STAT 1361 - Homework 6

Gordon Lu

4/1/2021

ISLR Conceptual Exercise 5:

5a)

$\hat{g}_2$ will have a smaller training RSS, since it will be a higher order polynomial which will have more degrees of freedom, thus producing a more flexible model.

5b)

$\hat{g}_1$ will have a smaller test RSS, as $\hat{g}_2$ could overfit with the extra degree of freedom, or to be more precise, it could overfit due to its high flexibility.

5c)

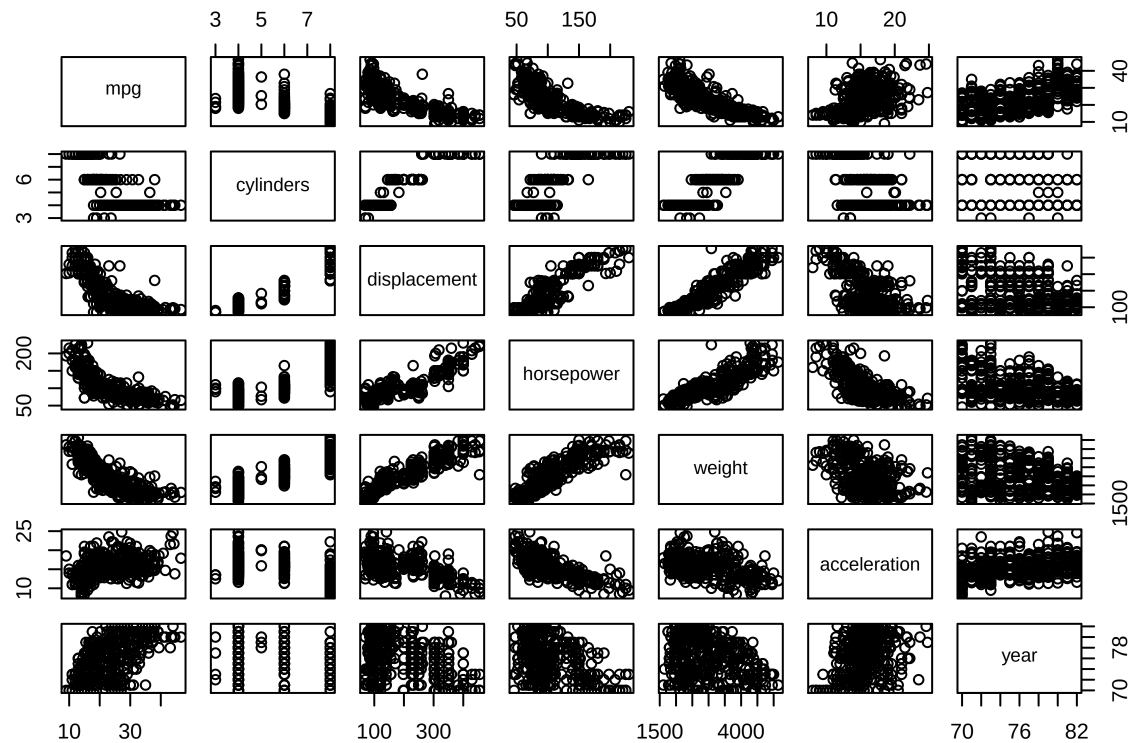
When $\lambda = 0$, the test and training RSS will be the same, so $\hat{g}_2 = \hat{g}_1$.

ISLR Applied Exercise 8:

```r
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.6.3
```

```r
pairs(Auto[1:7])
```



`mpg` appears to have a non-linear relationship with `horsepower`, `displacement`, and `weight`.

The relationship between `mpg` and `horsepower` will be explored below in-depth.

```
fit.1 = lm(mpg~horsepower, data = Auto)
fit.2 = lm(mpg~poly(horsepower,2), data = Auto)
fit.3 = lm(mpg~poly(horsepower,3), data = Auto)
fit.4 = lm(mpg~poly(horsepower,4), data = Auto)
fit.5 = lm(mpg~poly(horsepower,5), data = Auto)
fit.6 = lm(mpg~poly(horsepower,6), data = Auto)

anova(fit.1, fit.2, fit.3, fit.4, fit.5, fit.6)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ horsepower
## Model 2: mpg ~ poly(horsepower, 2)
## Model 3: mpg ~ poly(horsepower, 3)
## Model 4: mpg ~ poly(horsepower, 4)
## Model 5: mpg ~ poly(horsepower, 5)
## Model 6: mpg ~ poly(horsepower, 6)
##   Res.Df    RSS Df Sum of Sq        F    Pr(>F)
## 1    390 9385.9
## 2    389 7442.0  1   1943.89 104.6659 < 2.2e-16 ***
## 3    388 7426.4  1     15.59   0.8396  0.360083
## 4    387 7399.5  1     26.91   1.4491  0.229410
```

