HW7

Jon Griffith and Beatrice Lowe

2025-04-28

```
library(ISLR2)
```

```
## Warning: package 'ISLR2' was built under R version 4.4.3
```

```
library(boot)
```

3

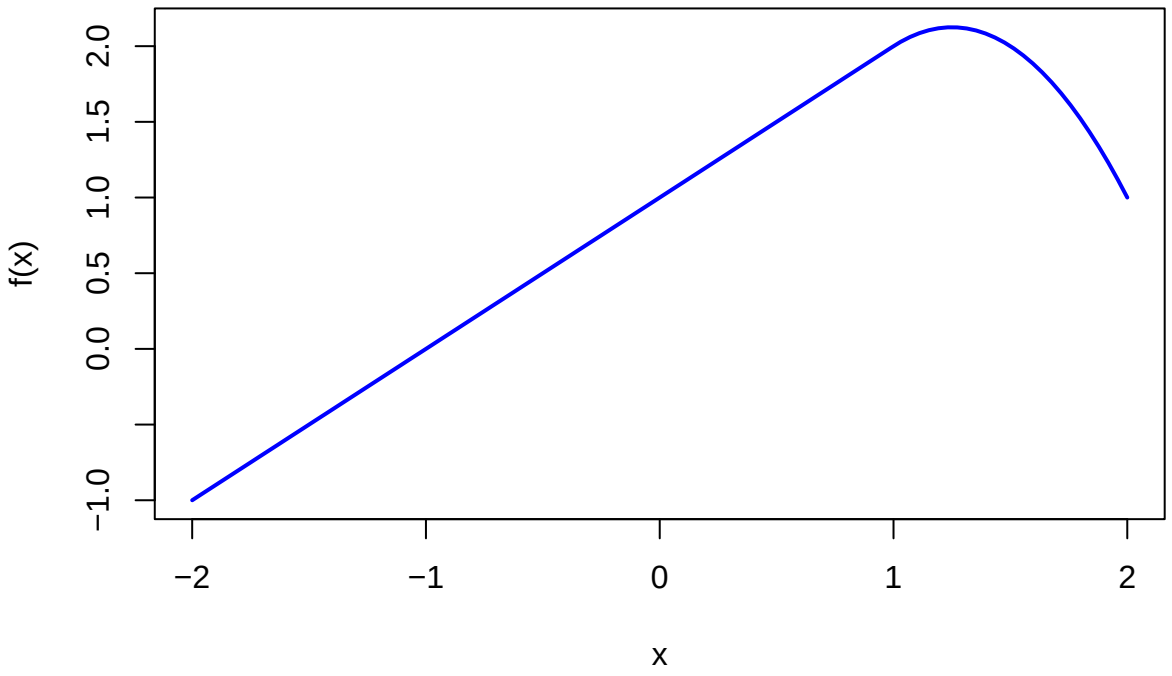
```
x <- seq(-2, 2, length.out=100)
B0 <- 1
B1 <- 1
B2 <- -2

b1 <- x
b2 <- (x-1)^2*I(x >= 1)

f <- function(x) {B0 + B1*b1 + B2*b2}

plot(x, f(x), type='l', lwd=2, col='blue',
     main = expression(f(x) == beta[0] + beta[1]*b[1](x) + beta[2]*b[2](x) + epsilon))
```

