Yonsei University, Department of Applied Statistics

Chapter 7 실습

12기 YBIGTA 신보현

April 6, 2019

1. Polynomial Regression and Step Functions

figure 7.1을 만들어보자.

```
knitr::opts_chunk$set(comment=NA, fig.width=4, fig.height=4,fig.align='center',message=FALSE,warning=FA
library(ISLR)
attach(Wage)
dim(Wage)
```

```
## [1] 3000   11
```

```
names(Wage)
```

```
##  [1] "year"       "age"        "maritl"     "race"       "education"
##  [6] "region"     "jobclass"   "health"     "health_ins" "logwage"
## [11] "wage"
```

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

다항 회귀를 시행할 때 주의할 점은, 예측 변수를 x, x^2, x^3 이런 식으로 설정하면 이의 상관계수가 1에 가깝기 때문에 다중 공선성의 문제가 발생할 수 있다. 위와 같이 승수를 그대로 올린 R코드는 아래와 같다.

```
poly.fit.raw = lm(wage~poly(age,4,raw=T),data=Wage)
coef(summary(poly.fit.raw))
```

```
                            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
```

만약에 위와 같이 다항회귀를 시행한다면 3차 다항회귀에서 3차까지의 계수와 4차 다항회귀에서 3차까지의 계수가 달라지게 된다. R 코드를 통해 직접 비교해보자.

```
poly.fit.raw.3 = lm(wage~poly(age,3,raw=T),data=Wage)
coef(summary(poly.fit.raw.3))
```

```
                            Estimate   Std. Error   t value     Pr(>|t|)
(Intercept)            -7.524391e+01 2.218373e+01 -3.391851 7.032236e-04
poly(age, 3, raw = T)1  1.018999e+01 1.605228e+00  6.348002 2.511372e-10
poly(age, 3, raw = T)2 -1.680286e-01 3.686021e-02 -4.558535 5.357433e-06
poly(age, 3, raw = T)3  8.494522e-04 2.702449e-04  3.143268 1.687063e-03
```

raw=T 옵션은 말 그대로 4승을 하라는 명령이다.
4차 다항 회귀를 시행할 때에, 3차 다항회귀의 결과를 그대로 유지하면서 4차 항에 대한 계수만 추가하고 싶을 것이다. 이를 위해서 기본 값으로 raw=F를 그대로 실행하면 되는데, 이는 서로 독립인(orthogonal) 벡터를 칼럼으로 하는 행렬을 반환하여 4차 다항 회귀를 실시 하였을 때, 그 아래 차수 계수는 그대로 유지하게 해준다. 결과를 통해 살펴보자.

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

```
poly.fit.3 = lm(wage~poly(age,3),data=Wage)
coef(summary(poly.fit.3))
```

```
               Estimate Std. Error    t value     Pr(>|t|)
(Intercept)    111.7036  0.7290826 153.211181 0.000000e+00
poly(age, 3)1  447.0679 39.9334995  11.195309 1.570802e-28
```

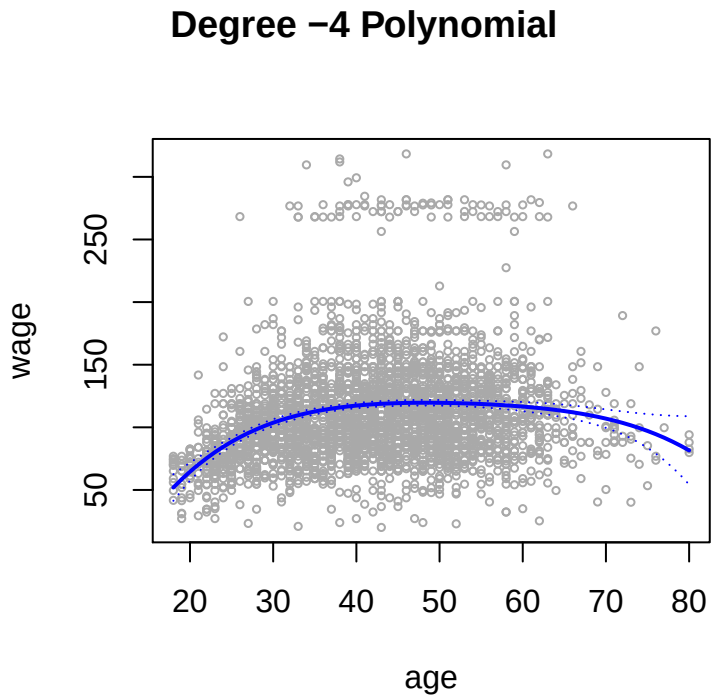
```
poly(age, 3)2 -478.3158 39.9334995 -11.977808 2.511784e-32
poly(age, 3)3  125.5217 39.9334995   3.143268 1.687063e-03
```

