Untitled

```r
library(ISLR2)
attach(Wage)
library(gam)
```

```
Loading required package: splines

Loading required package: foreach

Loaded gam 1.22-5
```

```r
library(glmnet)
```

```
Loading required package: Matrix

Loaded glmnet 4.1-8
```

```r
library(boot)
library(ggplot2)
library(leaps)
```

#labs #7.8.1 Polynomial Regression and Step Functions

#orthogonal polynomial

```r
fit <- lm(wage ~ poly(age, 4), data = Wage)
coef(summary(fit))
```

```
                 Estimate Std. Error    t value     Pr(>|t|)
(Intercept)     111.70361  0.7287409 153.283015 0.000000e+00
poly(age, 4)1   447.06785 39.9147851  11.200558 1.484604e-28
poly(age, 4)2  -478.31581 39.9147851 -11.983424 2.355831e-32
poly(age, 4)3   125.52169 39.9147851   3.144742 1.678622e-03
poly(age, 4)4   -77.91118 39.9147851  -1.951938 5.103865e-02
```

raw = TRUE

#it does not afect the ftted values obtained

```
fit2 <- lm(wage ~ poly(age, 4, raw = T), data = Wage)
coef(summary(fit2))
```

```
                              Estimate   Std. Error   t value     Pr(>|t|)
(Intercept)            -1.841542e+02 6.004038e+01 -3.067172 0.0021802539
poly(age, 4, raw = T)1  2.124552e+01 5.886748e+00  3.609042 0.0003123618
poly(age, 4, raw = T)2 -5.638593e-01 2.061083e-01 -2.735743 0.0062606446
poly(age, 4, raw = T)3  6.810688e-03 3.065931e-03  2.221409 0.0263977518
poly(age, 4, raw = T)4 -3.203830e-05 1.641359e-05 -1.951938 0.0510386498
```

#equivalent ways

```
fit2a <- lm(wage ~ age + I(age^2) + I(age^3) + I(age^4),
            data = Wage)
coef(fit2a)
```

```
  (Intercept)           age      I(age^2)      I(age^3)      I(age^4)
-1.841542e+02  2.124552e+01 -5.638593e-01  6.810688e-03 -3.203830e-05
```

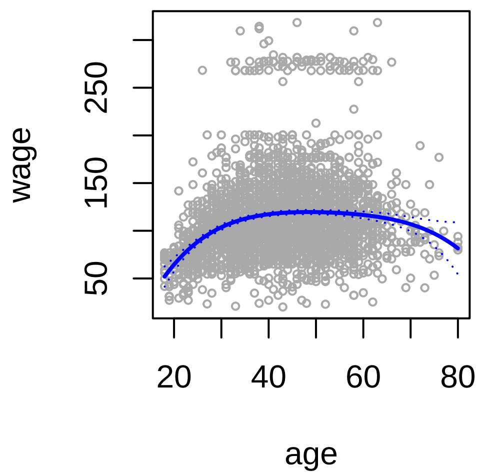
```
fit2b <- lm(wage ~ cbind(age, age^2, age^3, age^4),
            data = Wage)
```

```
agelims <- range(age)
age.grid <- seq(from = agelims[1], to = agelims[2]) # #create a grid of values for age
preds <- predict(fit, newdata = list(age = age.grid),
                 se = TRUE)
se.bands <- cbind(preds$fit + 2 * preds$se.fit,
                  preds$fit - 2 * preds$se.fit)
```

#plot the data and add the fit from the degree-4 polynomial

```
par(mfrow = c(1, 2), mar = c(4.5, 4.5, 1, 1),
    oma = c(0, 0, 4, 0))
plot(age, wage, xlim = agelims, cex = .5, col = "darkgrey")
title("Degree -4 Polynomial", outer = T)
lines(age.grid, preds$fit, lwd = 2, col = "blue")
matlines(age.grid, se.bands, lwd = 1, col = "blue", lty = 3)
```

Degree −4 Polynomial



#The ftted values obtained in either case are identical

```r
preds2 <- predict(fit2, newdata = list(age = age.grid),
                  se = TRUE)
max(abs(preds$fit - preds2$fit))
```

```
[1] 1.011813e-11
```

#ANOVA

```r
fit.1 <- lm(wage ~ age, data = Wage)
fit.2 <- lm(wage ~ poly(age, 2), data = Wage)
fit.3 <- lm(wage ~ poly(age, 3), data = Wage)
fit.4 <- lm(wage ~ poly(age, 4), data = Wage)
fit.5 <- lm(wage ~ poly(age, 5), data = Wage)
anova(fit.1, fit.2, fit.3, fit.4, fit.5)
```

```
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
```

```r
coef(summary(fit.5))
```

```
                Estimate Std. Error     t value     Pr(>|t|)
(Intercept)    111.70361  0.7287647 153.2780243 0.000000e+00
poly(age, 5)1  447.06785 39.9160847  11.2001930 1.491111e-28
poly(age, 5)2 -478.31581 39.9160847 -11.9830341 2.367734e-32
poly(age, 5)3  125.52169 39.9160847   3.1446392 1.679213e-03
poly(age, 5)4  -77.91118 39.9160847  -1.9518743 5.104623e-02
poly(age, 5)5  -35.81289 39.9160847  -0.8972045 3.696820e-01
```

```r
fit.1 <- lm(wage ~ education + age, data = Wage)
fit.2 <- lm(wage ~ education + poly(age, 2), data = Wage)
fit.3 <- lm(wage ~ education + poly(age, 3), data = Wage)
anova(fit.1, fit.2, fit.3)
```

```
Analysis of Variance Table

Model 1: wage ~ education + age
Model 2: wage ~ education + poly(age, 2)
Model 3: wage ~ education + poly(age, 3)
  Res.Df     RSS Df Sum of Sq        F Pr(>F)    
1   2994 3867992                                 
2   2993 3725395  1    142597 114.6969 <2e-16 ***
3   2992 3719809  1      5587   4.4936 0.0341 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
fit <- glm(I(wage > 250) ~ poly(age, 4), data = Wage,
           family = binomial)   #wage > 250 evaluates to a logical variable
```

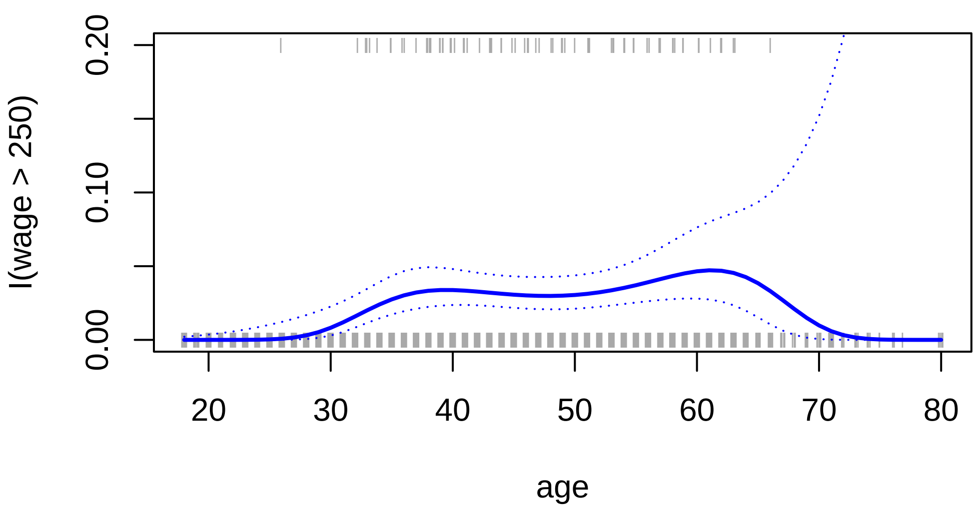
```r
preds <- predict(fit, newdata = list(age = age.grid), se = T)
```

```r
pfit <- exp(preds$fit) / (1 + exp(preds$fit))
se.bands.logit <- cbind(preds$fit + 2 * preds$se.fit,
                        preds$fit - 2 * preds$se.fit)
se.bands <- exp(se.bands.logit) / (1 + exp(se.bands.logit))
```

directly computed the probabilities by selecting the type = "response" option in the predict() function

```r
preds <- predict(fit, newdata = list(age = age.grid),
                 type = "response", se = T)
```

```r
plot(age, I(wage > 250), xlim = agelims, type = "n",
     ylim = c(0, .2))
points(jitter(age), I((wage > 250) / 5), cex = .5, pch = "|", col
       = "darkgrey")
lines(age.grid, pfit, lwd = 2, col = "blue")
matlines(age.grid, se.bands, lwd = 1, col = "blue", lty = 3)
```



fit a step function

```
table(cut(age, 4)) #df = 4
```

```
(17.9,33.5]   (33.5,49]   (49,64.5] (64.5,80.1] 
        750        1399         779          72 
```

