STATS 202 | HW:6 | Sagar Ganapaneni | SUID# 06167633

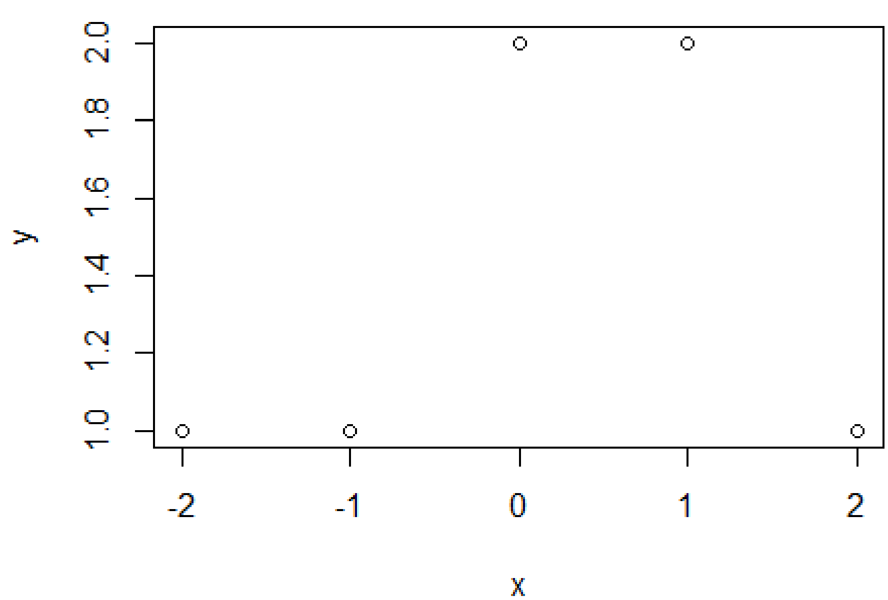
Problem: 1

Suppose we fit a curve with basis functions
b1(X)=I(0≤X≤2)−(X−1)I(1≤X≤2)b1(X)=I(0≤X≤2)−(X−1)I(1≤X≤2),
b2(X)=(X−3)I(3≤X≤4)+I(4≤X≤5)b2(X)=(X−3)I(3≤X≤4)+I(4≤X≤5).

We fit the linear regression model: Y=β0+β1b1(X)+β2b2(X)+ε,
and obtain coefficient estimates β^0=1, β^1=1 , β^2=3

Sketch the estimated curve between X=−2 and X=2. Note the intercepts, slopes, and other relevant information.

```r
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 -1 (y=1) (slope: zero)
- linear between -1 and 0 with y= 2+ x (slope 1, intercept: 2)
- constant between 0 and 1 (y=2) (slope: zero)
- linear between 1 and 2 with y= 3- x (slope: 1, intercept: 3)

Problem: 2

Consider two curves, $\hat{g}_1$ and $\hat{g}_2$, defined by

$$\hat{g}_1 = \arg\min_g \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g^{(3)}(x) \right]^2 dx \right),$$

$$\hat{g}_2 = \arg\min_g \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g^{(4)}(x) \right]^2 dx \right),$$

where $g^{(m)}$ represents the mth derivative of g.

(a) As $\lambda \to \infty$, will $\hat{g}_1$ or $\hat{g}_2$ have the smaller training RSS?

(b) As $\lambda \to \infty$, will $\hat{g}_1$ or $\hat{g}_2$ have the smaller test RSS?

(c) For $\lambda = 0$, will $\hat{g}_1$ or $\hat{g}_2$ have the smaller training and test RSS?

a. As λ→∞ will g^1 or g^2 have the smaller training RSS?

- As λ→∞, the weight of penalty term increases and with higher order of derivative in the penalty term, g^2 is more flexible compared to g^1. Hence the training error would be less for g^2

b. As λ→∞ will g^1 or g^2 have the smaller test RSS?

- As mentioned above we expect g^2 to be more flexible which can lead to model over fit on the training data. g^2 might have higher Test RSS compared to g^1. Hence g^1 likely to have low Test RSS value

c. For λ=0, will g^1 or g^2 have the smaller training and test RSS?

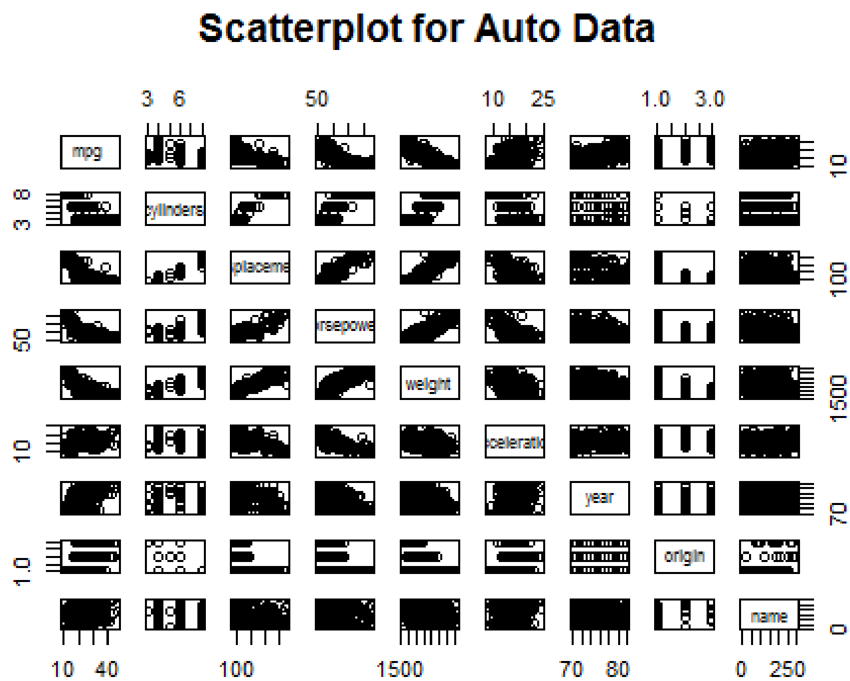
- With λ=0 ➔ g^1=g^2. Hence both will have same training and test RSS values.

Problem: 3

Find at least one non-linear estimate which does better than linear regression, and justify this using a t-test or by showing an improvement in the cross-validation error with respect to a linear model. You must also produce a plot of the predictor X vs. the non-linear estimate f^(X)

```
set.seed(12)
library(ISLR)

## Warning: package 'ISLR' was built under R version 3.2.5

library(glmnet)

## Warning: package 'glmnet' was built under R version 3.2.5

## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-5

library(boot)
data(Auto)
summary(Auto)

##       mpg          cylinders      displacement     horsepower   
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0  
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0  
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5  
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5  
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0  
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0  
##                                                                 
##      weight      acceleration        year           origin     
##  Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000  
##  1st Qu.:2225   1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000  
##  Median :2804   Median :15.50   Median :76.00   Median :1.000  
##  Mean   :2978   Mean   :15.54   Mean   :75.98   Mean   :1.577  
##  3rd Qu.:3615   3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000  
##  Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000  
##                                                                 
##                  name    
##  amc matador       :  5  
##  ford pinto        :  5  
##  toyota corolla    :  5  
##  amc gremlin       :  4  
##  amc hornet        :  4  
##  chevrolet chevette:  4  
##  (Other)           :365  
```

```r
pairs(Auto, main='Scatterplot for Auto Data')
```

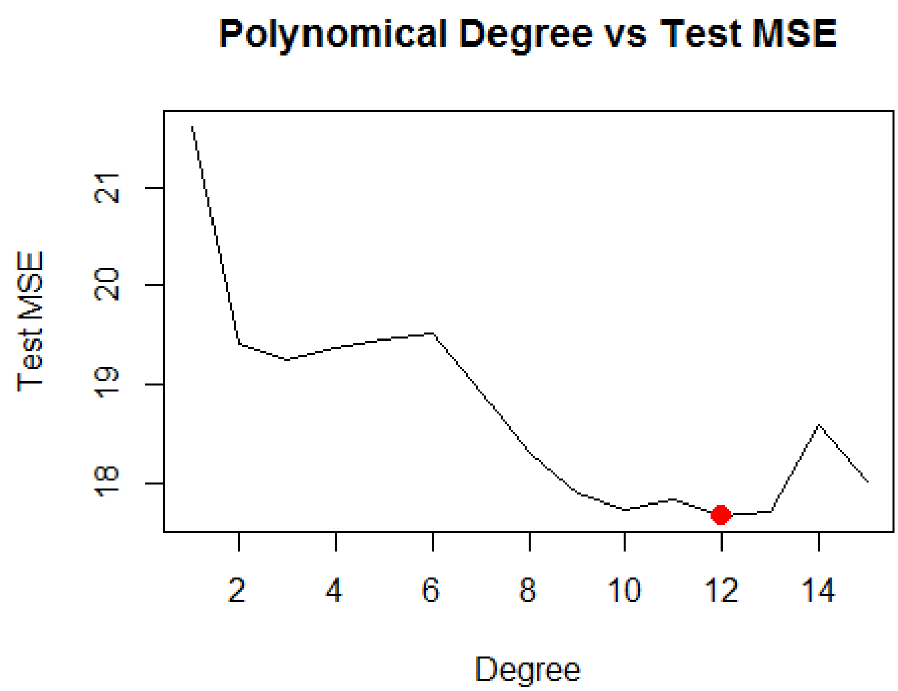


"mpg" is negatively correlated with "cylindes", "displacement", "horsepower" and "weight"

Let us start with Polynomial Models of mpg vs displacement

Polynomial function