$$f(x) = \beta_0 + \beta_1 b_1(x) + \beta_2 b_2(x) + \varepsilon$$



4

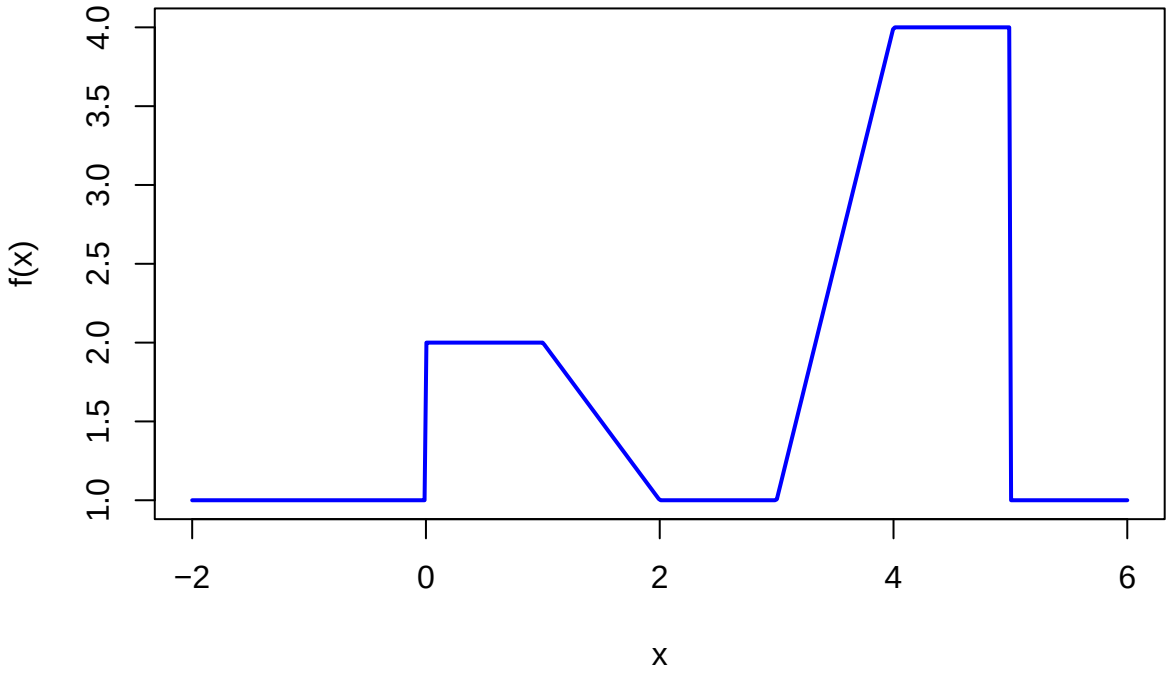
```
x <- seq(-2, 6, length.out=500)
B0 <- 1
B1 <- 1
B2 <- 3

b1 <- (I(0 <= x & x <= 2)-(x-1)*I(1 <= x & x <=2))
b2 <- (x-3)*I(3 <= x & x <= 4) + I(4 < x & x <= 5)

f <- function(x) {B0 + B1*b1 + B2*b2}

plot(x, f(x), type='l', lwd=2, col='blue',
     main = expression(f(x) == beta[0] + beta[1]*b[1](x) + beta[2]*b[2](x) + epsilon))
```

$$f(x) = \beta_0 + \beta_1 b_1(x) + \beta_2 b_2(x) + \varepsilon$$



6

(a)

```
set.seed(1)

cv.error.10 <- rep(NA, 10)
glm.fits <- vector('list', 10)

for (i in 1:10){
  glm.fits[[i]] <- glm(wage ~ poly(age, i), data=Wage)
  cv.error.10[i] <- cv.glm(Wage, glm.fits[[i]], K=10)$delta[1]
}
cat('Min MSE Degree',which.min(cv.error.10), '\n')
```

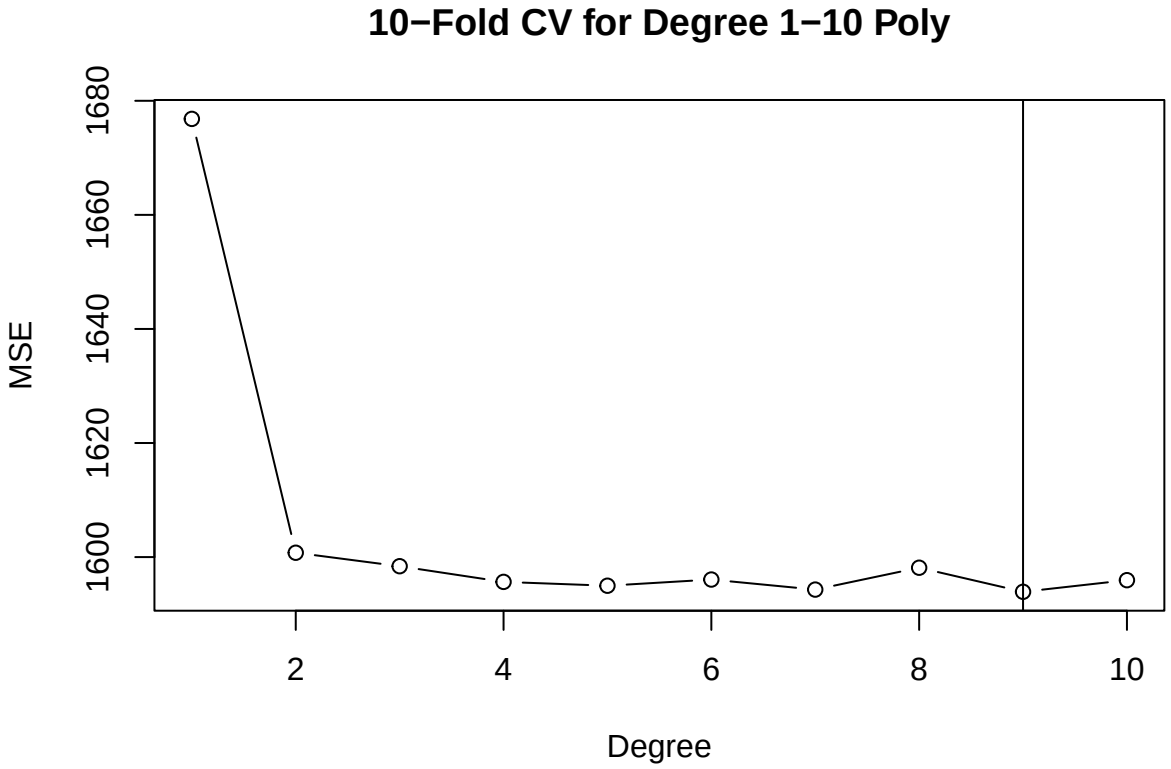
```
## Min MSE Degree 9
```

```
do.call(anova, c(glm.fits, test='F'))
```

```