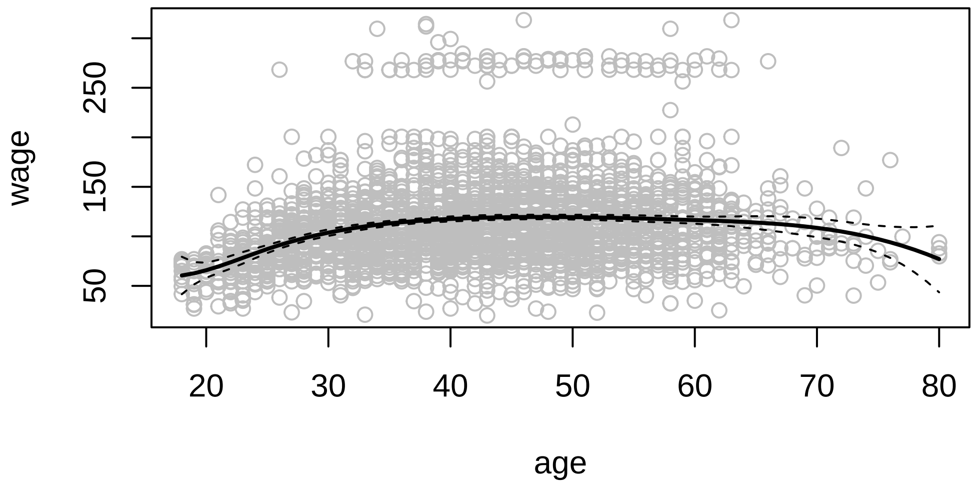
```
fit <- lm(wage ~ cut(age, 4), data = Wage)
coef(summary(fit))
```

```
                        Estimate Std. Error   t value     Pr(>|t|)
(Intercept)            94.158392   1.476069 63.789970 0.000000e+00
cut(age, 4)(33.5,49]   24.053491   1.829431 13.148074 1.982315e-38
cut(age, 4)(49,64.5]   23.664559   2.067958 11.443444 1.040750e-29
cut(age, 4)(64.5,80.1]  7.640592   4.987424  1.531972 1.256350e-01
```

Fitting wage to age using a regression spline is simple:

#7.8.2 Splines

```
library(splines)
fit <- lm(wage ~ bs(age, knots = c(25, 40, 60)), data = Wage) #prespecifed knots at ages 25,
pred <- predict(fit, newdata = list(age = age.grid), se = T)
plot(age, wage, col = "gray")
lines(age.grid, pred$fit, lwd = 2)
lines(age.grid, pred$fit + 2 * pred$se, lty = "dashed")
lines(age.grid, pred$fit - 2 * pred$se, lty = "dashed")
```




```
dim(bs(age, knots = c(25, 40, 60)))
```

```
[1] 3000    6
```

```
dim(bs(age, df = 6))
```

```
[1] 3000    6
```

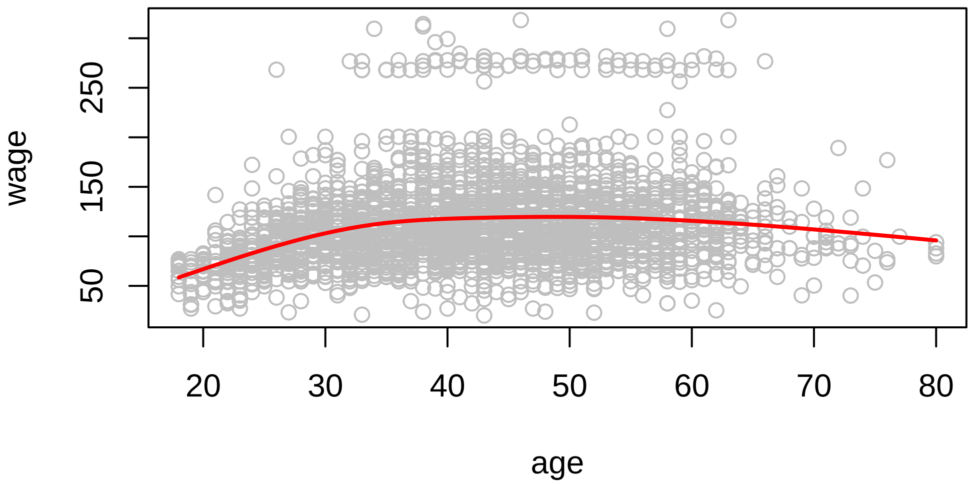
```
attr(bs(age, df = 6), "knots")
```

```
[1] 33.75 42.00 51.00
```

#bs() also has a degree argument, so we can ft splines of any degree. default degree = 3

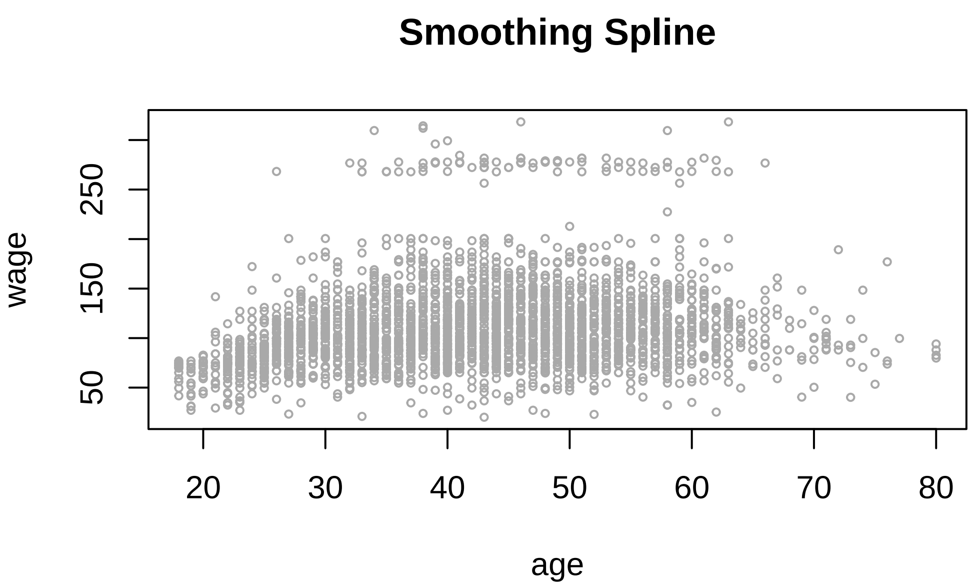
#In order to instead ft a natural spline, we use the ns() function

```
fit2 <- lm(wage ~ ns(age, df = 4), data = Wage)
pred2 <- predict(fit2, newdata = list(age = age.grid),se = T)
plot(age, wage, col = "gray")
lines(age.grid, pred2$fit, col = "red", lwd = 2)
```



#fit a smoothing spline, we use the smooth.spline()

```
plot(age, wage, xlim = agelims, cex = .5, col = "darkgrey")
title("Smoothing Spline")
```



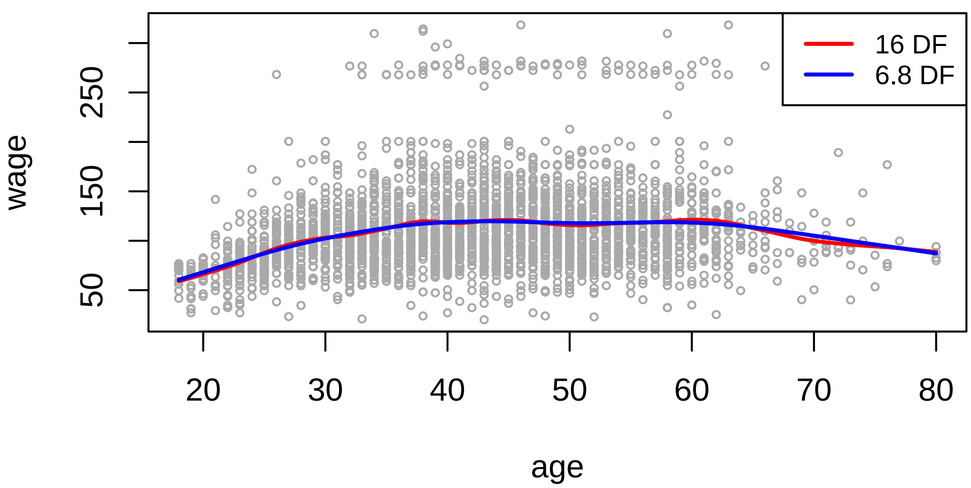

```
fit <- smooth.spline(age, wage, df = 16)
fit2 <- smooth.spline(age, wage, cv = TRUE)
```

```
Warning in smooth.spline(age, wage, cv = TRUE): cross-validation with
non-unique 'x' values seems doubtful
```

```
fit2$df
```

```
[1] 6.794596
```

```
plot(age, wage, xlim = agelims, cex = .5, col = "darkgrey")
lines(fit, col = "red", lwd = 2)
lines(fit2, col = "blue", lwd = 2)
legend("topright", legend = c("16 DF", "6.8 DF"),
       col = c("red", "blue"), lty = 1, lwd = 2, cex = .8)
```



#Notice that in the frst call to smooth.spline(), we specifed df = 16. The function then determines which value of leads to 16 degrees of freedom. In the second call to smooth.spline(), we select the smoothness level by crossvalidation; this results in a value of that yields 6.8 degrees of freedom.