```r
MSE <- rep(NA, 15)
for (i in 1:15) {
  fit <- glm(mpg ~ poly(displacement, i), data = Auto)
  MSE[i] <- cv.glm(Auto, fit, K = 10)$delta[1]
}
plot(1:15, MSE, xlab = "Degree", ylab = "Test MSE", type = "l", main = "Polynomical Degree vs Test MSE")
points(which.min(MSE), MSE[which.min(MSE)], col = "red", cex = 2, pch = 20)
```



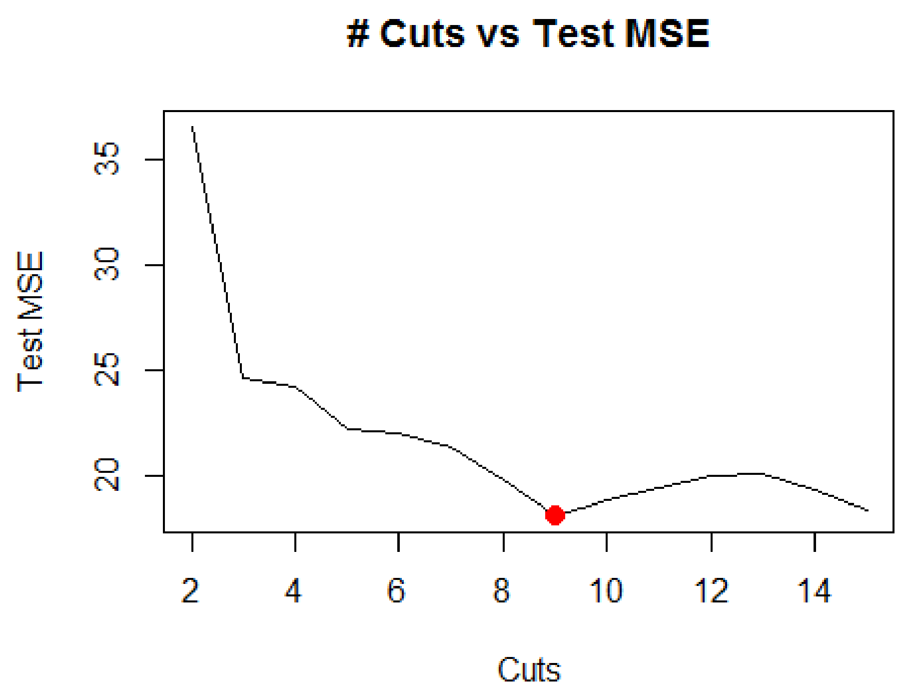
The optimal degree for the polynomial here is 12th degree with CV TEST MSE:

```
MSE[which.min(MSE)]

## [1] 17.66336
```

So we need to find a nonlinear model with Test MSE less than 17.66336

Step function

```
MSE <- rep(NA, 15)
for (i in 2:15) {
   Auto$dis_cut <- cut(Auto$displacement, i)
   fit <- glm(mpg ~ dis_cut, data = Auto)
   MSE[i] <- cv.glm(Auto, fit, K = 10)$delta[1]
}
plot(2:15, MSE[-1], xlab = "Cuts", ylab = "Test MSE", type = "l", main = "# Cuts vs Test MSE")
points(which.min(MSE), MSE[which.min(MSE)], col = "red", cex = 2, pch = 20)
```

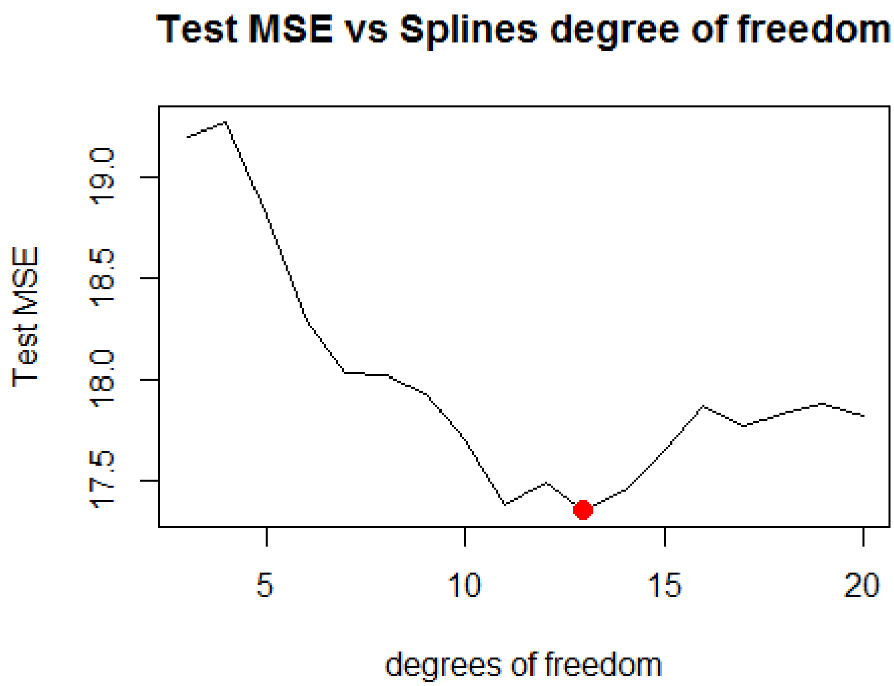


We may see that the error is minimum for 9 cuts with CV TEST MSE

```
MSE[which.min(MSE)]

## [1] 18.12879
```

Spline functions

```
library(splines)
MSE <- rep(NA, 20)
for (i in 3:20) {
  fit <- glm(mpg ~ ns(displacement, df = i), data = Auto)
  MSE[i] <- cv.glm(Auto, fit, K = 10)$delta[1]
}
plot(3:20, MSE[-c(1, 2)], xlab = "degrees of freedom", ylab = "Test MSE", type = "l", main="Test MSE vs Splines d
egree of freedom")
d.min <- which.min(MSE)
points(which.min(MSE), MSE[which.min(MSE)], col = "red", cex = 2, pch = 20)
```



We may see that the error is minimum for 13 degrees of freedom with CV TEST MSE

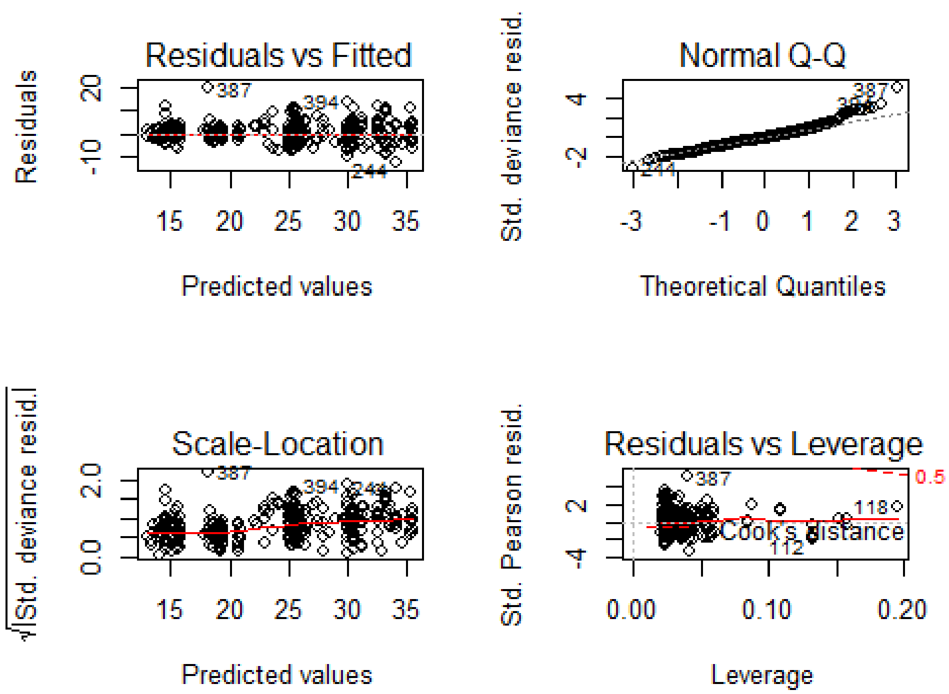
```
MSE[which.min(MSE)]

## [1] 17.34765
```

17.34765< 17.66336 Hence we found a nonlinear model which performs better than a Linear Model.

Potting the Nonlinear estimates

Plotting X vs. the non-linear estimate f^(X)

```
fit <- glm(mpg ~ ns(displacement, df = which.min(MSE)), data = Auto)
par(mfrow = c(2,2))
plot(fit)
```



GAM

```
library(gam)

## Warning: package 'gam' was built under R version 3.2.5

## Loaded gam 1.14

fit <- gam(mpg ~ s(displacement, 4) + s(horsepower, 4), data = Auto)
summary(fit)

##
## Call: gam(formula = mpg ~ s(displacement, 4) + s(horsepower, 4), data = Auto)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -11.2982  -2.1592  -0.4394   2.1247  17.0946
##
## (Dispersion Parameter for gaussian family taken to be 15.3543)
##
##     Null Deviance: 23818.99 on 391 degrees of freedom
## Residual Deviance: 5880.697 on 382.9999 degrees of freedom
## AIC: 2194.05
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                     Df  Sum Sq Mean Sq F value  Pr(>F)
```