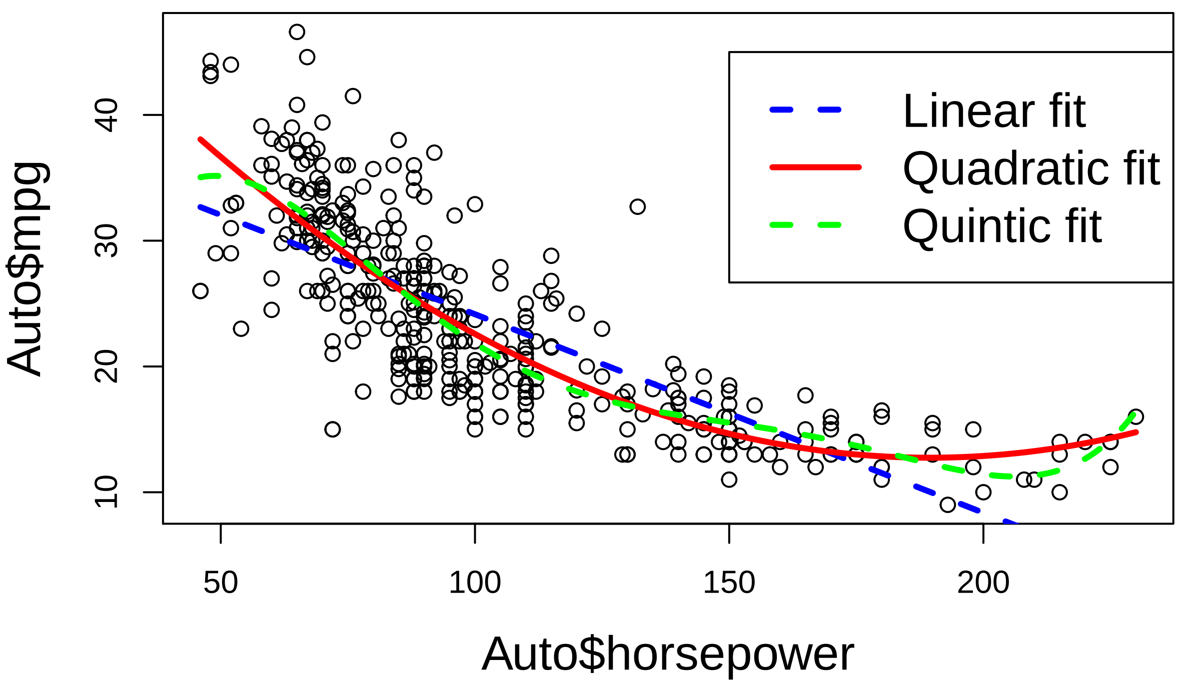
```
## 5    386 7223.4  1    176.15   9.4846  0.002221 **
## 6    385 7150.3  1     73.04   3.9326  0.048068 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Training RSS decreases over time. Quadratic polynomic is sufficient from ANOVA-perspective.

When taking into account the plot below and the ANOVA results, there is strong evidence of a non-linear relationship between `horsepower` and `mpg`.

I will now compare various fits to the data:

```
hplim = range(Auto$horsepower)
hp.grid = seq(from=hplim[1],to=hplim[2])
pred1 = predict(fit.1,newdata=list(horsepower=hp.grid))
pred2 = predict(fit.2,newdata=list(horsepower=hp.grid))
pred3 = predict(fit.5,newdata=list(horsepower=hp.grid))
plot(Auto$horsepower, Auto$mpg, xlim=hplim, cex.lab=1.5)
lines(hp.grid,pred1,lwd=3,col="blue",lty=2)
lines(hp.grid,pred2,lwd=3,col="red")
lines(hp.grid,pred3,lwd=3,col="green",lty=2)
legend(150,45,legend=c("Linear fit", "Quadratic fit", "Quintic fit"),
       col=c("blue", "red", "green"),lty=c(2,1,2), lwd=c(3,3,3),cex=1.5)
```



As can be seen, a quadratic model likely provides the best fit to the underlying data.

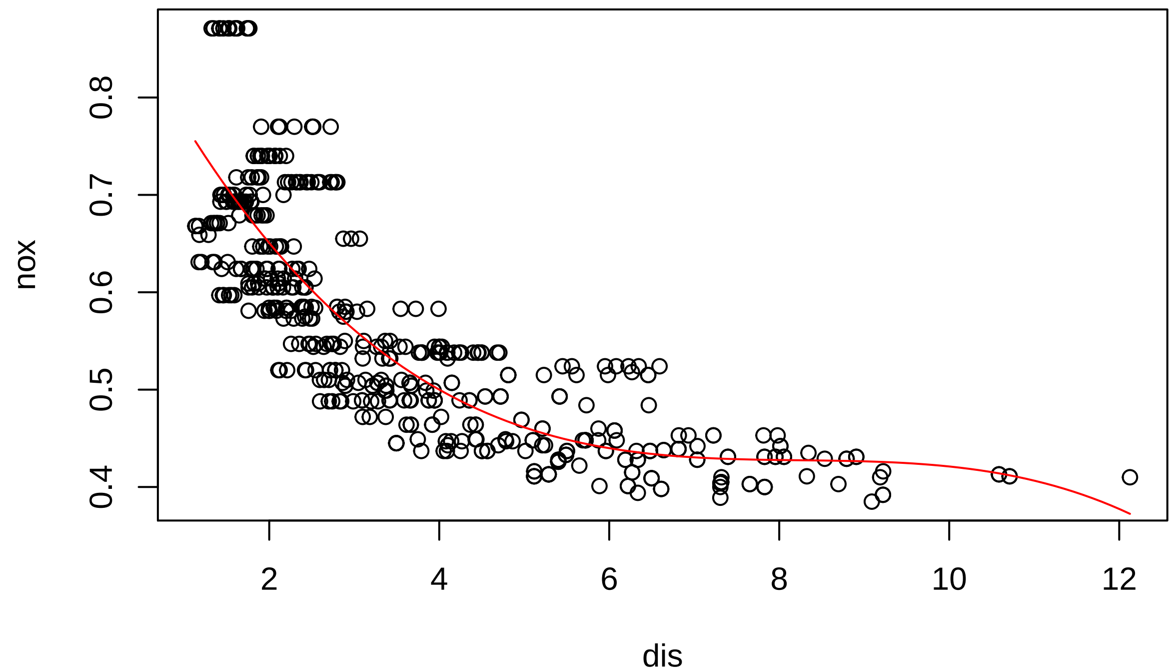
ISLR Applied Exercise 9:

9a)