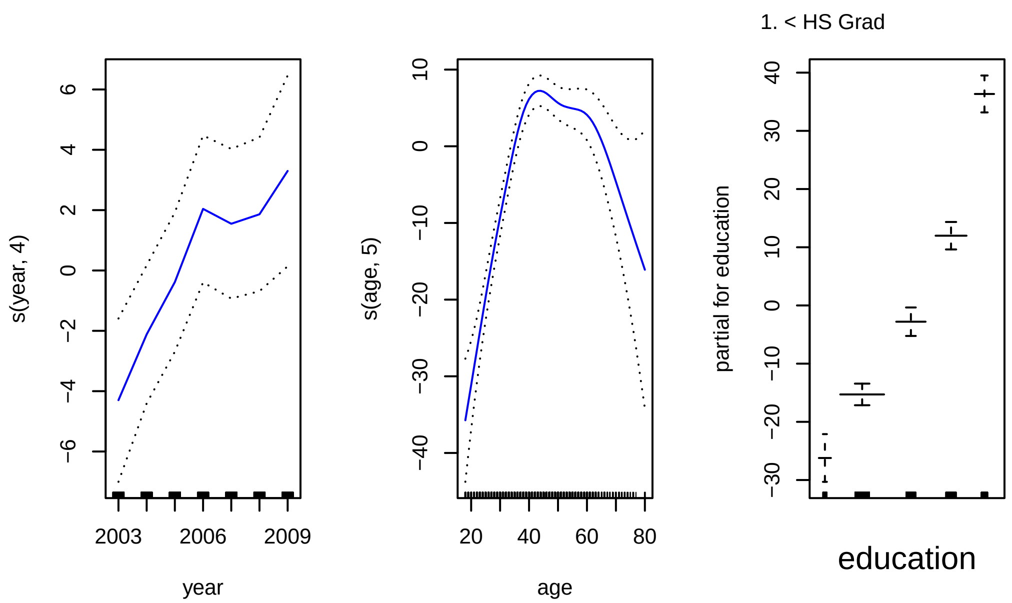
#7.8.3 GAMs

fit a GAM to predict wage using natural spline functions of year and age

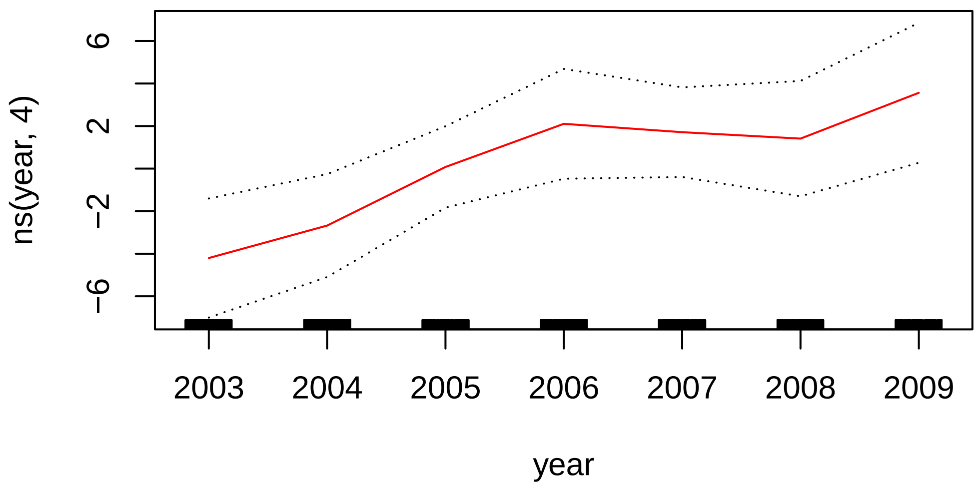
```
gam1 <- lm(wage ~ ns(year, 4) + ns(age, 5) + education ,
           data = Wage)
```

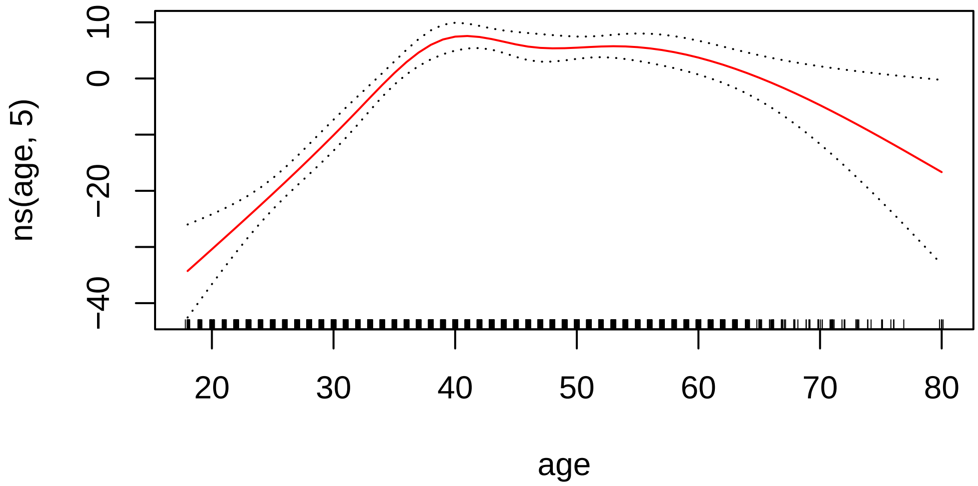
```
gam.m3 <- gam(wage ~ s(year, 4) + s(age, 5) + education, #year should have 4 degrees of freed
              data = Wage)
```

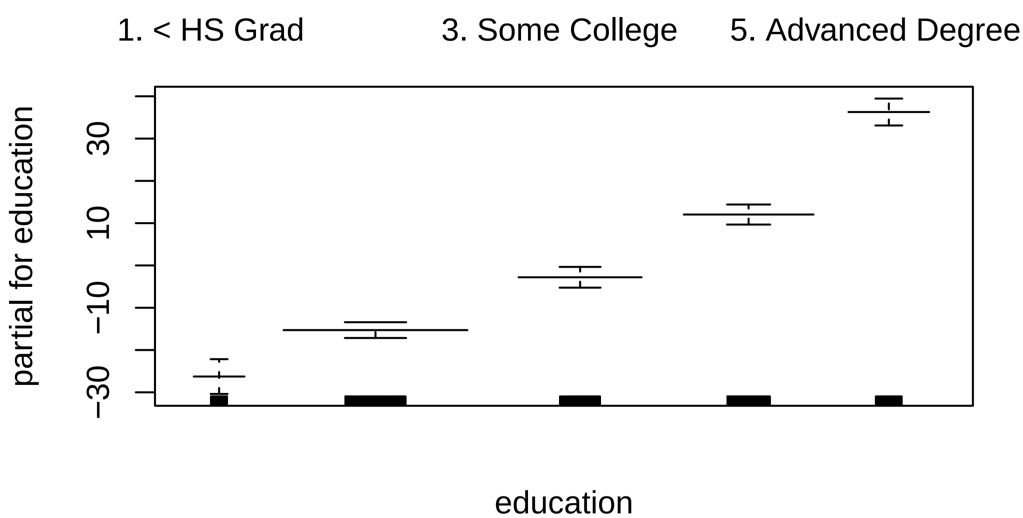
```
par(mfrow = c(1, 3))
plot(gam.m3, se = TRUE, col = "blue")
```




```
plot.Gam(gam1, se = TRUE, col = "red")
```







#:a GAM that excludes year (M1), #a GAM that uses a linear function of year (M2), #a GAM that uses a spline function of year (M3).

```
gam.m1 <- gam(wage ~ s(age, 5) + education ,
              data = Wage)
gam.m2 <- gam(wage ~ year + s(age, 5) + education ,
              data = Wage)
anova(gam.m1, gam.m2, gam.m3, test = "F")
```

```
Analysis of Deviance Table

Model 1: wage ~ s(age, 5) + education
Model 2: wage ~ year + s(age, 5) + education
Model 3: wage ~ s(year, 4) + s(age, 5) + education
```

```
  Resid. Df Resid. Dev Df Deviance       F    Pr(>F)    
1      2990    3711731                                  
2      2989    3693842  1  17889.2 14.4771 0.0001447 ***
3      2986    3689770  3   4071.1  1.0982 0.3485661    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

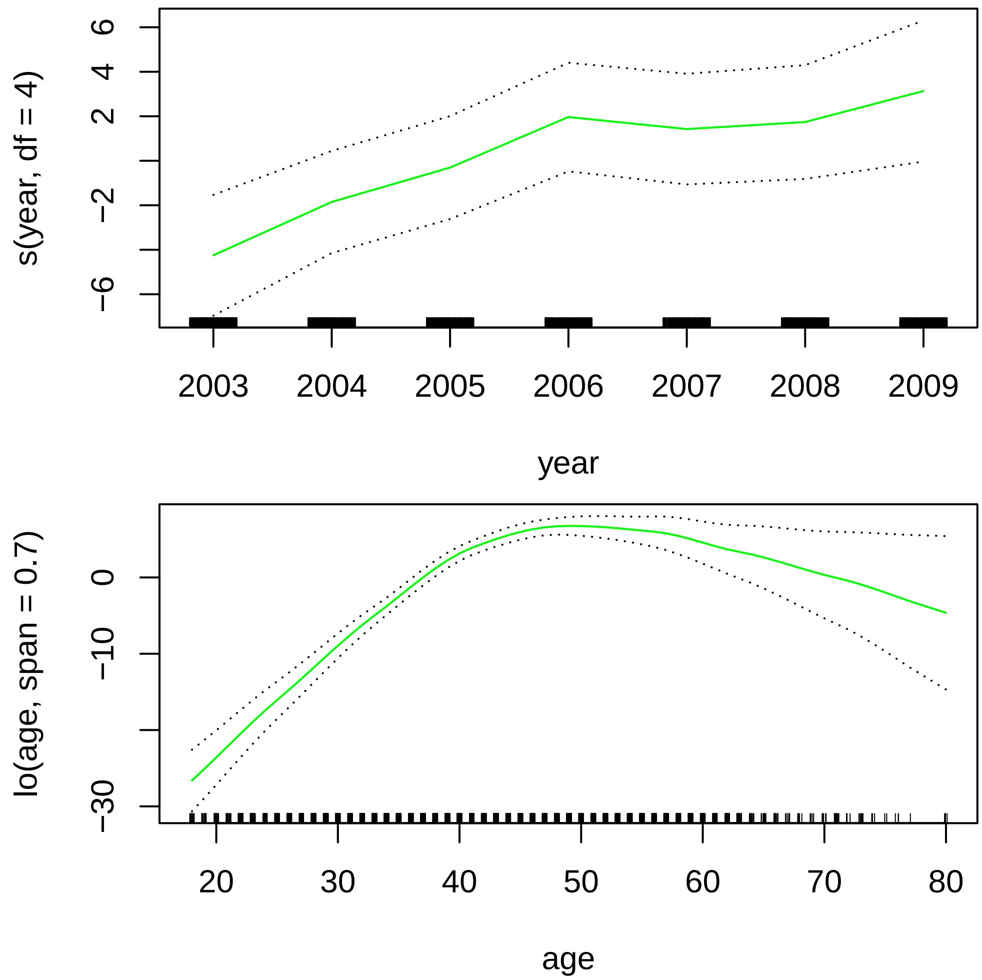
```r
summary(gam.m3)
```

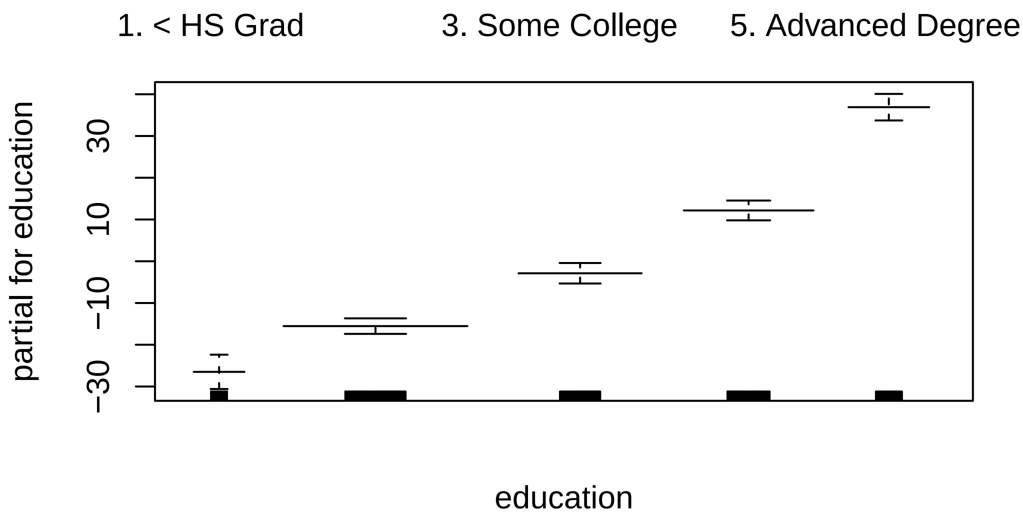
```