```
## s(displacement, 4)   1 15254.9 15254.9 993.524 < 2e-16 ***
## s(horsepower, 4)     1  1038.4  1038.4  67.632 3.1e-15 ***
## Residuals          383  5880.7    15.4
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                     Npar Df Npar F     Pr(F)
## (Intercept)
## s(displacement, 4)       3 13.613 1.863e-08 ***
## s(horsepower, 4)         3 15.606 1.349e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
library(DAAG)
```

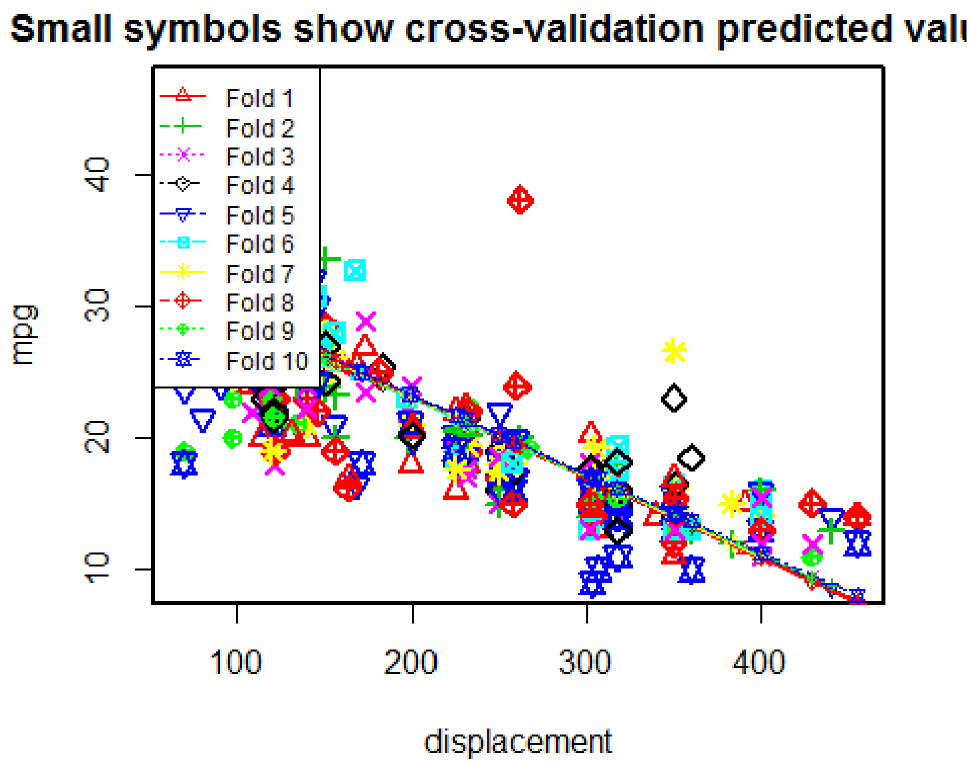
```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

```
lm_fit <- lm(mpg ~ displacement, data = Auto)
a<-cv.lm(data = Auto, form.lm = formula(mpg ~ displacement),m=10)
```

```
## Analysis of Variance Table
##
## Response: mpg
##               Df Sum Sq Mean Sq F value Pr(>F)
## displacement   1  15440   15440     719 <2e-16 ***
## Residuals    390   8379      21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
##
## fold 1
## Observations in test set: 39
##                 1      7     10      12     14    17     22     23     35
## displacement 307.00 454.00 390.00 340.000 455.0 199.0 107.00 104.00 225.00
## cvpred        16.62   7.66  11.56  14.611   7.6  23.2  28.81  28.99  21.62
## mpg           18.00  14.00  15.00  14.000  14.0  18.0  24.00  25.00  16.00
## CV residual    1.38   6.34   3.44  -0.611   6.4  -5.2  -4.81  -3.99  -5.62
##                56    61     74     81    108    120 125     165    185
## displacement 97.00 140.0 307.00 122.0 232.00 114.00 350 231.000 140.0
## cvpred       29.42  26.8  16.62  27.9  21.19  28.38  14  21.254  26.8
## mpg          27.00  20.0  13.00  22.0  18.00  20.00  11  21.000  25.0
## CV residual  -2.42  -6.8  -3.62  -5.9  -3.19  -8.38  -3  -0.254  -1.8
##                 189     192   196    208    252    254    260    269
## displacement 318.0000 225.00 85.00 130.00 302.00 200.00 200.00 119.00
## cvpred        15.9517  21.62 30.15  27.41  16.93  23.14  23.14  28.08
## mpg           16.0000  22.00 29.00  20.00  20.20  20.50  20.80  27.20
## CV residual    0.0483   0.38 -1.15  -7.41   3.27  -2.64  -2.34  -0.88
##                275   276   290   295    308  310    320    360     362
## displacement 131.00 163.0 350.0 86.00 173.00 98.0 120.00 141.00 168.000
## cvpred        27.35  25.4  14.0 30.09  24.79 29.4  28.02  26.74  25.093
## mpg           20.30  17.0  16.9 34.10  26.80 41.5  31.30  28.10  25.400
## CV residual   -7.05  -8.4   2.9  4.01   2.01 12.1   3.28   1.36   0.307
##                364    375    380    381
```

```
## displacement 231.00 105.00 120.00 107.00
## cvpred        21.25  28.93  28.02  28.81
## mpg           22.40  36.00  36.00  36.00
## CV residual    1.15   7.07   7.98   7.19
##
## Sum of squares = 942    Mean square = 24.1    n = 39
##
## fold 2
## Observations in test set: 40
##                   3      43    55      67     71   85     95      97
## displacement 318.00 383.000 72.00 304.000 400.00 97.0 440.00 360.000
## cvpred        16.01  12.164 30.59  16.844  11.16 29.1   8.79  13.527
## mpg           18.00  12.000 35.00  17.000  13.00 27.0  13.00  13.000
## CV residual    1.99  -0.164  4.41   0.156   1.84 -2.1   4.21  -0.527
##                  99    124    126    140    153    155     166    168
## displacement 250.00 156.00 198.00 302.00 225.00 250.00 262.000 97.000
## cvpred        20.04  25.61  23.12  16.96  21.52  20.04  19.331 29.104
## mpg           16.00  20.00  20.00  14.00  19.00  15.00  20.000 29.000
## CV residual   -4.04  -5.61  -3.12  -2.96  -2.52  -5.04   0.669 -0.104
##                 172    181     188   199    230    240    256    257
## displacement 134.00 121.00 305.000 91.00 400.00 97.000 140.00 225.00
## cvpred        26.91  27.68  16.784 29.46  11.16 29.104  26.56  21.52
## mpg           24.00  25.00  17.500 33.00  16.00 30.000  25.10  20.50
## CV residual   -2.91  -2.68   0.716  3.54   4.84  0.896  -1.46  -1.02
##                 272    273     284     285     297   304    309    319
## displacement 156.00 151.00 232.000 225.000 121.000 85.00 151.00 134.00
## cvpred        25.61  25.91  21.108  21.523  27.683 29.82  25.91  26.91
## mpg           23.20  23.80  20.200  20.600  27.400 31.80  33.50  29.80
## CV residual   -2.41  -2.11  -0.908  -0.923  -0.283  1.98   7.59   2.89
##                 328  330   348    370   377   379     396   397
## displacement 121.00 91.0 85.00 112.00 91.00 98.00 120.000 119.0
## cvpred        27.68 29.5 29.82  28.22 29.46 29.05  27.742  27.8
## mpg           36.40 44.6 37.00  34.00 31.00 36.00  28.000  31.0
## CV residual    8.72 15.1  7.18   5.78  1.54  6.95   0.258   3.2
##
## Sum of squares = 737    Mean square = 18.4    n = 40
##
## fold 3
## Observations in test set: 40
##                  16     47     63     77     91     94     104     105
## displacement 198.00 140.00 350.00 121.00 429.00 318.00 400.000 400.000
## cvpred        23.28  26.74  14.22  27.87   9.51  16.13  11.243  11.243
## mpg           22.00  22.00  13.00  18.00  12.00  14.00  11.000  12.000
## CV residual   -1.28  -4.74  -1.22  -9.87   2.49  -2.13  -0.243   0.757
##                 111    119    149    156    161    177    184     194
## displacement 108.00 116.00 116.00 250.00 231.00 232.00 116.00 200.000
## cvpred        28.64  28.17  28.17  20.18  21.31  21.25  28.17  23.161
```

```
## mpg           22.00  24.00  26.00  15.00  17.00  19.00  25.00  24.000
## CV residual   -6.64  -4.17  -2.17  -5.18  -4.31  -2.25  -3.17   0.839
##                202    214    215   219    221     227    228    229    232
## displacement 250.00 350.00 302.00 79.00 85.00 231.000 225.00 250.00 400.00
## cvpred        20.18  14.22  17.08 30.37 30.01  21.313  21.67  20.18  11.24
## mpg           18.50  13.00  13.00 36.00 33.50  20.500  19.00  18.50  15.50
## CV residual   -1.68  -1.22  -4.08  5.63  3.49  -0.813  -2.67  -1.68   4.26
##               245    264    265    274    283   294     300    307  318
## displacement 90.0 231.00 302.00 119.00 140.00 89.00 141.000 173.00 97.0
## cvpred       29.7  21.31  17.08  27.99  26.74 29.77  26.676  24.77 29.3
## mpg          43.1  17.70  18.10  23.90  22.30 31.90  27.200  28.80 34.3
## CV residual  13.4  -3.61   1.02  -4.09  -4.44  2.13   0.524   4.03  5.0
##                  341    342   350    353    374   383
## displacement 156.0000 173.00 91.00 98.000 140.00 91.00
## cvpred        25.7824  24.77 29.66 29.238  26.74 29.66
## mpg           25.8000  23.50 34.10 29.900  24.00 38.00
## CV residual    0.0176  -1.27  4.44  0.662  -2.74  8.34
##
## Sum of squares = 730    Mean square = 18.2    n = 40
##
## fold 4
## Observations in test set: 39
##                 24     31       39     48      51    80     90    123
## displacement 121.00 140.00 350.0000 250.00 116.000 96.00 318.00 121.00
## cvpred        27.86  26.71  14.0399  20.07  28.158 29.36  15.97  27.86
## mpg           26.00  28.00  14.0000  19.00  28.000 26.00  15.00  24.00
## CV residual   -1.86   1.29  -0.0399  -1.07  -0.158 -3.36  -0.97  -3.86
##                134   147    150      159    164    178    180    203
## displacement 250.00 90.00 120.00 318.0000 225.00 115.00 121.00 258.00
## cvpred        20.07 29.73  27.92  15.9705  21.58  28.22  27.86  19.59
## mpg           16.00 28.00  24.00  16.0000  18.00  23.00  22.00  17.50
## CV residual   -4.07 -1.73  -3.92   0.0295  -3.58  -5.22  -5.86  -2.09
##                216   217     222     243    255     281     287    288
## displacement 318.00 98.00 305.000 121.00 200.00 231.000 302.000 351.00
## cvpred        15.97 29.24  16.755  27.86  23.09  21.219  16.936  13.98
## mpg           13.00 31.50  17.500  21.50  20.20  21.500  17.600  16.50
## CV residual   -2.97  2.26   0.745  -6.36  -2.89   0.281   0.664   2.52
##                289    293    298    299    302    316    322   336
## displacement 318.00 360.00 183.00 350.00 105.00 151.00 108.00 122.0
## cvpred        15.97  13.44  24.12  14.04  28.82  26.05  28.64  27.8
## mpg           18.20  18.50  25.40  23.00  34.20  24.30  32.20  35.0
## CV residual    2.23   5.06   1.28   8.96   5.38  -1.75   3.56   7.2
##                  339  347   349    358    359    366     392
## displacement 135.000 97.0 89.00 119.00 120.00 200.00 151.000
## cvpred        27.011 29.3 29.79  27.98  27.92  23.09  26.046
## mpg           27.200 32.3 37.70  32.90  31.60  20.20  27.000
## CV residual    0.189  3.0  7.91   4.92   3.68  -2.89   0.954
```

```
##
## Sum of squares = 545    Mean square = 14    n = 39
##
## fold 5
## Observations in test set: 39
##                 8     41     49     53     58     78   82     89
## displacement 440.00 351.0000 250.00 88.0000 113.00 121.00 97.0 302.00
## cvpred         8.63  14.0473  20.19 30.0521  28.53  28.04 29.5  17.03
## mpg           14.00  14.0000  18.00 30.0000  24.00  22.00 28.0  14.00
## CV residual    5.37  -0.0473  -2.19 -0.0521  -4.53  -6.04 -1.5  -3.03
##                98    114   118    128    136    139    148    157
## displacement 225.00 155.00 68.00 232.00 225.00 318.00 90.00 400.00
## cvpred        21.72  25.97 31.27  21.29  21.72  16.06 29.93  11.07
## mpg           18.00  21.00 29.00  19.00  18.00  14.00 24.00  16.00
## CV residual   -3.72  -4.97 -2.27  -2.29  -3.72  -2.06 -5.93   4.93
##                 160    162    171    174   186    193    209    212
## displacement 351.0000 250.00 140.00 119.00 98.00 250.00 318.00 168.00
## cvpred        14.0473  20.19  26.89  28.17 29.44  20.19  16.06  25.18
## mpg           14.0000  16.00  23.00  24.00 26.00  22.00  13.00  16.50
## CV residual   -0.0473  -4.19  -3.89  -4.17 -3.44   1.81  -3.06  -8.68
##               223   244    250     268    270    282    329   335   352
## displacement 260.00 80.00 260.000 134.000 105.00 200.00 146.00 70.00 98.00
## cvpred        19.59 30.54  19.585  27.253  29.02  23.24  26.52 31.15 29.44
## mpg           17.00 21.50  19.900  27.500  30.90  19.80  30.00 23.70 34.40
## CV residual   -2.59 -9.04   0.315   0.247   1.88  -3.44   3.48 -7.45  4.96
##               356    363    369   376    390    395
## displacement 107.0 146.00 112.00 91.00 144.00 135.00
## cvpred        28.9  26.52  28.59 29.87  26.64  27.19
## mpg           33.7  24.20  27.00 37.00  32.00  32.00
## CV residual    4.8  -2.32  -1.59  7.13   5.36   4.81
##
## Sum of squares = 708    Mean square = 18.2    n = 39
##
## fold 6
## Observations in test set: 39
##                  2     13    19    20     38     64     75     86     87
## displacement 350.000 400.00 97.00 97.00 232.00 400.00 302.00 350.00 304.00
## cvpred        14.087  11.13 29.07 29.07  21.07  11.13  16.93  14.09  16.81
## mpg           15.000  15.00 27.00 26.00  18.00  14.00  13.00  13.00  14.00
## CV residual    0.913   3.87 -2.07 -3.07  -3.07   2.87  -3.93  -1.09  -2.81
##                88    100     102     106    117    131    138   144
## displacement 350.00 232.00 198.0000 360.000 400.00 122.00 350.00 97.00
## cvpred        14.09  21.07  23.0861  13.495  11.13  27.59  14.09 29.07
## mpg           13.00  18.00  23.0000  13.000  16.00  26.00  13.00 26.00
## CV residual   -1.09  -3.07  -0.0861  -0.495   4.87  -1.59  -1.09 -3.07
##               146    167   176   198   205    213    218   248    251
## displacement 83.00 302.00 90.00 90.00 85.00 350.00 111.00 85.00 318.00
```

```
## cvpred       29.89  16.93 29.48 29.48 29.78  14.09  28.24 29.78  15.98
## mpg          32.00  13.00 29.00 29.00 32.00  16.50  30.00 39.40  19.40
## CV residual   2.11  -3.93 -0.48 -0.48  2.22   2.41   1.76  9.62   3.42
##                261    262    266    321    324  327   333    334    343
## displacement 225.00 258.00 318.00 119.00 156.00 90.0 89.00 168.00 135.00
## cvpred        21.49  19.53  15.98  27.76  25.57 29.5 29.54  24.86  26.82
## mpg           18.60  18.10  17.50  37.00  27.90 43.4 29.80  32.70  30.00
## CV residual   -2.89  -1.43   1.52   9.24   2.33 13.9  0.26   7.84   3.18
##                351    361    373    378
## displacement 105.00 145.00 151.00 105.00
## cvpred        28.59  26.22  25.87  28.59
## mpg           34.70  30.70  27.00  38.00
## CV residual    6.11   4.48   1.13   9.41
##
## Sum of squares = 774    Mean square = 19.8    n = 39
##
## fold 7
## Observations in test set: 39
##                 11     18     37     40     42     44     46    57    59
## displacement 383.00 200.00 250.00 400.00 318.00 400.00 258.00 91.00 97.50
## cvpred        12.03  23.03  20.02  11.01  15.94  11.01  19.54 29.58 29.19
## mpg           15.00  21.00  19.00  14.00  14.00  13.00  18.00 26.00 25.00
## CV residual    2.97  -2.03  -1.02   2.99  -1.94   1.99  -1.54 -3.58 -4.19
##                110    129     191   197    201   206     207    210
## displacement 140.00 250.00 351.000 98.00 250.00 97.00 140.000 120.00
## cvpred        26.63  20.02  13.955 29.16  20.02 29.22  26.634  27.84
## mpg           21.00  15.00  14.500 24.50  18.00 28.00  26.500  19.00
## CV residual   -5.63  -5.02   0.545 -4.66  -2.02 -1.22  -0.134  -8.84
##                226    233    234    235   238    258    263    280
## displacement 250.00 351.00 97.000 151.00 98.00 232.00 305.00 98.000
## cvpred        20.02  13.96 29.218  25.97 29.16  21.11  16.72 29.158
## mpg           17.50  16.00 29.000  24.50 30.50  19.40  19.20 29.500
## CV residual   -2.52   2.04 -0.218  -1.47  1.34  -1.71   2.48  0.342
##                 286   296    303   312    314  326    338   365    367
## displacement 305.000 98.00 105.00 98.00 151.00 90.0 107.00 350.0 225.00
## cvpred        16.719 29.16  28.74 29.16  25.97 29.6  28.62  14.0  21.53
## mpg           17.000 35.70  34.50 32.10  28.00 44.3  32.40  26.6  17.60
## CV residual    0.281  6.54   5.76  2.94   2.03 14.7   3.78  12.6  -3.93
##                371    372     388     393  394
## displacement 112.00 135.00 156.000 140.000 97.0
## cvpred        28.32  26.93  25.673  26.634 29.2
## mpg           31.00  29.00  26.000  27.000 44.0
## CV residual    2.68   2.07   0.327   0.366 14.8
##
## Sum of squares = 972    Mean square = 24.9    n = 39
##
## fold 8
```

```
## Observations in test set: 39
##                 4      5     6     9    15    34    36    50
## displacement 304.00 302.0000 429.00 455.00 113.00 232.00 250.00 122.00
## cvpred        16.78  16.9021   9.13   7.54  28.47  21.19  20.08  27.92
## mpg           16.00  17.0000  15.00  14.00  24.00  19.00  17.00  23.00
## CV residual   -0.78   0.0979   5.87   6.46  -4.47  -2.19  -3.08  -4.92
##                70     73     83     92    107    113    121    141    145
## displacement 350.00 304.00 120.00 400.0 350.00 122.00 121.00 304.00 76.000
## cvpred        13.96  16.78  28.04  10.9  13.96  27.92  27.98  16.78 30.733
## mpg           12.00  15.00  23.00  13.0  12.00  19.00  19.00  14.00 31.000
## CV residual   -1.96  -1.78  -5.04   2.1  -1.96  -8.92  -8.98  -2.78  0.267
##               158    163    169    187    211    220    225    237    239
## displacement 350.00 258.00 140.00 101.0 156.00 122.00 302.0 140.00 98.00
## cvpred        13.96  19.59  26.82  29.2  25.84  27.92  16.9  26.82 29.39
## mpg           15.00  15.00  23.00  27.0  19.00  25.50  15.0  25.50 33.50
## CV residual    1.04  -4.59  -3.82  -2.2  -6.84  -2.42  -1.9  -1.32  4.11
##               242    267    278    291    301    306    345    354    357
## displacement 146.00 98.000 163.00 351.0 260.00 151.00 86.00 105.00 108.00
## cvpred        26.45 29.387  25.41  13.9  19.47  26.14 30.12  28.96  28.77
## mpg           22.00 30.000  16.20  15.5  23.90  28.40 39.00  33.00  32.40
## CV residual   -4.45  0.613  -9.21   1.6   4.43   2.26  8.88   4.04   3.63
##              384    386   387    389
## displacement 91.00 181.000 262.0 232.000
## cvpred       29.82  24.307  19.4  21.186
## mpg          32.00  25.000  38.0  22.000
## CV residual   2.18   0.693  18.6   0.814
##
## Sum of squares = 1055    Mean square = 27    n = 39
##
## fold 9
## Observations in test set: 39
##                21     32    60     62     65     68    72    84     93
## displacement 110.00 113.00 97.00 122.0 318.00 429.00  70.0 98.00 351.00
## cvpred        28.41  28.23 29.18  27.7  16.07   9.48  30.8 29.12  14.11
## mpg           25.00  25.00 23.00  21.0  15.00  11.00  19.0 28.00  13.00
## CV residual   -3.41  -3.23 -6.18  -6.7  -1.07   1.52 -11.8 -1.12  -1.11
##              109    116    132    142    151    152    170   173    179
## displacement 97.00 350.000 71.00 98.000 108.00 79.000 232.00 90.0 120.00
## cvpred       29.18  14.171 30.73 29.124  28.53 30.251  21.17 29.6  27.82
## mpg          20.00  15.000 32.00 29.000  26.00 31.000  20.00 25.0  23.00
## CV residual  -9.18   0.829  1.27 -0.124  -2.53  0.749  -1.17 -4.6  -4.82
##              182   190    195    204    224   241    259    271    277
## displacement 91.00 304.0 232.00 97.000 318.00 97.00 231.000 134.00 121.00
## cvpred       29.54  16.9  21.17 29.183  16.07 29.18  21.232  26.99  27.76
## mpg          33.00  15.5  22.50 29.500  15.50 30.50  20.600  21.10  21.60
## CV residual   3.46  -1.4   1.33  0.317  -0.57  1.32  -0.632  -5.89  -6.16
##              279    292   305   311   313  323  325   332    340   344
```

