# Bruce Campbell ST-617 Homework 2

Tue Jul 12 22:32:27 2016

# Chapter 7

## Problem 6

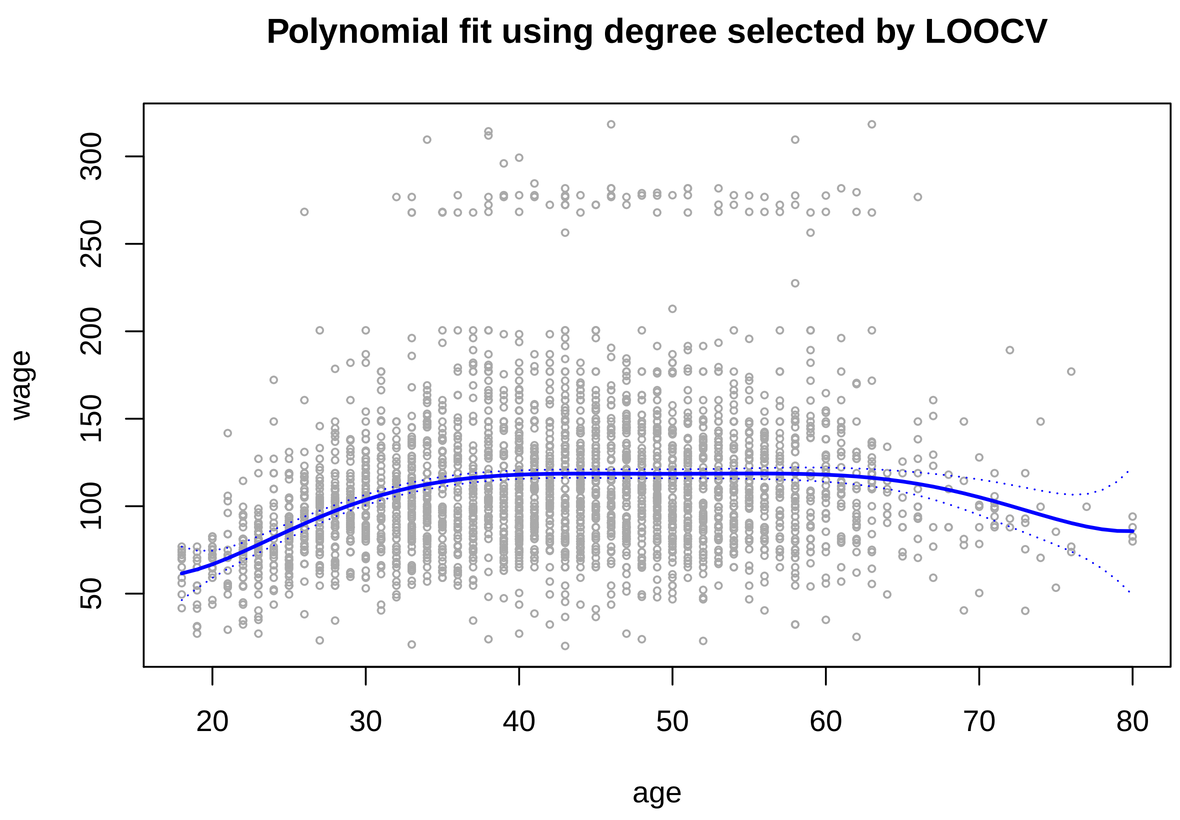
In this exercise, you will further analyze the Wage data set considered throughout this chapter.