Call: gam(formula = wage ~ s(year, 4) + s(age, 5) + education, data = Wage)
Deviance Residuals:
    Min      1Q  Median      3Q     Max 
-119.43  -19.70   -3.33   14.17  213.48 

(Dispersion Parameter for gaussian family taken to be 1235.69)

    Null Deviance: 5222086 on 2999 degrees of freedom
Residual Deviance: 3689770 on 2986 degrees of freedom
AIC: 29887.75 

Number of Local Scoring Iterations: NA 

Anova for Parametric Effects
             Df  Sum Sq Mean Sq F value    Pr(>F)    
s(year, 4)    1   27162   27162  21.981 2.877e-06 ***
s(age, 5)     1  195338  195338 158.081 < 2.2e-16 ***
education     4 1069726  267432 216.423 < 2.2e-16 ***
Residuals  2986 3689770    1236                      
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Anova for Nonparametric Effects
            Npar Df Npar F  Pr(F)    
(Intercept)                          
s(year, 4)        3  1.086 0.3537    
s(age, 5)         4 32.380 <2e-16 ***
education                            
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
preds <- predict(gam.m2, newdata = Wage)
```

#use local regression fts as building blocks in a GAM, using the lo() function #s() function, indicate that we would like to use a smoothing spline

```
gam.lo <- gam(
  wage ~ s(year, df = 4) + lo(age, span = 0.7) + education,
  data = Wage
)
plot(gam.lo, se = TRUE, col = "green")
```






```
library(interp)
```

#fits a two-term model, in which the frst term is an interaction between year and age

```
gam.lo.i <- gam(wage ~ lo(year, age, span = 0.5) + education ,
                data = Wage)
```

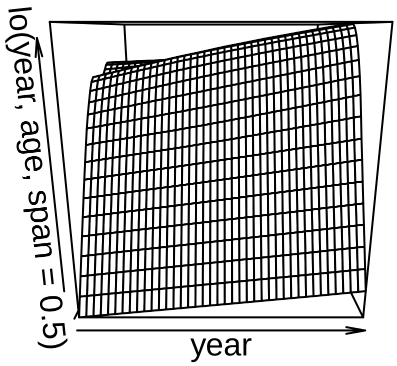
```
Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, :
liv too small.  (Discovered by lowesd)

Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, : lv
too small.  (Discovered by lowesd)

Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, :
liv too small.  (Discovered by lowesd)

Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, : lv
too small.  (Discovered by lowesd)
```

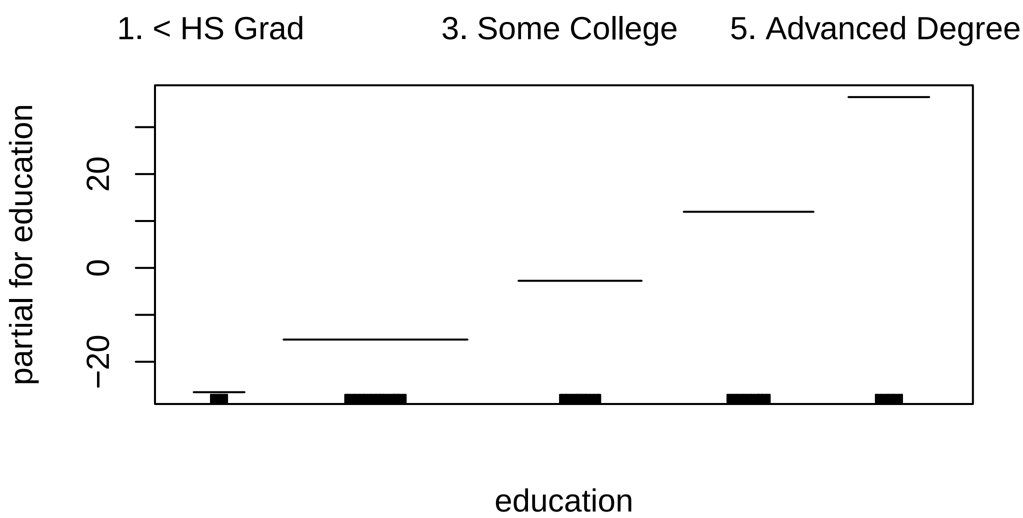
```
library(akima)
```

```
Attaching package: 'akima'
```

```
The following objects are masked from 'package:interp':

    aspline, bicubic, bicubic.grid, bilinear, bilinear.grid,
    franke.data, franke.fn, interp, interp2xyz, interpp
```

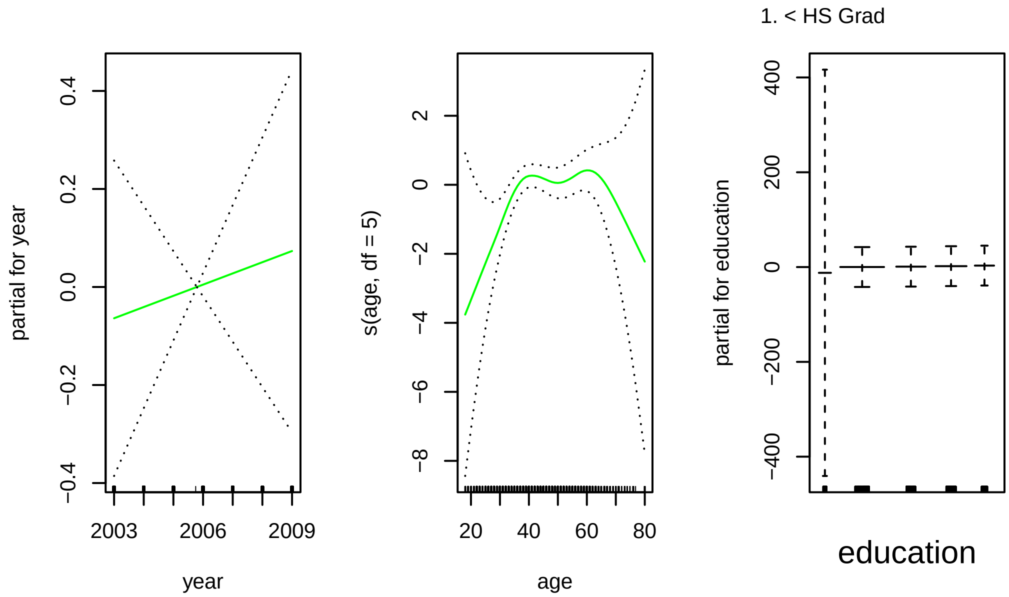
```
plot(gam.lo.i)
```





fit a logistic regression GAM

