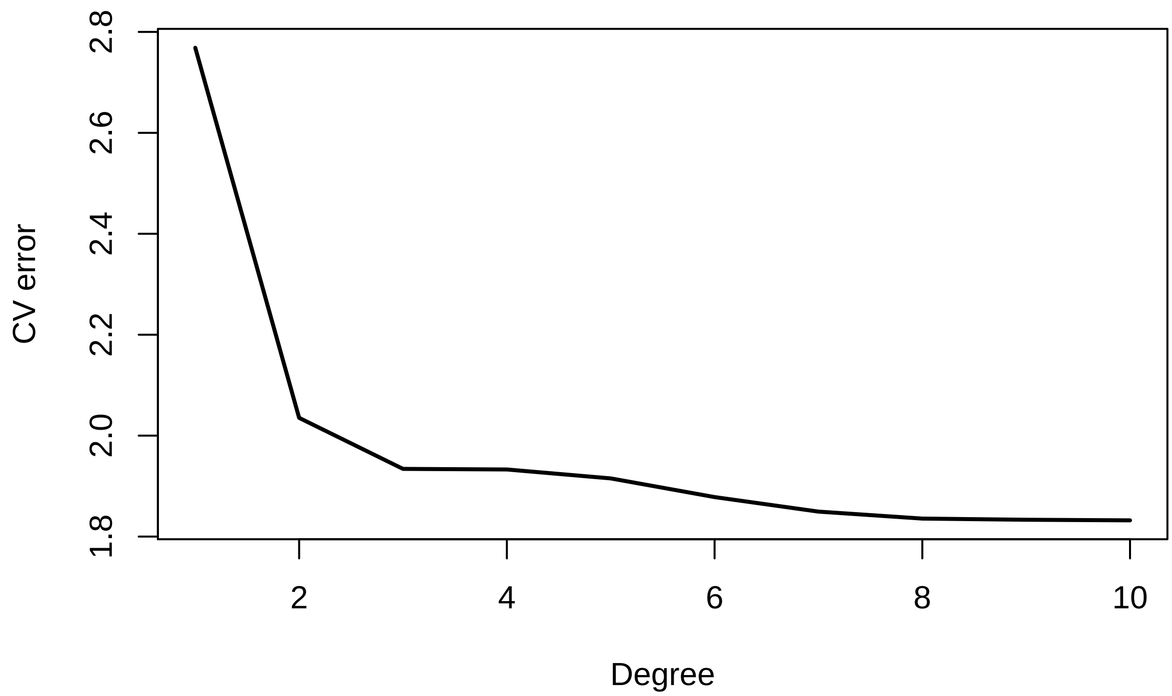
The plot shows that the curve looks like a good fit. From the model output we could also see that all three polynomial terms are significant.

part b

```
rss = rep(NA, 10)
for (i in 1:10) {
    lm1 = lm(nox ~ poly(dis, i), data = Boston)
    rss[i] = sum(lm1$residuals^2)
}
rss
```

```
##  [1] 2.768563 2.035262 1.934107 1.932981 1.915290 1.878257 1.849484
##  [8] 1.835630 1.833331 1.832171
```

```
plot(1:10, rss, xlab = "Degree", ylab = "CV error", type = "l", pch = 20,
    lwd = 2)
```



