StatR 201: Winter 2013

Homework 03

Rod Doe

Wednesday, February 13, 2013

Logistic Regression

Here is the code that I used to perform the analysis:

```
setwd("C:/Users/Rod/SkyDrive/R/201/Week05")
#challenger = read.csv("challenger.csv")
heart = read.csv("HosmerLemeshowHeart.csv")

str(heart)
table(heart$Age)
heart$AgeBin = cut(heart$Age, breaks = quantile(heart$Age, names = FALSE))
table(heart$AgeBin)
table(heart$Disease)
```

I initially did the fit of Disease versus Age:

```
fit = glm(Disease ~ Age, data = heart, family = binomial)
summary.glm(fit)
```

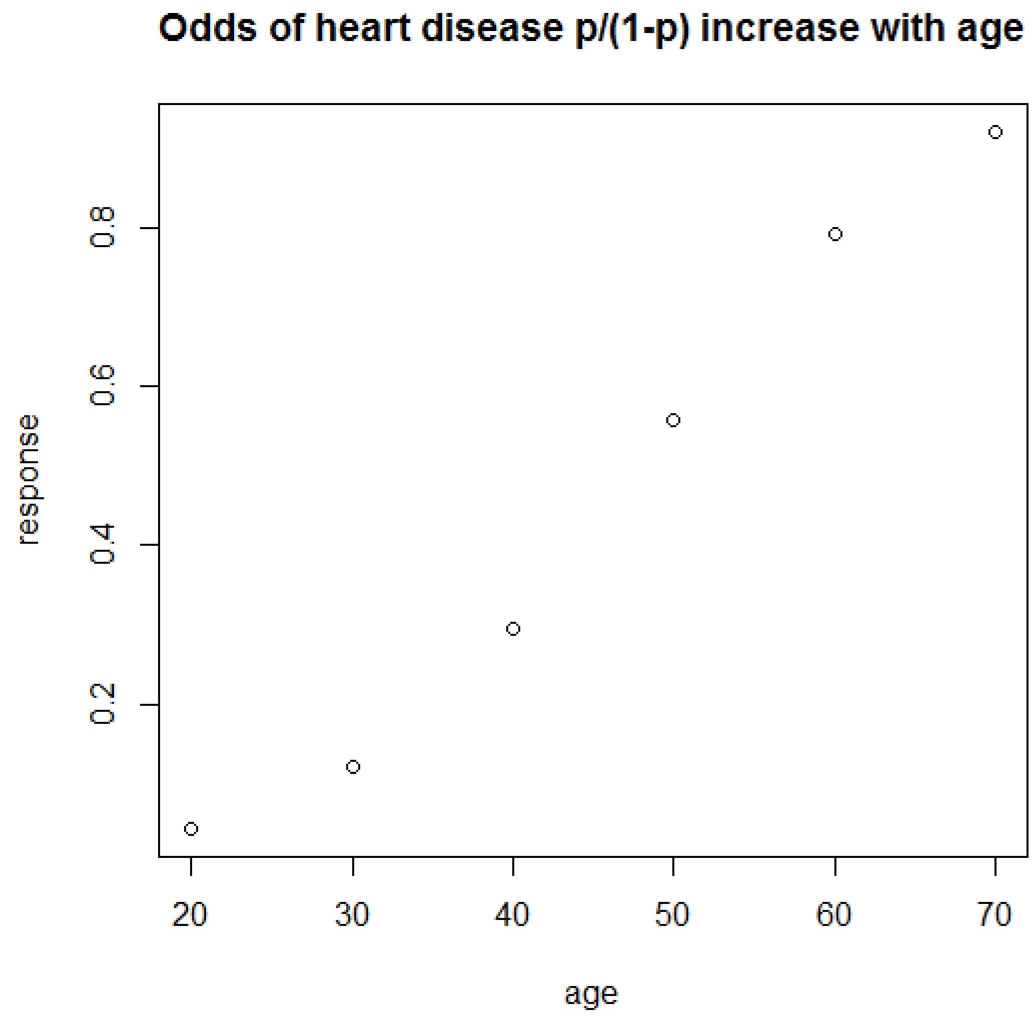
Here is the summary:

```
Call:
glm(formula = Disease ~ Age, family = binomial, data = heart)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9718  -0.8456  -0.4576   0.8253   2.2859

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.30945    1.13365  -4.683 2.82e-06 ***
Age          0.11092    0.02406   4.610 4.02e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 136.66  on 99  degrees of freedom
Residual deviance: 107.35  on 98  degrees of freedom
AIC: 111.35

Number of Fisher Scoring iterations: 4
```

Given only one covariate, that wasn't particularly interesting. I decided to test the predictive capability of the model:

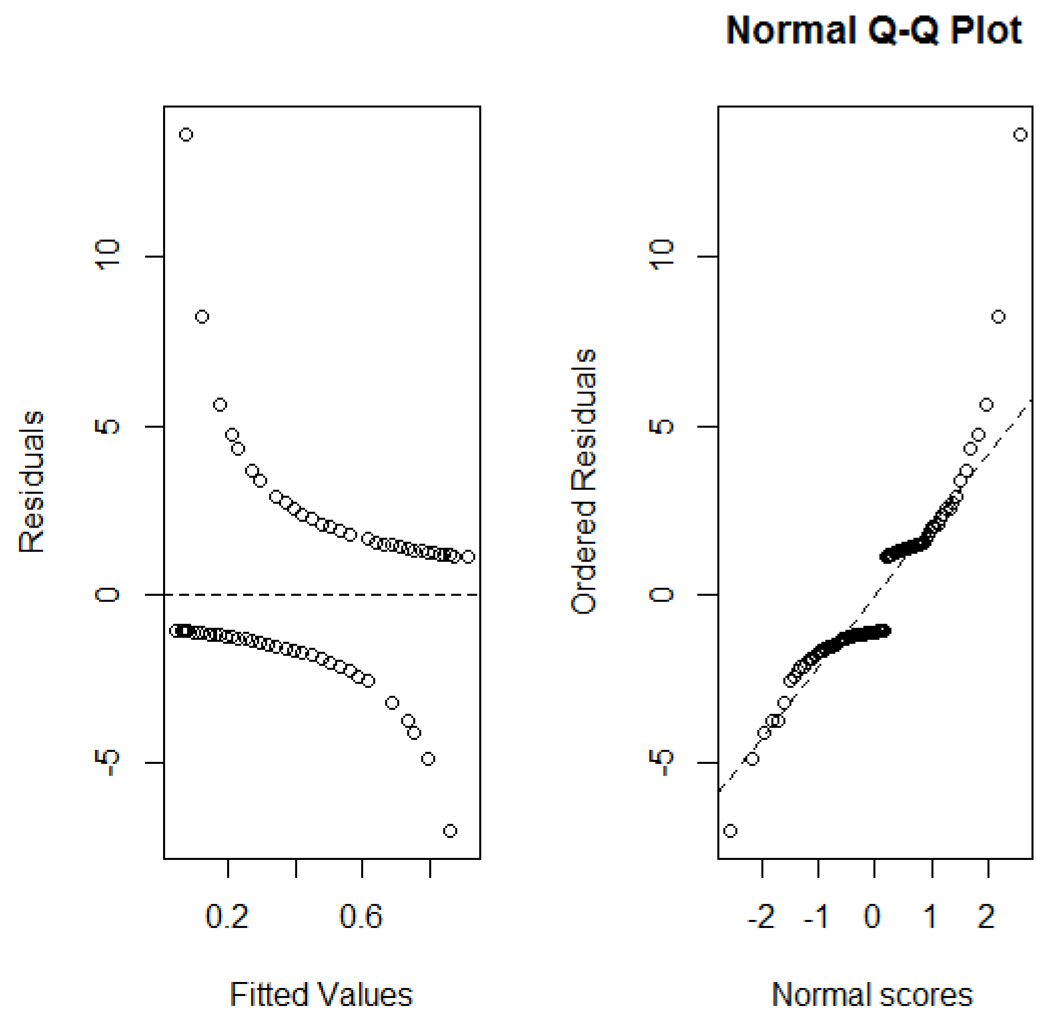
```
age = c(20, 30, 40, 50, 60, 70)
response=predict(fit, list(Age=age), type="response")
plot(age, response, main="Odds of heart disease p/(1-p) increase with age")
```



The non-linearity of that bothered me. I found this function that can be used to check models in a standard manner:

```
modelCheck <- function( model) {
        rs <- model$resid
        fv <- model$fitted
        par(mfrow = c(1, 2))
        plot( fv, rs, xlab="Fitted Values", ylab="Residuals")
        abline(h=0, lty=2)
        qqnorm(rs, xlab="Normal scores", ylab="Ordered Residuals")
        qqline(rs, lty=2)
        par(mfrow=c(1,1))
}
```

This fit did not look so good in the graphs produced by the modelCheck function. The non-normal residual suggested trouble in the model.