```
gam.lr <- gam( I(wage > 250) ~ year + s(age, df = 5) + education ,
               family = binomial, data = Wage)
par(mfrow = c(1, 3))
plot(gam.lr, se = T, col = "green")
```

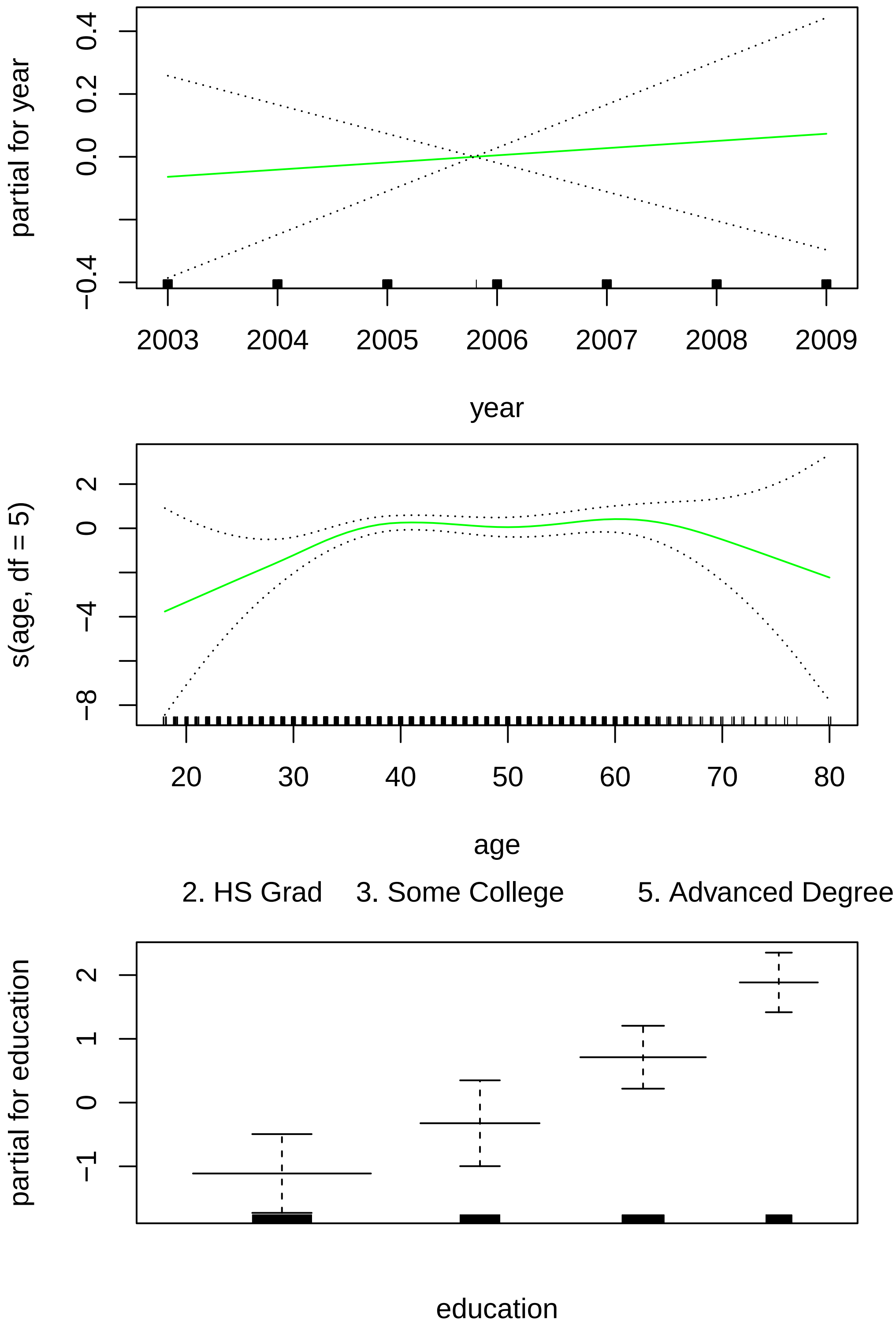



```
table(education, I(wage > 250))
```

```
education                   FALSE TRUE
  1. < HS Grad                268    0
  2. HS Grad                  966    5
  3. Some College             643    7
  4. College Grad             663   22
  5. Advanced Degree          381   45
```

#Hence, we ft a logistic regression GAM using all but this category

```
gam.lr.s <- gam( I(wage > 250) ~ year + s(age, df = 5) + education ,
                 family = binomial , data = Wage,
                 subset = (education != "1. < HS Grad") )
plot(gam.lr.s, se = T, col = "green")
```



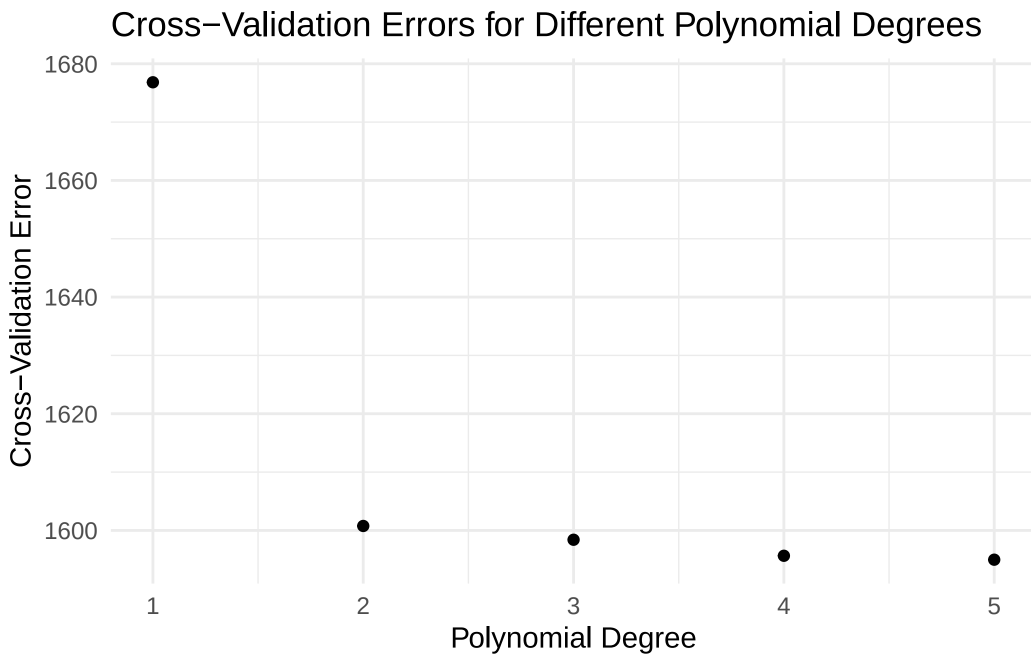
#6.a) Use cross-validation to select the optimal degree d for the polynomial.

```r
set.seed(1)
cv_errors <- rep(NA, 10)

for (d in 1:5) {
    model <- glm(wage ~ poly(age, d), data = Wage)
    cv_errors[d] <- cv.glm(Wage, model, K = 10)$delta[1]   #result different from python caus
}

optimal_degree <- which.min(cv_errors)
print(optimal_degree)
```

```
[1] 5
```

```r
cv_data <- data.frame(
  Degree = 1:5,
  CV_Error = cv_errors
)

ggplot(cv_data, aes(x = Degree, y = CV_Error))+
  geom_point() +
  labs(title = "Cross-Validation Errors for Different Polynomial Degrees",
       x = "Polynomial Degree",
       y = "Cross-Validation Error") +
  theme_minimal()
```

```
Warning: Removed 5 rows containing missing values or values outside the scale range
(`geom_point()`).
```



#6b)

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

for (k in 2:12) {
  Wage$age_cut <- cut(Wage$age, k)
  fit <- glm(wage ~ age_cut, data = Wage)  # Fit the step function model
  cv_errors[k] <- cv.glm(Wage, fit, K = 10)$delta[1]  # Perform 10-fold CV
}

optimal_cuts <- which.min(cv_errors[2:12]) + 1  # +1 because we started at k=2
print(paste("Optimal number of cuts:", optimal_cuts))
```

```
[1] "Optimal number of cuts: 11"
```

```
table(cut(age, 11)) #df = 11
```

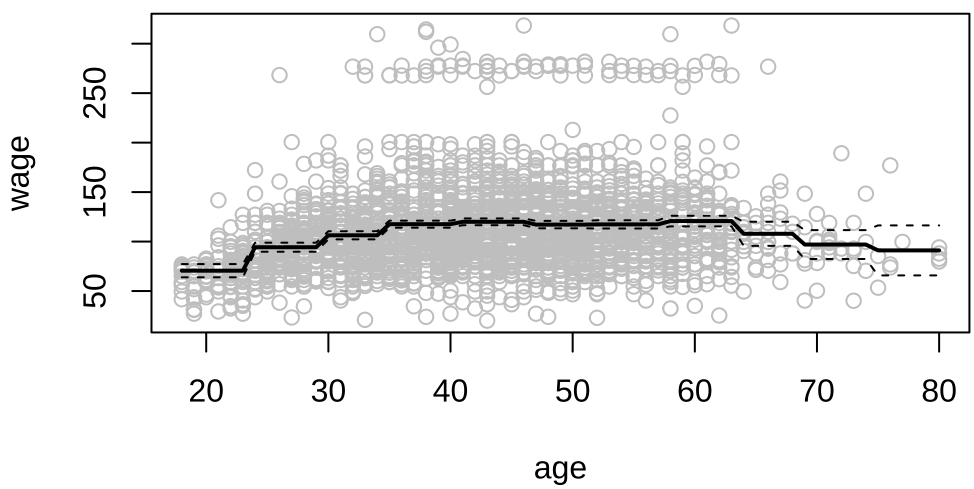
```
(17.9,23.6] (23.6,29.3] (29.3,34.9] (34.9,40.5] (40.5,46.2] (46.2,51.8]
        143         305         378         503         546         451
(51.8,57.5] (57.5,63.1] (63.1,68.7] (68.7,74.4] (74.4,80.1]
        368         223          43          30          10
```

```r
fit <- lm(wage ~ cut(age, 11), data = Wage)
coef(summary(fit))
```

```
                           Estimate Std. Error   t value     Pr(>|t|)
(Intercept)                70.58039   3.339779 21.133250 1.663885e-92
cut(age, 11)(23.6,29.3] 23.71870   4.047687  5.859815 5.139182e-09
cut(age, 11)(29.3,34.9] 35.76834   3.920945  9.122377 1.315720e-19
cut(age, 11)(34.9,40.5] 47.00149   3.784862 12.418282 1.459360e-34
cut(age, 11)(40.5,46.2] 49.39869   3.751726 13.166924 1.574538e-38
cut(age, 11)(46.2,51.8] 46.57080   3.832858 12.150410 3.415632e-33
cut(age, 11)(51.8,57.5] 46.78093   3.935540 11.886787 7.161447e-32
cut(age, 11)(57.5,63.1] 50.14208   4.278641 11.719159 4.806176e-31
cut(age, 11)(63.1,68.7] 37.29642   6.946084  5.369417 8.504251e-08
cut(age, 11)(68.7,74.4] 26.41431   8.020108  3.293510 1.000974e-03
cut(age, 11)(74.4,80.1] 20.49121  13.063619  1.568571 1.168538e-01
```