```r
library(MASS)
attach(Boston)
poly.fit = glm(nox ~ poly(dis, 3), data = Boston)
summary(poly.fit)
```

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

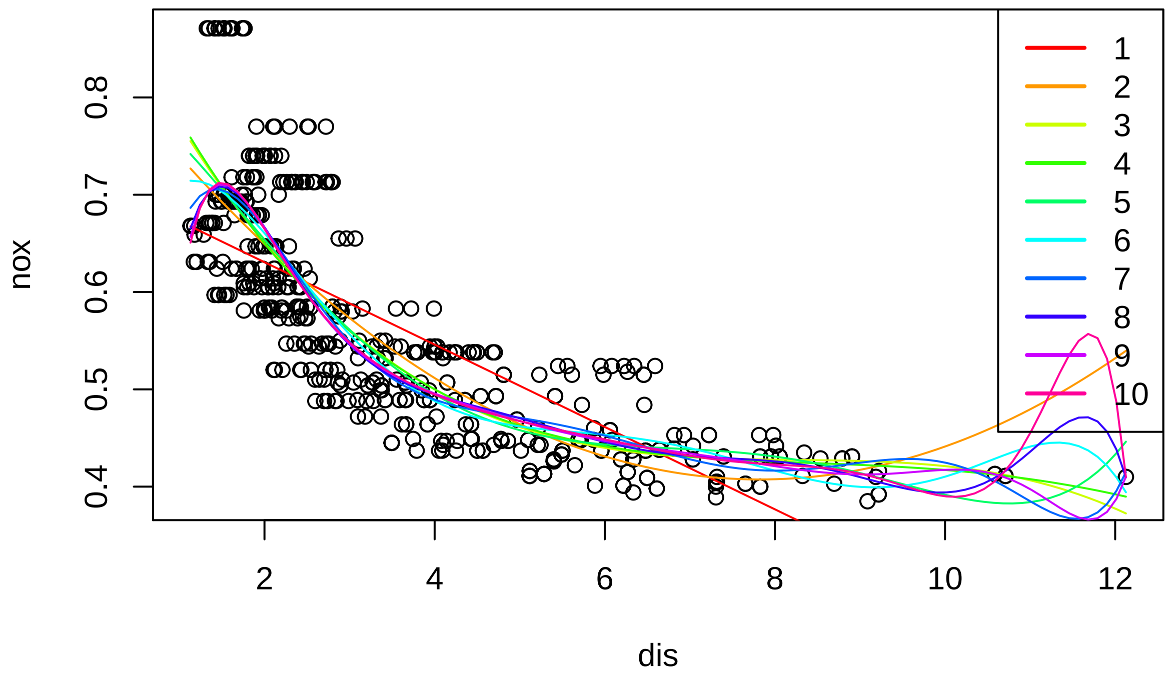
```r
plot(Boston[, c('dis', 'nox')])
pred = predict(poly.fit, data.frame(dis=seq(min(dis), max(dis), length.out = 100)))
lines(seq(min(dis), max(dis), length.out = 100), pred, col = "red")
```



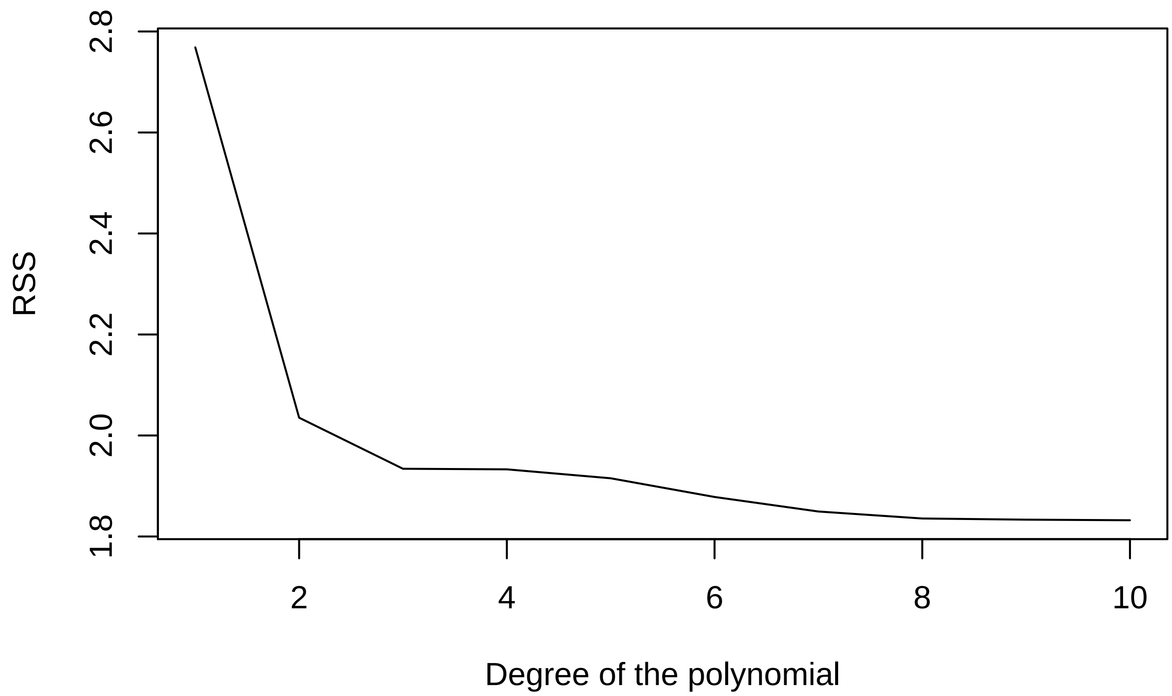
9b)