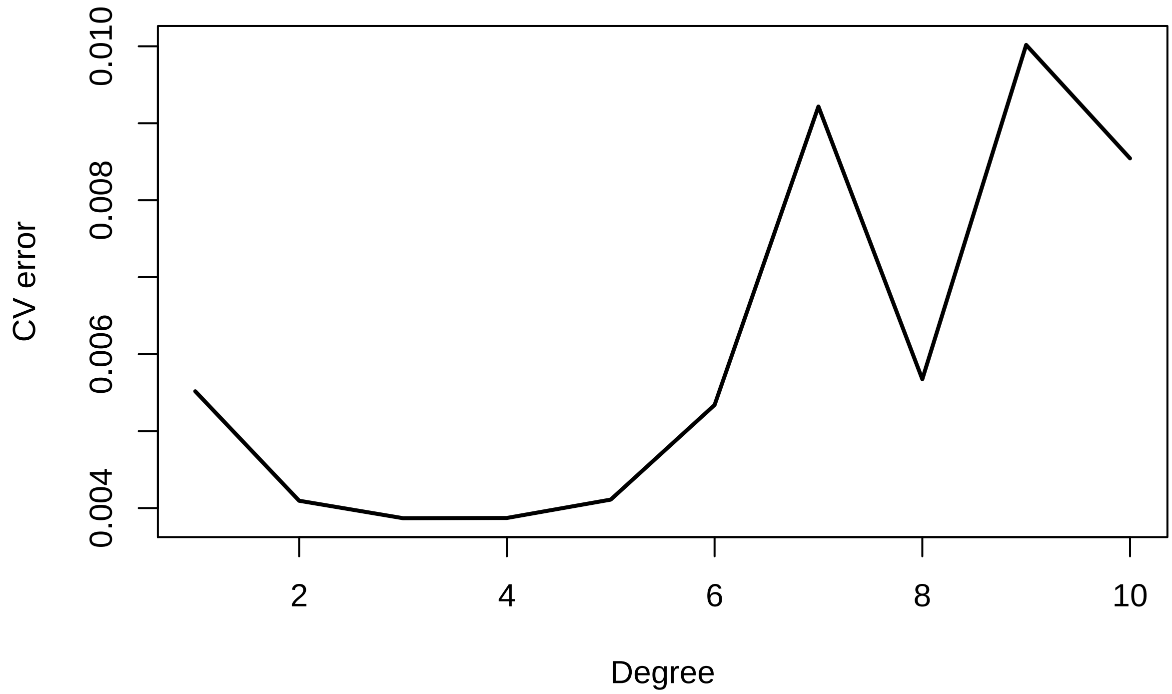
The plot shows RSS decreases as the degree of polynomial increases.

part c

```
set.seed(799)
cv.e = rep(NA, 10)
for (i in 1:10) {
    lm2 = glm(nox ~ poly(dis, i), data = Boston)
    cv.e[i] = cv.glm(Boston, lm2, K = 10)$delta[2]
}
cv.e
```

```
##  [1] 0.005517152 0.004095266 0.003868555 0.003871114 0.004109643
##  [6] 0.005339388 0.009217303 0.005674642 0.010017495 0.008543497
```

```r
plot(1:10, cv.e, xlab = "Degree", ylab = "CV error", type = "l", pch = 20,
    lwd = 2)
```



The plot shows 4 is the best polynomial degree.

part d

```r
lm3 <-lm(nox ~ bs(dis,df = 4),data = Boston)
summary(lm3)
```

```
##
## 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
```