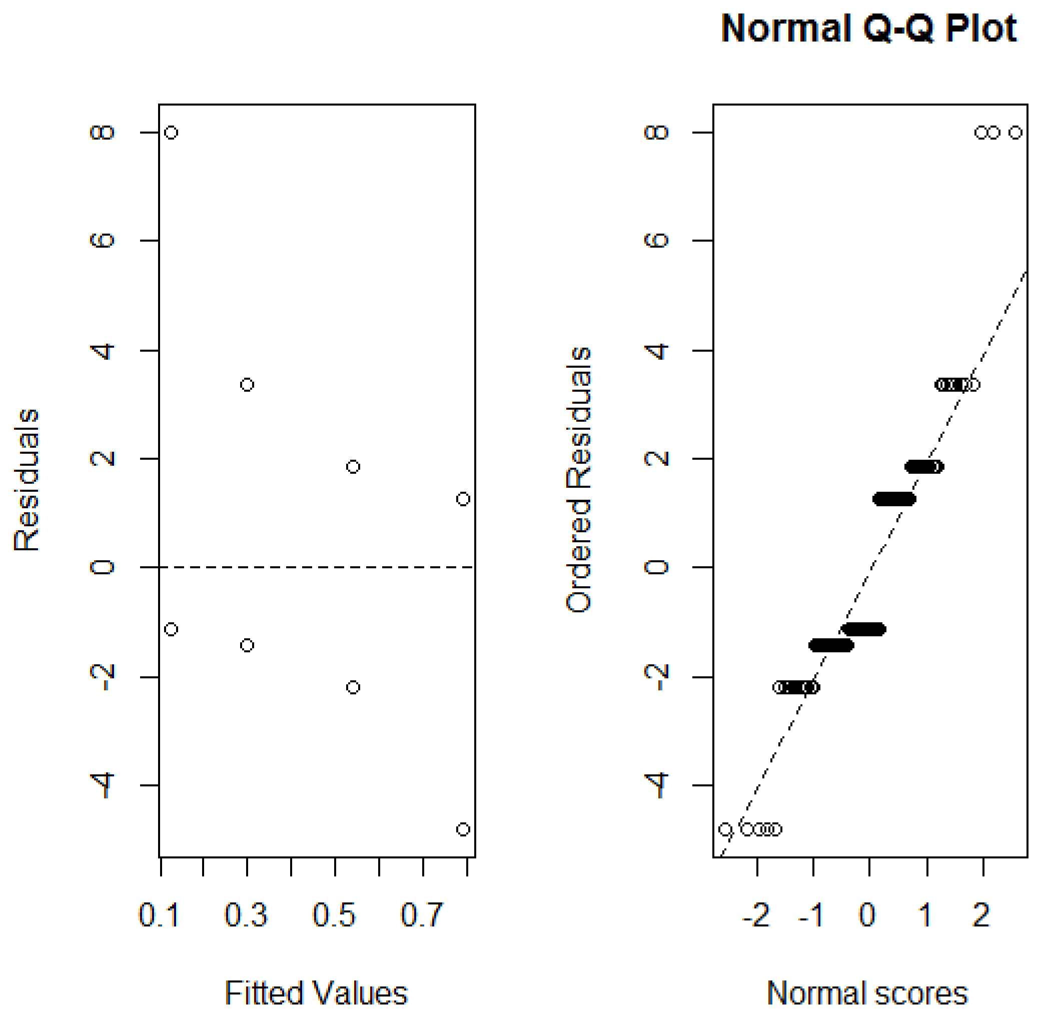
I then looked at the model check that was produced by fitting heart disease against AgeBin:
fit = glm(Disease ~ AgeBin, data = heart, family = binomial)
summary.glm(fit)