```
x = seq(min(dis), max(dis), length.out = 100)
cols = rainbow(10)
plot(Boston[, c('dis', 'nox')])
rss = c()
for(pwr in 1:10){
  poly.fit = glm(nox ~ poly(dis, pwr), data = Boston)
  pred = predict(poly.fit, data.frame(dis = x))
  lines(x, pred, col = cols[pwr])

  rss = c(rss, sum(poly.fit$residuals^2))
}
legend(x = 'topright', legend = 1:10, col = cols, lty = c(1, 1), lwd = c(2, 2))
```

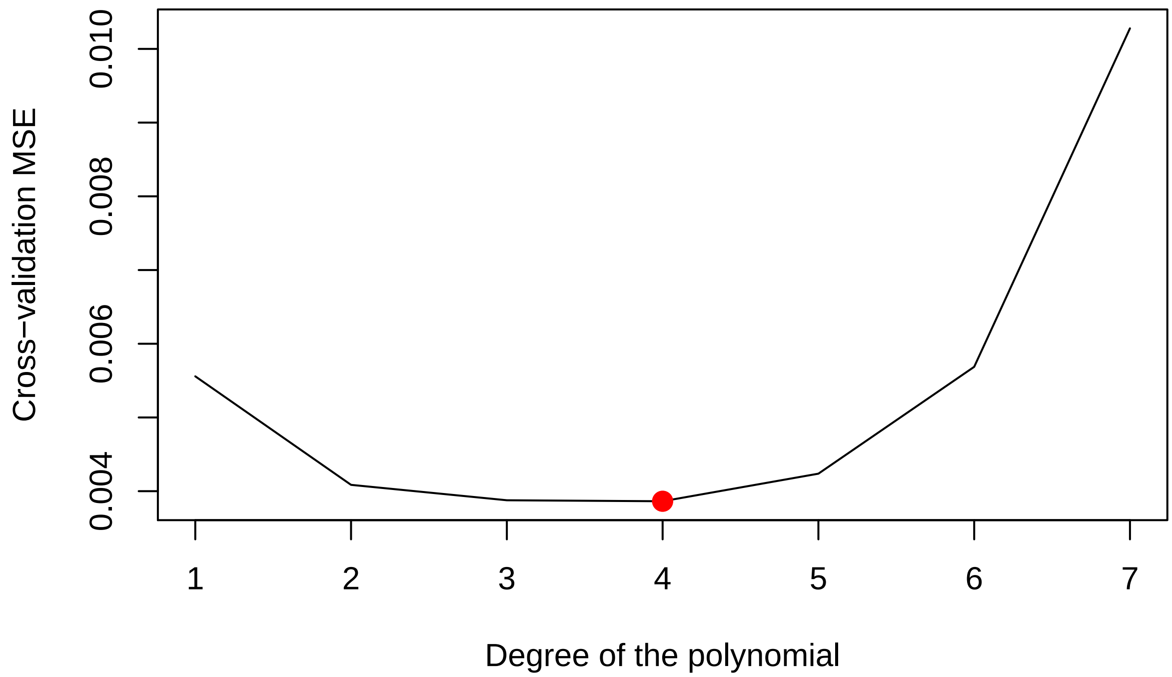



```r
plot(rss, xlab = "Degree of the polynomial", ylab = "RSS", type = "l")
```



9c)

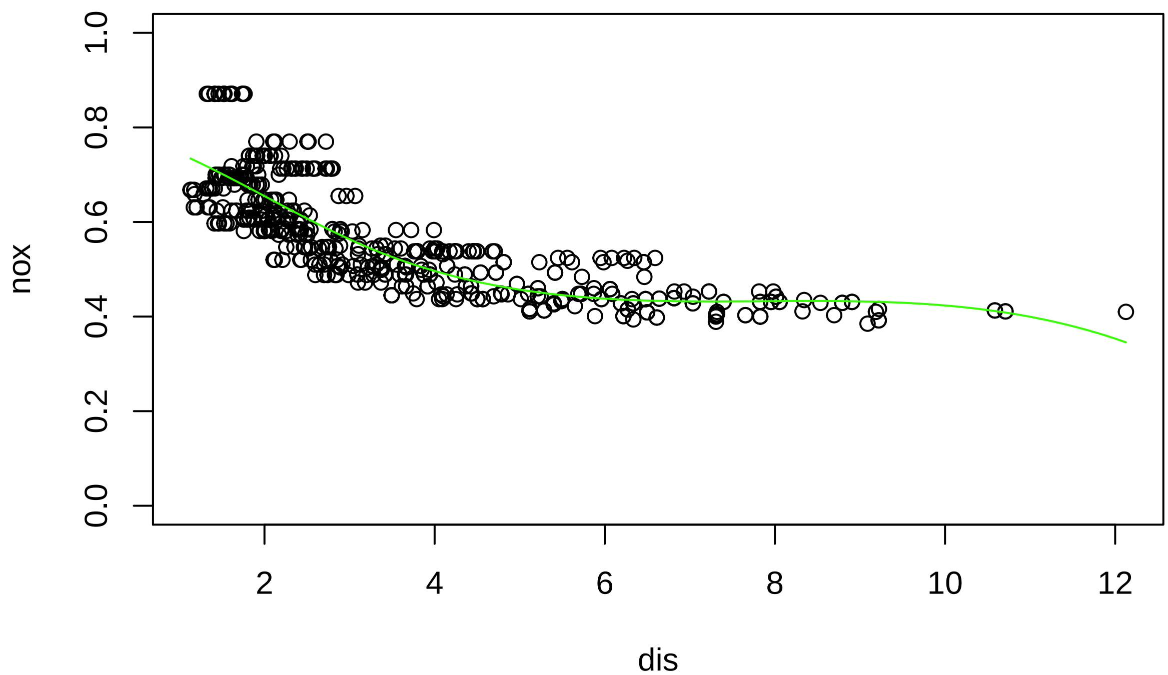
```
library(boot)
set.seed(1)
poly.mse = c()
for(degree in 1:7){
  poly.fit = glm(nox ~ poly(dis, degree, raw = T), data = Boston)
  mse = cv.glm(poly.fit, data = Boston, K = 10)$delta[1]
  poly.mse = c(poly.mse, mse)
}
plot(poly.mse, type = "l", xlab = "Degree of the polynomial", ylab = "Cross-validation MSE")
points(which.min(poly.mse), poly.mse[which.min(poly.mse)], col = "red", pch = 20, cex = 2)
```



Clearly, from the above graphic, the polynomial with the smallest MSE is the one with degree 4.

9d)

