Advanced Regression Analysis with Cross-Validation Techniques

Preethi Sree Allam

2023-04-11

Cross-Validation and the Bootstrap

The Validation Set Approach

```
library(ISLR2)
set.seed(1)
train <- sample(392, 196)
lm.fit <- lm(mpg ~ horsepower, data = Auto, subset = train)
attach(Auto)
mean((mpg - predict(lm.fit, Auto))[-train]^2)
```

```
## [1] 23.26601
```

```
lm.fit2 <- lm(mpg ~ poly(horsepower, 2), data = Auto,
              subset = train)
mean((mpg - predict(lm.fit2, Auto))[-train]^2)
```

```
## [1] 18.71646
```

```
lm.fit3 <- lm(mpg ~ poly(horsepower, 3), data = Auto,
              subset = train)
mean((mpg - predict(lm.fit3, Auto))[-train]^2)
```

```
## [1] 18.79401
```

```
set.seed(2)
train <- sample(392, 196)
lm.fit <- lm(mpg ~ horsepower, subset = train)
mean((mpg - predict(lm.fit, Auto))[-train]^2)
```

```
## [1] 25.72651
```

```
lm.fit2 <- lm(mpg ~ poly(horsepower, 2), data = Auto,
              subset = train)
mean((mpg - predict(lm.fit2, Auto))[-train]^2)
```

```
## [1] 20.43036
```

```
lm.fit3 <- lm(mpg ~ poly(horsepower, 3), data = Auto,
              subset = train)
mean((mpg - predict(lm.fit3, Auto))[-train]^2)
```

```
## [1] 20.38533
```

Leave-One-Out Cross-Validation

```
glm.fit <- glm(mpg ~ horsepower, data = Auto)
coef(glm.fit)
```

```
## (Intercept)  horsepower
##  39.9358610  -0.1578447
```

```
lm.fit <- lm(mpg ~ horsepower, data = Auto)
coef(lm.fit)
```

```
## (Intercept)  horsepower
##  39.9358610  -0.1578447
```

```
library(boot)
glm.fit <- glm(mpg ~ horsepower, data = Auto)
cv.err <- cv.glm(Auto, glm.fit)
cv.err$delta
```

```
## [1] 24.23151 24.23114
```

```
cv.error <- rep(0, 10)
for (i in 1:10) {
  glm.fit <- glm(mpg ~ poly(horsepower, i), data = Auto)
  cv.error[i] <- cv.glm(Auto, glm.fit)$delta[1]
}
cv.error
```

```
##  [1] 24.23151 19.24821 19.33498 19.42443 19.03321 18.97864 18.83305 18.96115
##  [9] 19.06863 19.49093
```

k-Fold Cross-Validation

```
set.seed(17)
cv.error.10 <- rep(0, 10)
for (i in 1:10) {
  glm.fit <- glm(mpg ~ poly(horsepower, i), data = Auto)
  cv.error.10[i] <- cv.glm(Auto, glm.fit, K = 10)$delta[1]
}
cv.error.10
```

```
##  [1] 24.27207 19.26909 19.34805 19.29496 19.03198 18.89781 19.12061 19.14666
##  [9] 18.87013 20.95520
```

Lab: Non-linear Modeling

```
library(ISLR2)
attach(Wage)
```

```
## The following object is masked from Auto:
##
##     year
```

Polynomial Regression and Step Functions

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

```
fit2 <- lm(wage ~ poly(age, 4, raw = T), data = Wage)
coef(summary(fit2))
```

```
##                               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
```

```
fit2a <- lm(wage ~ age + I(age^2) + I(age^3) + I(age^4),
            data = Wage)
coef(fit2a)
```

```
##   (Intercept)           age      I(age^2)      I(age^3)      I(age^4)
## -1.841542e+02  2.124552e+01 -5.638593e-01  6.810688e-03 -3.203830e-05
```

```
fit2b <- lm(wage ~ cbind(age, age^2, age^3, age^4),
            data = Wage)
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)
par(mfrow = c(1, 2), 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$fit, lwd = 2, col = "blue")
matlines(age.grid, se.bands, lwd = 1, col = "blue", lty = 3)
preds2 <- predict(fit2, newdata = list(age = age.grid),
                  se = TRUE)
max(abs(preds$fit - preds2$fit))
```

```
## [1] 6.88658e-11
```

```
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
```

```
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
```

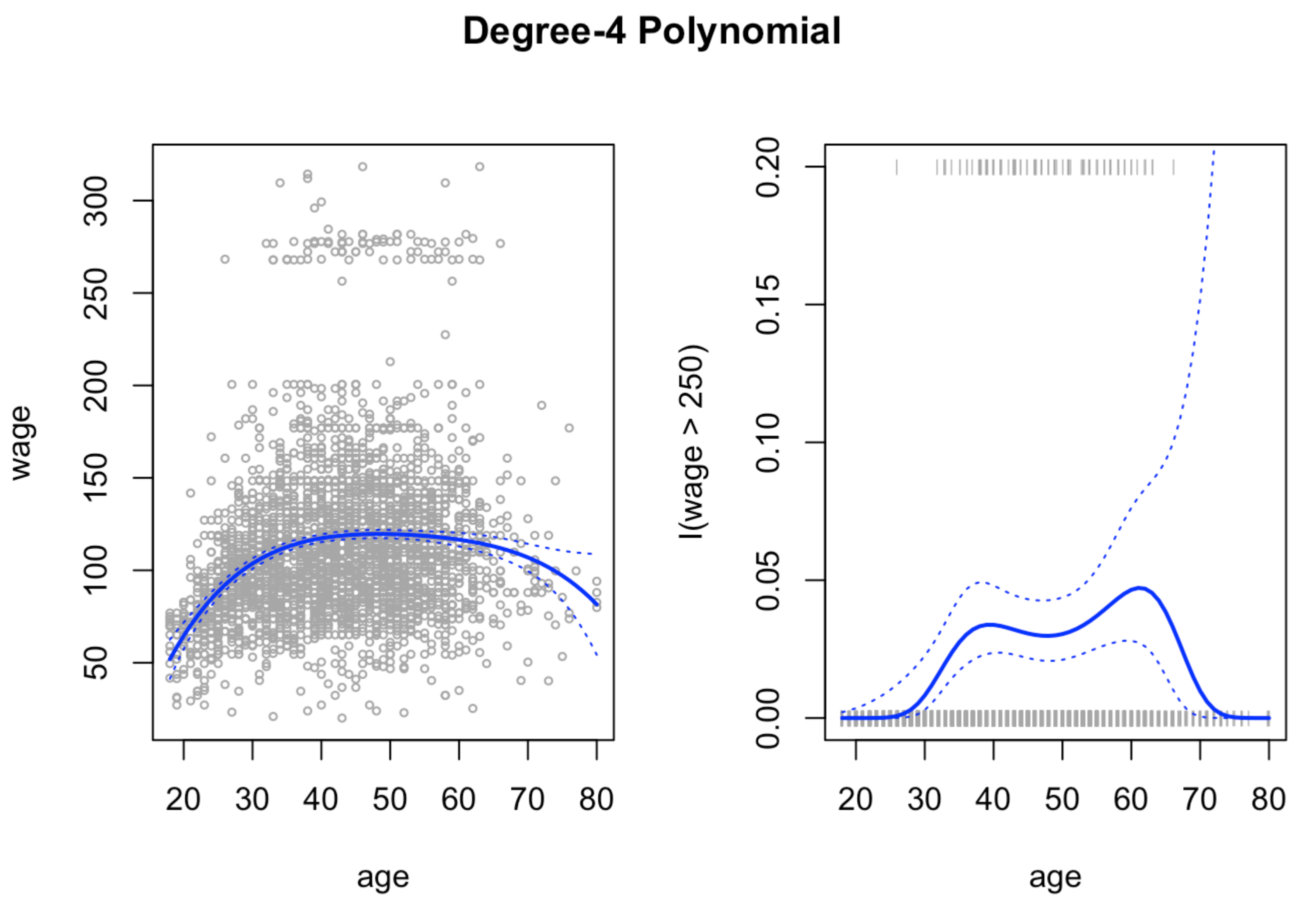
```
(-11.983)^2
```

```
## [1] 143.5923
```

```
fit.1 <- lm(wage ~ education + age, data = Wage)
fit.2 <- lm(wage ~ education + poly(age, 2), data = Wage)
fit.3 <- lm(wage ~ education + poly(age, 3), data = Wage)
anova(fit.1, fit.2, fit.3)
```