## Analysis of Deviance Table
##
## Model  1: wage ~ poly(age, i)
## Model  2: wage ~ poly(age, i)
## Model  3: wage ~ poly(age, i)
## Model  4: wage ~ poly(age, i)
## Model  5: wage ~ poly(age, i)
## Model  6: wage ~ poly(age, i)
## Model  7: wage ~ poly(age, i)
## Model  8: wage ~ poly(age, i)
```

```
## Model  9: wage ~ poly(age, i)
## Model 10: wage ~ poly(age, i)
##    Resid. Df Resid. Dev Df Deviance        F    Pr(>F)
## 1       2998    5022216
## 2       2997    4793430  1   228786 143.7638 < 2.2e-16 ***
## 3       2996    4777674  1    15756   9.9005  0.001669 **
## 4       2995    4771604  1     6070   3.8143  0.050909 .
## 5       2994    4770322  1     1283   0.8059  0.369398
## 6       2993    4766389  1     3932   2.4709  0.116074
## 7       2992    4763834  1     2555   1.6057  0.205199
## 8       2991    4763707  1      127   0.0796  0.777865
## 9       2990    4756703  1     7004   4.4014  0.035994 *
## 10      2989    4756701  1        3   0.0017  0.967529
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(1:10, cv.error.10, type='b',
     main = '10-Fold CV for Degree 1-10 Poly',
     xlab = 'Degree',
     ylab = 'MSE')
abline(v = which.min(cv.error.10))
```