### a)

Perform polynomial regression to predict wage using age. Use cross-validation to select the optimal degree d for the polynomial. What degree was chosen, and how does this compare to the results of hypothesis testing using ANOVA? Make a plot of the resulting polynomial fit to the data.

```r
rm(list = ls())
library(ISLR)
library(ggplot2)

attach(Wage)

max.poly <- 7

cvFunc <- function(age, wage, degree) {
    preds <- numeric(length(age))
    for (i in 1:length(age)) {
        # Here is where we eageclude i from the training set
        age.in <- age[-i]
        age.out <- age[i]

        # Single test point
        wage.in <- wage[-i]
        wage.out <- age[i]

        fit <- lm(wage.in ~ poly(age.in, degree = degree))
        preds[i] <- predict(fit, newdata = data.frame(age.in = age.out))
    }
    return(sum((wage - preds)^2))
}

cv.err <- data.frame(sq_err = numeric(max.poly))
for (i in 1:max.poly) {
    cv.err[i, 1] <- cvFunc(age, wage, degree = i)
}

best_degree <- which.min(cv.err$sq_err)

lmpoly.fit <- lm(wage ~ poly(age, degree = best_degree))
```

```r
agelims = range(age)
age.grid = seq(from = agelims[1], to = agelims[2])

preds = predict(lmpoly.fit, newdata = data.frame(age = age.grid), se = TRUE)

se.bands = cbind(preds$fit + 2 * preds$se.fit, preds$fit - 2 * preds$se.fit)

plot(age, wage, xlim = agelims, cex = 0.5, col = "darkgrey ")

lines(age.grid, preds$fit, lwd = 2, col = " blue")
matlines(age.grid, se.bands, lwd = 1, col = " blue", lty = 3)
title("Polynomial fit using degree selected by LOOCV")
```



```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)
