Nonlinear model

Mats Hansson

11 mars 2016

Here we explore the use of nonlinear models using some tolls in R.

```
library(ISLR)
attach(Wage)
```

Polynomials

First we will use polynomials, and focus on a single predictor age:

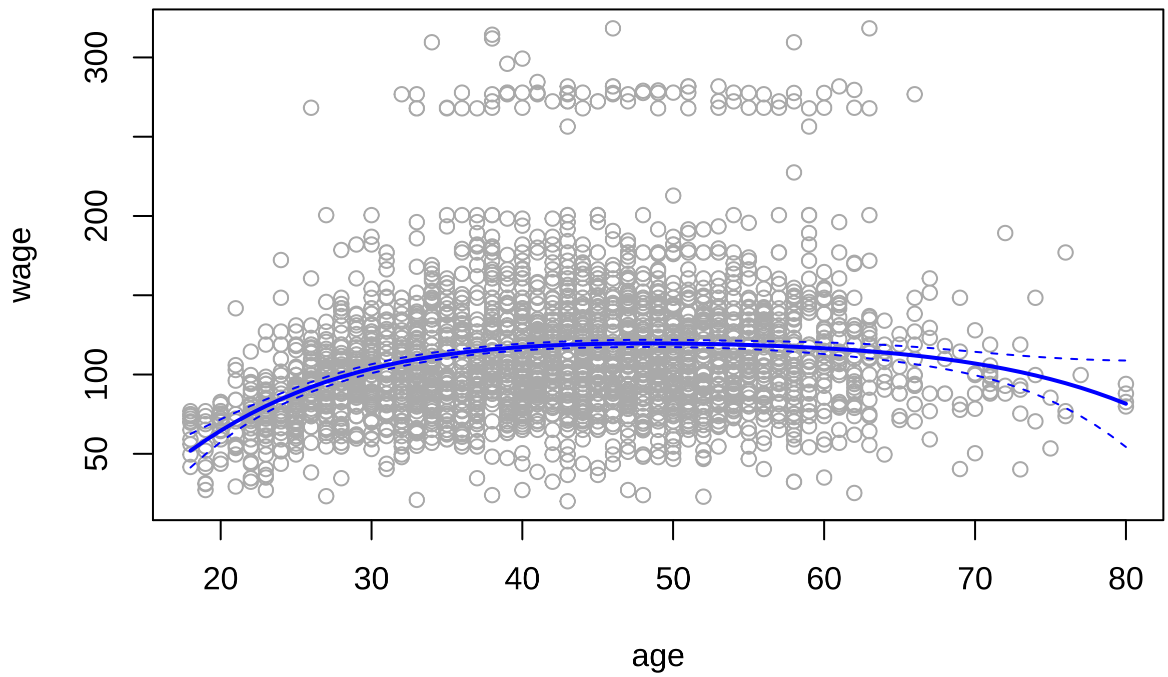
```
fit=lm(wage~poly(age,4), data=Wage)
summary(fit)
```

```
##
## Call:
## lm(formula = wage ~ poly(age, 4), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.707 -24.626  -4.993  15.217 203.693
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    111.7036     0.7287 153.283  < 2e-16 ***
## poly(age, 4)1  447.0679    39.9148  11.201  < 2e-16 ***
## poly(age, 4)2 -478.3158    39.9148 -11.983  < 2e-16 ***
## poly(age, 4)3  125.5217    39.9148   3.145  0.00168 **
## poly(age, 4)4  -77.9112    39.9148  -1.952  0.05104 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.91 on 2995 degrees of freedom
## Multiple R-squared:  0.08626,    Adjusted R-squared:  0.08504
## F-statistic: 70.69 on 4 and 2995 DF,  p-value: < 2.2e-16
```

The `poly()`function generates a basis of *orthogonal polynomials* . Lets make a plot of a fitted function, along with the standard errors of the fit.

```
agelims=range(age)
age.grid=seq(from=agelims[1], to=agelims[2])
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(age, wage, col="darkgrey")
lines(age.grid, preds$fit, lwd=2, col="blue")
matlines(age.grid, se.bands, col="blue", lty=2)
```