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

```
fit <- glm(I(wage > 250) ~ poly(age, 4), data = Wage,
           family = binomial)
preds <- predict(fit, newdata = list(age = age.grid), se = T)
pfit <- exp(preds$fit) / (1 + exp(preds$fit))
se.bands.logit <- cbind(preds$fit + 2 * preds$se.fit,
                        preds$fit - 2 * preds$se.fit)
se.bands <- exp(se.bands.logit) / (1 + exp(se.bands.logit))
preds <- predict(fit, newdata = list(age = age.grid),
                 type = "response", se = T)
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)
```




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

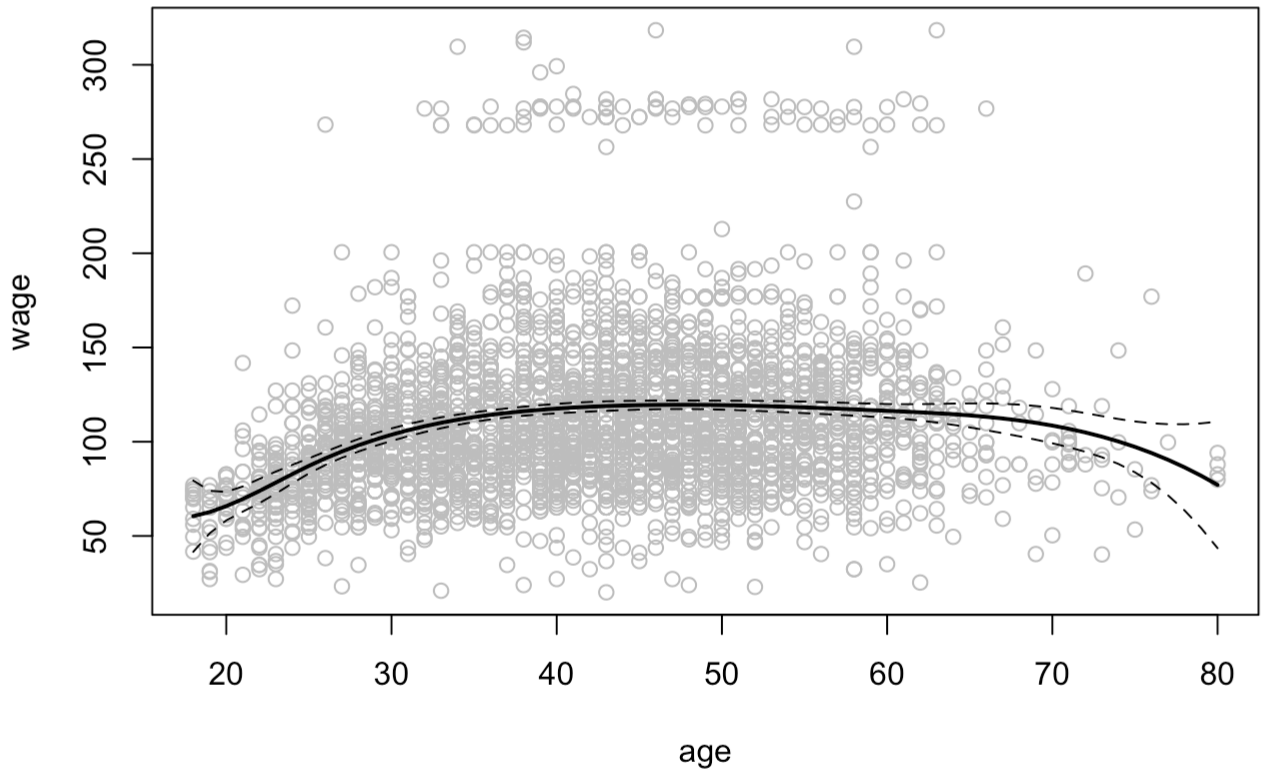
```
##
## (17.9,33.5]   (33.5,49]   (49,64.5] (64.5,80.1]
##         750        1399         779          72
```

```
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
```

Splines

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




```
dim(bs(age, knots = c(25, 40, 60)))
```

```
## [1] 3000    6
```