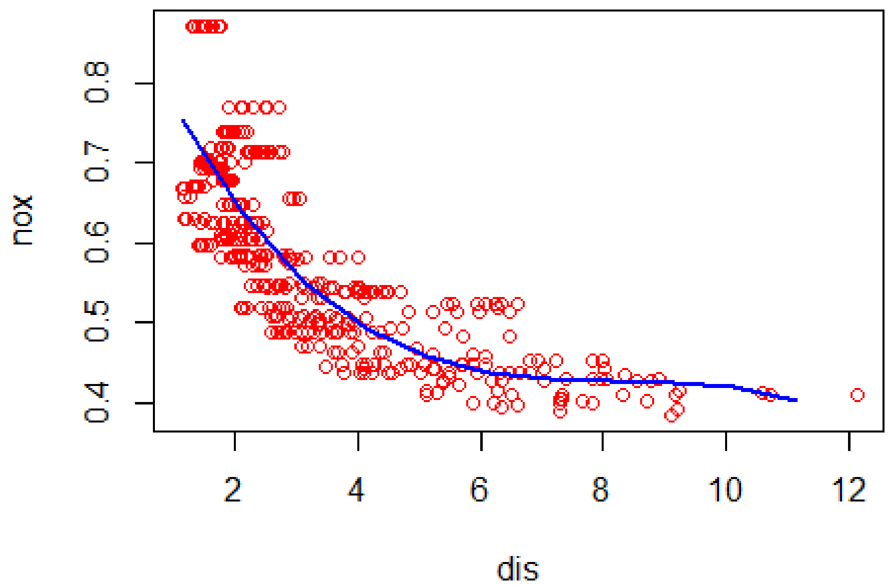
```
## displacement 89.00 267.000 91.00 89.00 86.00 86.0 85.0 97.00 151.000 79.00
## cvpred       29.66  19.096 29.54 29.66 29.84 29.8 29.9 29.18  25.979 30.25
## mpg          31.50  19.200 37.30 38.10 37.20 46.6 40.8 33.80  26.600 39.10
## CV residual   1.84   0.104  7.76  8.44  7.36 16.8 10.9  4.62   0.621  8.85
##                346    382
## displacement 81.00 108.00
## cvpred       30.13  28.53
## mpg          35.10  34.00
## CV residual   4.97   5.47
##
## Sum of squares = 1224    Mean square = 31.4    n = 39
##
## fold 10
## Observations in test set: 39
##                  25    26     27     28     29   30     45     52      54
## displacement 199.00 360.0 307.00 318.00 304.00 97.0 400.00 79.000 71.0000
## cvpred        23.31  13.7  16.87  16.21  17.05 29.4  11.32 30.477 30.9541
## mpg           21.00  10.0  10.00  11.00   9.00 27.0  13.00 30.000 31.0000
## CV residual   -2.31  -3.7  -6.87  -5.21  -8.05 -2.4   1.68 -0.477  0.0459
##                  66    69     76     79     96    101  103 112   115
## displacement 351.00 350.0 318.00 120.00 455.00 250.00 97.0  70 98.00
## cvpred        14.24  14.3  16.21  28.03   8.03  20.27 29.4  31 29.34
## mpg           14.00  13.0  14.00  21.00  12.00  18.00 26.0  18 26.00
## CV residual   -0.24  -1.3  -2.21  -7.03   3.97  -2.27 -3.4 -13 -3.34
##                 122    130    133    135    137   143    154    175
## displacement 318.00 79.000 140.00 258.00 302.00 79.00 250.00 171.00
## cvpred        16.21 30.477  26.84  19.79  17.17 30.48  20.27  24.98
## mpg           15.00 31.000  25.00  16.00  16.00 26.00  18.00  18.00
## CV residual   -1.21  0.523  -1.84  -3.79  -1.17 -4.48  -2.27  -6.98
##                  183    200   231  236   246   247   249   253     315
## displacement 107.000 225.00 350.0 97.0 98.00 78.00 91.00 231.0 140.000
## cvpred        28.805  21.76  14.3 29.4 29.34 30.54 29.76  21.4  26.835
## mpg           28.000  20.00  15.5 26.0 36.10 32.80 36.10  19.2  26.400
## CV residual   -0.805  -1.76   1.2 -3.4  6.76  2.26  6.34  -2.2  -0.435
##                 317     368   385    391
## displacement 225.00 112.000 91.00 135.00
## cvpred        21.76  28.507 29.76  27.13
## mpg           19.10  28.000 38.00  36.00
## CV residual   -2.66  -0.507  8.24   8.87
##
## Sum of squares = 797    Mean square = 20.4    n = 39
##
## Overall (Sum over all 39 folds)
##   ms
## 21.6
```