##11 is the same result with python

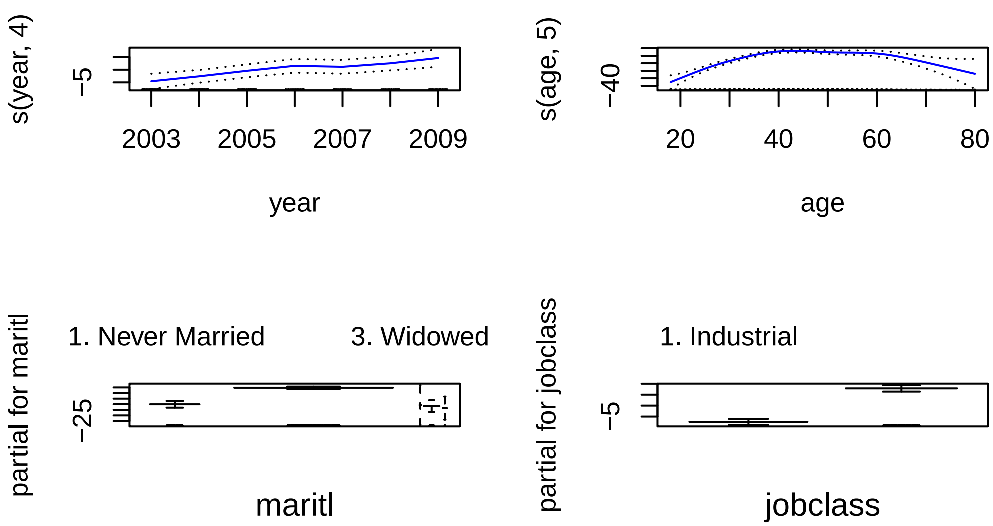
```r
age_grid <- seq(min(Wage$age), max(Wage$age), length.out = 100)
pred <- predict(fit, newdata = list(age = age.grid), se = T)
plot(age, wage, col = "gray")
lines(age.grid, pred$fit, lwd = 2)
lines(age.grid, pred$fit + 2 * pred$se, lty = "dashed")
lines(age.grid, pred$fit - 2 * pred$se, lty = "dashed")
```



#7. features such as marital status (maritl), job class (jobclass)

```r
gam1 <- gam(wage ~ s(year, 4) + s(age, 5) + maritl + jobclass,
           data = Wage)
```

```r
par(mfrow = c(2, 2))
plot(gam1, se = TRUE, col = "blue")
```

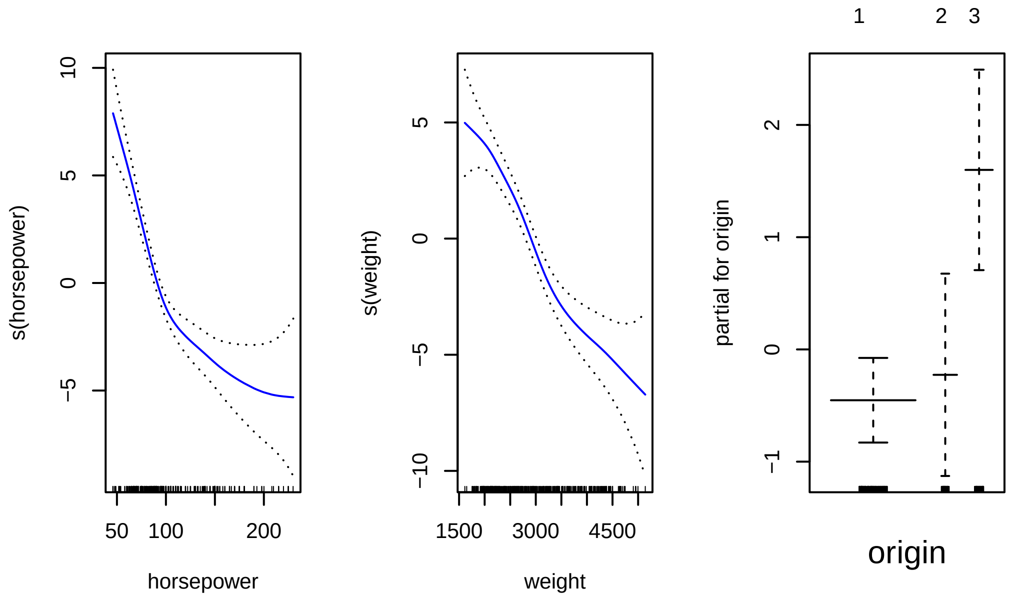


#same results with python

#8). Fit some of the non-linear models investigated in this chapter to the Auto data set. Is there evidence for non-linear relationships in this data set? Create some informative plots to justify your answer

```r
Auto$origin <- factor(Auto$origin)
```

```r
gam2 <- gam(mpg ~ s(horsepower) + s(weight) + origin,
           data = Auto)
par(mfrow = c(1, 3))
plot(gam2, se = TRUE, col = "blue")
```



#9. This question uses the variables dis (the weighted mean of distances to fve 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.

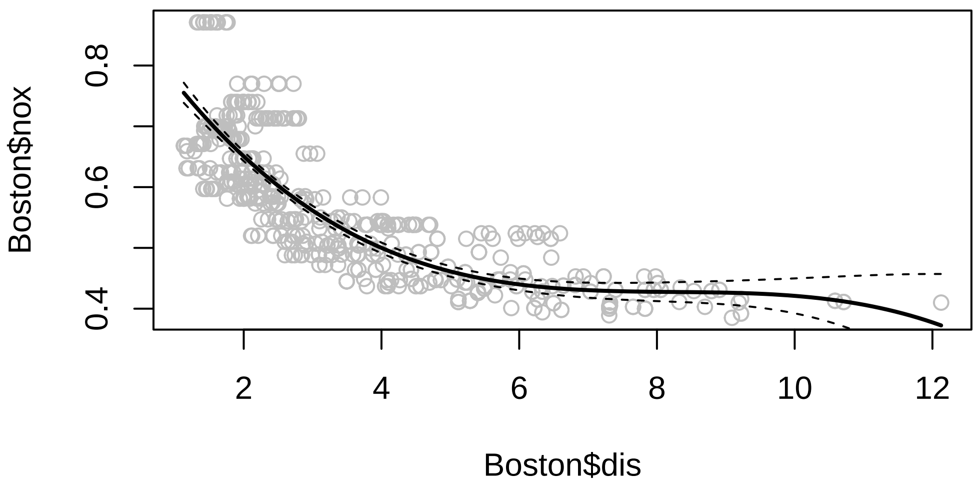
#a)Use the poly() function to fit a cubic polynomial regression to predict nox using dis

```
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
```

```
dis_grid <- seq(min(Boston$dis), max(Boston$dis), length.out = 100)
pred <- predict(fit, newdata = list(dis = dis_grid),
                 type = "response", se = T)
plot(Boston$dis, Boston$nox, col = "gray")
lines(dis_grid, pred$fit, lwd = 2)
lines(dis_grid, pred$fit + 2 * pred$se, lty = "dashed")
lines(dis_grid, pred$fit - 2 * pred$se, lty = "dashed")
```

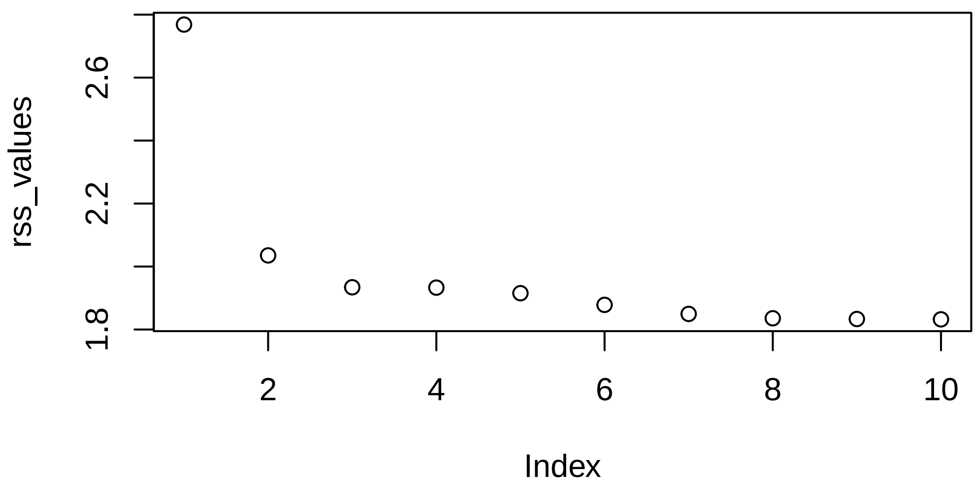


#9b)

```
set.seed(1)

dis_grid <- seq(min(Boston$dis), max(Boston$dis), length.out = 100)
rss_values <- rep(NA, 10)
for (d in 1:10) {
  fit <- glm(nox ~ poly(dis, d), data = Boston)
  preds <- predict(fit, newdata = list(dis = dis_grid))
  rss_values[d] <- sum(residuals(fit)^2)
}
print(rss_values)
```

```
 [1] 2.768563 2.035262 1.934107 1.932981 1.915290 1.878257 1.849484 1.835630
 [9] 1.833331 1.832171
```