즉, poly(age,4,raw=T)은 age 칼럼을 그대로 1승, 2승, 3승, 4승을 한 행렬이고 poly(age,4,raw=F)은 poly(age,4,raw=T) 칼럼의 선형 결합이기 때문에 결국은 둘이 동일한 예측을 하는 동일한 모델이고 parameterization에서만 차이를 보인다. 예를 들어 이 둘의 fitted values은 동일하다.

```
all.equal(fitted(poly.fit),fitted(poly.fit.raw))


[1] TRUE
```

이제 figure 7.1의 왼쪽 그림을 그려보자.

```
agelims =range(age)
age.grid=seq (from=agelims [1], to=agelims [2])
preds=predict (poly.fit ,newdata =list(age=age.grid),se=TRUE)
se.bands=cbind(preds$fit +2* preds$se.fit ,preds$fit -2* preds$se.fit)
par(mfrow =c(1,1) ,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)
```



이제 5차 다항 회귀까지 적합을 하며 어떤 다항 회귀가 가장 적합한지 알아본다. 이를 위해 anova() 함수를 사용을 하는데, 이는 모델 $\mathcal{M}_1$으로도 데이터를 설명하기에 충분하다는 귀무가설과 좀 더 복잡한 모델인 $\mathcal{M}_2$가 필요하다는 대립가설에 대한 가설 검정이다. 이를 사용하기 위해서는 $\mathcal{M}_1$가 $\mathcal{M}_2$에 반드시 포함되어 있어야 한다. 즉, 한쪽의 예측변수가 다른 쪽의 예측변수에 모두 포함되어야 한다.

```
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
```

모델 1과 모델 2를 비교하는 검정에서 p-value가 상당히 유의하게 나왔으므로, age를 선형 예측 변수로 했을 때의 단순 선형 회귀는 별로 유의하지 않음을 확인할 수 있다. 모델 3과 모델 4를 비교하는 검정의 p-value는 0.05를 살짝 넘으므로 근소한 차이로 귀무가설을 기각하지 못한다. 하지만 모델 4와 모델 5를 비교하는 검정에서는 p-value가 크게 나와서 모델 5, 즉 5차 다항 회귀는 이 데이터에 적절하지 않음을 통계적으로 확인할 수 있다.
이는 또한 5차 다항회귀를 했을 때의 5차 항의 계수가 유의하지 않은 결과로부터도 확인할 수 있다.

```
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
```

다음으로 개인이 연 $250,000 이상을 버는지에 대한 분류 문제로 들어가보자. 로지스틱 회귀는 glm 함수에서 family='binomial' 옵션을 둠으로써 실행할 수 있다.

```
fit.logistic = glm(I(wage>250)~poly(age,4),data=Wage,family='binomial')
pred.logistic = predict(fit.logistic,newdata=list(age=age.grid),se=T)
summary(fit.logistic)
```

```

Call:
glm(formula = I(wage > 250) ~ poly(age, 4), family = "binomial",
    data = Wage)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
```

```
-0.3110  -0.2607  -0.2488  -0.1791   3.7859

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    -4.3012     0.3451 -12.465  < 2e-16 ***
poly(age, 4)1  71.9642    26.1176   2.755  0.00586 **
poly(age, 4)2 -85.7729    35.9043  -2.389  0.01690 *
poly(age, 4)3  34.1626    19.6890   1.735  0.08272 .
poly(age, 4)4 -47.4008    24.0909  -1.968  0.04912 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 730.53  on 2999  degrees of freedom
Residual deviance: 701.22  on 2995  degrees of freedom
AIC: 711.22

Number of Fisher Scoring iterations: 9
```

I(wage>250)은 wage 값이 250이상인 데이터에 대해서는 TRUE, 그렇지 않으면 FALSE를 반환한다. 로지스틱 회귀로 추정한 계수는 logit에 대한 추정 계수이다. 다시 말해서,

$$log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X + \cdots + \beta_4 X^4$$

따라서 추정된 계수나 신뢰구간도 모두 logit에 관한 것이므로 이를 우리가 추정하고자 했던 $p(X) = Pr(Y = 1 \mid X) = \frac{exp(\boldsymbol{X\beta})}{1 + exp(\boldsymbol{X\beta})}$ 에 대하여 변경해야 한다.

```
pfit = exp(pred.logistic$fit)/(1+exp(pred.logistic$fit))
se.bands.logit = cbind(pred.logistic$fit +2* pred.logistic$se.fit , pred.logistic$fit -2*pred.logistic$
se.bands = exp(se.bands.logit)/(1+exp(se.bands.logit))
```

하지만 predict 함수에서 type='response'을 통해 추정된 확률을 뽑아낼 수도 있다.