fit.6 = lm(wage ~ poly(age, 6), data = Wage)
fit.7 = lm(wage ~ poly(age, 7), data = Wage)
anova(fit.1, fit.2, fit.3, fit.4, fit.5, fit.6, fit.7)
```

```
## 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)
## Model 6: wage ~ poly(age, 6)
## Model 7: wage ~ poly(age, 7)
##   Res.Df     RSS Df Sum of Sq        F    Pr(>F)
## 1   2998 5022216
## 2   2997 4793430  1    228786 143.6926 < 2.2e-16 ***
## 3   2996 4777674  1     15756   9.8956  0.001673 **
## 4   2995 4771604  1      6070   3.8125  0.050966 .
## 5   2994 4770322  1      1283   0.8055  0.369516
## 6   2993 4766389  1      3932   2.4697  0.116165
## 7   2992 4763834  1      2555   1.6049  0.205311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

According to the ANOVA results, the fourth order polynomial is sufficient. Higher order terms are not justified by the F test comparing fit.5 against fit.4

Leave one out cross validation has selected 6 as the best order, but looking at the cross validation error we see that the error for the fourth order model is very close to the fith and sixth order model. We could justifyably choose 4 from these results as the best model based on the desire to have a low error and a parsimonious model.

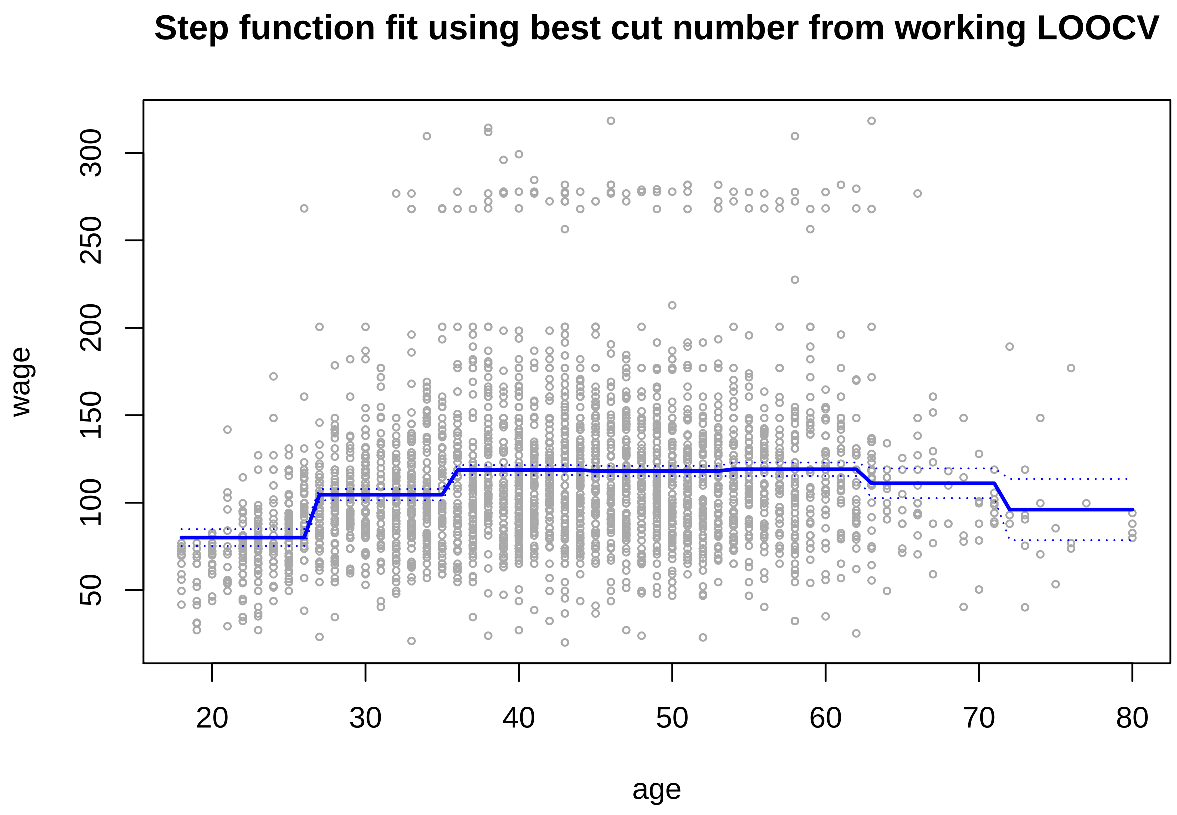
```r
library(pander)
pander(cv.err)
```

| sq_err |
|---|
| 5028705 |
| 4801588 |
| 4787879 |
| 4783788 |
| 4784636 |
| 4782357 |
| 4782436 |

### b)

Fit a step function to predict wage using age, and perform crossvalidation to choose the optimal number of cuts. Make a plot of the fit obtained.

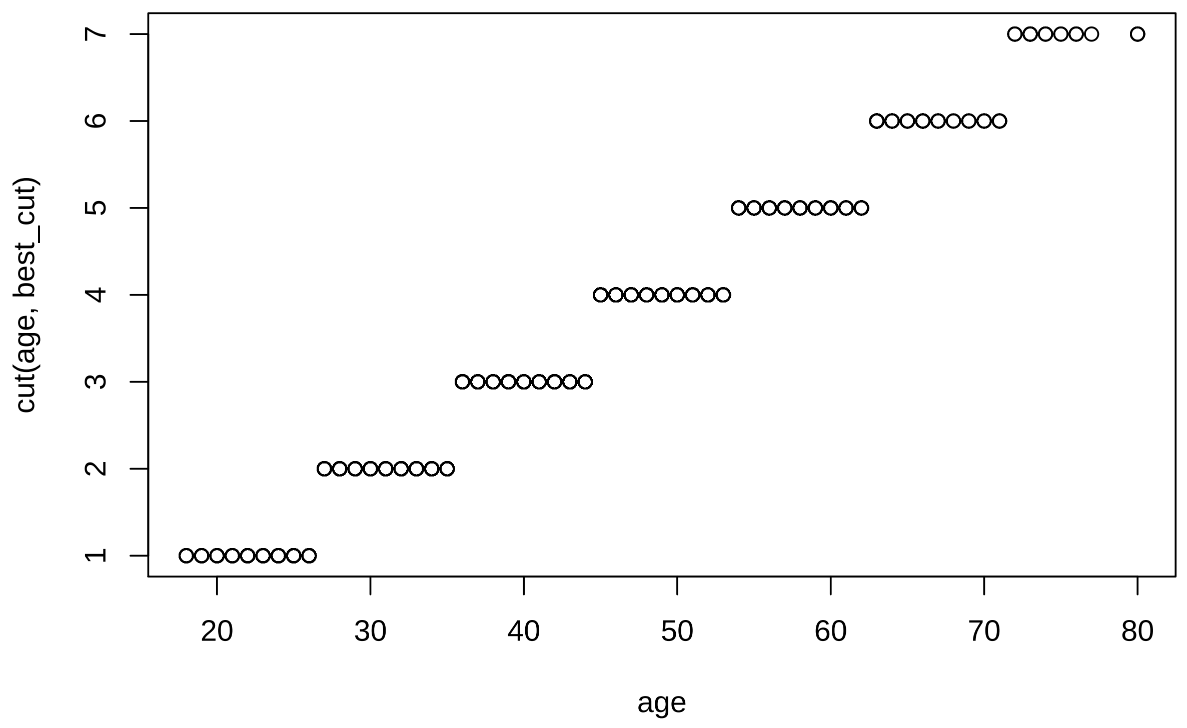
```r
max.cut <- 10
age = Wage$age
wage = Wage$wage