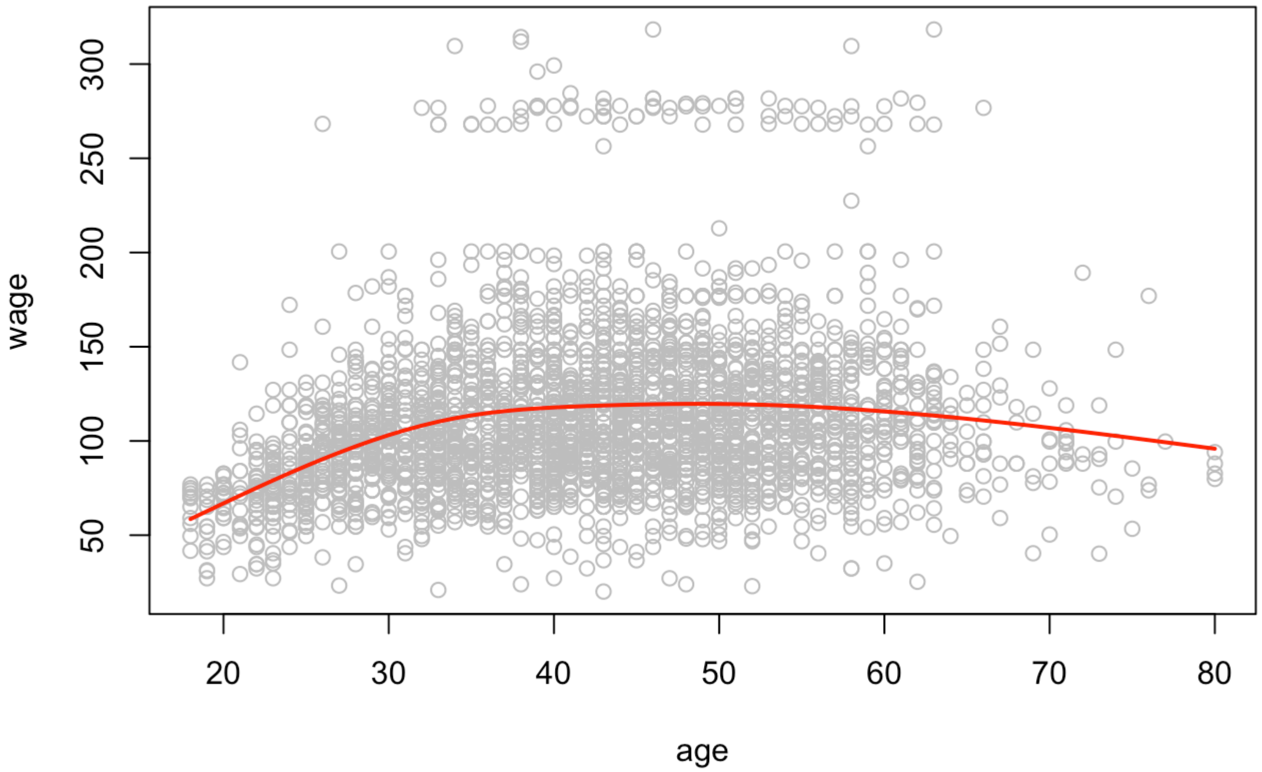
```
dim(bs(age, df = 6))
```

```
## [1] 3000    6
```

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

```
## [1] 33.75 42.00 51.00
```

```
fit2 <- lm(wage ~ ns(age, df = 4), data = Wage)
pred2 <- predict(fit2, newdata = list(age = age.grid),
                 se = T)
plot(age, wage, col = "gray")
lines(age.grid, pred2$fit, col = "red", lwd = 2)
```



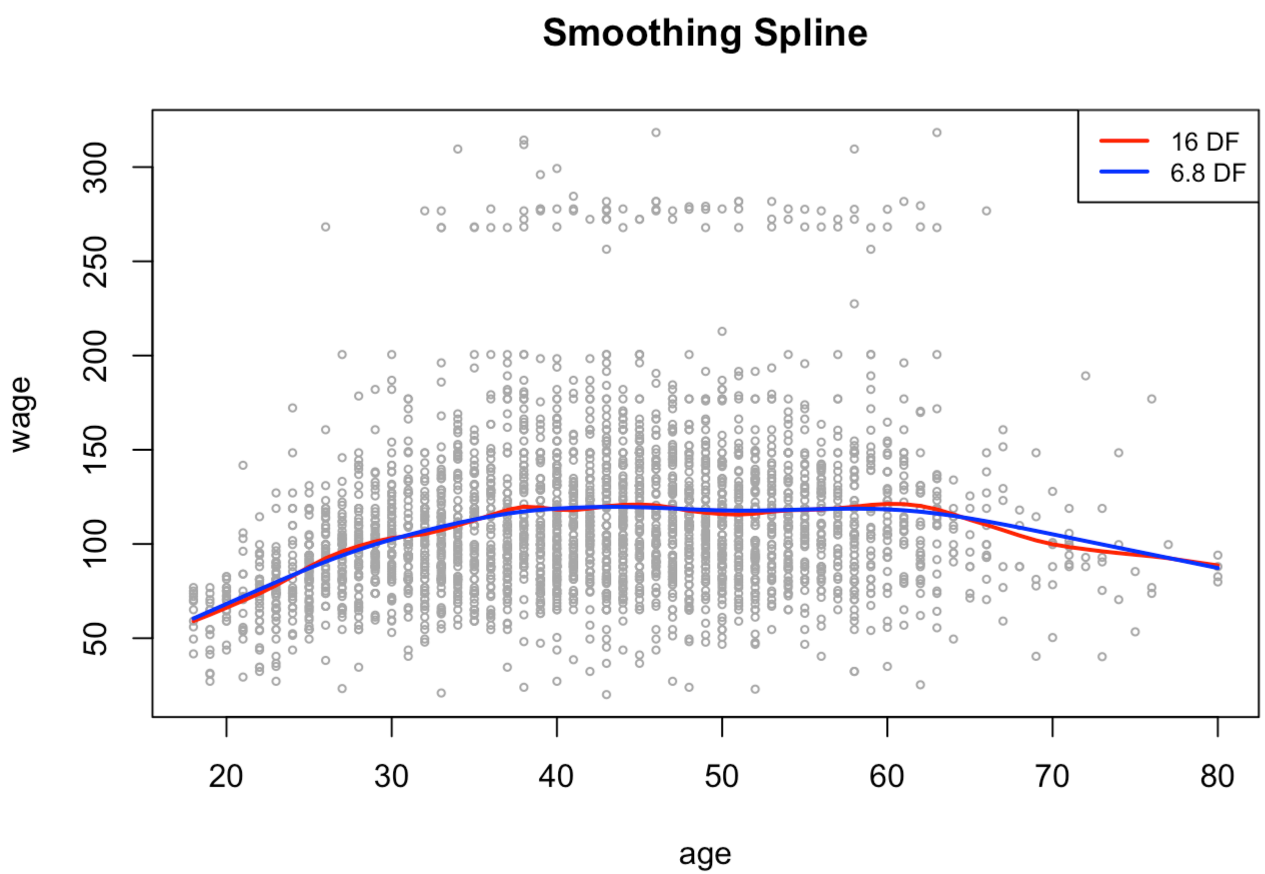
```
plot(age, wage, xlim = agelims, cex = .5, col = "darkgrey")
title("Smoothing Spline")
fit <- smooth.spline(age, wage, df = 16)
fit2 <- smooth.spline(age, wage, cv = TRUE)
```

```
## Warning in smooth.spline(age, wage, cv = TRUE): cross-validation with
## non-unique 'x' values seems doubtful
```