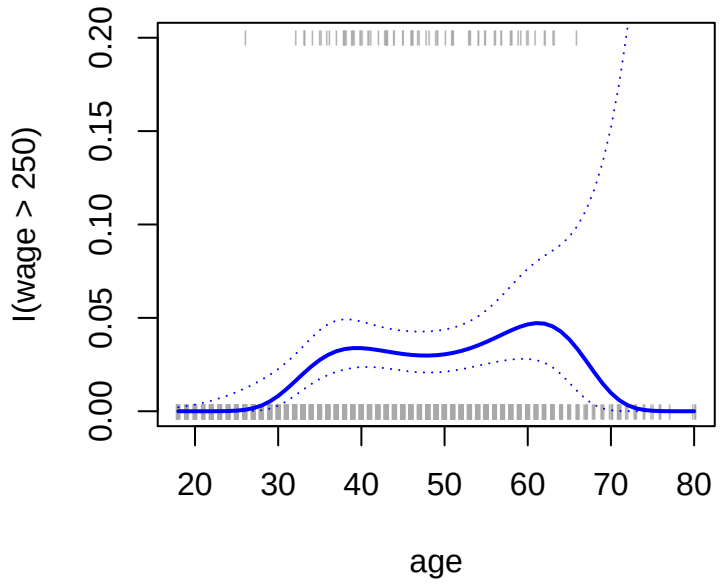
```
preds = predict(fit.logistic,newdata=list(age=age.grid),se=T,type='response')
names(preds)


[1] "fit"            "se.fit"         "residual.scale"
```

```
all.equal(preds$fit,pfit)
```

```
[1] TRUE
```

마지막으로 그래프로 그려본다.

```
par(mfrow =c(1,1) ,mar=c(4.5 ,4.5 ,1 ,1) ,oma=c(0,0,4,0))
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)
```



step function을 적합하기 위해서, cut() 함수를 사용한다.

```
table(cut(age,4))
```

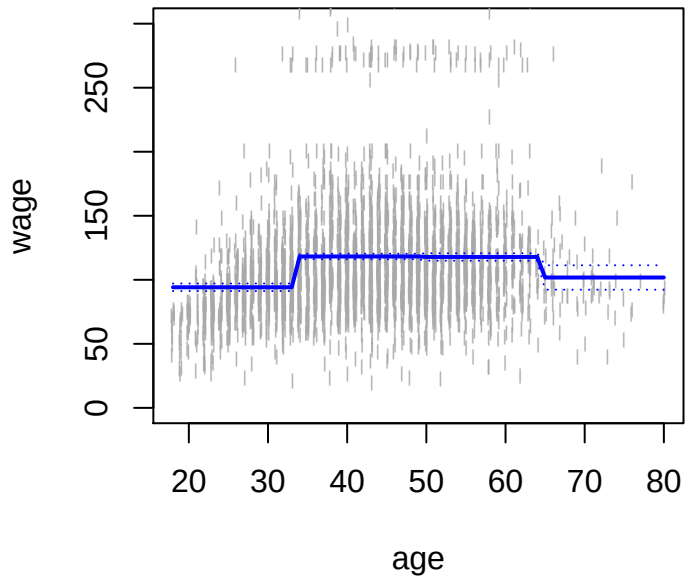
```
(17.9,33.5]   (33.5,49]   (49,64.5] (64.5,80.1]
        750        1399         779          72
```

```
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
```

여기서 cut() 함수는 자동적으로 33.5, 49, 64.5의 cutpoints을 선택했다. 직접 cutpoints을 정하고 싶을 때에는, breaks 옵션을 사용하면 된다. 그래프를 그리면 아래와 같다.

```
preds.cut = predict(fit,newdata=list(age=age.grid),se=T)
se.bands.cut = cbind(preds.cut$fit +2* preds.cut$se.fit , preds.cut$fit -2*preds.cut$se.fit)
plot(age ,wage ,xlim=agelims ,type ="n",ylim=c(0,300))
points(jitter(age), wage ,cex =.5, pch ="|",         col =" darkgrey ")
lines(age.grid ,preds.cut$fit ,lwd =2, col ="blue")
matlines(age.grid ,se.bands.cut ,lwd =1, col ="blue",lty =3)
```



로지스틱 회귀도 위와 동일하게 하면 된다.