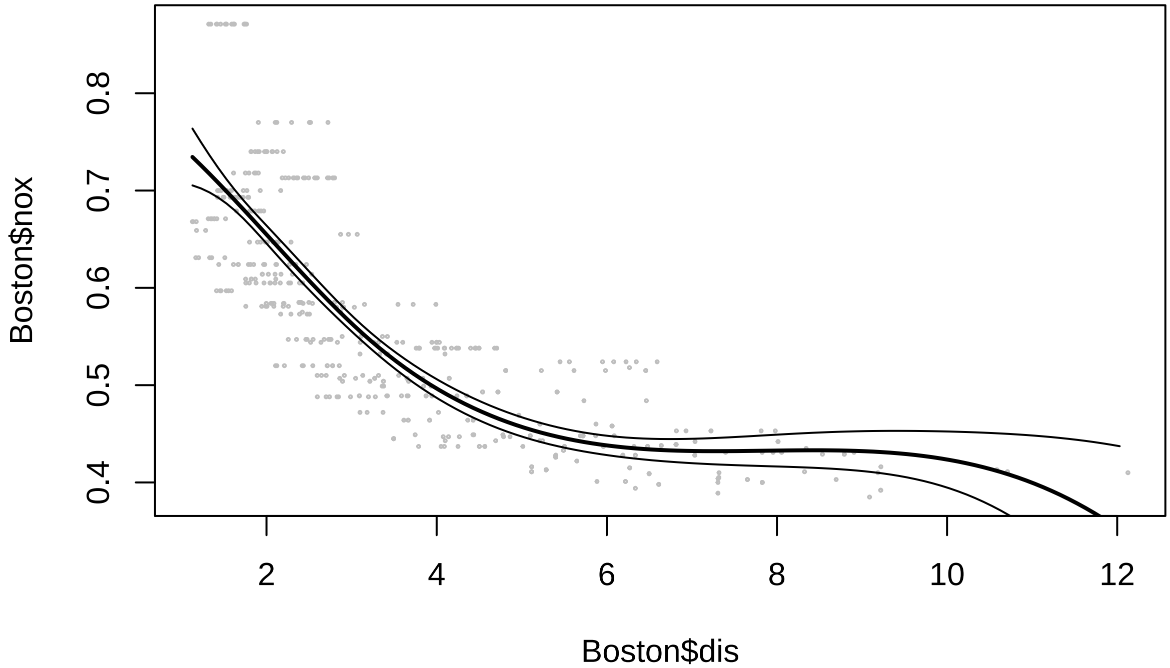
```r
dim(bs(Boston$dis,df = 4))
```

```
## [1] 506   4
```

```r
attr(bs(Boston$dis,df = 4),"knots")
```

```
##     50%
## 3.20745
```

```r
pred2 <- predict(lm3, newdata = list(dis = dis.grid), se = T)
plot(x = Boston$dis, y = Boston$nox, cex = 0.2, col = "grey");lines(dis.grid,pred2$fit, lwd = 2);lines(d
```