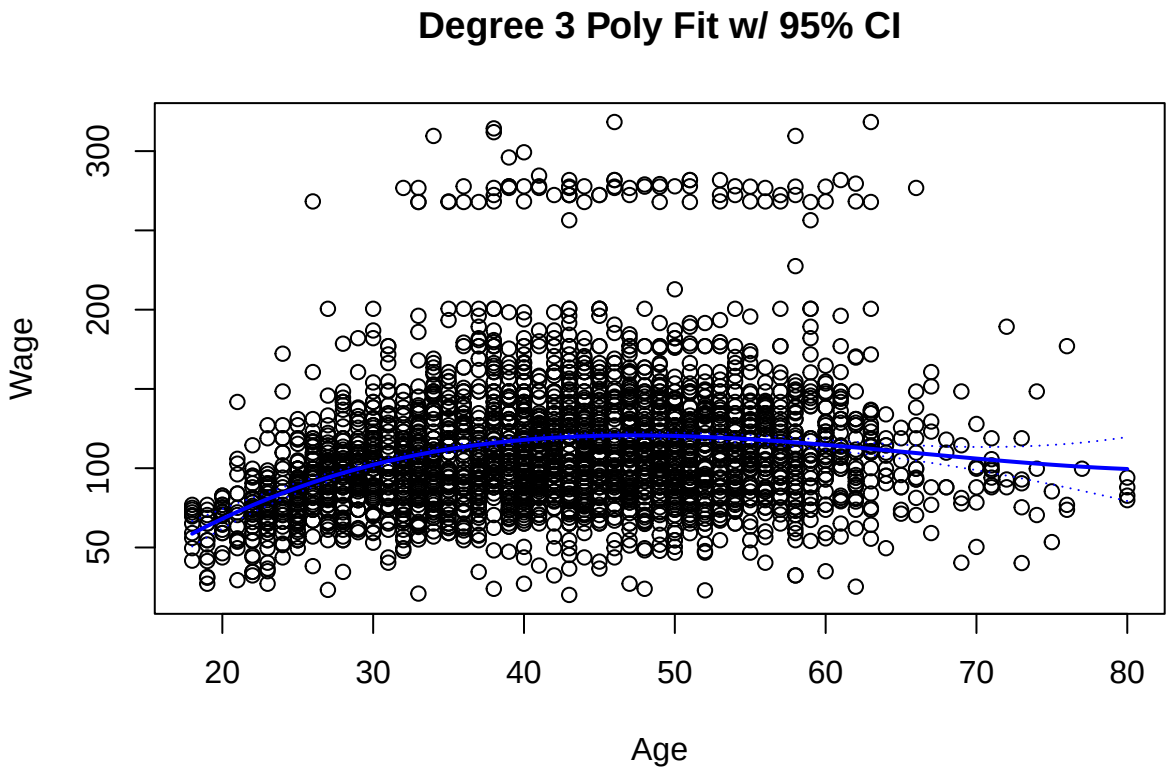


We see from 10-fold CV that the minimum MSE comes from the degree 9 polynomial, but when we plot all corresponding MSE's to each degree, we see that degree 9 only has a marginal advantage over all but degree 1. Using the ANOVA test, we see that all polynomials above degree three, except for nine, are not statistically significant and we deduce from this result that degree three polynomial is probably the optimal degree. We plot this below.

```
agelims <- range(Wage$age)
age.grid <- seq(from=agelims[1], to=agelims[2])
preds <- predict(glm(wage ~ poly(age, 3), data=Wage), newdata=list(age=age.grid), se=TRUE)
```

```r
se.bands <- cbind(preds$fit + 2*preds$se.fit, preds$fit - 2*preds$se.fit)

plot(Wage$age, Wage$wage,
     main = 'Degree 3 Poly Fit w/ 95% CI',
     xlab = 'Age',
     ylab = 'Wage')
lines(age.grid, preds$fit, lwd=2, col='blue')
matlines(age.grid, se.bands, lwd=1, col='blue', lty=3)
```



(b)

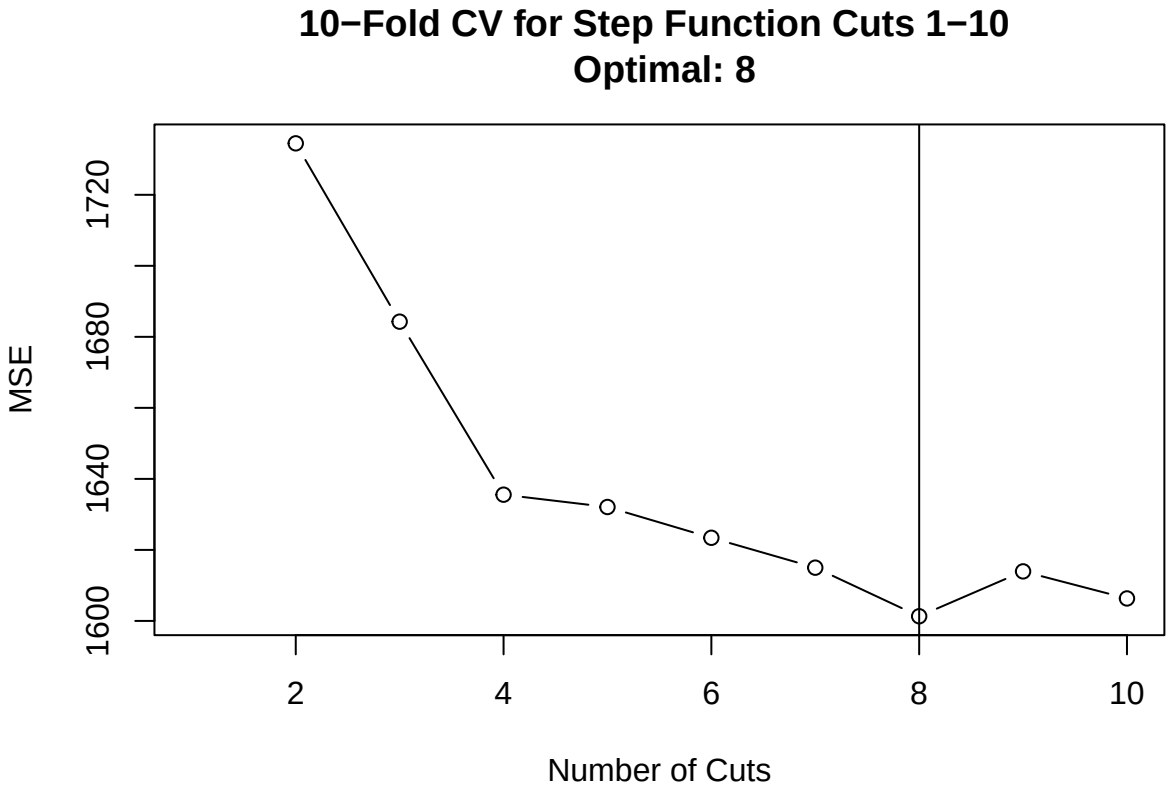
```r
set.seed(1)

cv.error.10 <- rep(NA, 10)

for (i in 2:10){
  Wage$age.cut = cut(Wage$age, i)
  cv.error.10[i] <- cv.glm(Wage, glm(wage ~ age.cut, data=Wage), K=10)$delta[1]
}
cat('Min MSE Cut',which.min(cv.error.10), '\n')
```

```
## Min MSE Cut 8
```