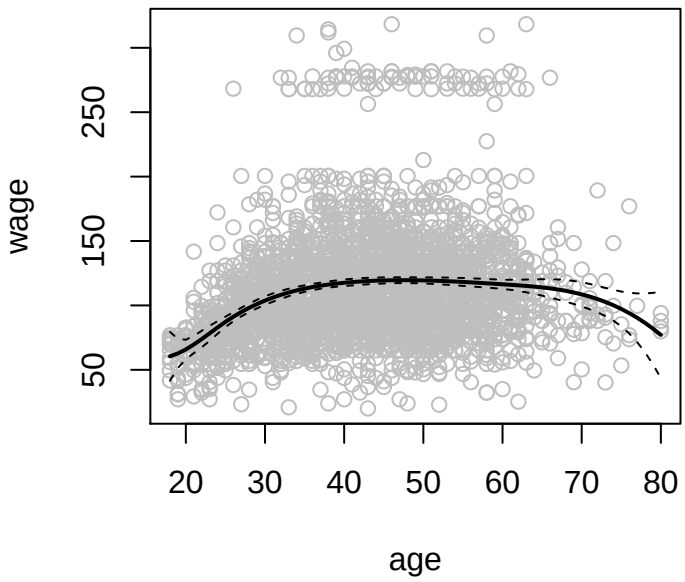
2. Splines

regression splines을 적합하기 위해서 splines library을 이용한다. 7.4단원에서, 적절한 basis 함수를 통해 regression splines을 적합할 수 있음을 확인했다. bs() 함수는 명시된 knots와 함께 전체 basis functions을 생성한다. 기본값으로, cubic splines가 실행된다.

```
library(splines)
fit = lm(wage~bs(age,knots=c(25,40,60)),data=Wage)
names(fit)
```

```
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "xlevels"       "call"          "terms"         "model"
```

```
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')
```



bs() 함수에 대해서 살펴보자. bs(age,knots=c(25,40,60))을 통해 knots를 명시하여 age에 대한 basis function을 만들었다. bs() 함수는 df을 명시하여 basis function의 개수를 정할 수도 있다. cubic spline에서 df는 $3 + 4 = 7$

인데 절편항으로 인해 자유도 1이 사용이 되어 df가 6이 된다. 따라서 bs(age,df=6) 이라 하면 knots가 3개인 cubic spline에 대한 basis function을 구하는 것과 동일하다. 그런데 이를 시행해보면 knots의 위치는 균일하게 생성됨을 확인할 수 있다.

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

```
  25%   50%   75%
33.75 42.00 51.00
```

bs() 함수는 또한 degree 옵션도 있어서, cubic spline이 아닌 2차, 4차 spline을 할 수도 있다.
natural spline을 적합하기 위해서는 ns() 함수를 이용한다.

```
fit2 = lm(wage~ns(age,df=4),data=Wage)
names(fit2)
```

```
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "xlevels"       "call"          "terms"         "model"
```

```
pred2 = predict(fit2,newdata=list(age=age.grid),se=T)
#lines(age.grid, pred2$fit, col="red", lwd=2)
```

bs 함수와 마찬가지로 knots을 구체적으로 명시할 수 있다.
smoothing spline을 적합하기 위해서, smooth.spline() 함수를 이용한다. figure 7.8은 아래의 코드로 만들어졌다.

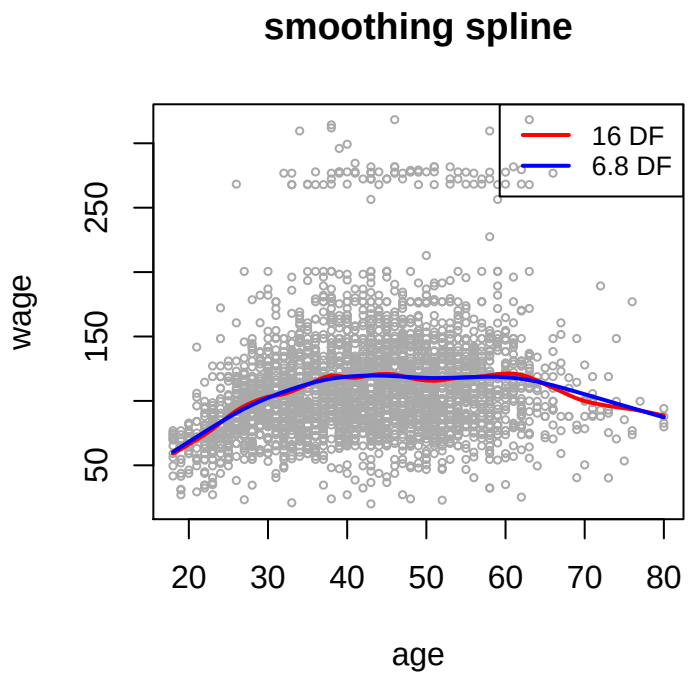
```
plot(age,wage,xlim=agelims,cex=.5,col='darkgrey')
title('smoothing spline')
fit = smooth.spline(age,wage,df=16)
fit2 = smooth.spline(age,wage,cv=TRUE)
names(fit2)
```

```
 [1] "x"          "y"          "w"          "yin"        "tol"
 [6] "data"       "no.weights" "lev"        "cv.crit"    "pen.crit"
[11] "crit"       "df"         "spar"       "ratio"      "lambda"
[16] "iparms"     "auxM"       "fit"        "call"
```

```
fit2$df
```

```
[1] 6.794596
```

```
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=0.8)
```