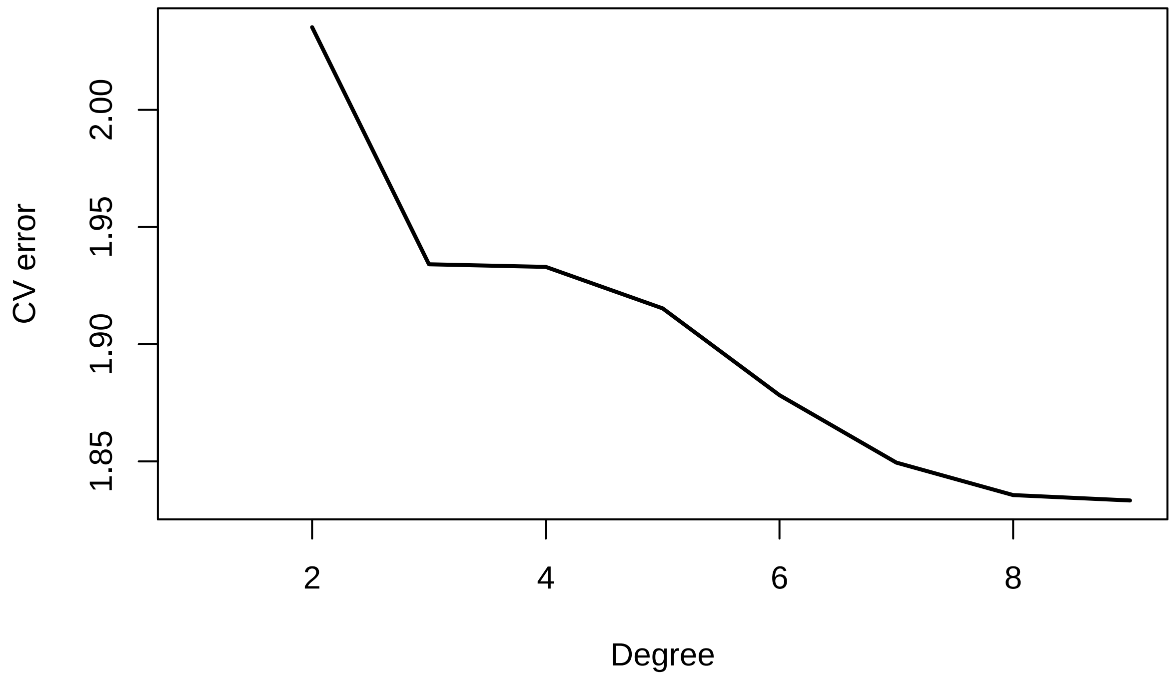


part e

```r
set.seed(7995)
rss = rep(NA, 7)
for (i in 2:9) {
    lm1 = lm(nox ~ poly(dis, i), data = Boston)
    rss[i] = sum(lm1$residuals^2)
}
rss
```

```
## [1]       NA 2.035262 1.934107 1.932981 1.915290 1.878257 1.849484 1.835630
## [9] 1.833331
```

```r
plot(1:9, rss, xlab = "Degree", ylab = "CV error", type = "l", pch = 20,
    lwd = 2)
```



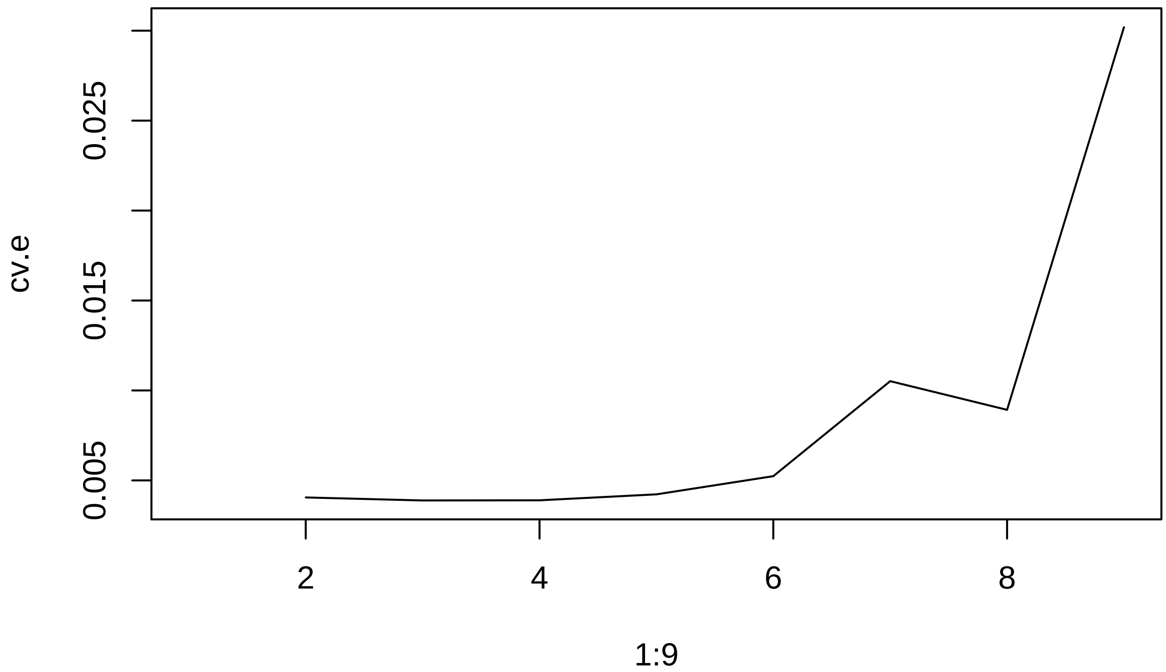
The plot shows RSS decreases as the degree of polynomial increases.

part f

```
cv.e = rep(NA, 7)
for (i in 2:9) {
    lm1 = glm(nox ~ poly(dis, i), data = Boston)
    cv.e[i] = cv.glm(Boston, lm1, K = 10)$delta[2]
}
cv.e
```

```
## [1]          NA 0.004053681 0.003885955 0.003893759 0.004225818 0.005235671
## [7] 0.010517236 0.008923080 0.030193322
```

```
plot(1:9, cv.e,type = "l")
```