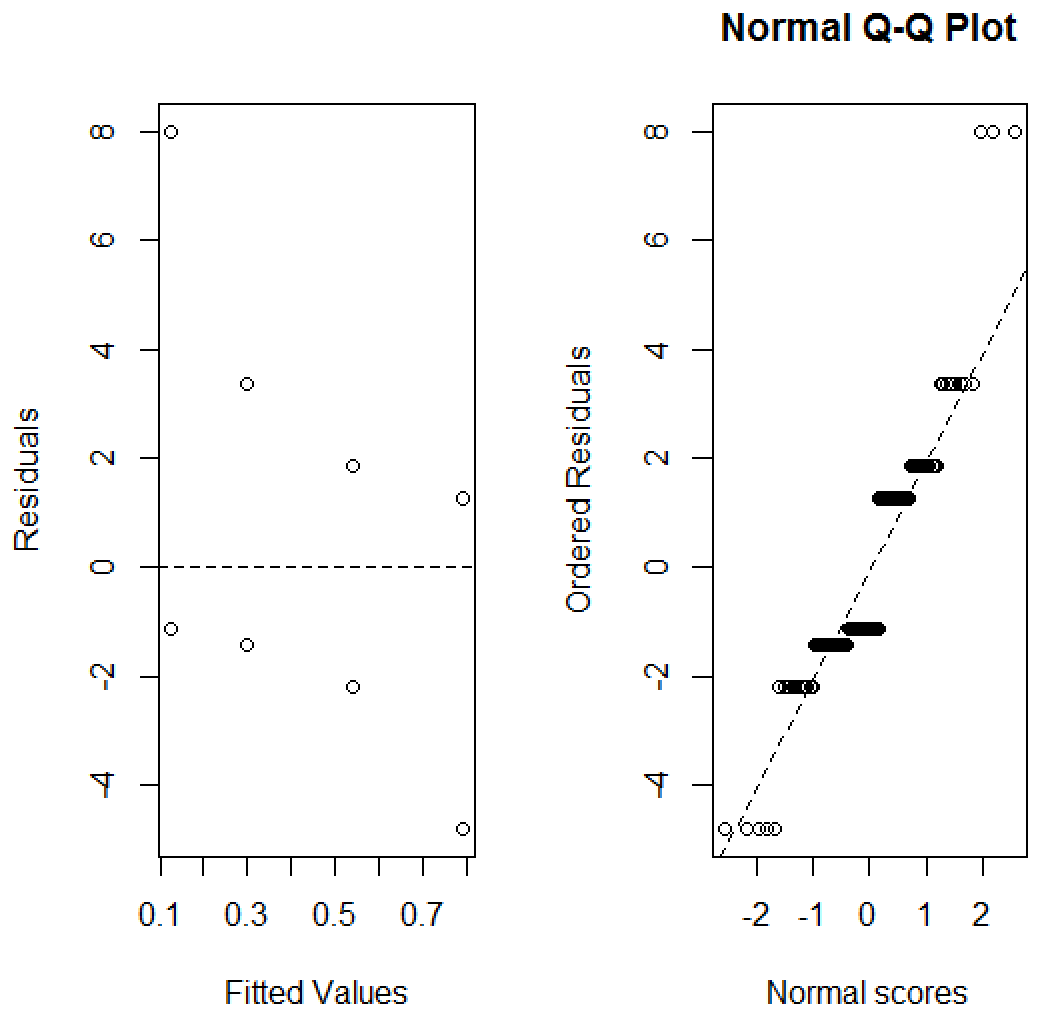
Here is the summary:

```
glm(formula = Disease ~ AgeBin, family = binomial, data = heart)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7712  -0.8383  -0.5168   0.6835   2.0393

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.9459     0.6172  -3.153  0.00162 **
```

```
AgeBin(34.8,44]    1.0809     0.7474   1.446  0.14810
AgeBin(44,55]      2.1130     0.7408   2.852  0.00434 **
AgeBin(55,69]      3.2809     0.7960   4.122 3.76e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 135.53  on 98  degrees of freedom
Residual deviance: 108.57  on 95  degrees of freedom
  (1 observation deleted due to missingness)
AIC: 116.57

Number of Fisher Scoring iterations: 4
```

The qqplot produced by modelCheck looked better here:





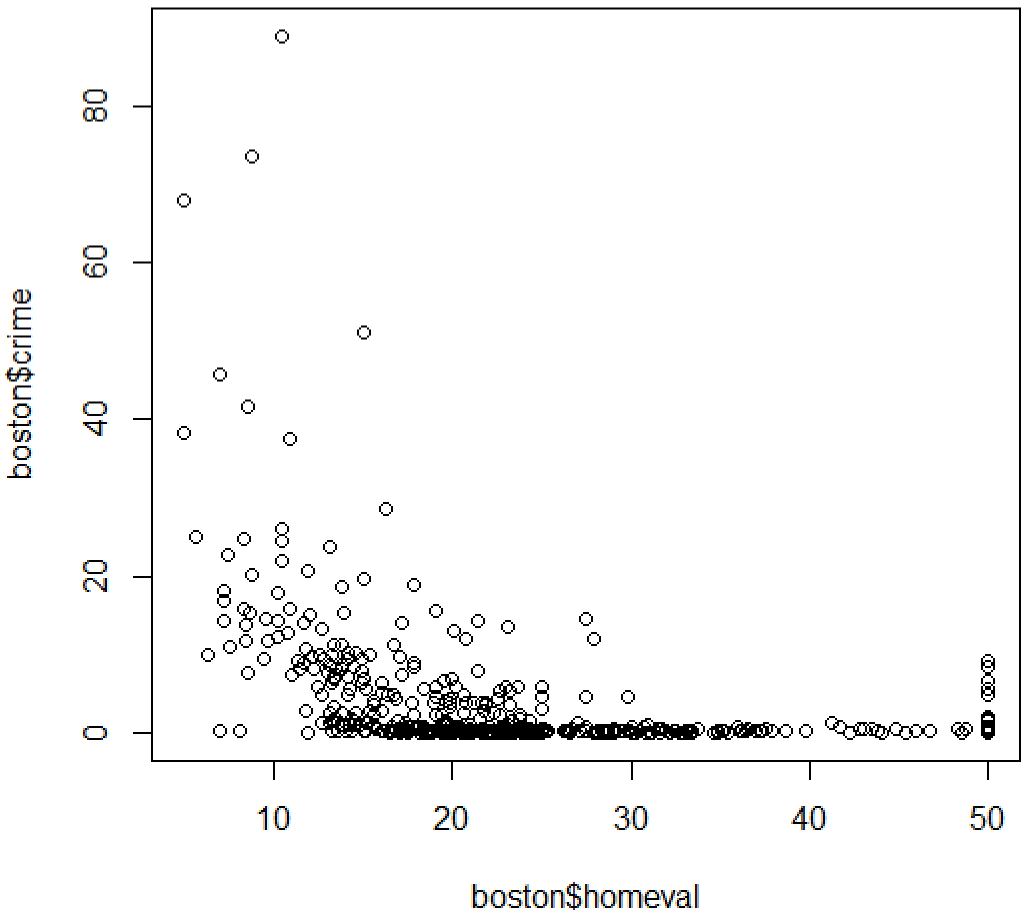
The Q-Q plot was a bit more linear when Disease was plotted against AgeBin.