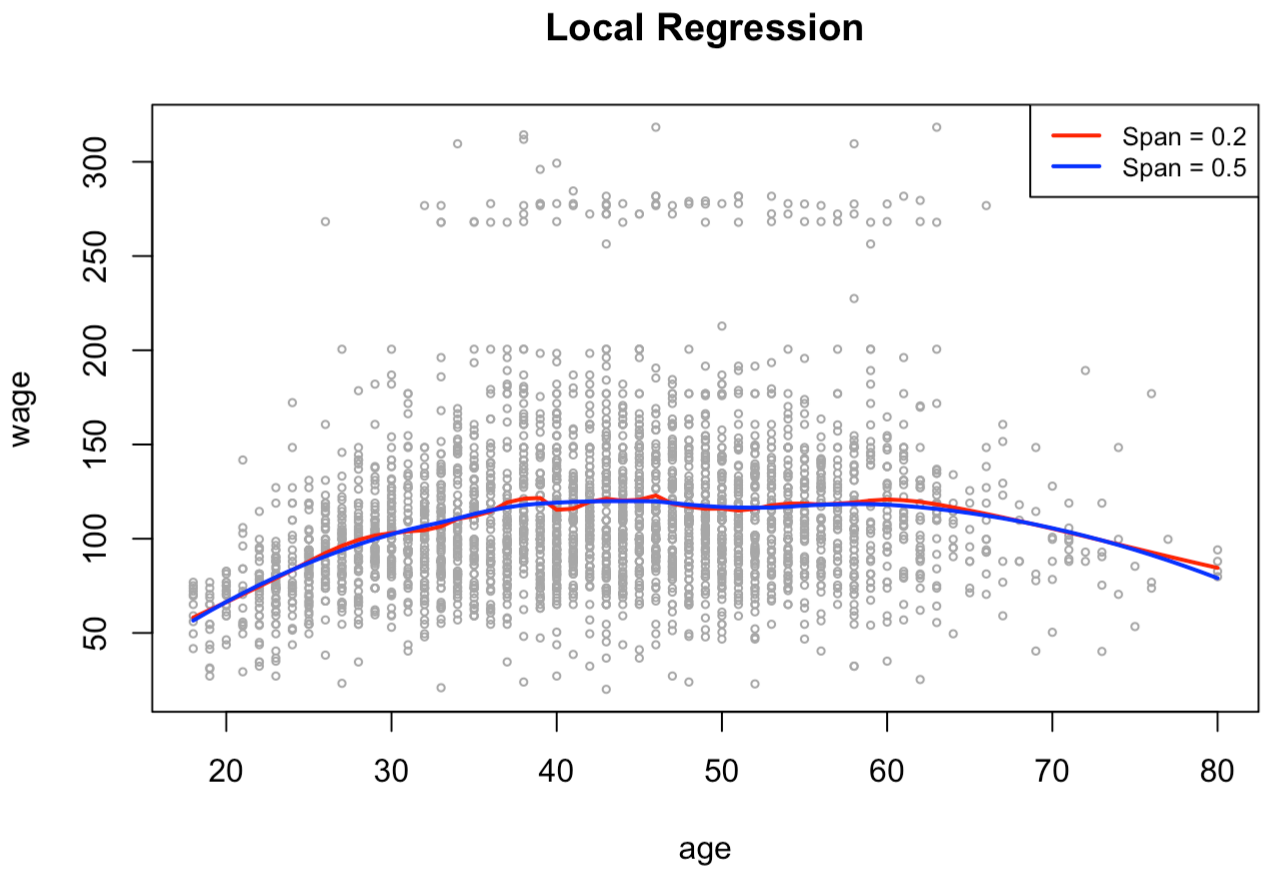
```
fit2$df
```

```
## [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 = .8)
```




```
plot(age, wage, xlim = agelims, cex = .5, col = "darkgrey")
title("Local Regression")
fit <- loess(wage ~ age, span = .2, data = Wage)
fit2 <- loess(wage ~ age, span = .5, data = Wage)
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 = .8)
```



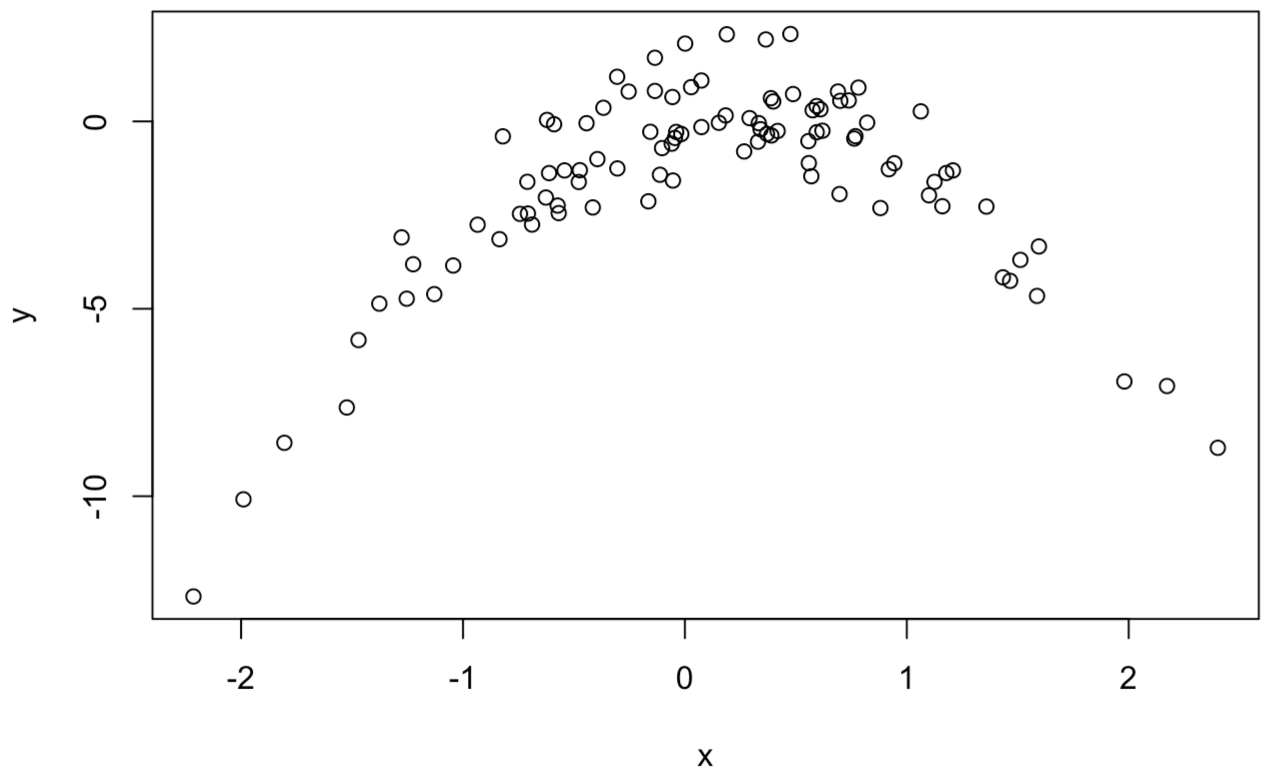

```r
library(ISLR)
```

```
##
## Attaching package: 'ISLR'
```

```
## The following objects are masked from 'package:ISLR2':
##
##     Auto, Credit
```

```
set.seed(1)
x=rnorm(100)
y=x - 2*x^2 + rnorm(100)
```

```
plot(x,y)
```