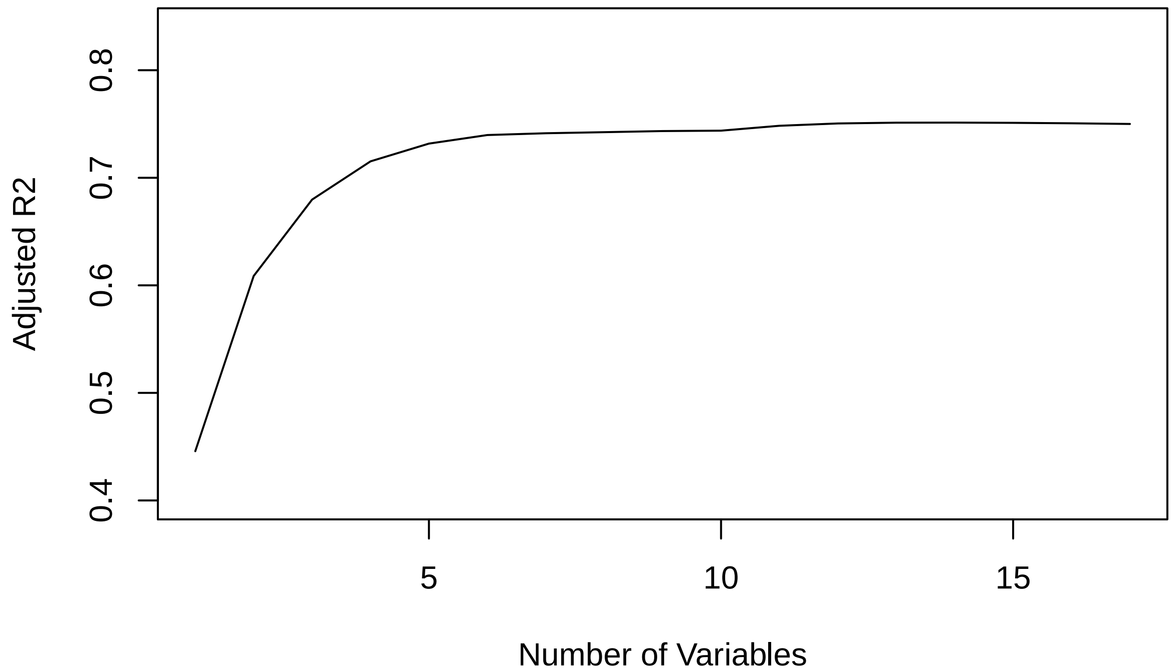


When degree of freedom is 4, we obtain the min cv error.

7.9.10

part a

```
attach(College)
train <- sample(1:nrow(College), nrow(College)/2)
train.c <- College[train,]
test.c <- College[-train,]
f1_reg <- regsubsets(Outstate~., data = train.c, nvmax = 17, method = "forward")
f1_summary <- summary(f1_reg)
plot(f1_summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted R2",
    type = "l", ylim = c(0.4, 0.84))
```