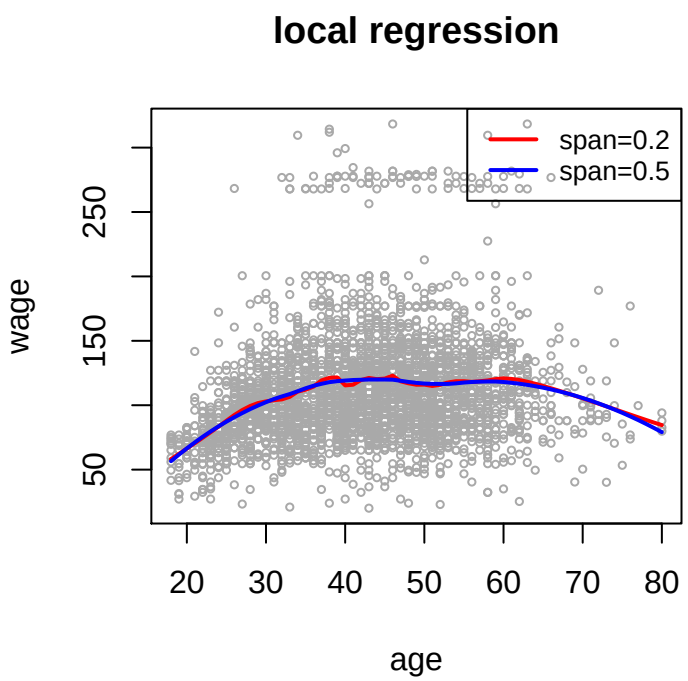
처음에는 df=16이라고 명시를 했지만 그 다음에는 CV를 통해 df=6.8인 λ로 smoothness parameter을 정했다. local regression을 실행하기 위해서, loess() 함수를 이용한다.

```
plot(age,wage,xlim=agelims,cex=0.5,col='darkgrey')
title('local regression')
fit = loess(wage~age,span=0.2,data=Wage)
fit2 = loess(wage~age,span=0.5,data=Wage)
names(fit2)
```

```
 [1] "n"          "fitted"     "residuals" "enp"        "s"
 [6] "one.delta"  "two.delta"  "trace.hat" "divisor"    "robust"
[11] "pars"       "kd"         "call"      "terms"      "xnames"
[16] "x"          "y"          "weights"
```

```
lines(age.grid,predict(fit,data.frame(age=age.grid)),col='red',lwd=2)
```

```
lines(age.grid,predict(fit2,data.frame(age=age.grid)),col='blue',lwd=2)
legend('topright',legend=c("span=0.2","span=0.5"),col=c("red","blue"),lty=1,lwd=2,cex=0.8)
```



3. GAMs

이제 wage을 예측하기 위해서 year, age, education을 예측 변수로 활용해본다. GAMs은 예측 변수의 형태만 다를뿐, 최소자승추정법을 이용하는 것은 동일하므로 lm() 함수를 이용한다. 더욱 일반적인 형태의 GAMs을 적합하기 위해서 (smoothing splines이나 basis functions으로 표현될 수 없는 성분)는 gam library을 사용한다. gam library에 있는 s() 함수는 smoothing spline을 사용하고 싶을 때 사용한다. year, age변수 각각 자유도가 4, 5가 되도록 설정하고 education 변수는 범주형 변수이기 때문에 그냥 둔다.