```
library(splines)
library(MASS)
spline.fit = lm(nox ~ bs(dis, df = 4), data = Boston)
x = seq(min(Boston[, "dis"]), max(Boston[, "dis"]), length.out = 100)
y = predict(spline.fit, data.frame(dis = x))
plot(Boston[, c("dis", "nox")], ylim = c(0, 1))
lines(x, y, col = cols[4])
```

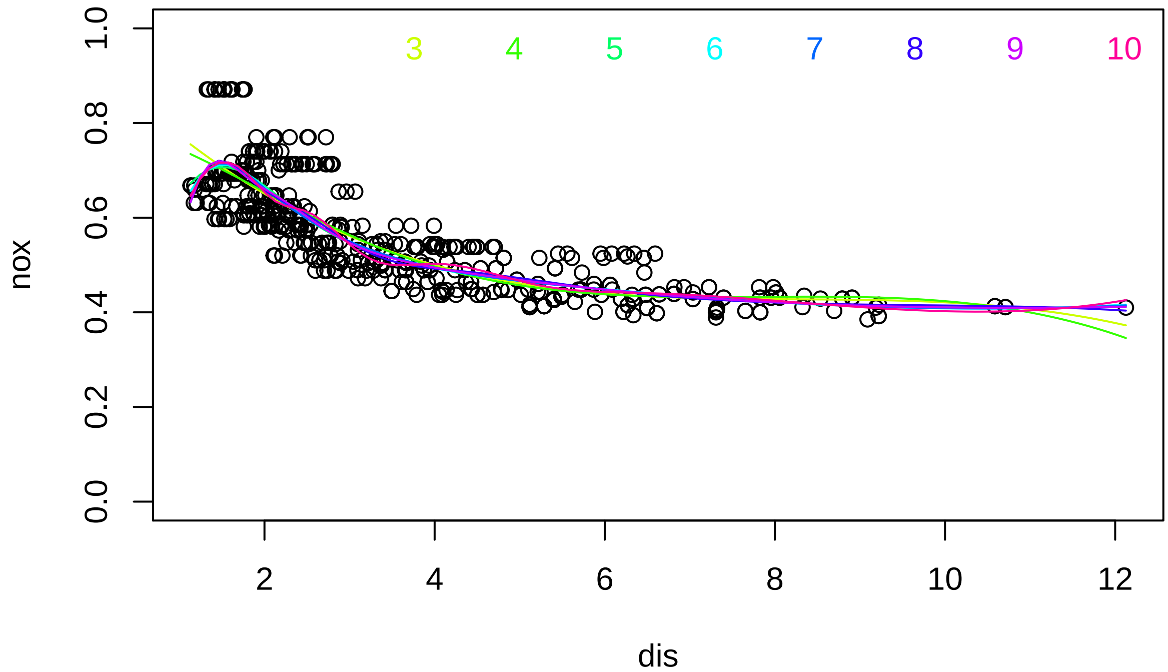


From the above plot, the polynomial with 4 degrees of freedom is shown above. I went ahead and used 100 knots just so the curve was as smooth as possible; using something like 5 knots makes it a lot jumpier/jagged. I figured 100 knots is a safe number to have in order to get the full power of the model while still making it look nice and ensuring it fits the data.

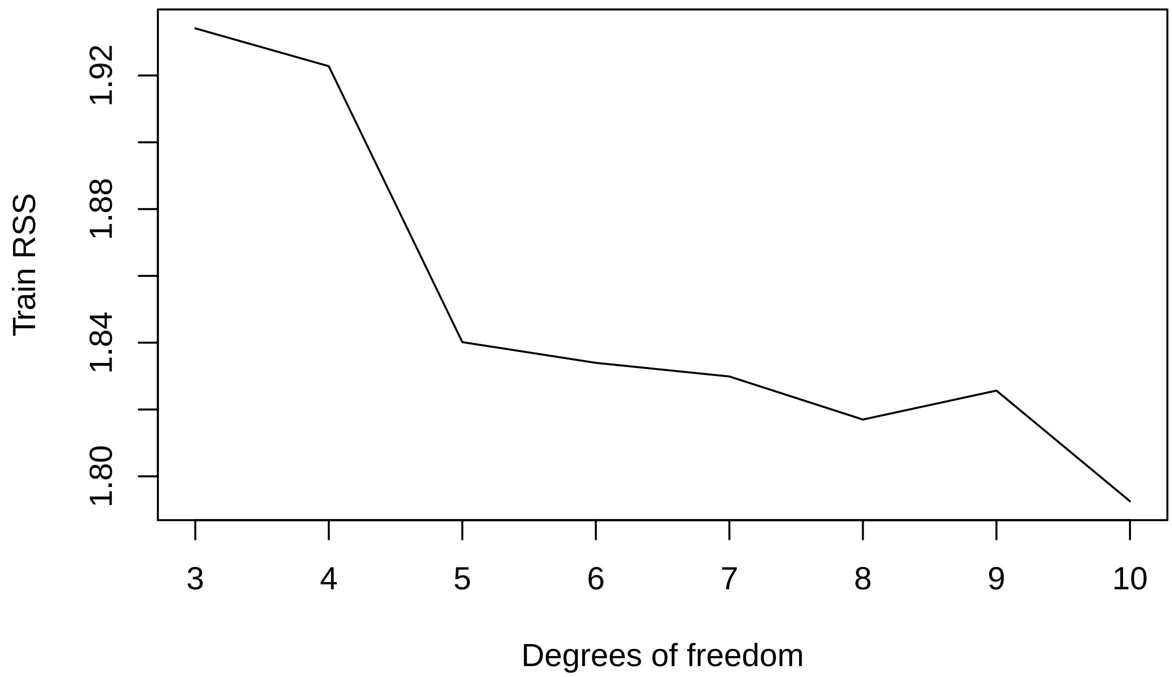
9e)