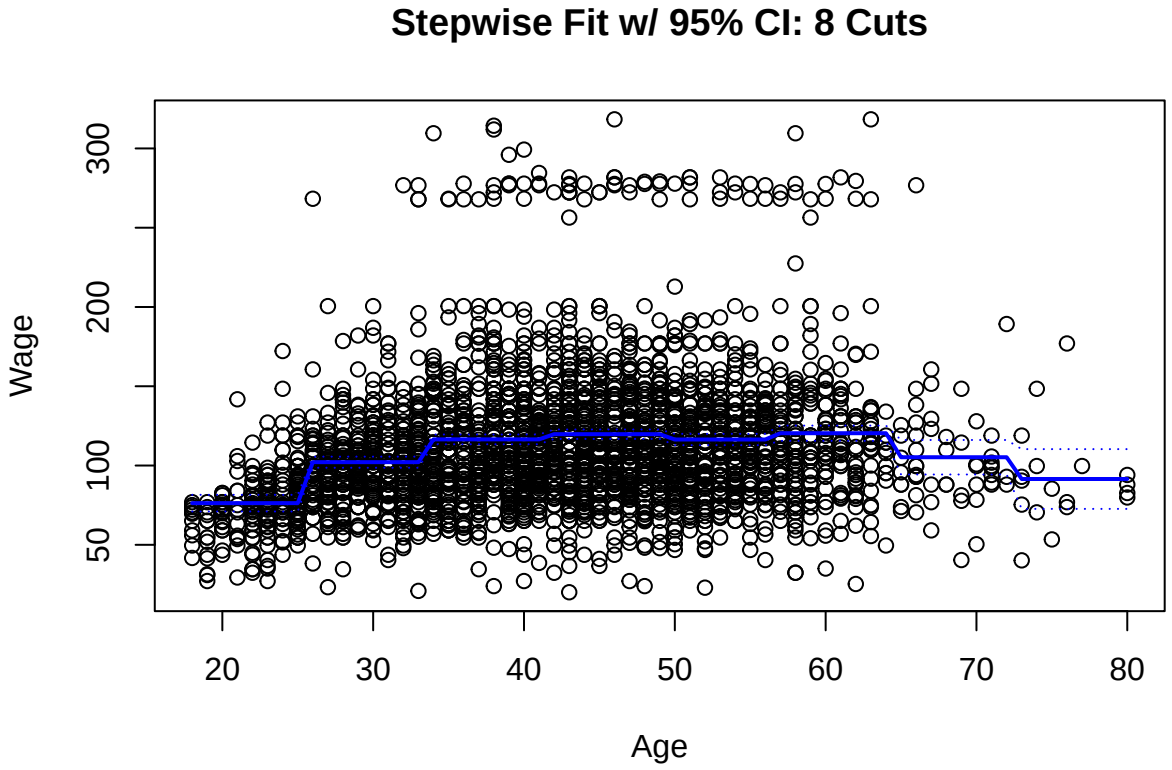
```r
plot(1:10, cv.error.10, type='b',
     main = '10-Fold CV for Step Function Cuts 1-10 \n Optimal: 8',
     xlab = 'Number of Cuts',
     ylab = 'MSE')
abline(v = which.min(cv.error.10))
```



We see from our 10-fold CV results testing cuts 2-10 that the optimal number of cuts based on MSE is eight cuts. We plot this below.

```
agelims <- range(Wage$age)
age.grid <- seq(from=agelims[1], to=agelims[2])
preds <- predict(glm(wage ~ cut(age, 8), data=Wage), newdata=list(age=age.grid), se=TRUE)
se.bands <- cbind(preds$fit + 2*preds$se.fit, preds$fit - 2*preds$se.fit)

plot(Wage$age, Wage$wage,
     main = 'Stepwise Fit w/ 95% CI: 8 Cuts',
     xlab = 'Age',
     ylab = 'Wage')
lines(age.grid, preds$fit, lwd=2, col='blue')
matlines(age.grid, se.bands, lwd=1, col='blue', lty=3)
```



9

(a)

```
head(Boston)
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio lstat medv
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3  4.98 24.0
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8  9.14 21.6
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8  4.03 34.7
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7  2.94 33.4
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7  5.33 36.2
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7  5.21 28.7
```

