Tutorials for ECO3080

Topic 6: Models with Nonlinearity

Long Ma

School of Management and Economics, CUHKSZ

2022.11

1. Polynomial Regression
2. Step Function
3. Regression Spline
4. Local Regression and GAM

Polynomial Regression

- The regression function is:

$$wage_i = \beta_0 + \beta_1 age_i + \beta_2 age_i^2 + \beta_3 age_i^3 + \beta_4 age_i^4 + \varepsilon_i \quad (1)$$

  where there is only one variable "age" but its higher order terms are included. Hence, this function is **nonlinear** in "age". However, this function is **linear** in parameters β_0, β_1, ..., β_4 which means that OLS estimation is still feasible.

- What about the following regression function?

$$wage_i = \beta_0 + \beta_1 age_i^{\beta_2} + \varepsilon_i \quad (2)$$

  where β_2 is on the power of "age" and this is inherently **nonlinear** in parameters. Fortunately, ε_i is additive which means Nonlinear Least Square can be used to do estimation.

Polynomial Regression

- Focusing on equation (1), we can write the code:

```
library(ISLR)
polyfit1<-lm(wage~poly(age,4,raw=T),data=Wage)
coef(summary(polyfit1))
```

- The results are shown below:

```
                             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
```

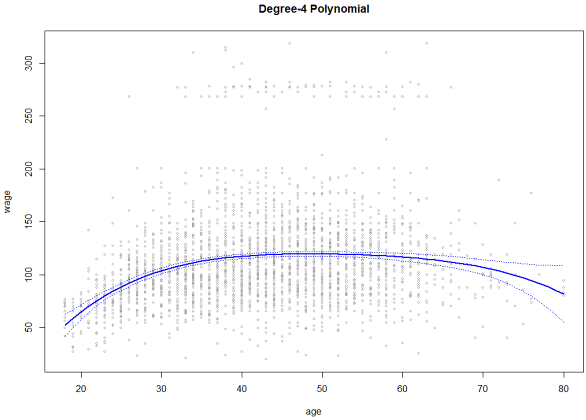
Polynomial Regression

- If we want to draw the "fitting" graph:

```
attach(Wage)
 agelims<-range(age)
 agegrid<-seq(from=agelims[1],to=agelims[2])
 polypreds<-predict(polyfit1,
                    newdata=list(age=agegrid),
                    se=T)
 sebs<-cbind(polypreds$fit+1.96*polypreds$se.fit,
             polypreds$fit-1.96*polypreds$se.fit)
 plot(age,wage,xlim=agelims,cex=0.5,col="darkgrey")
 title("Degree-4Polynomial")
 lines(agegrid,polypreds$fit,lwd=2,col="blue")
 matlines(agegrid,sebs,lwd=1,col="blue",lty=3)
detach(Wage)
```

Polynomial Regression

- The fitting plot is:



Polynomial Regression

- Consider the classification context:

$$\ln\left[\frac{P(wage_i > 250|\boldsymbol{X})}{P(wage_i \leq 250|\boldsymbol{X})}\right] = \boldsymbol{x}_i'\boldsymbol{\beta} \tag{3}$$

where

$$\boldsymbol{x}_i = \begin{bmatrix} 1 \\ age_i \\ age_i^2 \\ age_i^3 \\ age_i^4 \end{bmatrix} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{bmatrix}$$

- You need to be careful to deal with the standard error of your estimation. You have to do a transformation, and then you can draw the dash lines denoting the confidence interval.

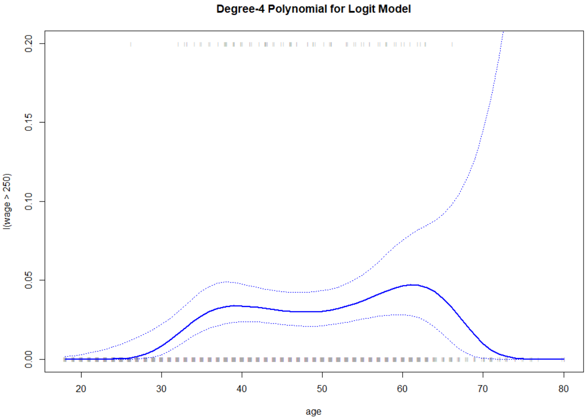
Polynomial Regression

- The code for logit regression:

```r
attach(Wage)
 logitpolyfit<-glm(I(wage>250)~poly(age,4,raw = T),
                   data=Wage,
                   family=binomial(link="logit"))
 logitpreds<-predict(logitpolyfit,
                     newdata=list(age=agegrid),
                     se=T)
 pfit<-exp(logitpreds$fit)/(1+exp(logitpreds$fit))
 sebslogit<-cbind(logitpreds$fit+1.96*logitpreds$se.fit,
                  logitpreds$fit-1.96*logitpreds$se.fit)
 seb2<-exp(sebslogit)/(1+exp(sebslogit))
 plot(age,I(wage>250),xlim=agelims,type="n",
      ylim=c(0,0.2))
 points(jitter(age),I((wage>250)/5),
        cex=0.5,pch="|",col="darkgrey")
 lines(agegrid,pfit,lwd=2,col="blue")
 matlines(agegrid,seb2,lwd=1,col="blue",lty=3)
detach(Wage)
```

Polynomial Regression

- The fitting plot is:



Step Function

- The regression function is:

$$wage_i = \beta_0 + \beta_1 I(age_i < c_1) + \beta_2 I(c_1 \leq age_i < c_2) + \beta_3 I(c_2 \leq age_i < c_3) + \beta_4 I(c_3 \leq age_i) + \varepsilon_i \tag{4}$$