Problem: 4

November 15, 2016

(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.

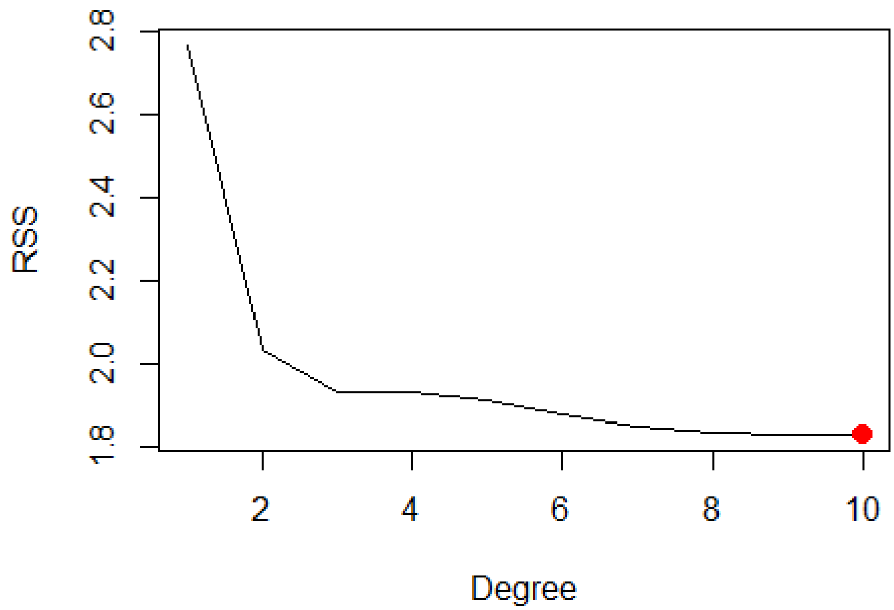
```
set.seed(12)
par(mfrow = c(1,1))

library(MASS)
library(boot)
library(splines)
df<-Boston
fit <- lm(nox ~ poly(dis, 3), data = df)
summary(fit)

##
## Call:
## lm(formula = nox ~ poly(dis, 3), data = df)
##
## 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

dis_range = range(df$dis)
dis_seq = seq(from = dis_range[1], to = dis_range[2])
prediction = predict(fit, list(dis = dis_seq))
plot(nox ~ dis, data = df,col="red")
lines(dis_seq, prediction, lwd = 2, col = 'blue')
```



By looking at p values we can conclude that all polynomial terms are significant.