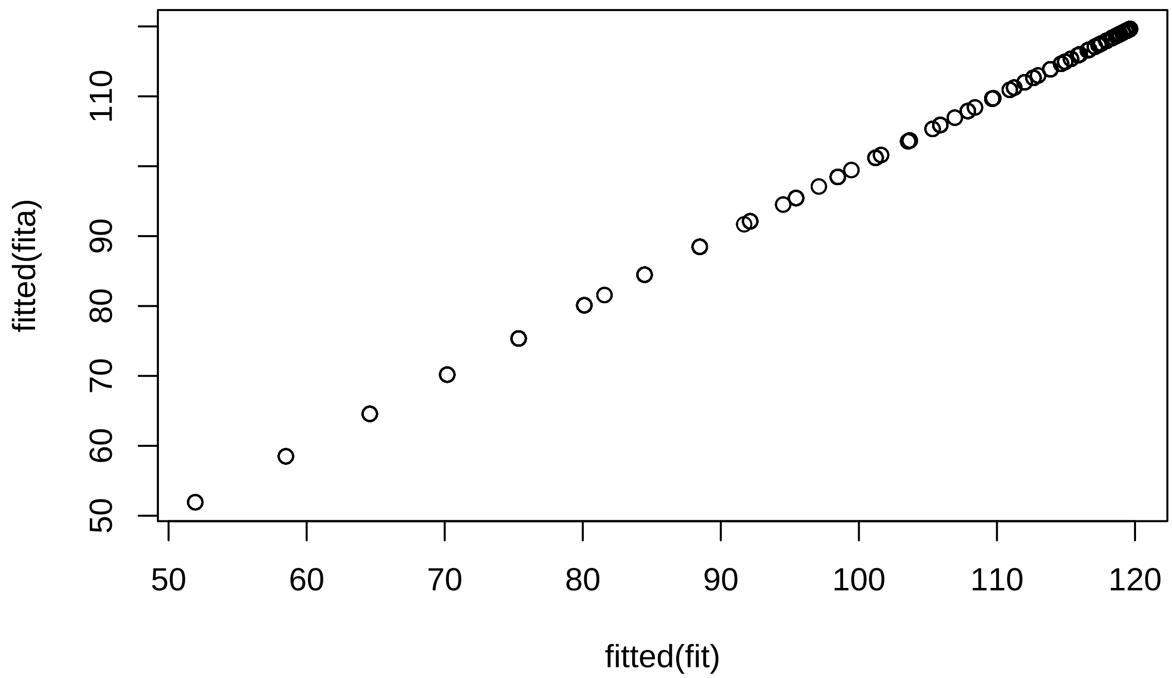
There are others more direct ways of doing this in R. For example

```
fita=lm(wage~age+I(age^2)+I(age^3)+I(age^4), data=Wage)
summary(fita)
```

```
##
## Call:
## lm(formula = wage ~ age + I(age^2) + I(age^3) + I(age^4), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.707 -24.626  -4.993  15.217 203.693
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.842e+02  6.004e+01  -3.067 0.002180 **
## age          2.125e+01  5.887e+00   3.609 0.000312 ***
## I(age^2)    -5.639e-01  2.061e-01  -2.736 0.006261 **
## I(age^3)     6.811e-03  3.066e-03   2.221 0.026398 *
## I(age^4)    -3.204e-05  1.641e-05  -1.952 0.051039 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.91 on 2995 degrees of freedom
## Multiple R-squared:  0.08626,    Adjusted R-squared:  0.08504
## F-statistic: 70.69 on 4 and 2995 DF,  p-value: < 2.2e-16
```

Here `I()` is a *wrapper* function; we need it becouse `age^2` means something to formula language, while `I(age^2)` is protected. The coffecient are different to those we got before! However, the fits are thes same.

```
plot(fitted(fit), fitted(fita))
```



By using orthogonal polynomials in this simple way, it turns out that we can separatley test for each coeffecient. So if we look at the summary again, we can see that the linear, quadratic and cubic terms are significant, but not quartic.

```
summary(fit)
```

```
##
## Call:
## lm(formula = wage ~ poly(age, 4), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.707 -24.626  -4.993  15.217 203.693
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    111.7036     0.7287 153.283  < 2e-16 ***
## poly(age, 4)1  447.0679    39.9148  11.201  < 2e-16 ***
## poly(age, 4)2 -478.3158    39.9148 -11.983  < 2e-16 ***
## poly(age, 4)3  125.5217    39.9148   3.145  0.00168 **
## poly(age, 4)4  -77.9112    39.9148  -1.952  0.05104 .
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.91 on 2995 degrees of freedom
## Multiple R-squared:  0.08626,    Adjusted R-squared:  0.08504
## F-statistic: 70.69 on 4 and 2995 DF,  p-value: < 2.2e-16
```

This only work with linear regression,and if there is a single predictor. In general we would use `anova()` as this next example demonstrates.

```r
fita=lm(wage~education, data=Wage)
fitb=lm(wage~education+age, data=Wage)
fitc=lm(wage~education+poly(age,2), data=Wage)
fits=lm(wage~education+poly(age,3), data=Wage)
anova(fita,fitb, fitc, fits)
```

```
## Analysis of Variance Table
##
## Model 1: wage ~ education
## Model 2: wage ~ education + age
## Model 3: wage ~ education + poly(age, 2)
## Model 4: wage ~ education + poly(age, 3)
##   Res.Df     RSS Df Sum of Sq        F Pr(>F)
## 1   2995 3995721
## 2   2994 3867992  1    127729 102.7378 <2e-16 ***
## 3   2993 3725395  1    142597 114.6969 <2e-16 ***
## 4   2992 3719809  1      5587   4.4936 0.0341 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Polynomial logistics regression

Now we fit a logistic regression model to a binary respone variable, constructed from `wage`. We code the big earners (>250K) as 1, else 0.

```r
fit=glm(wage>250~poly(age, 3), data=Wage, family = binomial)

summary(fit)
```

```
##
## Call:
## glm(formula = wage > 250 ~ poly(age, 3), family = binomial, data = Wage)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2808  -0.2736  -0.2487  -0.1758   3.2868
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -3.8486     0.1597 -24.100  < 2e-16 ***
## poly(age, 3)1  37.8846    11.4818   3.300 0.000968 ***
## poly(age, 3)2 -29.5129    10.5626  -2.794 0.005205 **
```

```
## poly(age, 3)3   9.7966     8.9990   1.089 0.276317
## ---
## 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: 707.92  on 2996  degrees of freedom
## AIC: 715.92
##
## Number of Fisher Scoring iterations: 8
```

```
preds=predict(fit, list(age=age.grid), se=T)

se.bands=preds$fit+cbind(fit=0, lower=-2*preds$se.fit, upper=2*preds$se.fit)
```

we have done the computations on the logit scale. To transform we need to apply the inverse logit mapping.

$$p = \frac{e^{\eta}}{1+e^{\eta}}.$$

We can do this simultaneous for all three columns of `se.bands`:

```
prob.bands=exp(se.bands)/(1+exp(se.bands))
matplot(age.grid, prob.bands, col="blue", lwd=c(2,1,1), lty=c(1,2,2), type="l", ylim=c(0,.1))
points(jitter(age), I(wage>250)/10, pch="l", cex=0.5)
```