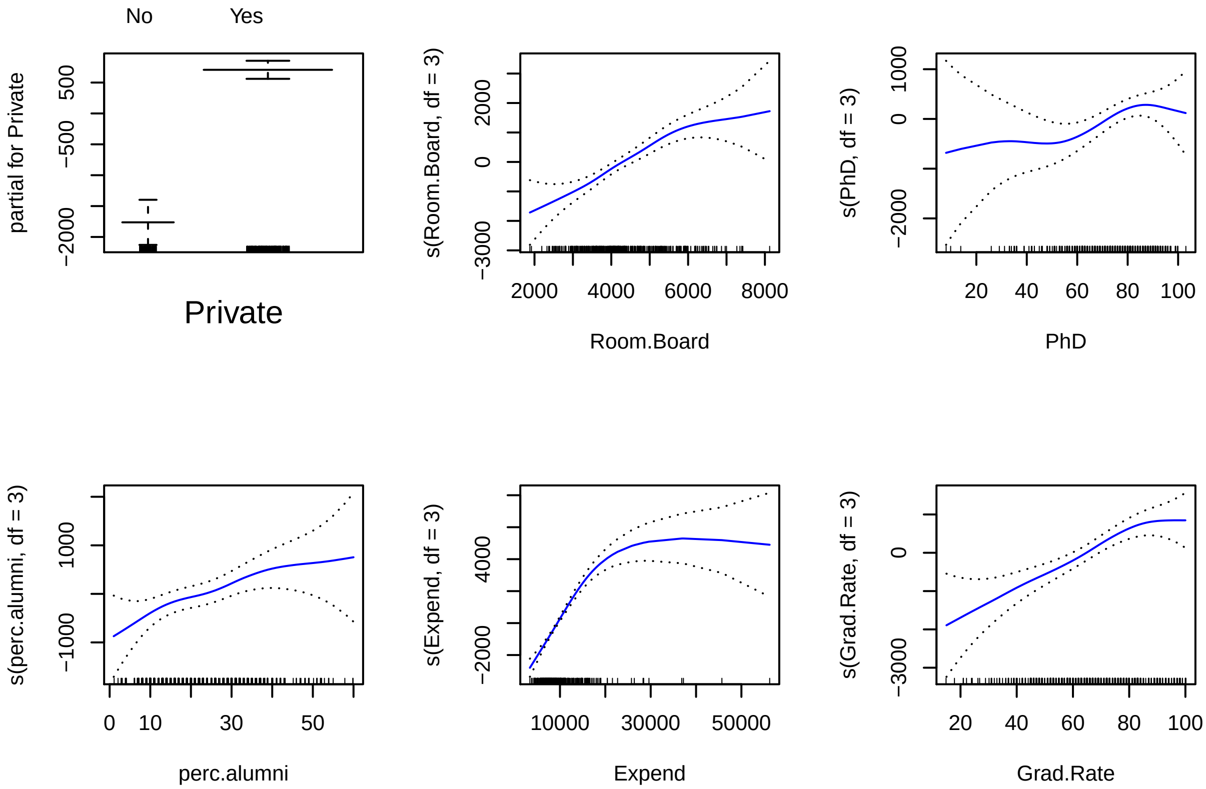
The model with 6 variables seems to be the best fit. The fitting terms are:

```
f1 = regsubsets(Outstate ~ ., data = College, method = "forward")
coefi = coef(f1, id = 6)
names(coefi)
```

```
## [1] "(Intercept)" "PrivateYes"  "Room.Board"  "PhD"         "perc.alumni"
## [6] "Expend"      "Grad.Rate"
```

part b

```
f1_gam = gam(Outstate ~ Private + s(Room.Board, df = 3) + s(PhD, df = 3) +
    s(perc.alumni, df = 3) + s(Expend, df = 3) + s(Grad.Rate, df = 3), data = train.c)
par(mfrow = c(2, 3))
plot(f1_gam, se = T, col = "blue")
```



part c

```r
pre_gam <- predict(f1_gam, newdata = test.c)
er_gam <- mean((test.c$Outstate - pre_gam)^2)
SS_tot <- mean((test.c$Outstate - mean(test.c$Outstate))^2)
rss <- 1- er_gam/SS_tot
rss
```

```
## [1] 0.7766641
```

Using GAM, we obtained r square 0.80, which is better than regression fit.