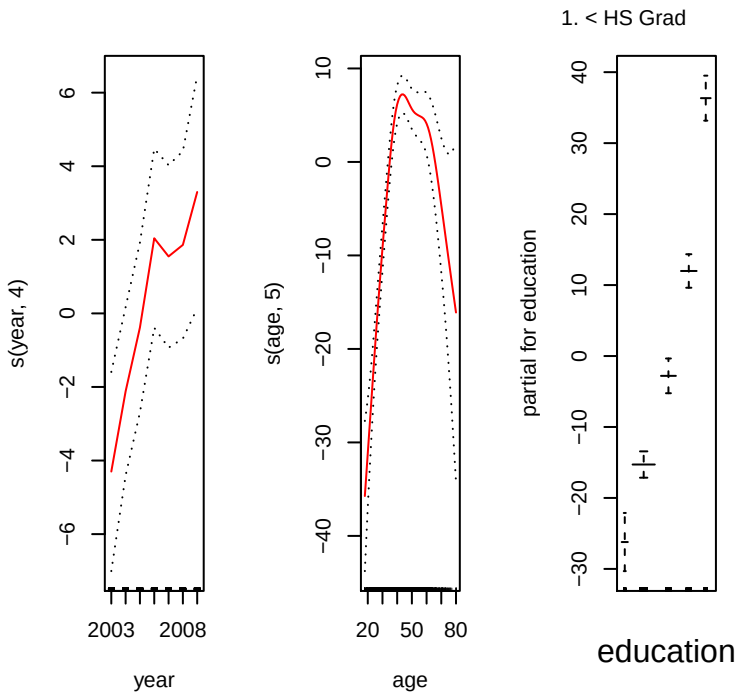
```
gam1 = lm(wage~ns(year,4)+ns(age,5)+education,data=Wage)
library(gam)
gam.mod = gam(wage~s(year,4) + s(age,5) + education, data=Wage)
names(gam.mod)
```

```
 [1] "smooth.frame"        "coefficients"        "residuals"
 [4] "fitted.values"       "effects"             "weights"
 [7] "rank"                "assign"              "qr"
[10] "smooth"              "nl.df"               "df.residual"
[13] "var"                 "additive.predictors" "R"
```

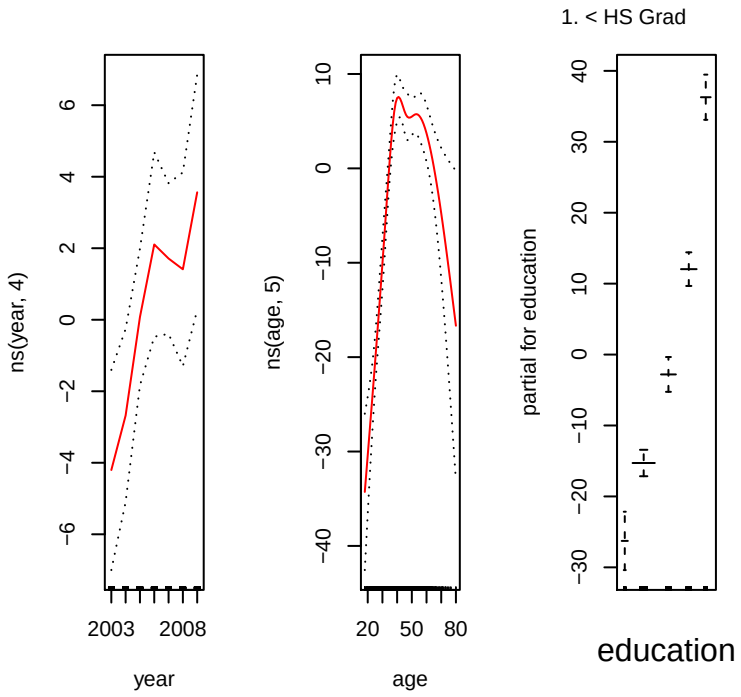
```
[16] "rank"                "family"              "deviance"
[19] "aic"                 "null.deviance"       "iter"
[22] "prior.weights"       "y"                   "df.null"
[25] "nl.chisq"            "model"               "call"
[28] "formula"             "terms"               "data"
[31] "offset"              "control"             "method"
[34] "contrasts"           "xlevels"
```

gam() 함수로 생성한 객체는 plot() 함수를 실행해도 자동으로 plot.Gam()으로 인식되지만 lm() 함수로 생성한 객체는 plot.Gam()으로 직접 실행해야 한다.

```
par(mfrow=c(1,3))
plot(gam.mod,se=T,col='red')
```




```
plot.Gam(gam1,se=T,col='red')
```



year 변수를 선형으로 적합하는 것도 괜찮아 보이니, 이를 anova를 통하여 모델들간에 비교를 해보자. year 변수를 포함하지 않은 모델, year 변수를 선형으로 포함한 모델, year 변수를 smoothing spline으로 포함한 모델을 비교한다.

```
gam.m1 = gam(wage~s(age,5)+education,data=Wage)
gam.m2 = gam(wage~year+s(age,5)+education,data=Wage)
gam.m3 = gam(wage~s(year,4)+s(age,5)+education,data=Wage)
anova(gam.m1,gam.m2,gam.m3)

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  Pr(>Chi)
1      2990    3711731
2      2989    3693842  1  17889.2 0.0001419 ***
3      2986    3689770  3   4071.1 0.3483897
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

결과를 모면 모델 1 vs 모델 2 가설에서 p-value가 매우 유의하게 나온 것으로 보아, 모델 2, 즉, year을 선형으로

포함한 모델이 성능이 가장 좋게 나왔다.