Adventures with Splines

I plotted the "unturned stones" in the set of covariates to see what would be a good candidate for a spline treatment.

```
m0 = lm(homeval ~ lowSES + rooms, data=boston)
plot(boston$homeval, boston$crime)
plot(boston$homeval, boston$biglots)
plot(boston$homeval, boston$river)
plot(boston$homeval, boston$nox) # interesting
plot(boston$homeval, boston$old)
plot(boston$homeval, boston$jobdist) # interesting, but done in class
plot(boston$homeval, boston$tax)
plot(boston$homeval, boston$teachratio)
plot(boston$homeval, boston$crime) #interesting
```

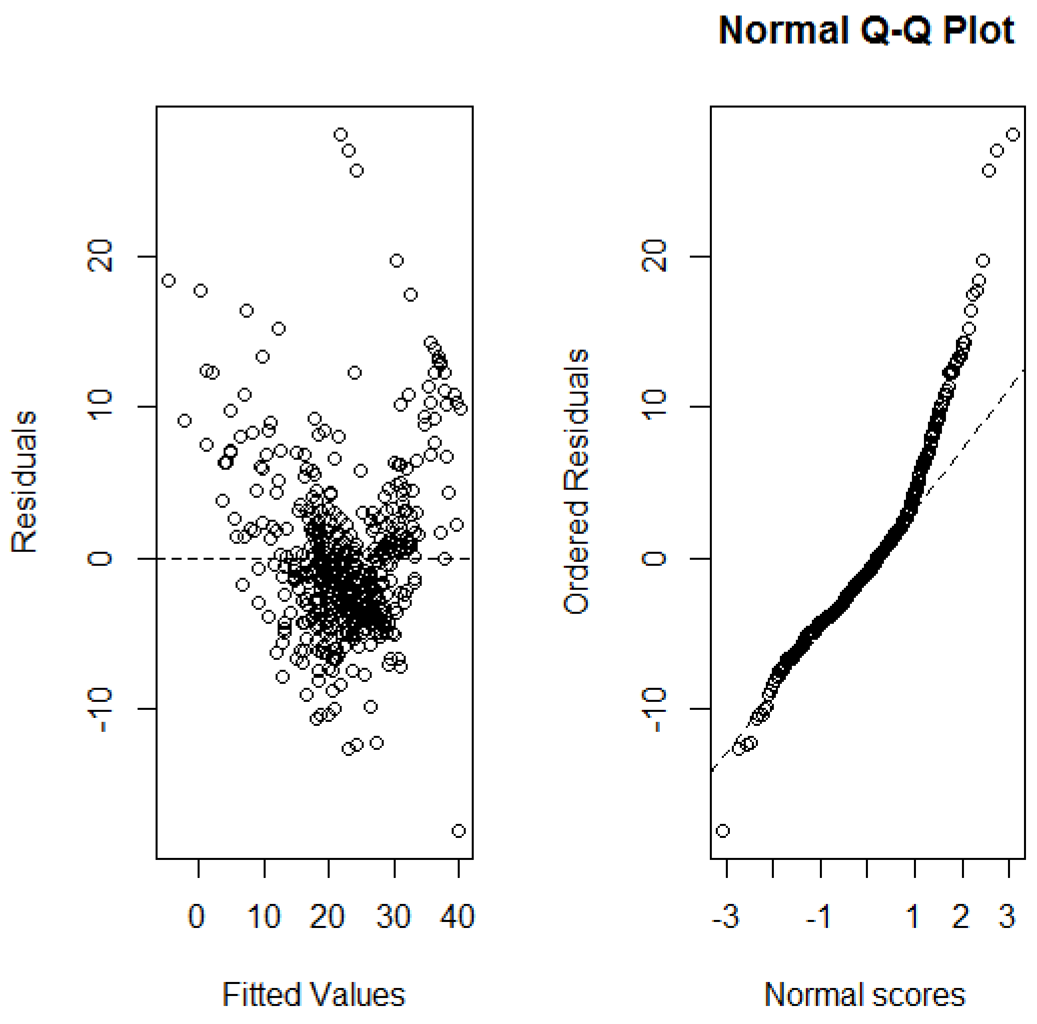
Of these, crime looked most like something that could be fit with a spline. There was a lot of variation at for low home values, and flattened out at the upper end (gated communities, no doubt).



Since the crime values became linear at high home values, it made sense to use a natural cubic spline using the ns function.