(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.

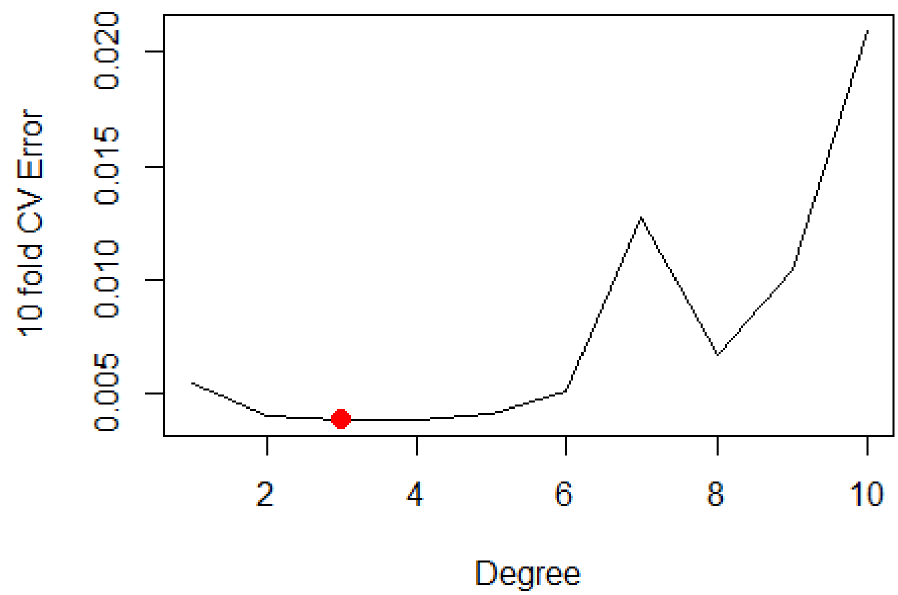
```
RSS <- rep(NA, 10)
for (i in 1:10) {
  fit <- lm(nox ~ poly(dis, i), data = Boston)
  RSS[i] <- sum(fit$residuals^2)
}
plot(1:10, RSS, xlab = "Degree", ylab = "RSS", type = "l")
points(which.min(RSS), RSS[which.min(RSS)], col = "red", cex = 2, pch = 20)
```



RSS monotonically decreases as we increase degree of polynomial. minimum RSS is achieved at degree 10

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

```
MSE <- rep(NA, 10)
for (i in 1:10) {
  fit <- glm(nox ~ poly(dis, i), data = Boston)
  MSE[i] <- cv.glm(Boston, fit, K = 10)$delta[1]
}
plot(1:10, MSE, xlab = "Degree", ylab = "10 fold CV Error", type = "l")
points(which.min(MSE), MSE[which.min(MSE)], col = "red", cex = 2, pch = 20)
```



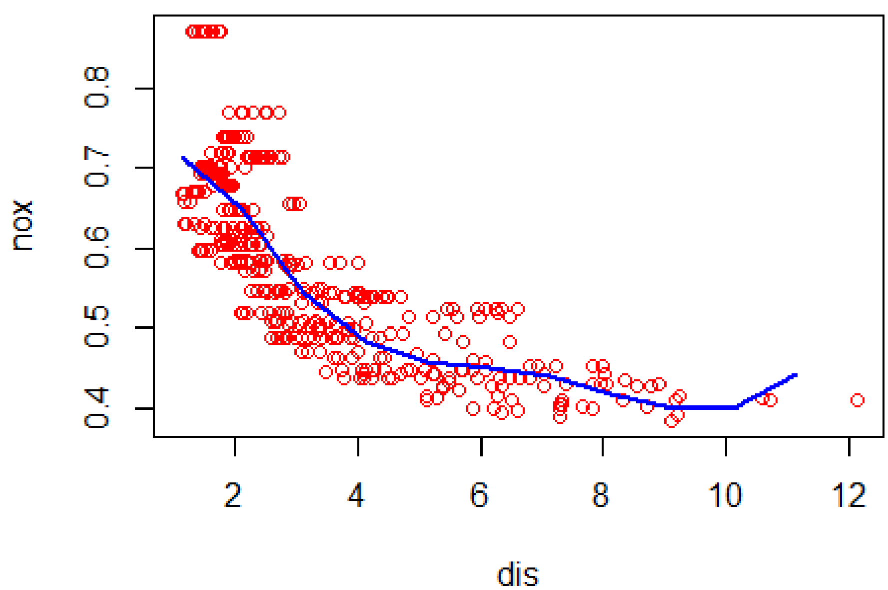
We can say that polynomial of degree 4 minimizes the test MSE.

(d) Use the "bs()" function to fit a regression spline to predict "nox" using "dis". Report the output for the fit using 7 degrees of freedom (3 knots)

```
fit <- lm(nox ~ bs(dis, df = 4, knots = c(3, 7, 11)), data = df)
summary(fit)

##
## Call:
## lm(formula = nox ~ bs(dis, df = 4, knots = c(3, 7, 11)), data = df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.130710 -0.039850 -0.008357  0.027792  0.188518
##
## Coefficients:
##                                      Estimate Std. Error t value
## (Intercept)                          0.714346   0.015846  45.081
## bs(dis, df = 4, knots = c(3, 7, 11))1 -0.006626   0.024307  -0.273
## bs(dis, df = 4, knots = c(3, 7, 11))2 -0.296980   0.018293 -16.234
## bs(dis, df = 4, knots = c(3, 7, 11))3 -0.222840   0.033763  -6.600
## bs(dis, df = 4, knots = c(3, 7, 11))4 -0.379811   0.042317  -8.975
## bs(dis, df = 4, knots = c(3, 7, 11))5 -0.222959   0.086870  -2.567
```

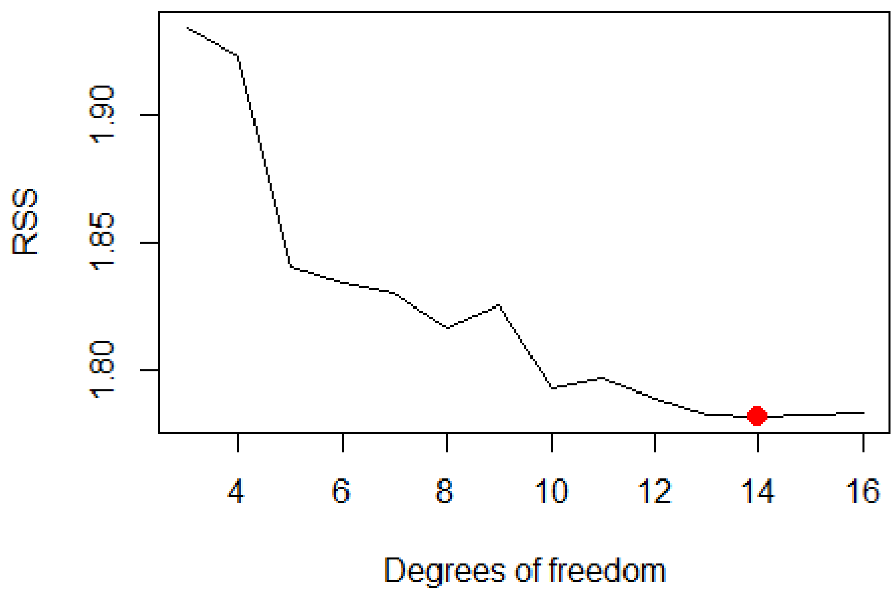
```
## bs(dis, df = 4, knots = c(3, 7, 11))6 -0.304346   0.063378  -4.802
##                                        Pr(>|t|)
## (Intercept)                            < 2e-16 ***
## bs(dis, df = 4, knots = c(3, 7, 11))1   0.7853
## bs(dis, df = 4, knots = c(3, 7, 11))2  < 2e-16 ***
## bs(dis, df = 4, knots = c(3, 7, 11))3 1.05e-10 ***
## bs(dis, df = 4, knots = c(3, 7, 11))4  < 2e-16 ***
## bs(dis, df = 4, knots = c(3, 7, 11))5   0.0106 *
## bs(dis, df = 4, knots = c(3, 7, 11))6 2.08e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06137 on 499 degrees of freedom
## Multiple R-squared:  0.7229, Adjusted R-squared:  0.7196
## F-statistic:   217 on 6 and 499 DF,  p-value: < 2.2e-16
```

```r
pred <- predict(fit, list(dis = dis_seq))
plot(nox ~ dis, data = Boston, col = "red")
lines(dis_seq, pred, col = "blue", lwd = 2)
```



(e) 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.

```
RSS <- rep(NA, 16)
for (i in 3:16) {
  fit <- lm(nox ~ bs(dis, df = i), data = Boston)
  RSS[i] <- sum(fit$residuals^2)
}
plot(3:16, RSS[-c(1, 2)], xlab = "Degrees of freedom", ylab = "RSS", type = "l")
points(which.min(RSS), RSS[which.min(RSS)], col = "red", cex = 2, pch = 20)
```



We may see that RSS decreases until 14 and then slightly increases after that. Minimum RSS is achieved with splines of degrees of freedom 14

(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.