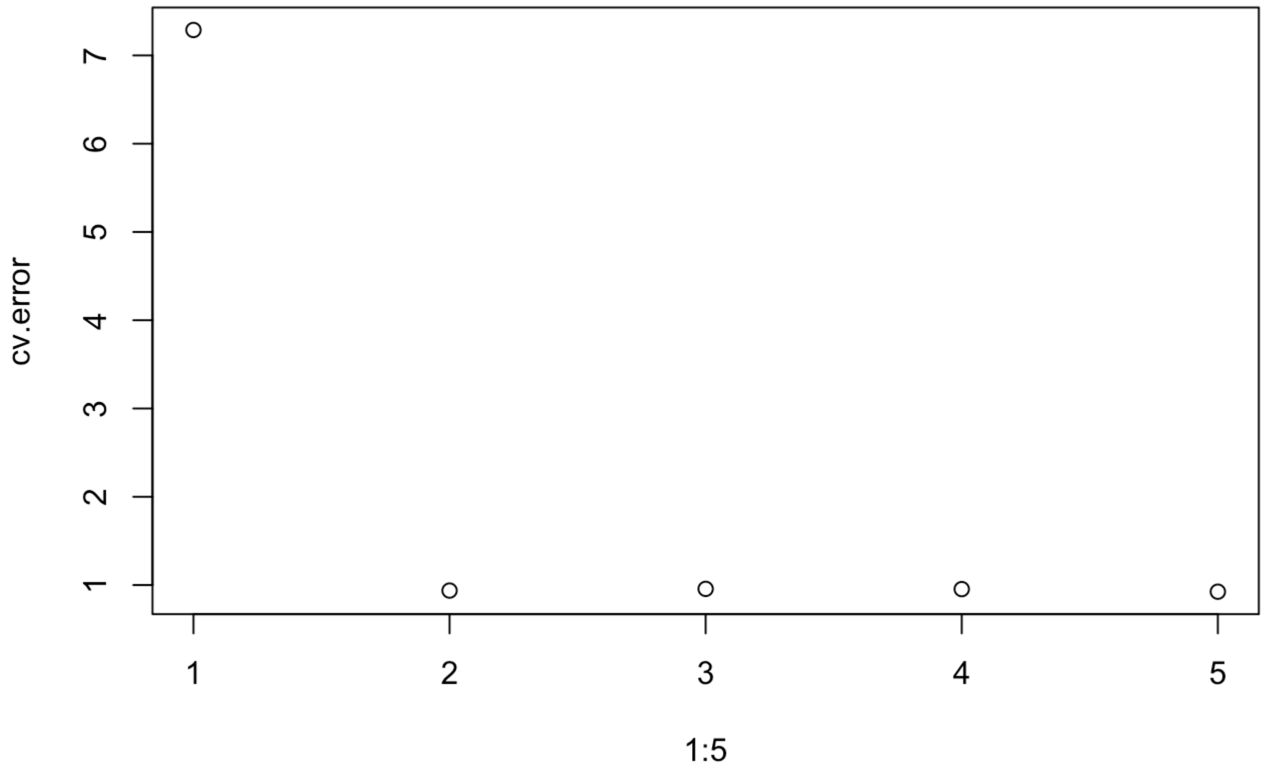
```r
library(boot)
loocv.fn = function(df, exponent) {
  glm.fit = glm(y~poly(x, exponent), df)
}
set.seed(1)
df = data.frame(x,y)
cv.error=rep(0,5)
for (i in 1:5) {
  print(i)
  glm.fit=glm(y~poly(x, i))
  print(cv.glm(df, glm.fit)$delta)
}
```

```
## [1] 1
## [1] 7.288162 7.284744
## [1] 2
## [1] 0.9374236 0.9371789
## [1] 3
## [1] 0.9566218 0.9562538
## [1] 4
## [1] 0.9539049 0.9534453
## [1] 5
## [1] 0.9247836 0.9243911
```

```r
set.seed(10)
cv.error=rep(0,5)
for (i in 1:5) {
  print(i)
  glm.fit=glm(y~poly(x, i))
  cv.error[i] = cv.glm(df, glm.fit)$delta[1]
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

```
plot(1:5, cv.error)
```



We get the same answer because LOOCV only leaves out one data point out Cycles through all points one at a time to train.

Because original equation was quadratic, the quadratic fit will match the best