```
plot(Boston[, c("dis", "nox")], ylim = c(0, 1))
x = seq(min(Boston[, "dis"]), max(Boston[, "dis"]), length.out = 100)
rss = c()
for(df in 3:10){
  spline.fit = lm(nox ~ bs(dis, df = df), data = Boston)
  y = predict(spline.fit, data.frame(dis = x))
  lines(x, y, col = cols[df])

  rss = c(rss, sum(spline.fit$residuals^2))
}
legend(x = "topright", legend = 3:10, text.col = cols[3:10], text.width = 0.5, bty = "n", horiz = T)
```




```
plot(3:10, rss, xlab = "Degrees of freedom", ylab = "Train RSS", type = "l")
```

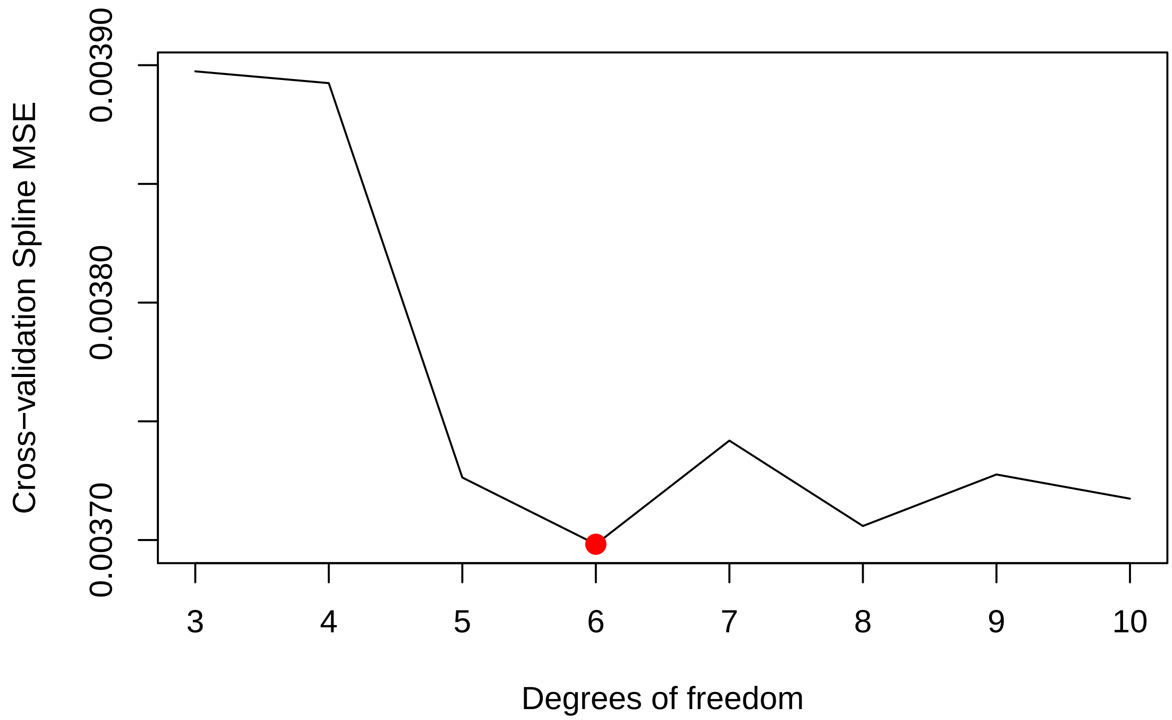


The model with the lowest training RSS is the one with 10 degrees of freedom. However, it's important to note in the above graph the scale on the y-axis does not have a large range. As such, all models are very close to being the same; the one with 10 df is only marginally better.

9f)