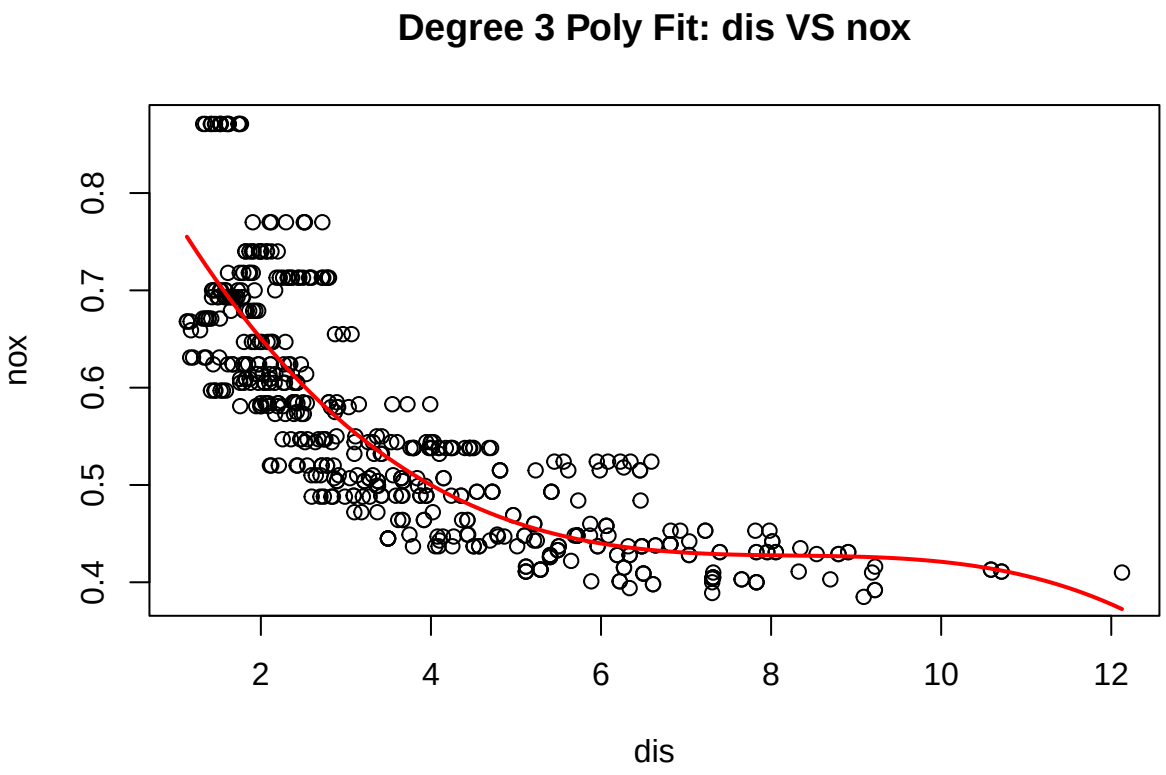
```
poly.fit <- glm(nox ~ poly(dis, 3), data=Boston)
summary(poly.fit)
```

```
##
## Call:
## glm(formula = nox ~ poly(dis, 3), data = Boston)
##
## 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
```

```
##
## (Dispersion parameter for gaussian family taken to be 0.003852802)
##
##     Null deviance: 6.7810  on 505  degrees of freedom
## Residual deviance: 1.9341  on 502  degrees of freedom
## AIC: -1370.9
##
## Number of Fisher Scoring iterations: 2
```

```r
dis.range <- range(Boston$dis)
dis.grid <- seq(dis.range[1], dis.range[2], length.out=100)
preds <- predict(poly.fit, newdata=list(dis = dis.grid))

plot(Boston$dis, Boston$nox,
     main='Degree 3 Poly Fit: dis VS nox',
     xlab='dis',
     ylab='nox')
lines(dis.grid, preds, lwd=2, col='red')
```