```
summary(glm.fit)
```

```
##
## Call:
## glm(formula = y ~ poly(x, i))
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5500     0.0952 -16.282  < 2e-16 ***
## poly(x, i)1   6.1888     0.9520   6.501  3.8e-09 ***
## poly(x, i)2 -23.9483     0.9520 -25.156  < 2e-16 ***
## poly(x, i)3   0.2641     0.9520   0.277    0.782
## poly(x, i)4   1.2571     0.9520   1.321    0.190
## poly(x, i)5   1.4802     0.9520   1.555    0.123
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9062565)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  85.188  on 94  degrees of freedom
## AIC: 281.76
##
## Number of Fisher Scoring iterations: 2
```

The linear and quadratic have significance p-value < 0.05

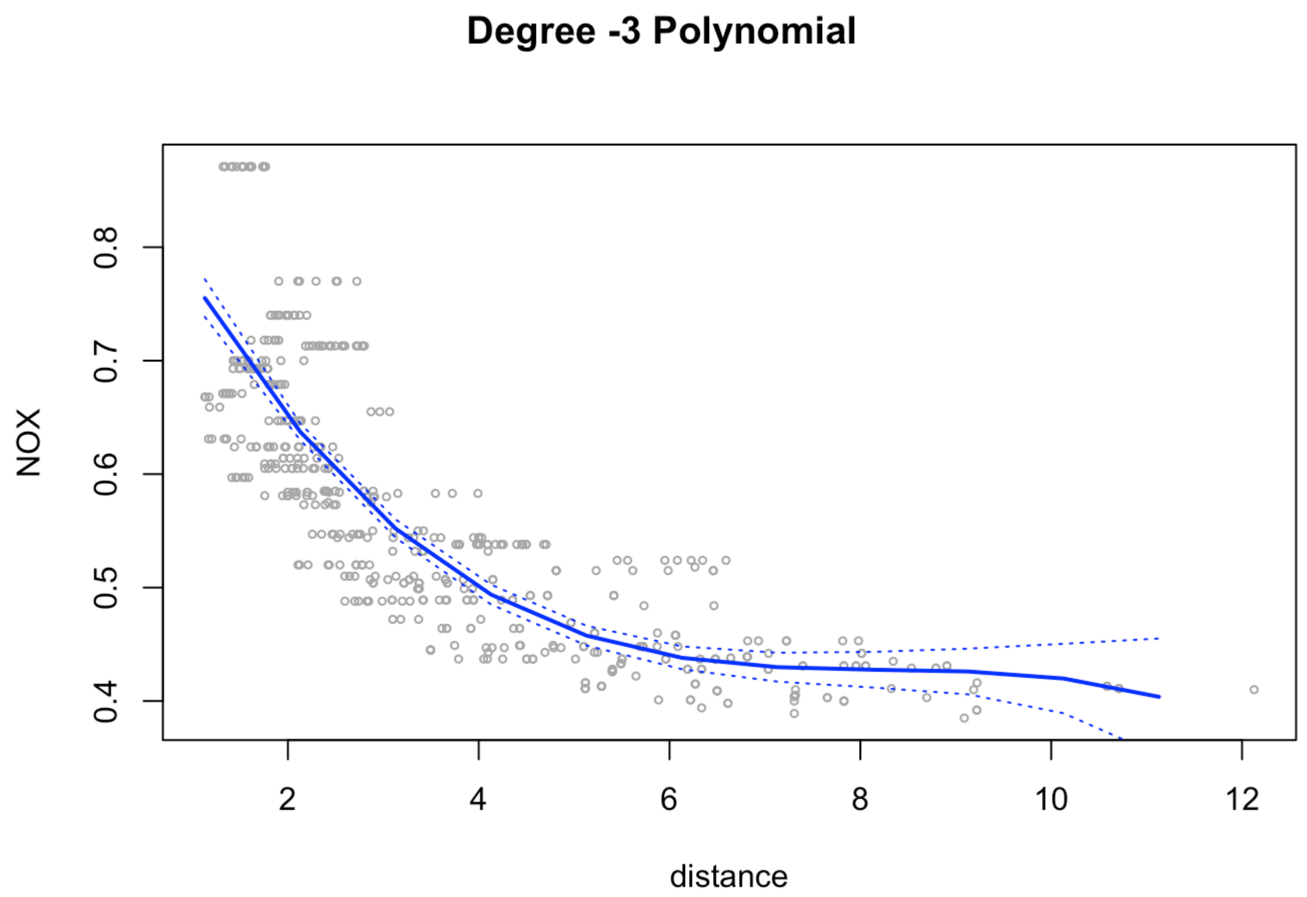
```
attach(Boston)
```