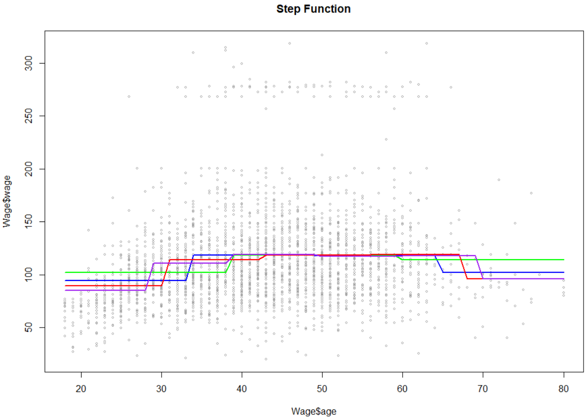
- The code for step regression is:

```r
attach(Wage)
 stepfit<-lm(wage~cut(age,4),data=Wage)
 coef(summary(stepfit))
detach(Wage)
```

```
                        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
```

Step Function

- Compare cases with different number of knots.



1. Polynomial Regression
2. Step Function
3. Regression Spline
4. Local Regression and GAM

Regression Spline

- The linear spline with 3 knots:

$$wage_i = \begin{cases} \beta_{1,0} + \beta_{1,1} age_i + \varepsilon_{1,i}, & age_i \in (-\infty, c_1) \\ \beta_{2,0} + \beta_{2,1} age_i + \varepsilon_{2,i}, & age_i \in [c_1, c_2) \\ \beta_{3,0} + \beta_{3,1} age_i + \varepsilon_{3,i}, & age_i \in [c_2, c_3) \\ \beta_{4,0} + \beta_{4,1} age_i + \varepsilon_{4,i}, & age_i \in [c_3, +\infty) \end{cases} \tag{5}$$

- Subject to:

$$\beta_{1,0} + \beta_{1,1} c_1 = \beta_{2,0} + \beta_{2,1} c_1$$

$$\beta_{2,0} + \beta_{2,1} c_2 = \beta_{3,0} + \beta_{3,1} c_2$$

$$\beta_{3,0} + \beta_{3,1} c_3 = \beta_{4,0} + \beta_{4,1} c_3$$

- Degree of freedom: 8 - 3 = 5

Regression Spline

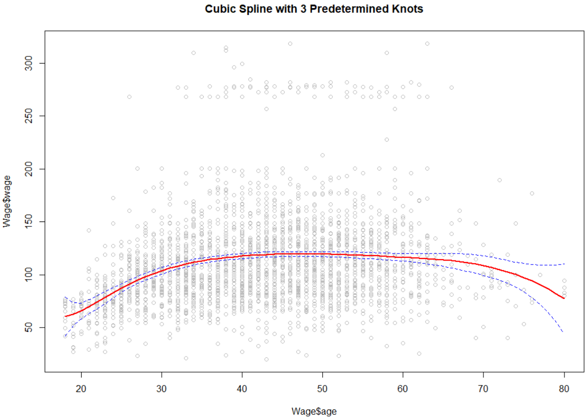
- The code for cubic spline with 3 predetermined knots:

```
install.packages("splines")
library(splines)
splinefit<-lm(wage~bs(age,knots=c(25,40,60)),data=Wage)
splinepred<-predict(splinefit,
                    newdata=list(age=agegrid),
                    se=T)
par(mfrow = c(1, 1))
plot(Wage$age,Wage$wage,col="gray")
lines(agegrid,splinepred$fit,lwd=2,col="red")
lines(agegrid,splinepred$fit+1.96*splinepred$se,
      lty="dashed",col="blue")
lines(agegrid,splinepred$fit-1.96*splinepred$se,
      lty="dashed",col="blue")
```

- What is the DF in this case? (16 - 9 = 7) (1 + 6 = 7)

Regression Spline

- The fitting plot is:



Regression Spline

- If you cannot decide the location of knots initially, then you can just set the degree of freedom and let the computer choose the location automatically.

```
splinefit<-lm(wage~bs(age,df=6),data=Wage)
```

Why df is 6 here?

- If you want to use quadratic spline or linear spline, then you can set the parameter "degree" to equal 2 or 1.

```
linearfit<-lm(wage~bs(age,df=4,degree=1),data=Wage)
```

Why df is 4 here?

Regression Spline

- Natural Cubic Spline:

```
splinefit2<-lm(wage~ns(age,df=4),data=Wage)
```

Why df is 4 here?

- Smoothing Spline

```
smoothfit<-smooth.spline(age,wage,df=16)
smoothfit2<-smooth.spline(age,wage,cv=T)
```

Choosing df is equivalent to choosing λ.

1. Polynomial Regression
2. Step Function
3. Regression Spline
4. **Local Regression and GAM**

Local Regression

- Local regression is a different approach for fitting flexible non-linear functions, which involves computing the fit at a target point x_0 using only the nearby training observations.
- The code for local regression is:

```
localfit<-loess(wage~age,span=0.2,degree=0,data=Wage)
```

GAM

- Generalized additive models (GAMs) provide a general framework for extending a standard linear model by allowing non-linear functions of each of the variables, while maintaining additivity.
- The code for GAM is:

```
install.packages("gam")
library(gam)
gam.m3<-gam(wage~s(year,4)+s(age,5)+education,data=Wage)
```

We want to include year, age and education in our model. Two of them are transformed into nonlinear expression and one of them remains linear.

Summary

- Cross Validation (CV) is always feasible when choosing the best model. The only different is the parameter we care:
    1. For polynomial regression: the degree of power;
    2. For step function: the number of cut-offs;
    3. For "traditional" spline: the number of cut-offs and the degree of power;
    4. For smooth spline: the parameter of penalty term;
    5. For local regression: the width of the bin and the degree of power;
    6. ....
- There are other methods for choosing the best model: ANOVA (shown in the codes).

Thanks!