```
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: 2

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
```

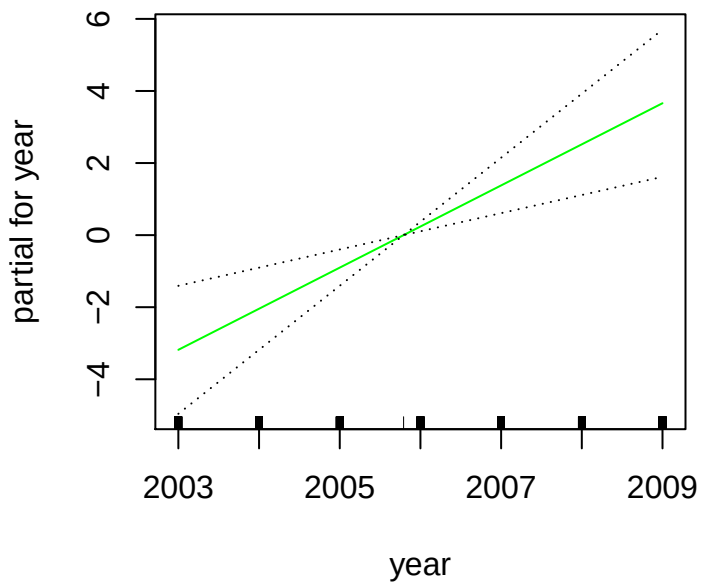
위 결과를 봐도, s(year,4)에 대한 p-value가 매우 높은 것으로 보아, 해당 변수는 통계적으로 유의미하지 않음을 확인할 수 있다.

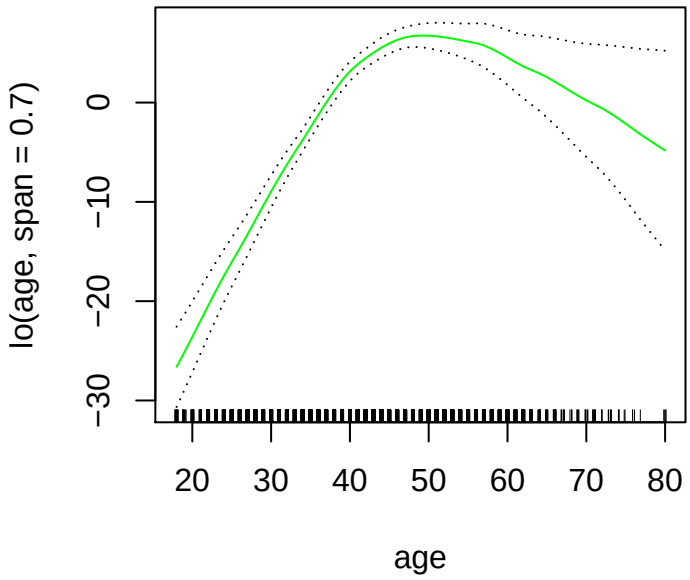
gam 모델로 예측을 할 수도 있다.

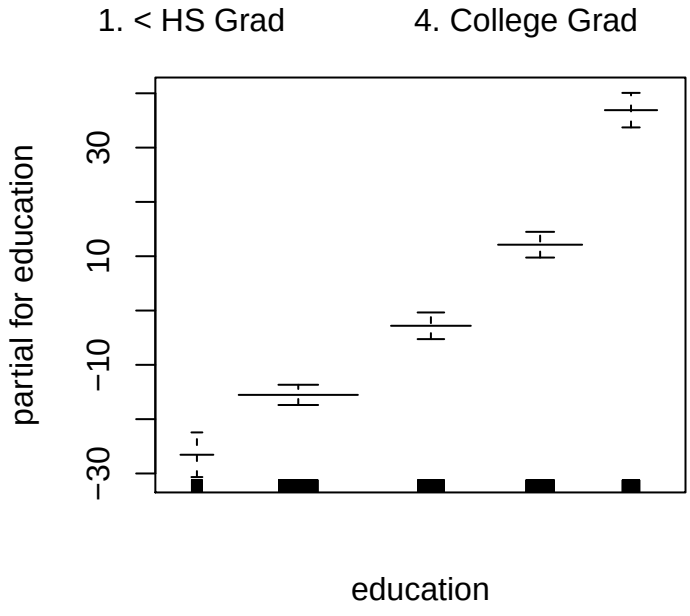
```
pred = predict(gam.m2, newdata=Wage)
```

gam 모델에서 local regression 적합을 사용할 수 있다.