```
## The following object is masked from Wage:
##
##     age
```

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

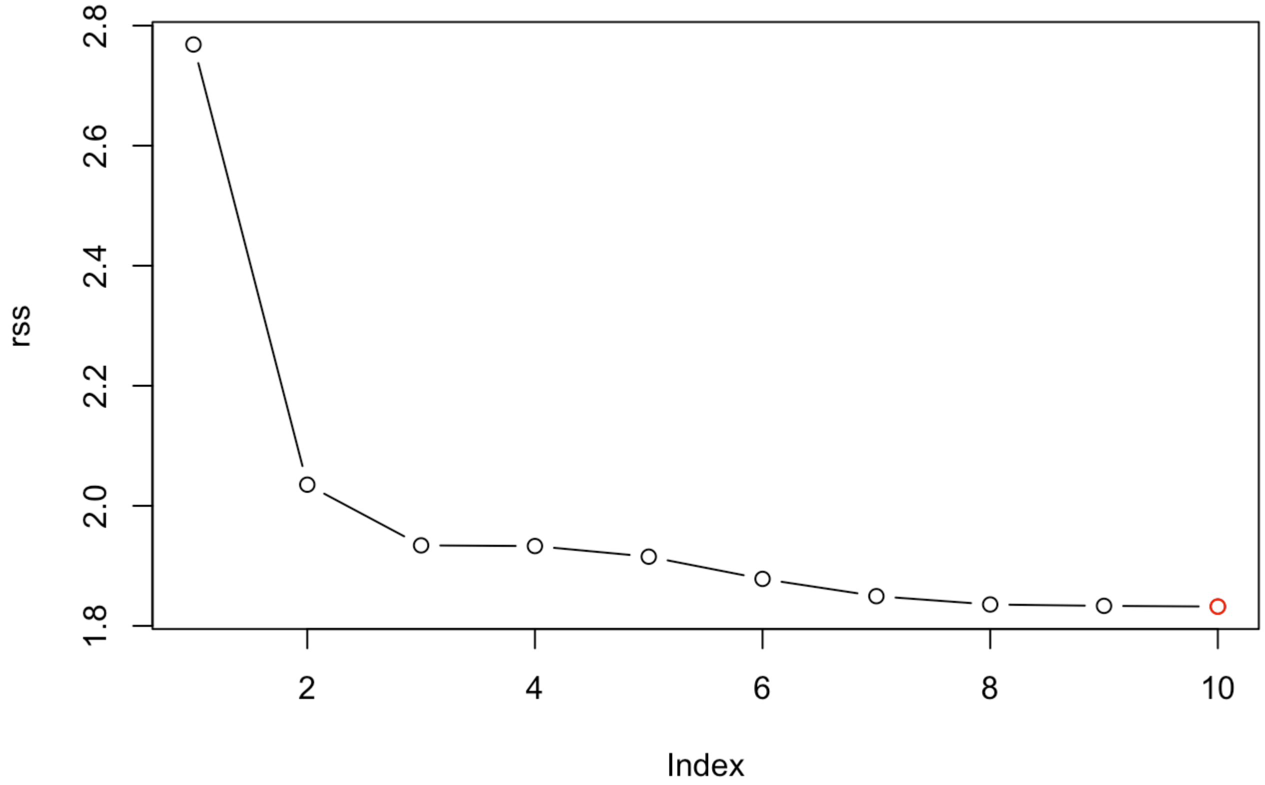
```
##
## Call:
## lm(formula = nox ~ poly(dis, 3), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.121130 -0.040619 -0.009738  0.023385  0.194904
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.554695   0.002759 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
##
## Residual standard error: 0.06207 on 502 degrees of freedom
## Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131
## F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
dislims  = range(dis)
dis.grid = seq(from=dislims[1], to=dislims[2])
preds    = predict(fit, newdata=list(dis=dis.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 (dis , nox , xlim = dislims ,xlab = 'distance',ylab = 'NOX', cex = .5, col = "darkgrey")
title ("Degree -3 Polynomial", outer = T)
lines (dis.grid, preds$fit , lwd = 2, col = "blue")
matlines (dis.grid , se.bands, lwd = 1, col = "blue", lty = 3)
```



Model we choose fits the data well.

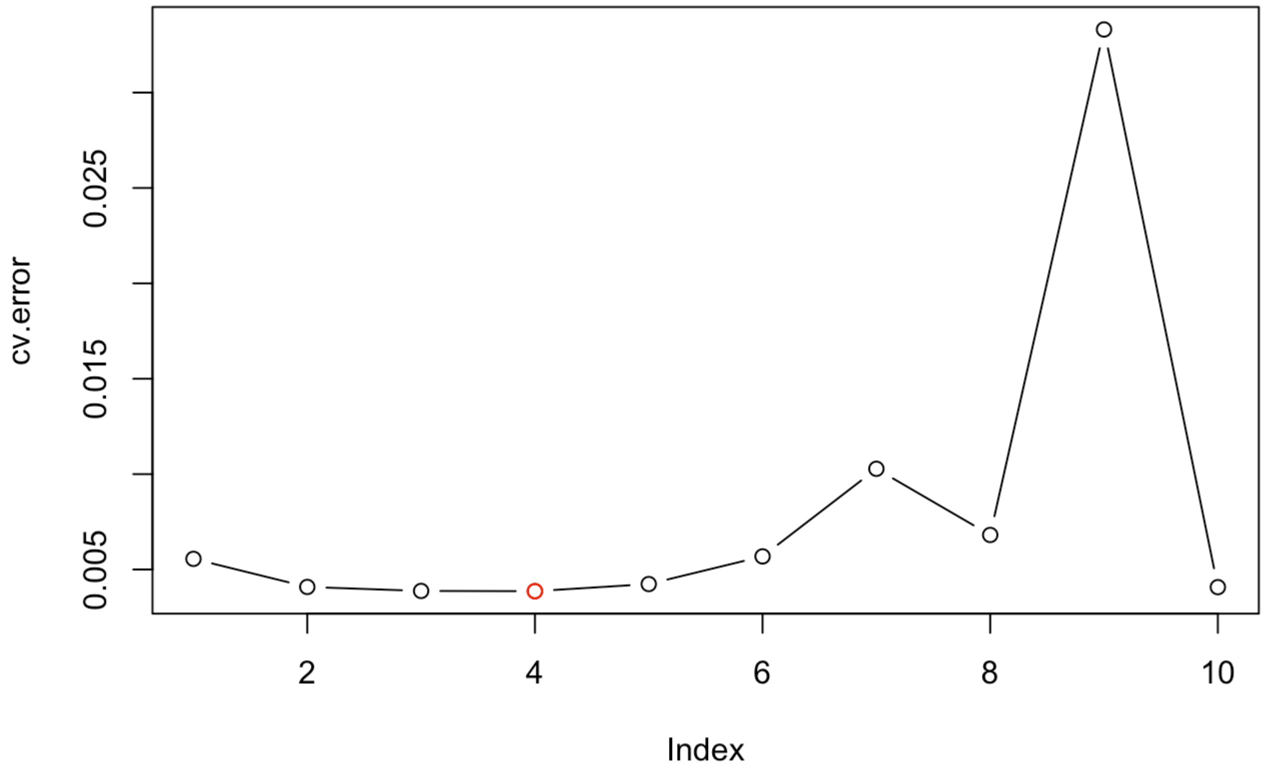
```
rss = rep(0,10)
for (i in 1:10) {
  lm.fit = lm(nox~poly(dis,i), data=Boston)
  rss[i] = sum(lm.fit$residuals^2)
}
plot(rss, type='b')
points(which.min(rss), rss[which.min(rss)], col='red')
```




```
rss
```

```
##  [1] 2.768563 2.035262 1.934107 1.932981 1.915290 1.878257 1.849484 1.835630
##  [9] 1.833331 1.832171
```

```
set.seed(1)
cv.error = rep(0,10)
for (i in 1:10) {
  glm.fit     = glm(nox~poly(dis,i), data=Boston)
  cv.error[i] = cv.glm(Boston, glm.fit, K=10)$delta[1]
}
plot(cv.error, type='b')
points(which.min(cv.error), cv.error[which.min(cv.error)], col='red')
```



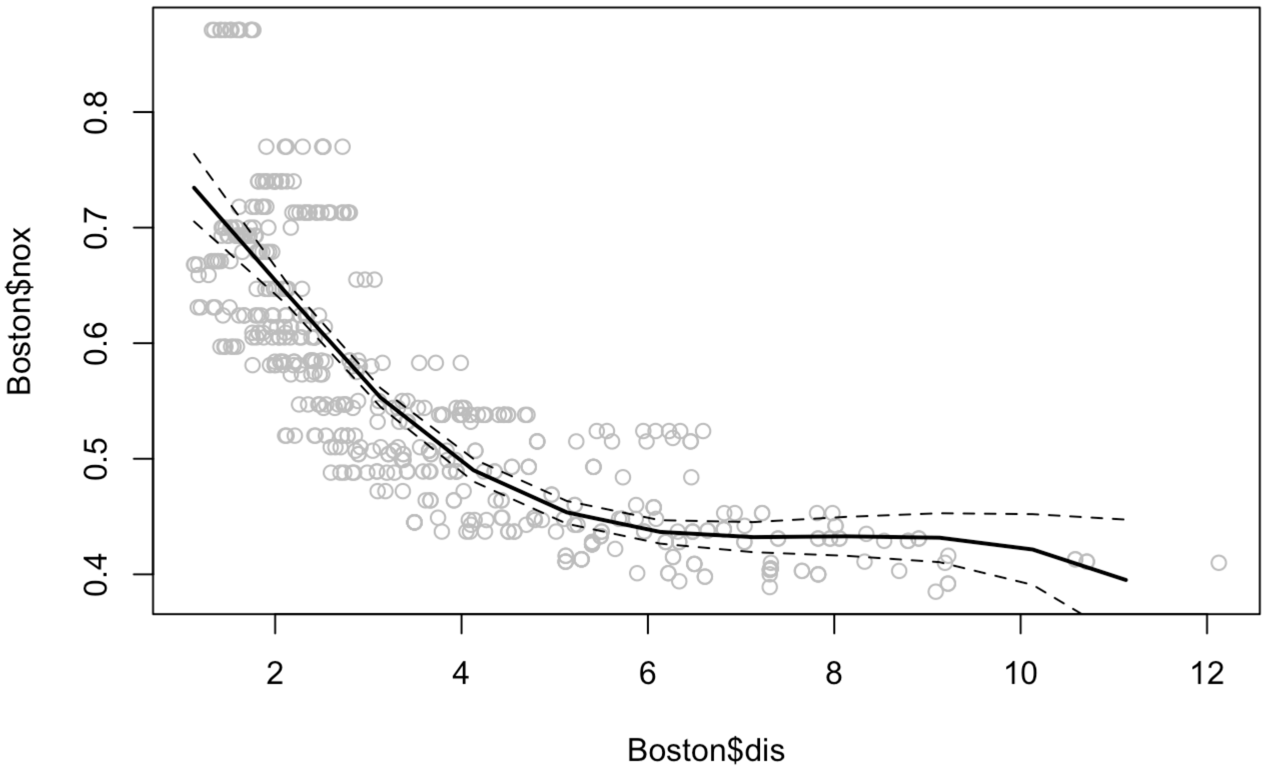
Model with the degree 4 shows minimum error and with degree greater than 6 resulting overfitting.