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

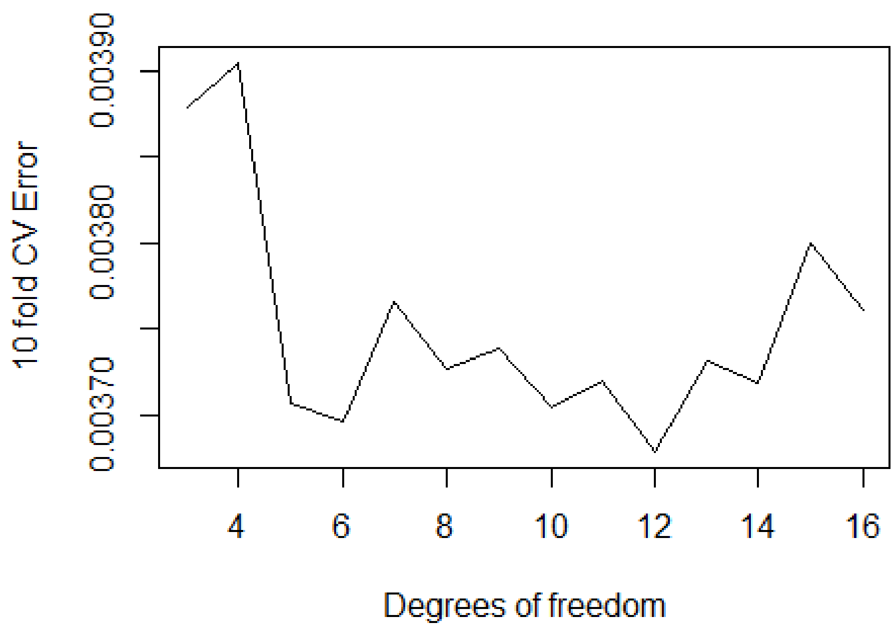
```
## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(3.1675, .Names =
## "50%"), : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(3.1675, .Names =
## "50%"), : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(3.1827, .Names =
## "50%"), : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(3.1827, .Names =
## "50%"), : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(2.3727,
## 4.36263333333333: some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(2.3727,
## 4.36263333333333: some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(2.4212,
## 4.23913333333333: some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(2.4212,
## 4.23913333333333: some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = structure(c(2.10035, 3.2157,
## 5.16495: some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(2.10035, 3.2157,
## 5.16495: some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(2.0754, 3.1323,
## 5.11735: some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(2.0754, 3.1323,
## 5.11735: some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.96376, 2.66502,
## 3.9175, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.96376, 2.66502,
## 3.9175, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.9512, 2.6403,
## 3.9454, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.9512, 2.6403,
## 3.9454, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.8651,
## 2.41306666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.8651,
## 2.41306666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.86156666666667,
## 2.38403333333333, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.86156666666667,
## 2.38403333333333, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = structure(c(1.78037142857143,
## 2.2044, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.78037142857143,
## 2.2044, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.81317142857143,
## 2.25881428571429, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.81317142857143,
## 2.25881428571429, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.743225, 2.0754,
## 2.4999, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.743225, 2.0754,
## 2.4999, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.6732,
## 2.00496666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.6732,
## 2.00496666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.6723,
## 2.00613333333333, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.6723,
## 2.00613333333333, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.6362, 1.9865,
## 2.288, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.6362, 1.9865,
## 2.288, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = structure(c(1.64668, 1.96376,
## 2.28422, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.64668, 1.96376,
## 2.28422, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.60816363636364,
## 1.87607272727273, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.60816363636364,
## 1.87607272727273, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.59590909090909,
## 1.87607272727273, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.59590909090909,
## 1.87607272727273, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.58948333333333,
## 1.8301, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.58948333333333,
## 1.8301, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.60973333333333,
## 1.8651, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.60973333333333,
## 1.8651, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.5523,
## 1.79772307692308, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.5523,
## 1.79772307692308, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = structure(c(1.5895, 1.8195,
## 2.0407, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.5895, 1.8195,
## 2.0407, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.5311,
## 1.78037142857143, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = structure(c(1.5311,
## 1.78037142857143, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

plot(3:16, CVError[-c(1, 2)], xlab = "Degrees of freedom", ylab = "10 fold CV Error", type = "l")
```



minimum CV error is achieved at degrees od fredom12

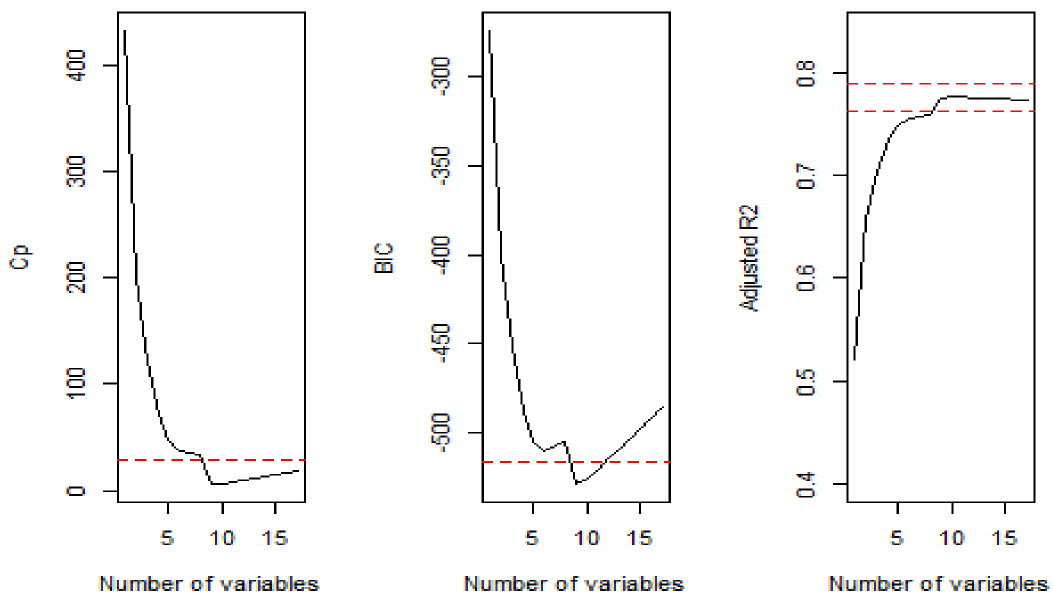
Problem 5:

This question relates to the College data set.

(a) Split the data into a training set and a test set. Using out-of-state tuition as the response and the other variables as the predictors, perform forward stepwise selection on the training set in order to identify a satisfactory model that uses just a subset of the predictors.

```
library(ISLR)

## Warning: package 'ISLR' was built under R version 3.2.5

library(leaps)

## Warning: package 'leaps' was built under R version 3.2.5

library(gam)

## Warning: package 'gam' was built under R version 3.2.5

## Loading required package: splines

## Loading required package: foreach

## Loaded gam 1.14

data("College")
attach(College)
train <- sample(length(Outstate), length(Outstate) / 2)
test <- -train
College_train <- College[train, ]
College_test <- College[test, ]
fit <- regsubsets(Outstate ~ ., data = College_train, nvmax = 17, method = "forward")
fit.summary <- summary(fit)
par(mfrow = c(1, 3))
plot(fit.summary$cp, xlab = "Number of variables", ylab = "Cp", type = "l")
min_cp <- min(fit.summary$cp)
std_cp <- sd(fit.summary$cp)
abline(h = min_cp + 0.2 * std_cp, col = "red", lty = 2)
abline(h = min_cp - 0.2 * std_cp, col = "red", lty = 2)
plot(fit.summary$bic, xlab = "Number of variables", ylab = "BIC", type='l')
min_bic <- min(fit.summary$bic)
std_bic <- sd(fit.summary$bic)
abline(h = min_bic + 0.2 * std_bic, col = "red", lty = 2)
abline(h = min_bic - 0.2 * std_bic, col = "red", lty = 2)
plot(fit.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R2", type = "l", ylim = c(0.4, 0.84))
max_adjR2 <- max(fit.summary$adjr2)
std_adjR2 <- sd(fit.summary$adjr2)
```

```
abline(h = max_adjR2 + 0.2 * std_adjR2, col = "red", lty = 2)
abline(h = max_adjR2 - 0.2 * std_adjR2, col = "red", lty = 2)
```

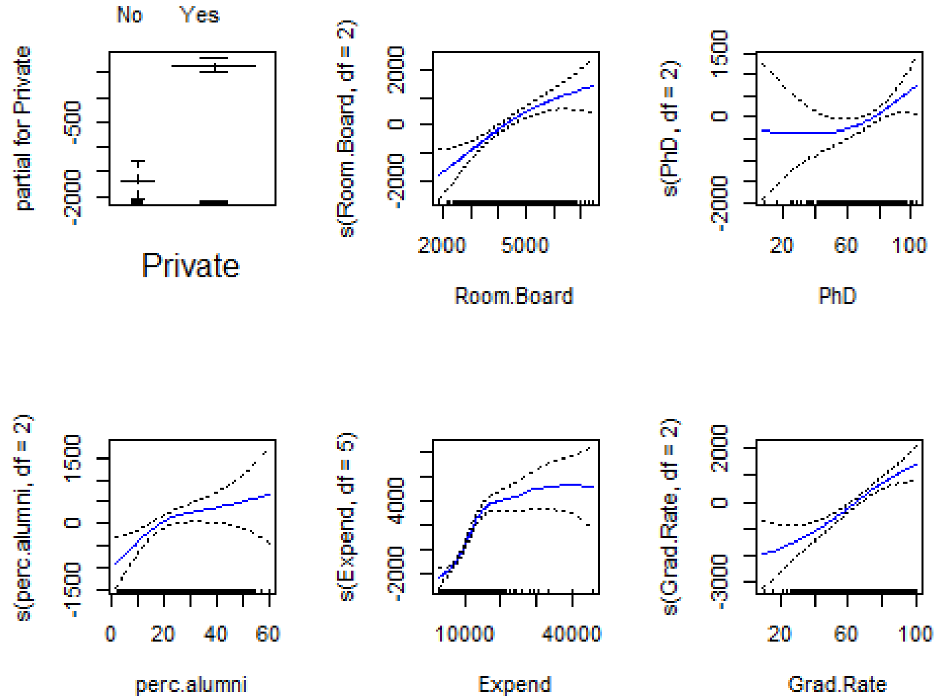


Cp, BIC and AdjR^2 show that size 8 is the minimum size for the subset for which the scores are within 0.2 standard deviations of optimum.

```
fit <- regsubsets(Outstate ~ ., data = College, method = "forward")
coeffs <- coef(fit, id = 6)
names(coeffs)

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

(b) Fit a GAM on the training data, using out-of-state tuition as the response and the features selected in the previous step as the predictors. Plot the results, and explain your findings.