```
gam.lo = gam(wage~year+lo(age,span=0.7)+education,data=Wage)
plot(gam.lo,se=T,col='green')
```

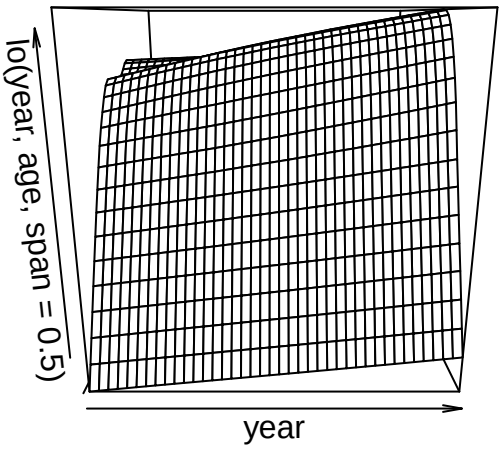


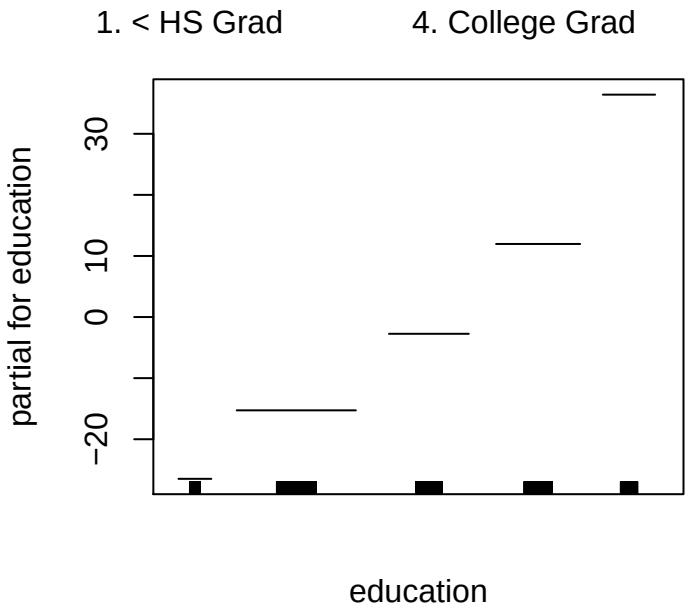




교호작용 term도 추가할 수 있다.

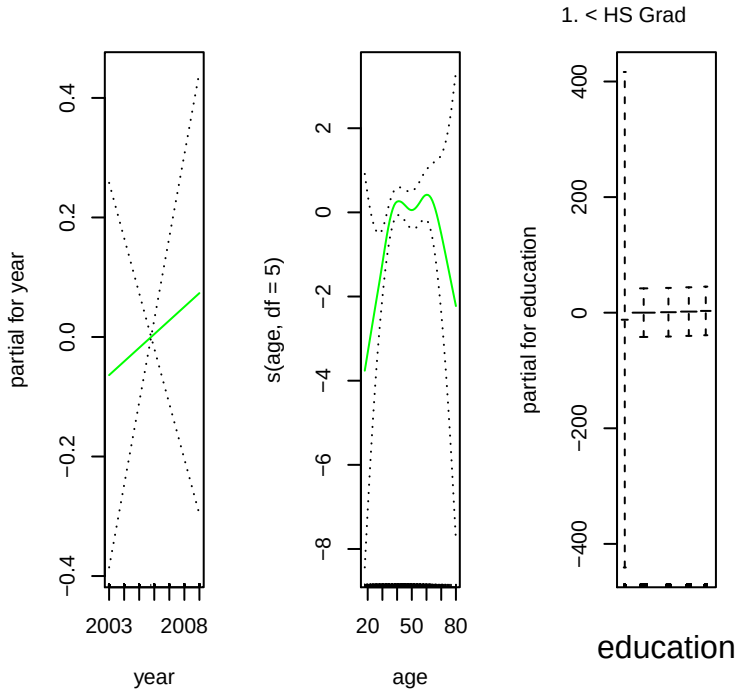
```
gam.lo.int = gam(wage~lo(year,age,span=0.5)+education,data=Wage)
library(akima)
plot(gam.lo.int)
```





로지스틱 회귀를 시행하기 위해서 반응 변수에 대해서 I()를 사용하고 family=binomial을 명시해야 한다.

```
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
```

고소득 집단 중 < HS Grad인 데이터는 없으므로 해당 데이터를 제외하고 적합하면 신뢰구간이 위 아래로 뻗은 모양과 같은 이상한 모양이 나오지 않을 것이다.

해결해야할 점

1. regression spline에서 bs() 함수에서 값이 어떻게 만들어지는지. ns() 함수도 마찬가
2. regression spline에서 자유도, knots을 어떻게 정할까?
3. smoothing spline에서는 무튼 λ을 CV로 정하니 상관 없음.
4. gam에서 각 변수에 어떤 함수를 쓸지 어떻게 정할까?
5. local regression에서 span, weight을 어떻게 정할까?