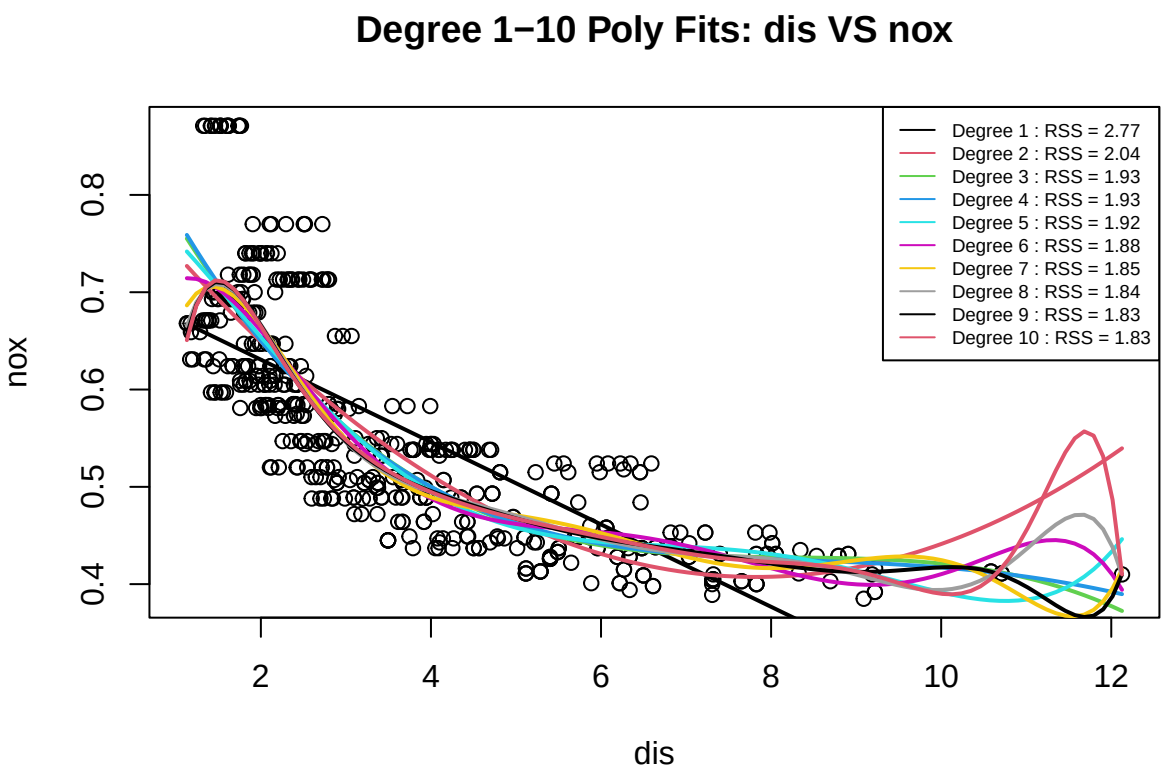
(b)

```r
dis_range <- range(Boston$dis)
dis.grid <- seq(dis_range[1], dis_range[2], length.out=100)
rss_vec <- rep(NA, 10)
pred_mat <- matrix(NA, nrow=100, ncol=10)

for (i in 1:10){
  fit <- glm(nox ~ poly(dis, i), data=Boston)
  pred_mat[,i] <- predict(fit, newdata=list(dis = dis.grid))
  rss_vec[i] <- sum(fit$residuals^2)
```

```
}
```

```
plot(Boston$dis, Boston$nox,
     main='Degree 1-10 Poly Fits: dis VS nox',
     xlab='dis',
     ylab='nox')
matlines(dis.grid, pred_mat, lty=1, lwd=2, col=c(1:10))
legend('topright', legend=paste('Degree',1:10, ': RSS =', round(rss_vec[1:10],2)), col=c(1:10), lty=1,
```



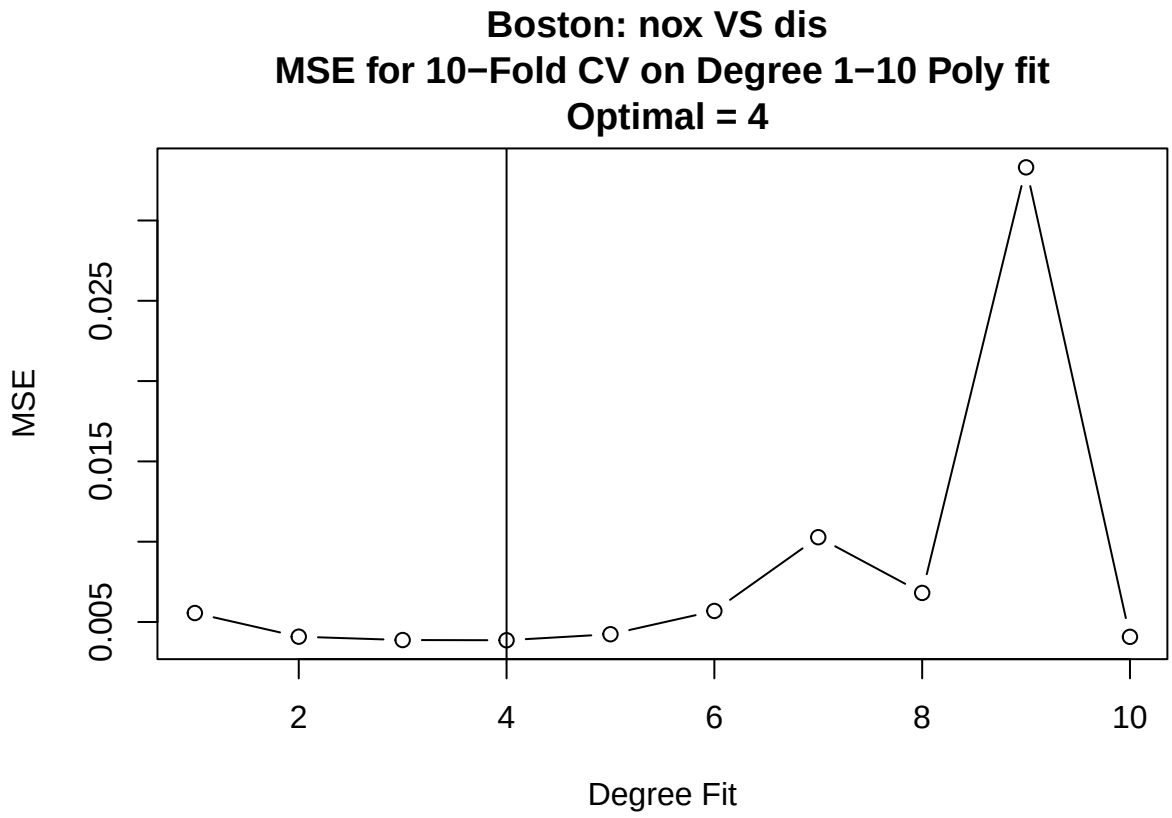
We see in the above plot of 10 different poly fits, from degree one to degree ten, that degree one does poorly but degree two through ten are relatively close. The best degrees are degree nine and degree ten, though it is important to note that tail behavior becomes more extreme at higher order polynomials. This holds true especially for the degree nine and degree ten fit, suggesting overfitting.