part d

```r
summary(f1_gam)
```

```
##
## Call: gam(formula = Outstate ~ Private + s(Room.Board, df = 3) + s(PhD,
##     df = 3) + s(perc.alumni, df = 3) + s(Expend, df = 3) + s(Grad.Rate,
##     df = 3), data = train.c)
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -6961.4 -1062.2   -49.1  1143.1  8215.3
##
## (Dispersion Parameter for gaussian family taken to be 3322191)
##
##     Null Deviance: 5881471368 on 387 degrees of freedom
```

```
## Residual Deviance: 1232531617 on 370.9997 degrees of freedom
## AIC: 6945.973
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                         Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private                  1 1618339297 1618339297 487.130 < 2.2e-16 ***
## s(Room.Board, df = 3)    1 1187687929 1187687929 357.501 < 2.2e-16 ***
## s(PhD, df = 3)           1  357668038  357668038 107.660 < 2.2e-16 ***
## s(perc.alumni, df = 3)   1  224956218  224956218  67.713 3.243e-15 ***
## s(Expend, df = 3)        1  539220737  539220737 162.309 < 2.2e-16 ***
## s(Grad.Rate, df = 3)     1   96789142   96789142  29.134 1.207e-07 ***
## Residuals              371 1232531617    3322191
## ---
## 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(Room.Board, df = 3)        2  2.021   0.13399
## s(PhD, df = 3)               2  2.426   0.08976 .
## s(perc.alumni, df = 3)       2  0.978   0.37692
## s(Expend, df = 3)            2 33.484 4.285e-14 ***
## s(Grad.Rate, df = 3)         2  1.647   0.19406
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

7.9.11

part a

```
set.seed(7911)
X1 = rnorm(100)
X2 = rnorm(100)
eps = rnorm(100, sd = 0.1)
Y = -2.1 + 1.3 * X1 + 0.54 * X2 + eps
```

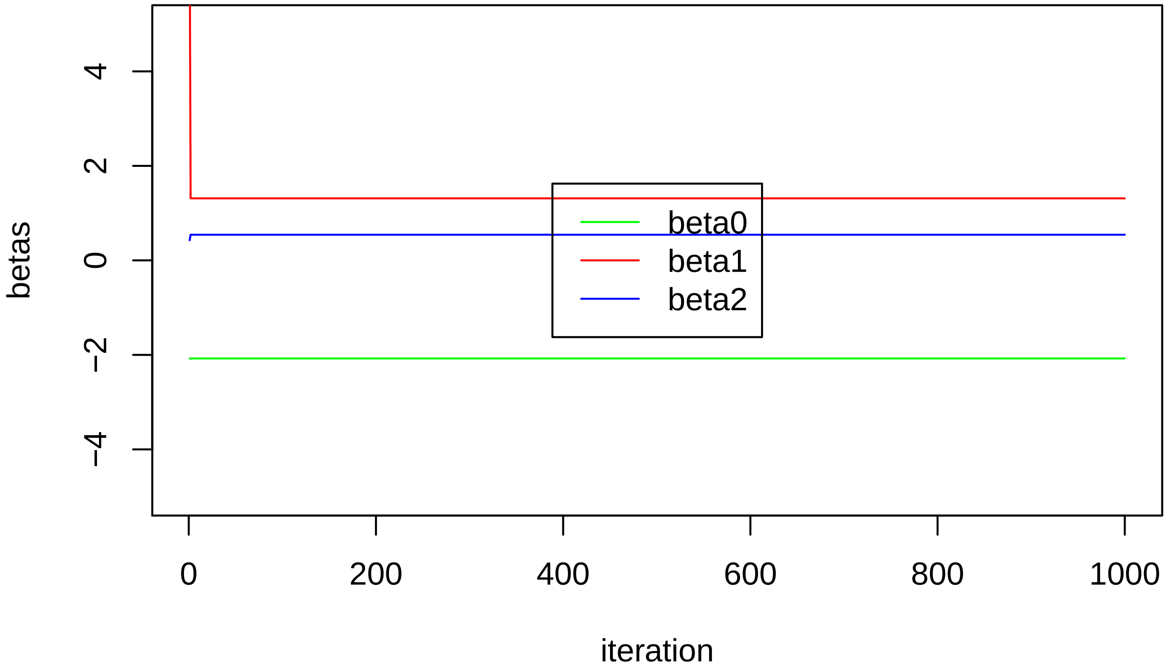
part b

```
beta0 <- rep(NA,1000)
beta1 <- rep(NA,1000)
beta2 <- rep(NA,1000)
beta1[1] <- 7
```

part c-e