Let’s get a baseline model perspective with my recycled modelCheck function:

```
m0 = lm(homeval ~ lowSES + rooms, data=boston)
modelCheck(m0)
```



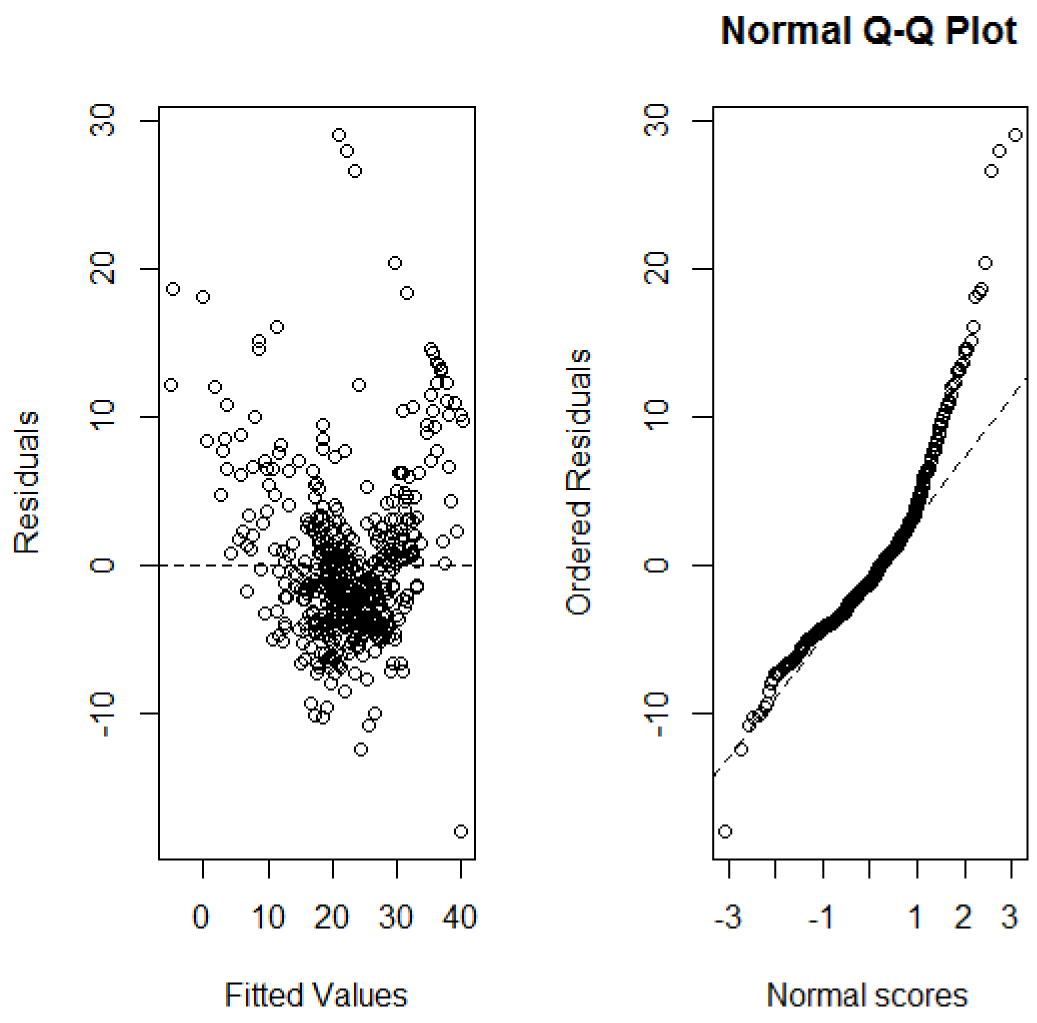
Let’s add crime to the m0 model:

```
m1 = lm(homeval ~ lowSES + rooms + crime, data=boston)
summary.lm(m1) #p-value crime = 0.001

Call:
lm(formula = homeval ~ lowSES + rooms + crime, data = boston)

Residuals:
    Min      1Q  Median      3Q     Max 
-17.925  -3.566  -1.157   1.906  29.024 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
```