(c)

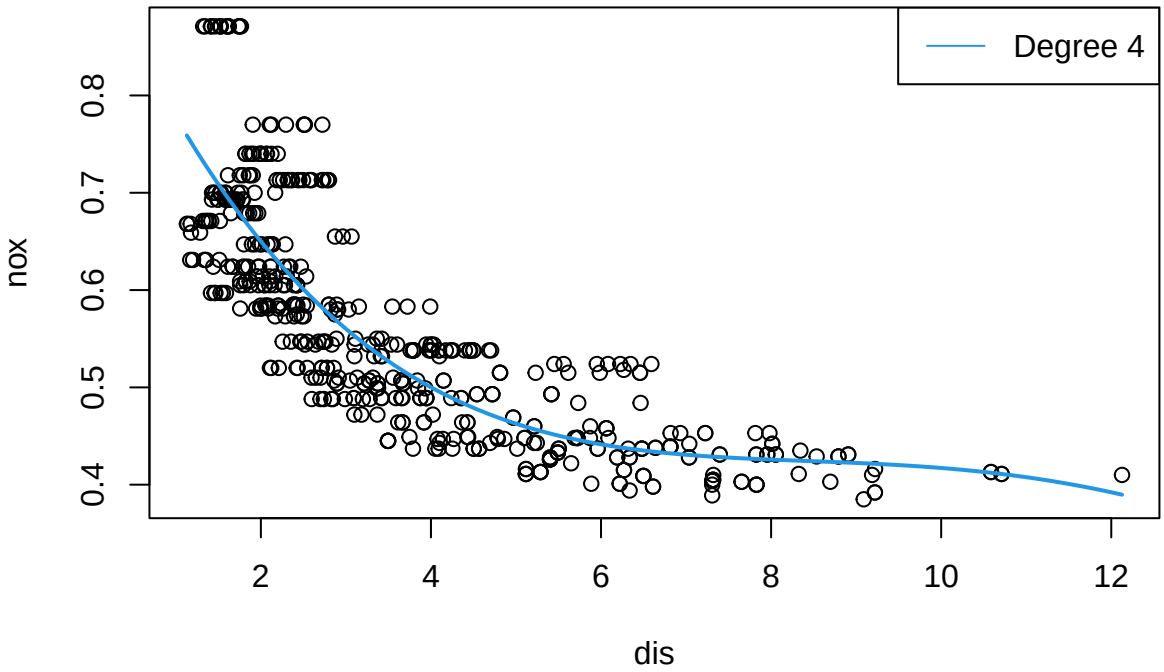
```
set.seed(1)

cv.error.10 <- rep(NA, 10)
for (i in 1:10){
  fit <- glm(nox ~ poly(dis, i), data=Boston)
  cv.error.10[i] <- cv.glm(Boston, fit, K=10)$delta[1]
}
```

```
plot(1:10, cv.error.10, type='b',
     main='Boston: nox VS dis \n MSE for 10-Fold CV on Degree 1-10 Poly fit \n Optimal = 4',
     xlab='Degree Fit',
     ylab='MSE')
abline(v=which.min(cv.error.10))
```




```
plot(Boston$dis, Boston$nox,
     main='Degree 4 Poly Fit: dis VS nox',
     xlab='dis',
     ylab='nox')
matlines(dis.grid, pred_mat[,4], lty=1, lwd=2, col=4)
legend('topright', legend=paste('Degree 4'), col=4, lty=1)
```



Using 10-fold CV on degrees one through ten polynomial fits, we obtained four degrees as the optimal fit based on minimizing MSE. It performed relatively well when just fitting the data and recording the RSS, relative to the other fits in the plot in (b). But on that same plot, you can see extreme tail behavior for higher order polynomials which would point to overfitting. Thus, four degrees makes sense since the tail behavior was modest while still giving accurate predictions.