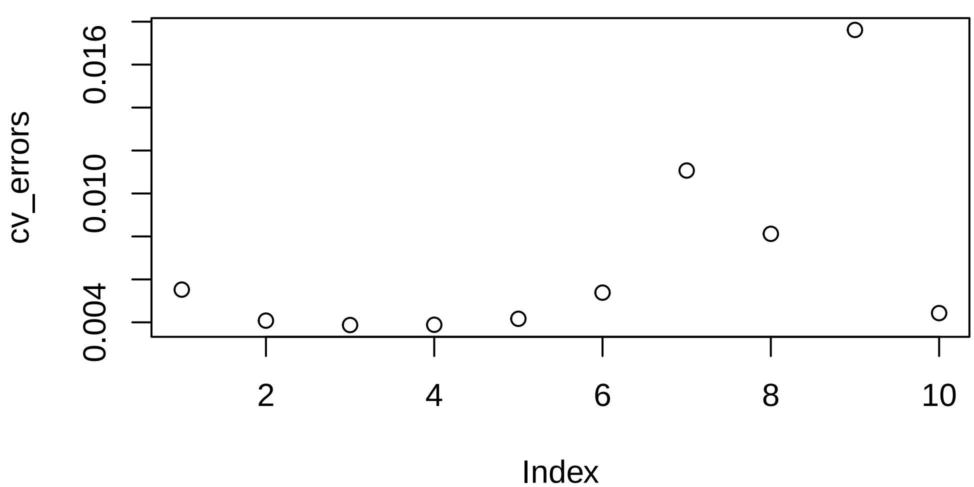
```r
plot(rss_values)
```



#d = 10 , 1.832171

#9c)

```r
set.seed(1)
cv_errors <- rep(NA, 10)
for (d in 1:10) {
  model <- glm(nox ~ poly(dis, d), data = Boston)
  cv_errors[d] <- cv.glm(Boston, model, K = nrow(Boston))$delta[1]
}
plot(cv_errors)
```



```r
optimal_degree <- which.min(cv_errors)
print(optimal_degree)
```

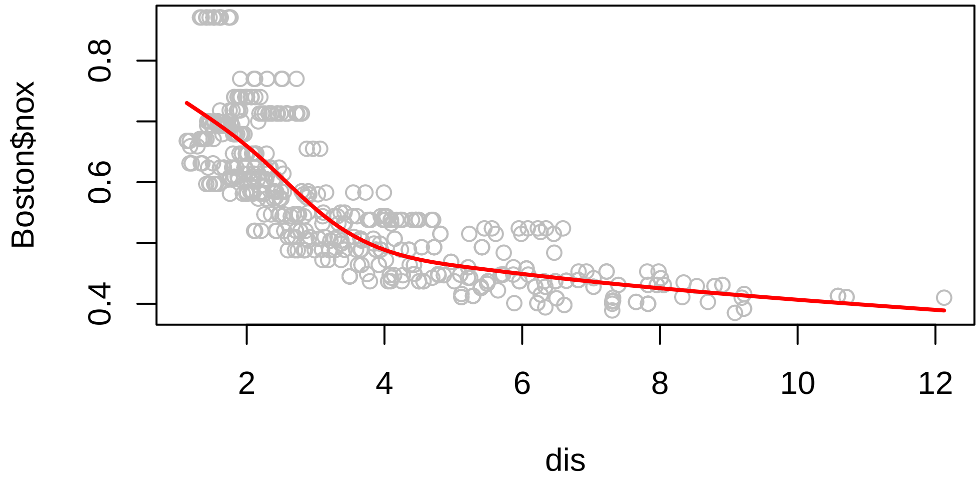
```
[1] 3
```

#optimal d = 3 with the least cv_error #9d)

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

```
[1] 3.20745
```

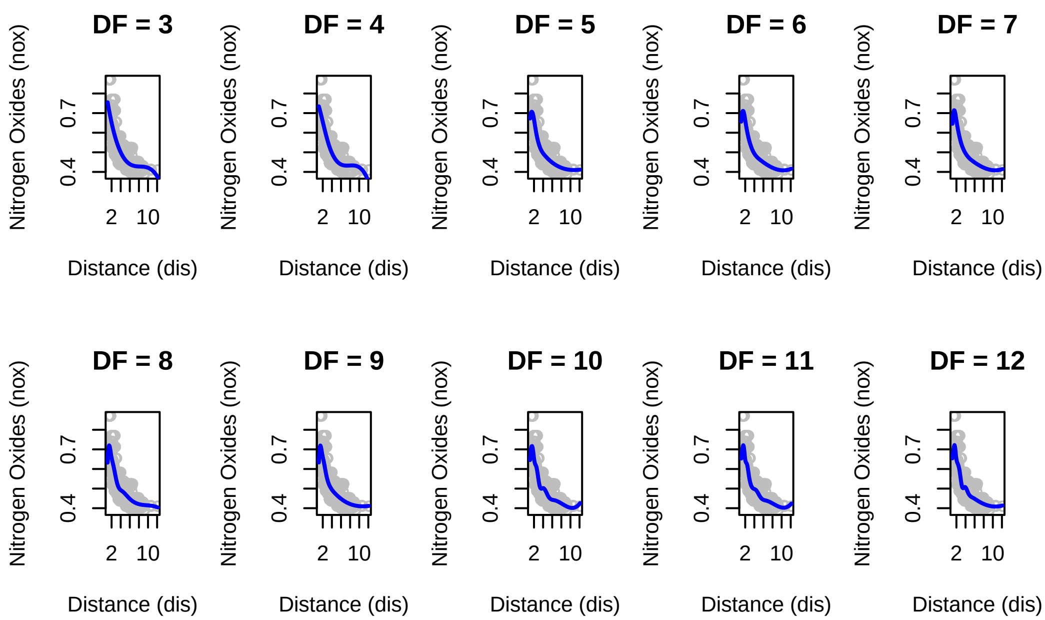
```r
dis <- Boston$dis
dis.grid <- seq(min(Boston$dis), max(Boston$dis), length.out = 100)
fit <- lm(nox ~ ns(dis, df = 4), data = Boston)
pred <- predict(fit, newdata = list(dis = dis.grid),se = T)
plot(dis, Boston$nox, col = "gray")
lines(dis.grid, pred$fit, col = "red", lwd = 2)
```



#e)

```r
dis_grid <- seq(min(Boston$dis), max(Boston$dis), length.out = 100)

rss_spline <- rep(NA, 10)

par(mfrow = c(2, 5))
```

```r
# Loop over degrees of freedom from 3 to 12 (for spline)
for (df in 3:12) {

  fit_spline <- lm(nox ~ bs(dis, df = df), data = Boston)
  preds_spline <- predict(fit_spline, newdata = list(dis = dis_grid))
  rss_spline[df - 2] <- sum(residuals(fit_spline)^2)

  plot(Boston$dis, Boston$nox, col = "gray", main = paste("DF =", df),
       xlab = "Distance (dis)", ylab = "Nitrogen Oxides (nox)")
  lines(dis_grid, preds_spline, col = "blue", lwd = 2)
}
```




```r
par(mfrow = c(5, 2))

# Print RSS values for each degree of freedom
print(rss_spline)
```

```
 [1] 1.934107 1.922775 1.840173 1.833966 1.829884 1.816995 1.825653 1.792535
 [9] 1.796992 1.788999
```

#9f) Perform cross-validation or another approach in order to select the best degrees of freedom

```
#set.seed(1)
#for (df in 3:15) {

  #fit_spline <- glm(nox ~ bs(dis, df = df), data = Boston)
  #cv_result <- suppressWarnings(cv.glm(Boston, fit_spline))
  #cv_error_k[df - 2] <- cv_result$delta[1]
#}

#print("Cross-validation errors (MSE) for different df values (3 to 15):")
#print(cv_error_k)

# Finding the best degrees of freedom based on the lowest MSE
#best_df_index <- which.min(cv_error_k) + 2
#best_df_index
```

#10 #a)

```
y <- College$Outstate
x <- College[, -which(names(College) == "Outstate")]
```

```
set.seed(66)
train <- sample(1:nrow(x), nrow(x) / 2)
test <- (-train)
y.test <- y[test]
y.test <- y[test]
x.train <- x[train, ]
x.test <- x[test, ]
y.train <- y[train]
```

```
train_data <- cbind(x.train, Outstate = y.train)
test_data <- cbind(x.test, Outstate = y.test)
```

```
regfit.fwd <- regsubsets(Outstate ~ ., data = train_data, nvmax = 17, method = "forward")
summary(regfit.fwd)
```

```
Subset selection object
Call: regsubsets.formula(Outstate ~ ., data = train_data, nvmax = 17,
    method = "forward")
17 Variables  (and intercept)
            Forced in Forced out
```