```
for(i in 1:1000){
  a <- Y - beta1[i]*X1
```

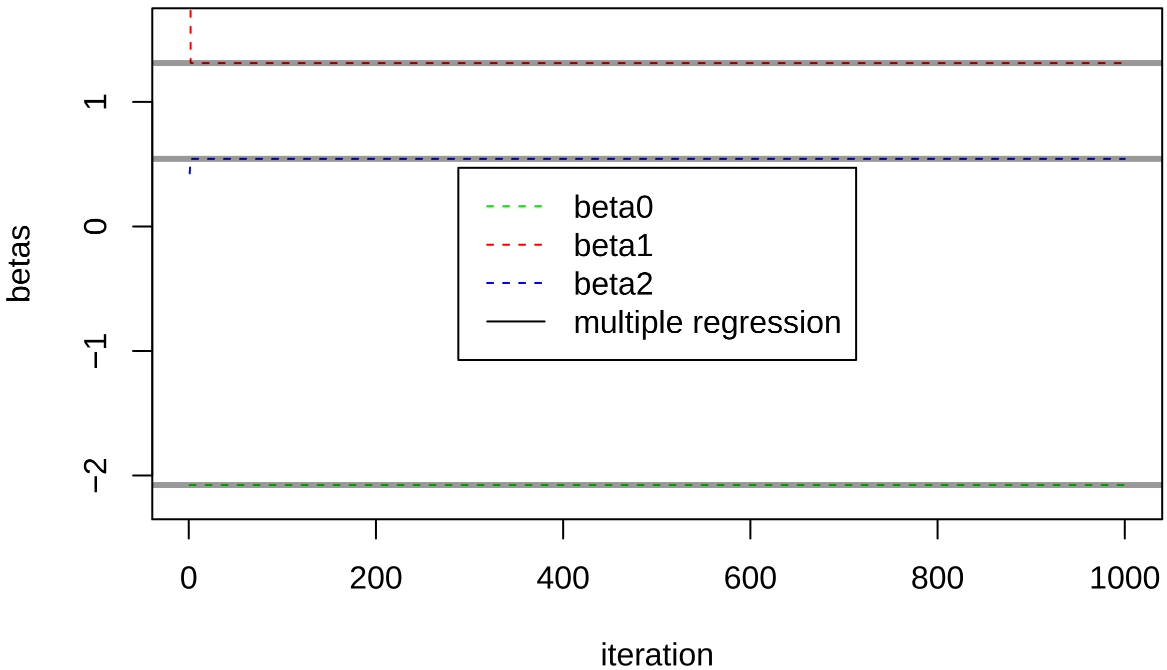
```
  lm4<- lm(a~X2)
  beta2[i] <- lm4$coeff[2]
  b <- Y - beta2[i]*X2
  fm5 <- lm(b~X1)
  if(i < 1000){
  beta1[i+1] <- fm5$coef[2]
  }
  beta0[i] <- fm5$coef[1]
}
plot(1:1000, beta0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(-5, 5), col = "green");lin
```



After one iteration, we obtain beta0 is -2.075059 beta1 is 1.311948 beta2 is 0.5428701

part f

```
lm.fit = lm(Y ~ X1 + X2)
plot(1:1000, beta0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(-2.2,
    1.6), col = "green",lty = "dashed",)
lines(1:1000, beta1, col = "red",lty = "dashed",)
lines(1:1000, beta2, col = "blue",lty = "dashed",)
abline(h = lm.fit$coef[1],  lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))
abline(h = lm.fit$coef[2],  lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))
abline(h = lm.fit$coef[3],  lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))
legend("center", c("beta0", "beta1", "beta2", "multiple regression"), lty = c(2,
    2, 2, 1), col = c("green", "red", "blue", "black"))
```



The estimated multiple regression coefficients are shown with black lines, which match with the result from part e

part g

When Y and X are linearly related, one backfitting iterations is required in order to obtain a ???good??? approximation to the multiple re- gression coefficient estimates.