```r
library(boot)
set.seed(1)
spline.mse = c()
for(df in 3:10){
  Boston.model = model.frame(nox ~ bs(dis, df = df), data = Boston)
  names(Boston.model) = c("nox", "bs.dis")

  spline.fit = glm(nox ~ bs.dis, data = Boston.model)
  mse = cv.glm(spline.fit, data = Boston.model, K = 10)$delta[1]
  spline.mse = c(spline.mse, mse)
}
plot(3:10, spline.mse, type = "l", xlab = "Degrees of freedom", ylab = "Cross-validation Spline MSE")
x = which.min(spline.mse)
points(x + 2, spline.mse[x], col = "red", pch = 20, cex = 2)
```



Clearly, from the above plot, the model with 6 degrees of freedom has the smallest MSE by a very small margin compared to the other models.

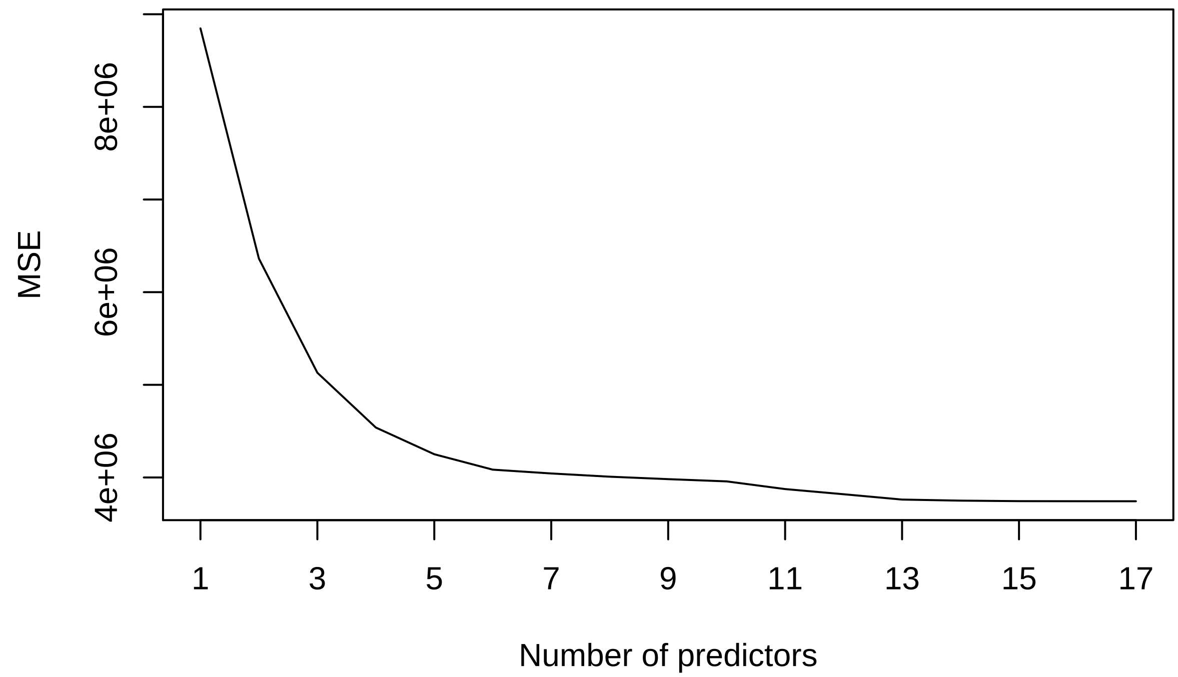
ISLR Applied Exercise 10:

10a)

```
library(ISLR)
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.6.3
```

```
set.seed(1)
train = sample(1:nrow(College), 500)
test = -train
forward = regsubsets(Outstate ~ ., data = College, method = "forward", nvmax = 17)
plot(1 / nrow(College) * summary(forward)$rss, type = "l", xlab = "Number of predictors", ylab = "MSE",
axis(side = 1, at = seq(1, 17, 2), labels = seq(1, 17, 2))
```




```r
which(summary(forward)$which[7, -1])
```

```
##  PrivateYes  Room.Board    Personal         PhD perc.alumni      Expend
##           1           9          11          12          15          16
##   Grad.Rate
##          17
```

10b)