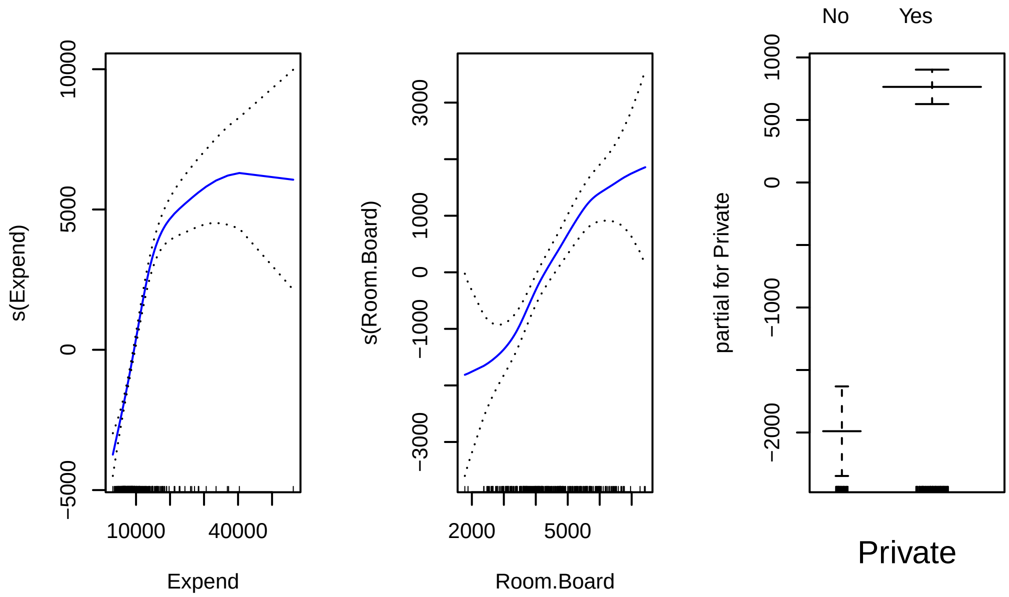
```
PrivateYes      FALSE      FALSE
Apps            FALSE      FALSE
Accept          FALSE      FALSE
Enroll          FALSE      FALSE
Top10perc       FALSE      FALSE
Top25perc       FALSE      FALSE
F.Undergrad     FALSE      FALSE
P.Undergrad     FALSE      FALSE
Room.Board      FALSE      FALSE
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 ) "*"          "*"    "*"
```

#Room.Board #expend #PrivateYes the predictors change if the training sample change #but the 2/3 of the predictors are the same with python

#(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 fndings

```r
gam <- gam(Outstate ~ s(Expend) + s(Room.Board) + Private,
           data = test_data)

par(mfrow = c(1, 3))
plot(gam, se = TRUE, col = "blue")
```



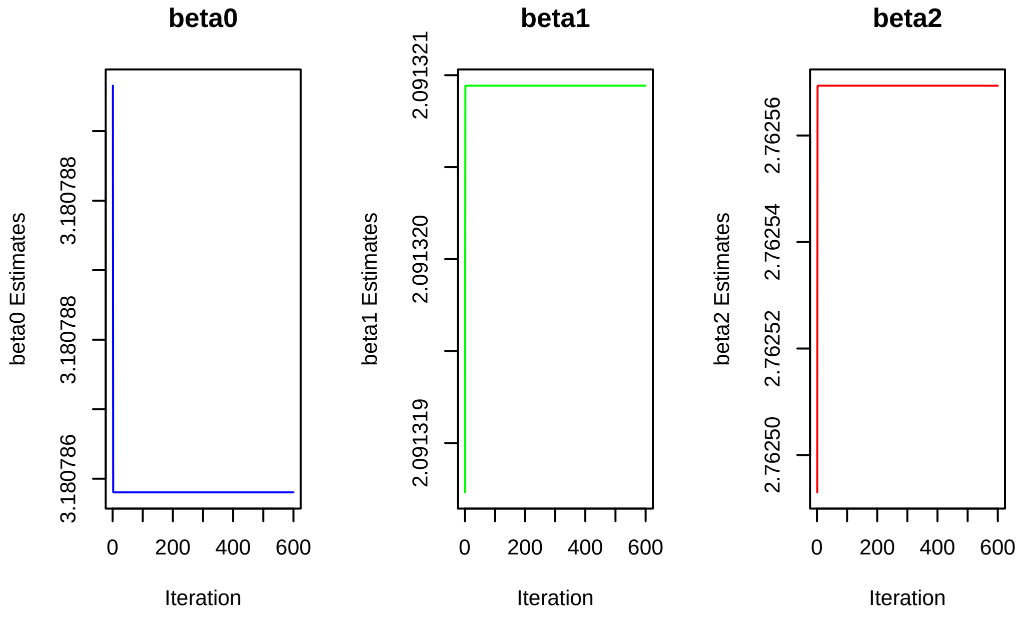
#11.

```r
# a)
set.seed(66)
X1 <- rnorm(100, mean = 0, sd = 1)
X2 <- rnorm(100, mean = 0, sd = 1)
Y <- 3 + 2 * X1 + 3 * X2 + rnorm(100, mean = 0, sd = 1)
```

```r
#b)
beta1 <- 0

#c)
a <- Y - beta1 * X1
beta2 <- lm(a ~ X2)$coef[2]

#d)
a <- Y - beta2 * X2
beta1 <- lm(a ~ X1)$coef[2]


#e)
iterations <- 600
beta0_vals <- beta1_vals <- beta2_vals <- numeric(iterations)
beta2 <- 0
```

```r
for (i in 1:iterations) {
  # (c) Keep beta1 fixed, fit Y - beta1*X1 = beta0 + beta2*X2 + error
  a <- Y - beta1 * X1
  beta2 <- lm(a ~ X2)$coef[2]  # Update beta2

  # (d) Keep beta2 fixed, fit Y - beta2*X2 = beta0 + beta1*X1 + error
  a <- Y - beta2 * X2
  beta1 <- lm(a ~ X1)$coef[2]  # Update beta1

  # Store beta estimates for each iteration
  beta0_vals[i] <- lm(a ~ X1)$coef[1]
  beta1_vals[i] <- beta1
  beta2_vals[i] <- beta2
}
```

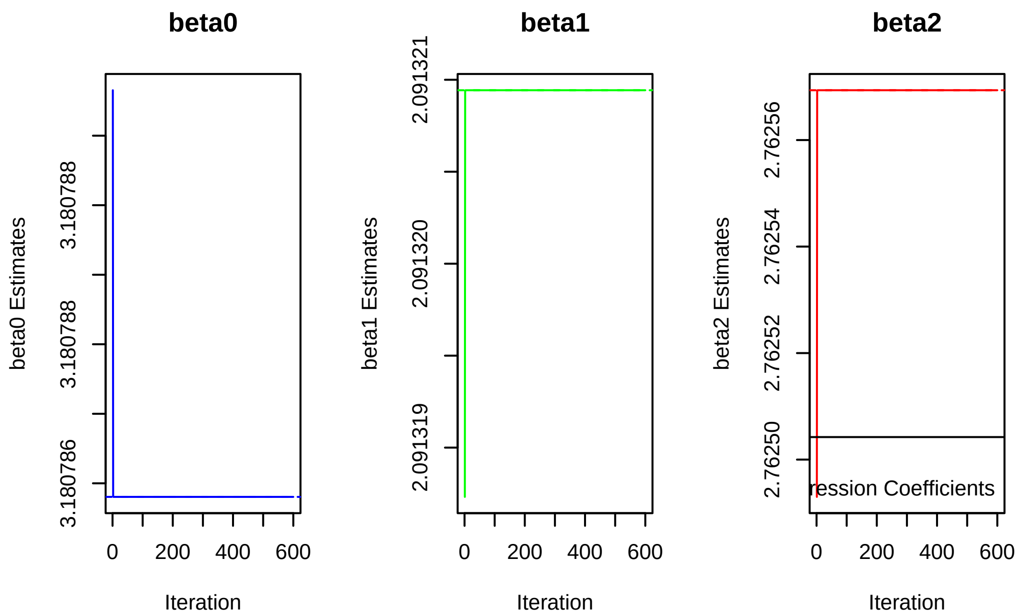
```r
par(mfrow = c(1, 3))

plot(1:iterations, beta0_vals, type = "l", col = "blue",
     ylab = "beta0 Estimates", xlab = "Iteration", main = "beta0")
plot(1:iterations, beta1_vals, type = "l", col = "green",
     ylab = "beta1 Estimates", xlab = "Iteration", main = "beta1")
plot(1:iterations, beta2_vals, type = "l", col = "red",
     ylab = "beta2 Estimates", xlab = "Iteration", main = "beta2")
```



```r
# f)
fit <- lm(Y ~ X1 + X2)
summary(fit)
```

```
Call:
lm(formula = Y ~ X1 + X2)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.6423 -0.7321 -0.0059  0.8477  3.1870 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   3.1808     0.1058   30.06   <2e-16 ***
X1            2.0913     0.1094   19.12   <2e-16 ***
X2            2.7626     0.1211   22.81   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.056 on 97 degrees of freedom
Multiple R-squared:  0.8985,	Adjusted R-squared:  0.8965 
F-statistic: 429.5 on 2 and 97 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(1, 3))
plot(1:iterations, beta0_vals, type = "l", col = "blue",
     ylab = "beta0 Estimates", xlab = "Iteration", main = "beta0")
abline(h = coef(fit)[1], col = "blue", lty = 2)
plot(1:iterations, beta1_vals, type = "l", col = "green",
     ylab = "beta1 Estimates", xlab = "Iteration", main = "beta1")
abline(h = coef(fit)[2], col = "green", lty = 2)
plot(1:iterations, beta2_vals, type = "l", col = "red",
     ylab = "beta2 Estimates", xlab = "Iteration", main = "beta2")
abline(h = coef(fit)[3], col = "red", lty = 2)

legend("bottomright", legend = c("Backfitting", "Multiple Regression Coefficients"),
       col = c("black", "black"), lty = c(1, 2))
```




```
tolerance <- 0.00001

fit <- lm(Y ~ X1 + X2)
beta0_mlr <- coef(fit)[1]
beta1_mlr <- coef(fit)[2]
beta2_mlr <- coef(fit)[3]

# Check convergence in each iteration
for (i in 1:iterations) {
  if (abs(beta0_vals[i] - beta0_mlr) < tolerance &&
      abs(beta1_vals[i] - beta1_mlr) < tolerance &&
      abs(beta2_vals[i] - beta2_mlr) < tolerance) {
    cat("Convergence reached at iteration", i, "\n")
    break
  }
}
```

```
Convergence reached at iteration 2
```

```
if (i == iterations) {
  cat("No convergence within the tolerance of", tolerance, "after", iterations, "iterations.\
}
```