```
library(splines)
fit  = lm(nox~bs(dis, df=4),data=Boston)
pred = predict(fit, newdata =list(dis=dis.grid), se=T)
attr(bs(Boston$dis, df=4),"knots")
```

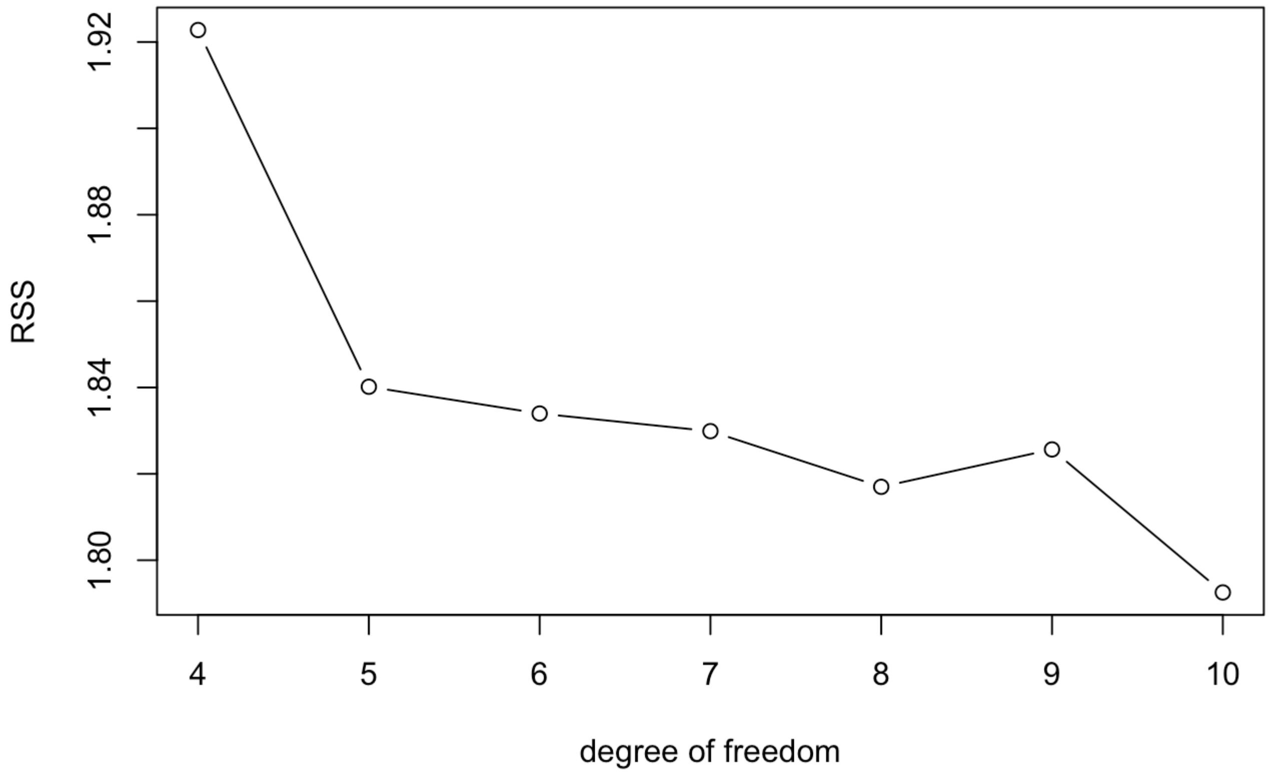
```
## [1] 3.20745
```

knots are calculated on the given degrees of freedom.

```
plot(Boston$dis, Boston$nox, col='gray')
lines(dis.grid, pred$fit ,lwd=2)
lines(dis.grid, pred$fit +2*pred$se, lty='dashed')
lines(dis.grid, pred$fit -2*pred$se, lty='dashed')
```




```
set.seed(1)
rss = rep(0,7)
for (i in 4:10) {
  fit      = lm(nox~bs(dis, df=i), data=Boston)
  rss[i-3] = sum(fit$residuals^2)
}
plot(4:10, rss, xlab='degree of freedom', ylab='RSS', type='b')
```



for the range 4 to 10 the rss lowest at 10 ,the rss is decreasing as the degree of freedom increases but not monotonically.

```
fit.4  = lm(nox~bs(dis, df=4), data=Boston)
fit.5  = lm(nox~bs(dis, df=5), data=Boston)
fit.6  = lm(nox~bs(dis, df=6), data=Boston)
fit.7  = lm(nox~bs(dis, df=7), data=Boston)
fit.8  = lm(nox~bs(dis, df=8), data=Boston)
fit.9  = lm(nox~bs(dis, df=9), data=Boston)
fit.10 = lm(nox~bs(dis, df=10), data=Boston)
anova(fit.4, fit.5, fit.6, fit.7, fit.8, fit.9, fit.10)
```

```
## Analysis of Variance Table
##
## Model 1: nox ~ bs(dis, df = 4)
## Model 2: nox ~ bs(dis, df = 5)
## Model 3: nox ~ bs(dis, df = 6)
## Model 4: nox ~ bs(dis, df = 7)
## Model 5: nox ~ bs(dis, df = 8)
## Model 6: nox ~ bs(dis, df = 9)
## Model 7: nox ~ bs(dis, df = 10)
##   Res.Df    RSS Df Sum of Sq       F    Pr(>F)
## 1    501 1.9228
## 2    500 1.8402  1  0.082602 22.8102 2.359e-06 ***
## 3    499 1.8340  1  0.006207  1.7140  0.191074
## 4    498 1.8299  1  0.004081  1.1271  0.288918
## 5    497 1.8170  1  0.012889  3.5593  0.059796 .
## 6    496 1.8256  1 -0.008657
## 7    495 1.7925  1  0.033118  9.1453  0.002623 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

df with 10 seems to be better choice.