```
fit <- gam(Outstate ~ Private + s(Room.Board, df = 2) + s(PhD, df = 2) + s(perc.alumni, df = 2) + s(Expend, df = 5) + s(Grad.Rate, df = 2), data=College_train)
par(mfrow = c(2, 3))
plot(fit, se = T, col = "blue")
```



(c) Evaluate the model obtained on the test set, and explain the results obtained.

```
preds <- predict(fit, College_test)
err <- mean((College_test$Outstate - preds)^2)
err

## [1] 3706367

tss <- mean((College_test$Outstate - mean(College_test$Outstate))^2)
rss <- 1 - err / tss
rss

## [1] 0.7764983
```

We obtain a test R^2 of 0.77 using GAM with 6 predictors.

```
summary(fit)

##
## Call: gam(formula = Outstate ~ Private + s(Room.Board, df = 2) + s(PhD,
##     df = 2) + s(perc.alumni, df = 2) + s(Expend, df = 5) + s(Grad.Rate,
##     df = 2), data = College_train)
## Deviance Residuals:
##      Min      1Q   Median      3Q      Max
```

```
## -7457.88 -1066.58    10.69  1205.80  4269.38
##
## (Dispersion Parameter for gaussian family taken to be 3322454)
##
##     Null Deviance: 6102207011 on 387 degrees of freedom
## Residual Deviance: 1239276076 on 373.0002 degrees of freedom
## AIC: 6944.09
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                         Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private                  1 1650304201 1650304201 496.712 < 2.2e-16 ***
## s(Room.Board, df = 2)    1 1241956325 1241956325 373.807 < 2.2e-16 ***
## s(PhD, df = 2)           1  439487340  439487340 132.278 < 2.2e-16 ***
## s(perc.alumni, df = 2)   1  220881872  220881872  66.481 5.431e-15 ***
## s(Expend, df = 5)        1  617867577  617867577 185.967 < 2.2e-16 ***
## s(Grad.Rate, df = 2)     1  145332274  145332274  43.742 1.301e-10 ***
## Residuals              373 1239276076    3322454
## ---
## 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 = 2)        1  2.6037   0.10745
## s(PhD, df = 2)               1  2.6635   0.10352
## s(perc.alumni, df = 2)       1  4.7013   0.03077 *
## s(Expend, df = 5)            4 15.5625 8.735e-12 ***
## s(Grad.Rate, df = 2)         1  3.1876   0.07501 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By Looking at the p values: We can say there is a

- Strong of non-linear relationship between "Outstate" and "Expend"",
- Moderately strong non-linear relationship between “Outstate" and "Grad.Rate"" or "PhD".

Problem 6

In Section 7.7, it was mentioned that GAMs are generally fit using a *back fitting* approach. The idea behind back fitting is actually quite simple. We will now explore back fitting in the context of multiple linear regression. Suppose that we would like to perform multiple linear regression, but we do not have software to do so. Instead, we only have software to perform simple linear regression. Therefore, we take the following iterative approach: we repeatedly hold all but one coefficient estimate fixed at its current value, and update only that coefficient estimate using a simple linear regression. The process is continued until *convergence*—that is, until the coefficient estimates stop changing. We now try this out on a toy example.

Part (a) Generate a response Y and two predictors X1 and X2, with n = 100.

```
N = 100
X1 = rnorm(N)
X2 = rnorm(N)
e = rnorm(100, sd = 1)
Y = 1 + 2*X1 + 3*X2
```

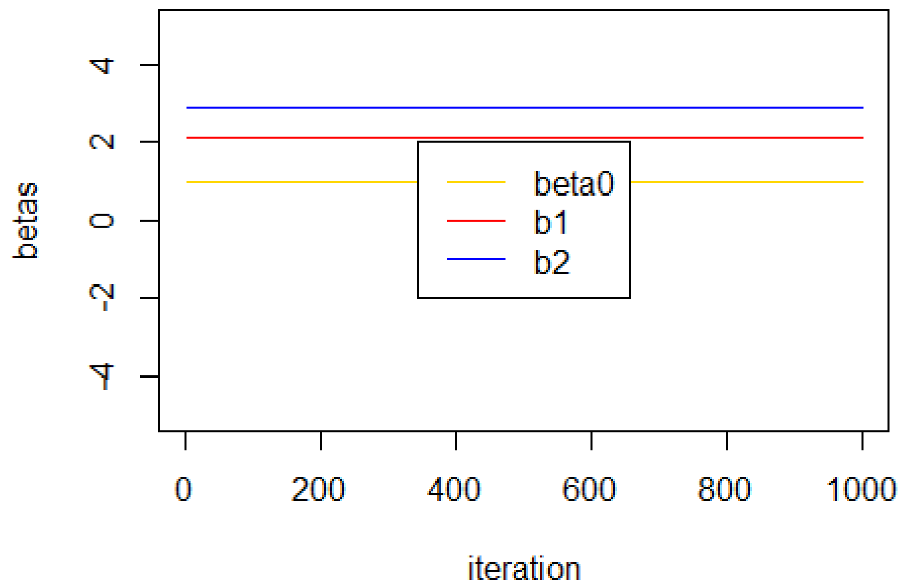
Part (b)(c)

```
b1 <-12
a <- Y - b1*X1
b2 <- lm(a ~ X2)$coef[2]

a<-Y - b2*X2
b1 <- lm(a ~ X1)$coef[2]
```

part (d)(e) Accumulate results of 1000 iterations in the beta arrays.

```
b0 <-rep(NA, 1000)
for (i in 1:1000) {
  a = Y - b1[i] * X1
  b2[i] = lm(a ~ X2)$coef[2]
  a = Y - b2[i] * X2
  lm_fit = lm(a ~ X1)
  if (i < 1000) {
    b1[i + 1] = lm_fit$coef[2]
  }
  b0[i] = lm_fit$coef[1]
}
plot(1:1000, b0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(-5,
  5), col = "gold")
lines(1:1000, b1, col = "red")
lines(1:1000, b2, col = "blue")
```

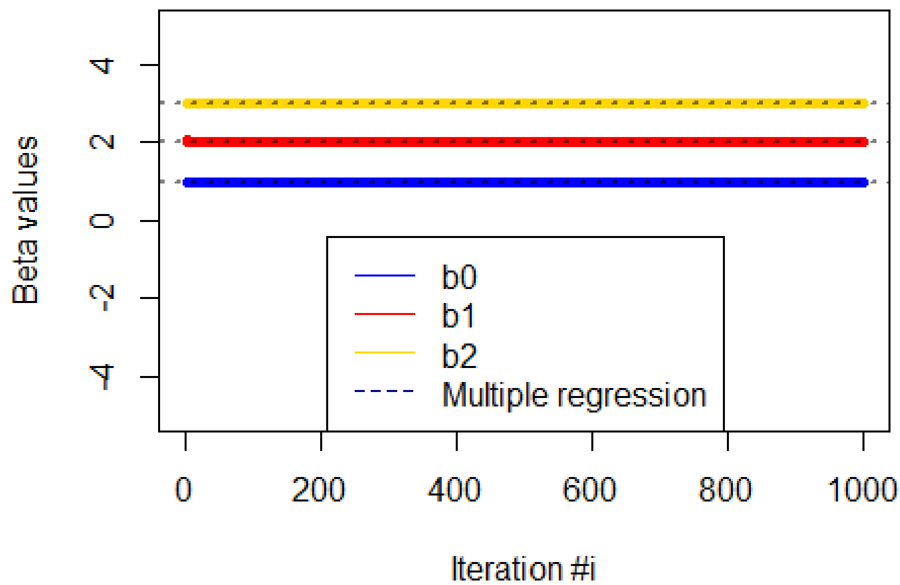
```r
legend("center", c("beta0", "b1", "b2"), lty = 1, col = c("gold", "red",
  "blue"))
```



The coefficients quickly attain their least square values.

part (f)

```r
lm_fit = lm(Y ~ X1 + X2)
plot (1:1000, b0, lwd = 5, type = 'l', xlab = 'Iteration #i', ylab = 'Beta values', ylim = c(-5, 5), col = 'blue')
lines(1:1000, b1, lwd = 5, col = 'red')
lines(1:1000, b2, lwd = 5, col = 'gold')
abline(h = lm_fit$coef[1], lty = 'dotted', lwd = 2, col = rgb(0, 0, 0, alpha = 0.5))
abline(h = lm_fit$coef[2], lty = 'dotted', lwd = 2, col = rgb(0, 0, 0, alpha = 0.5))
abline(h = lm_fit$coef[3], lty = 'dotted', lwd = 2, col = rgb(0, 0, 0, alpha = 0.5))
legend('bottom', c('b0', 'b1', 'b2', 'Multiple regression'), lty = c(1, 1, 1, 2), col = c('blue', 'red', 'gold', 'dark blue'))
```



Overlap of Dotted lines with solid line indicates that estimated multiple regression coefficients match exactly with the coefficients obtained using back fitting.