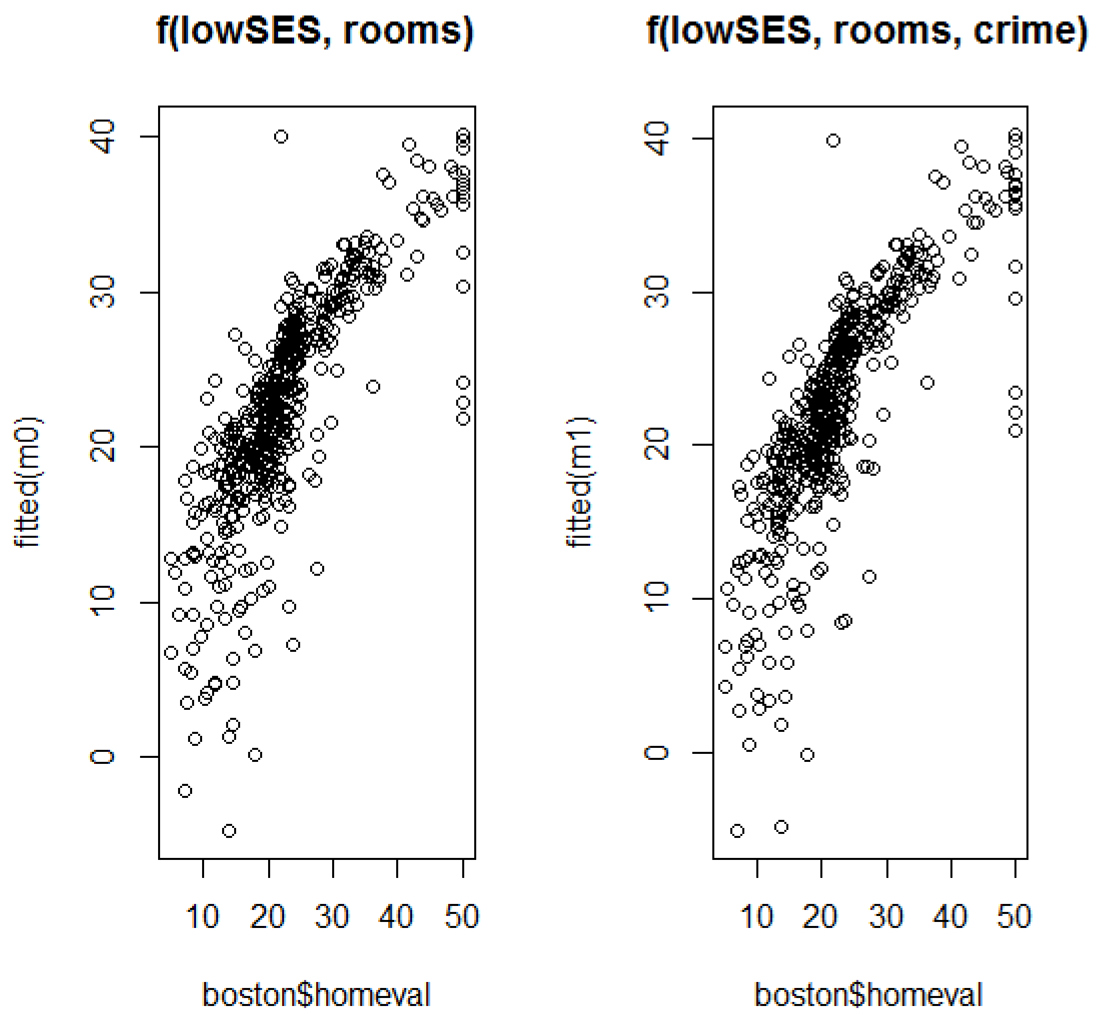
```
(Intercept) -2.56225    3.16602  -0.809  0.41873
lowSES      -0.57849    0.04767 -12.135  < 2e-16 ***
rooms        5.21695    0.44203  11.802  < 2e-16 ***
crime       -0.10294    0.03202  -3.215  0.00139 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.49 on 502 degrees of freedom
Multiple R-squared: 0.6459,     Adjusted R-squared: 0.6437
F-statistic: 305.2 on 3 and 502 DF,  p-value: < 2.2e-16
```

Do a model check of the new model:

```
modelCheck(m1)
```



Let's visually compare the results of the two models:



Visually, addition of crime seems to have little effect.

Let's add natural splines of crime of increasing degrees of freedom to the model.

```
library(splines)
m2  = lm(homeval ~ lowSES + rooms + ns(crime, df=2), data=boston)
m3  = lm(homeval ~ lowSES + rooms + ns(crime, df=3), data=boston)
m4  = lm(homeval ~ lowSES + rooms + ns(crime, df=4), data=boston)
m5  = lm(homeval ~ lowSES + rooms + ns(crime, df=5), data=boston)
m6  = lm(homeval ~ lowSES + rooms + ns(crime, df=6), data=boston)
m7  = lm(homeval ~ lowSES + rooms + ns(crime, df=7), data=boston)
```

Calculate the likelihood ratio for each model:

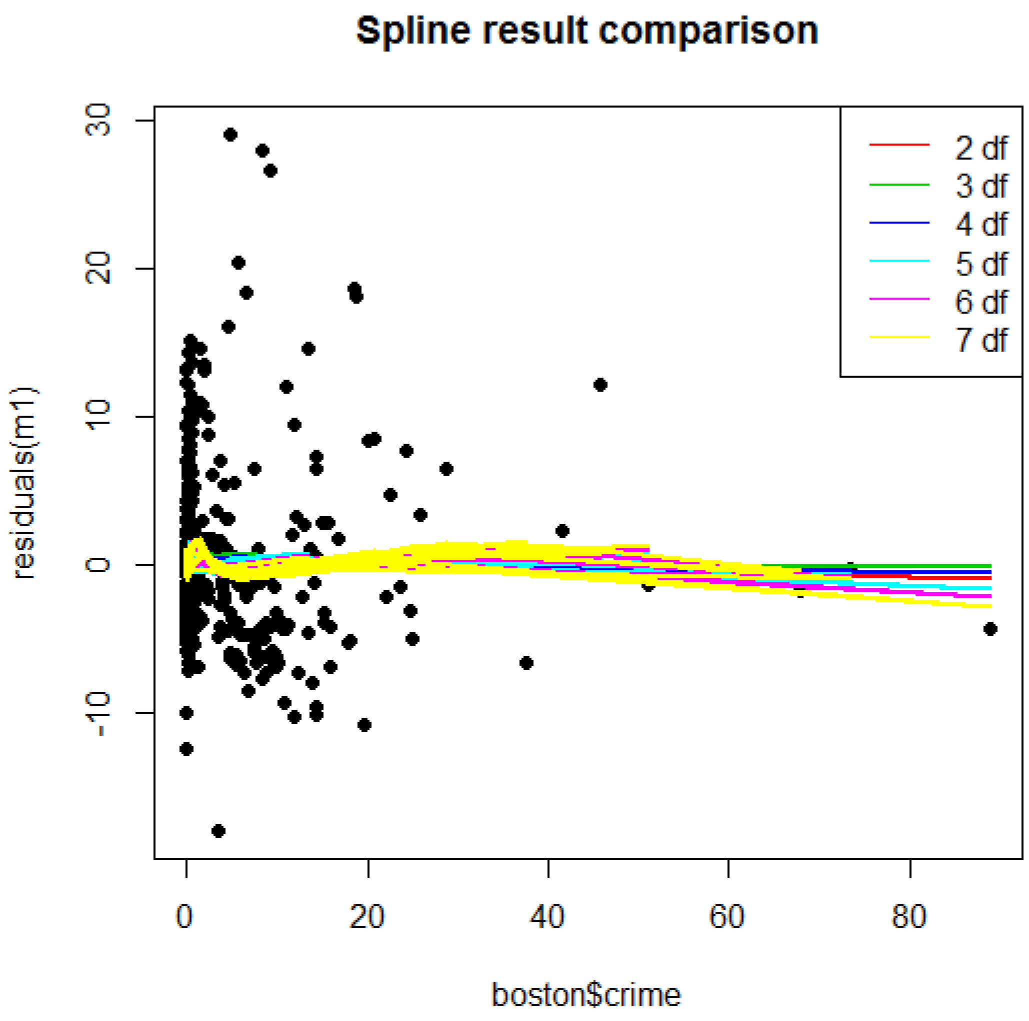
```
library(lmtest)
lrtest(m0, m1, m2, m3, m4, m5, m6, m7)
Likelihood ratio test
```

```
Model 1: homeval ~ lowSES + rooms
Model 2: homeval ~ lowSES + rooms + crime
Model 3: homeval ~ lowSES + rooms + ns(crime, df = 2)
Model 4: homeval ~ lowSES + rooms + ns(crime, df = 3)
Model 5: homeval ~ lowSES + rooms + ns(crime, df = 4)
Model 6: homeval ~ lowSES + rooms + ns(crime, df = 5)
Model 7: homeval ~ lowSES + rooms + ns(crime, df = 6)
Model 8: homeval ~ lowSES + rooms + ns(crime, df = 7)
  #Df  LogLik Df   Chisq Pr(>Chisq)
1   4 -1582.8
2   5 -1577.6  1 10.3107   0.001323 **
3   6 -1577.5  1  0.1280   0.720551
4   7 -1576.5  1  2.1669   0.141008
5   8 -1576.9  1  0.9084   0.340530
6   9 -1574.2  1  5.4594   0.019463 *
7  10 -1573.9  1  0.5445   0.460591
8  11 -1572.7  1  2.5102   0.113113
```

From this, it appears that generation of a natural spline for the crime covariate did not improve the model.

Here is the spline through the residuals:

```
plot(boston$crime, residuals(m1), pch=19, main="Spline result
comparison")
lines(boston$crime, fitted(m2) - fitted(m1), col=2, lwd=2)
lines(boston$crime, fitted(m3) - fitted(m1), col=3, lwd=2)
lines(boston$crime, fitted(m4) - fitted(m1), col=4, lwd=2)
lines(boston$crime, fitted(m5) - fitted(m1), col=5, lwd=2)
lines(boston$crime, fitted(m6) - fitted(m1), col=6, lwd=2)
lines(boston$crime, fitted(m7) - fitted(m1), col=7, lwd=2)
legend("topright", legend=paste(c(2:7), "df"), lty=1, col=2:7)
```


Spline result comparison
residuals(m1)
boston$crime
30
20
10
0
-10
0
20
40
60
80
2 df
3 df
4 df
5 df
6 df
7 df