```r
library(gam)
```

```
## Warning: package 'gam' was built under R version 3.6.3
```

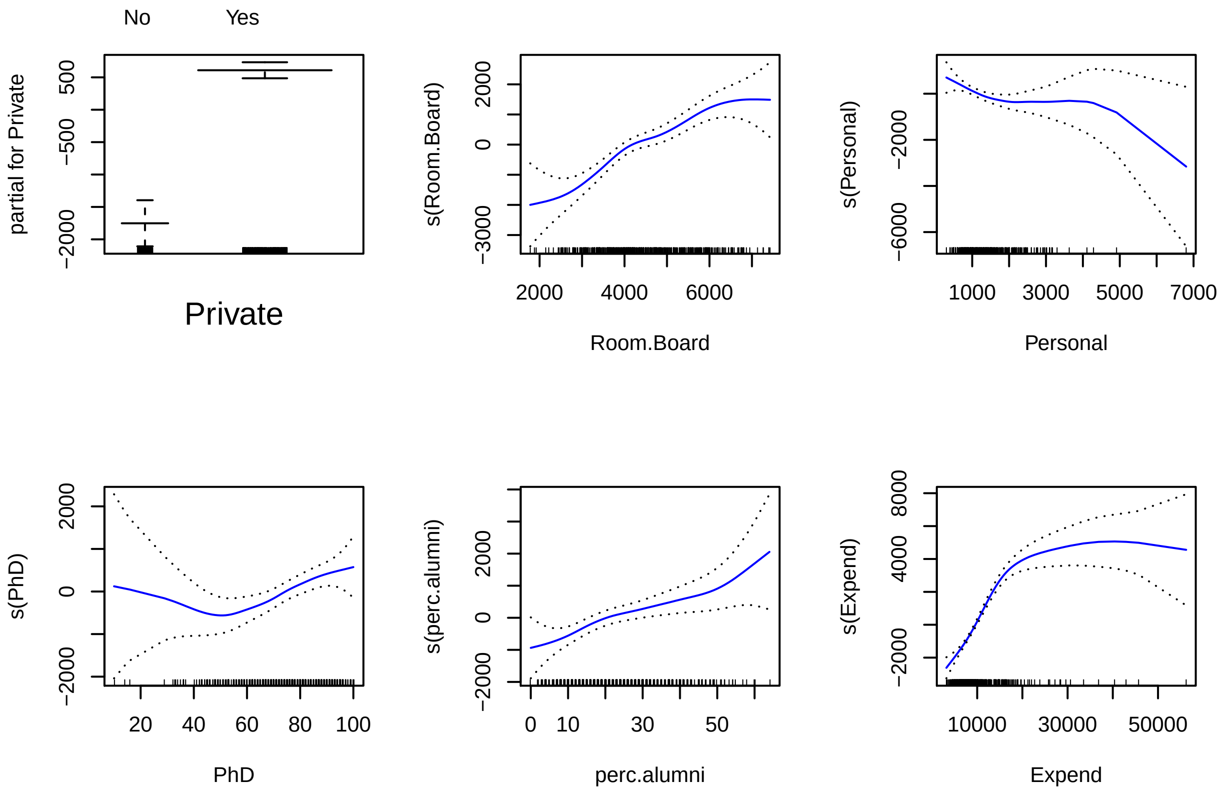
```
## Loading required package: foreach
```

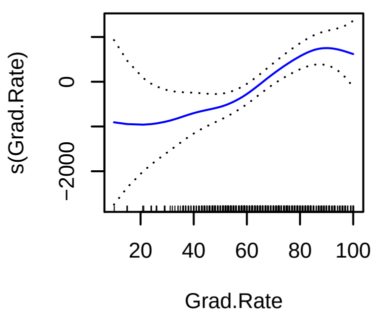
```
## Loaded gam 1.16.1
```

```r
gam.fit = gam(Outstate ~ Private + s(Room.Board) + s(Personal) + s(PhD) + s(perc.alumni) + s(Expend) +
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts argument
## ignored
```

```r
par(mfrow = c(2, 3))
plot(gam.fit, se = T, col = "blue")
```





Unfortunately, `Private` is a qualitative predictor, so we couldn't fit a smooth spline onto it, but we did so with the other 6 predictors which were quantitative. It doesn't look like any of the splines are too complex, as in they don't look like they overfit. All of the models look different from each other, which is a good indicator that they all have different relationships with out-of-state tuition, reducing concerns with collinearity.

10c)

```
gam.pred = predict(gam.fit, College[test, ])
gam.mse = mean((College[test, "Outstate"] - gam.pred)^2)
gam.mse
```

```
## [1] 3300729
```

```
gam.tss = mean((College[test, "Outstate"] - mean(College[test, "Outstate"]))^2)
test.rss = 1 - gam.mse / gam.tss
test.rss
```

```
## [1] 0.7598627
```

We can see the test MSE is lower than the training MSE, which shows the model performs better on the test set. As such, we don't have any concerns of overfitting. In addition, the correlation is about 0.8, so the model explains a good amount of variance in out-of-state tuition.

10d)

```r
summary(gam.fit)
```

```
##
## Call: gam(formula = Outstate ~ Private + s(Room.Board) + s(Personal) +
##     s(PhD) + s(perc.alumni) + s(Expend) + s(Grad.Rate), data = College[train,
##     ])
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -7288.57 -1159.85    38.48  1293.29  7166.37
##
## (Dispersion Parameter for gaussian family taken to be 3537534)
##
##     Null Deviance: 8686699532 on 499 degrees of freedom
## Residual Deviance: 1676789008 on 473.9995 degrees of freedom
## AIC: 8985.709
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                 Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private          1 2220985265 2220985265 627.834 < 2.2e-16 ***
## s(Room.Board)    1 1810874632 1810874632 511.903 < 2.2e-16 ***
## s(Personal)      1   88823255   88823255  25.109 7.663e-07 ***
## s(PhD)           1  527206393  527206393 149.032 < 2.2e-16 ***
## s(perc.alumni)   1  352602187  352602187  99.675 < 2.2e-16 ***
## s(Expend)        1  723775260  723775260 204.599 < 2.2e-16 ***
## s(Grad.Rate)     1   75294938   75294938  21.285 5.102e-06 ***
## Residuals      474 1676789008    3537534
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                Npar Df  Npar F    Pr(F)
## (Intercept)
## Private
## s(Room.Board)        3  3.0225  0.02941 *
## s(Personal)          3  1.5591  0.19852
## s(PhD)               3  1.5090  0.21138
## s(perc.alumni)       3  0.8017  0.49336
## s(Expend)            3 27.5082 2.22e-16 ***
## s(Grad.Rate)         3  1.6580  0.17524
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the above output, all the predictors are statistically significant, so we will assume all the predictors have a non-linear relationship with out-of-state tuition.