cvFunc <- function(age, wage, cut_val) {
    preds <- numeric(length(age))
    for (i in 1:length(age)) {
        # Here is where we eageclude i from the training set
        age.in <- age[-i]
        age.out <- age[i]

        # Single test point
        wage.in <- wage[-i]
        wage.out <- age[i]

        # fit <- lm(wage.in ~ cut(age.in, cut_val) )

        # See below - Error in model.frame.default(Terms, newdata, na.action =
        # na.action, xlev = object$xlevels) : factor cut(age.in, cut_val) has new
        # level preds[i]<- predict(fit, newdata=data.frame(age.in=age.out))
        # preds[i]<- predict(fit, newdata=data.frame(age.in=cut(age
        # ,best_cut)[age.out]))
    }
    return(sum((wage - preds)^2))
}


cv.err <- data.frame(sq_err = numeric(max.cut))

for (i in 1:max.cut) {
    cv.err[i, 1] <- cvFunc(age, wage, cut_val = i)
}

best_cut <- 7  # LOOCV algorithm error which.min(cv.err$sq_err)

lmcut.fit <- lm(wage ~ cut(age, best_cut), Wage)


agelims = range(age)
age.grid = seq(from = agelims[1], to = agelims[2])

preds = predict(lmcut.fit, newdata = data.frame(age = age.grid), se = TRUE)

se.bands = cbind(preds$fit + 2 * preds$se.fit, preds$fit - 2 * preds$se.fit)

plot(age, wage, xlim = agelims, cex = 0.5, col = "darkgrey ")

lines(age.grid, preds$fit, lwd = 2, col = " blue")
matlines(age.grid, se.bands, lwd = 1, col = " blue", lty = 3)
title("Step function fit using best cut number from working LOOCV")
```



There was one successful run of the LOOCV for the step function regression - then I ran into the error captured in the code comments above. Much effort was expended to get it working again, but we ran out of time so the best_cut was hard coded to be the value I recall from the one run that worked.

Here's what we learned from debugging - the cut function creates a factor variable

```
plot(age, cut(age, best_cut))
```



The expression used in the plot `predict (lmcut.fit ,newdata =data.frame(age=age.grid),se=TRUE)` is able to convert the data frame to a factor and calculate the predicted values. For other cases like a single value in LOOCV or a split of the data as in a validation set we got the error that reads like there was a problem converting the data to a factor. I checked the type of the data in all cases and verified it was an int. I tried converting the input data to a factor (see code) but that did not work either.