Lab 5: Linear Regression Diagnostics

Hao Wang

February 13, 2017

1. Load data and essential packages

```
#install.packages(c('lmtest', 'car', 'faraway', 'MASS', 'ggplot2','grid','gridExtra'))
library(lmtest)
library(car)
library(faraway)
library(MASS)
library(ggplot2)
mydata <- Prestige
names(mydata)
```

```
## [1] "education" "income"    "women"     "prestige"  "census"    "type"
```

```
set.seed(99) #set seed of random sample
n <-nrow(mydata)
n1 <- floor(n/1.5) #train
n2 <- n -n1        #test

train=sample(n, n1, replace = F) #create a random sample
Train=data.frame(mydata[train,], row.names=NULL)#select data.frame of Train by row
Test =data.frame(mydata[-train,], row.names =NULL  )#select data.drame of Test by row
```

2. Matrix Notation

$$\hat{Y} = X\beta$$

Hat Matrix

$$\hat{Y} = X\beta$$
$$\hat{Y} = X(X'X)^{-1}X'Y$$
$$\hat{Y} = HY$$
$$H = X(X'X)^{-1}X'$$

H is the hat matrix, which turns Y into $\hat{Y}$. Hat matrix measures high leveage points. I is the identity matrix with all elements as 1.

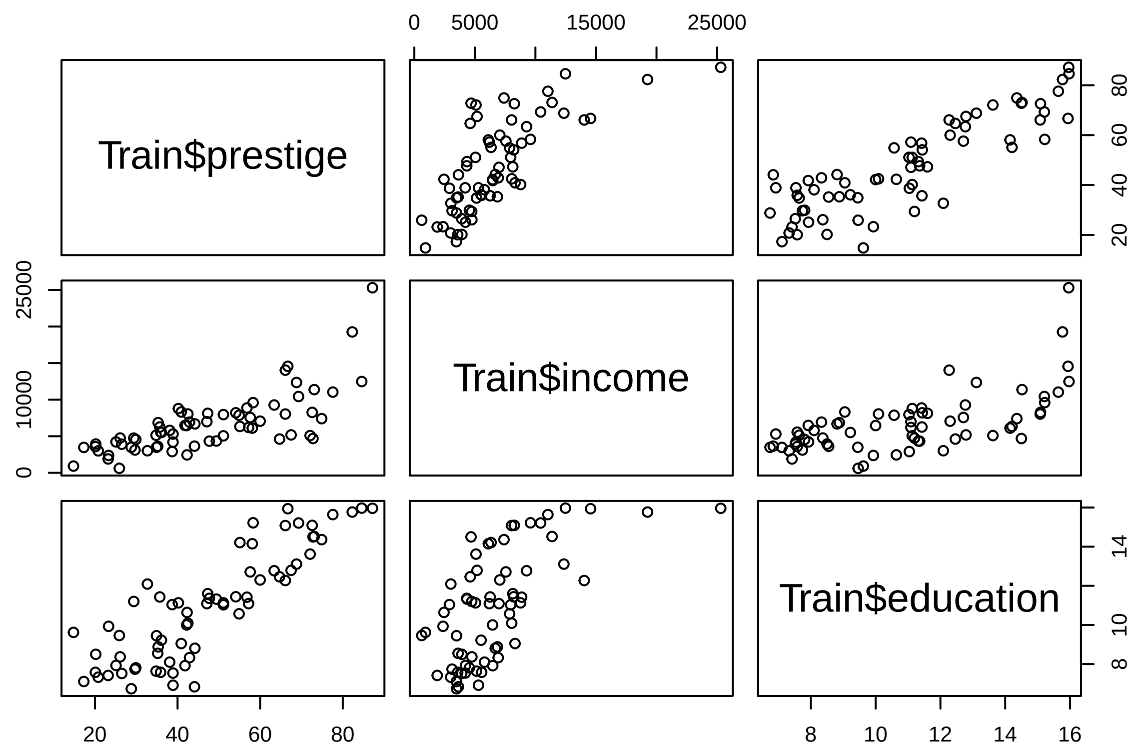
- *H is symmetric: $H = H'$ and $(I - H)' = (I - H)$
- *H is idempotent: $H^2 = H$ and $(I - H)(I - H) = (I - H)$

Residuals

$$e = Y - \hat{Y} = Y - HY = (I - H)Y$$

3. Check one-to-one bivariate relations

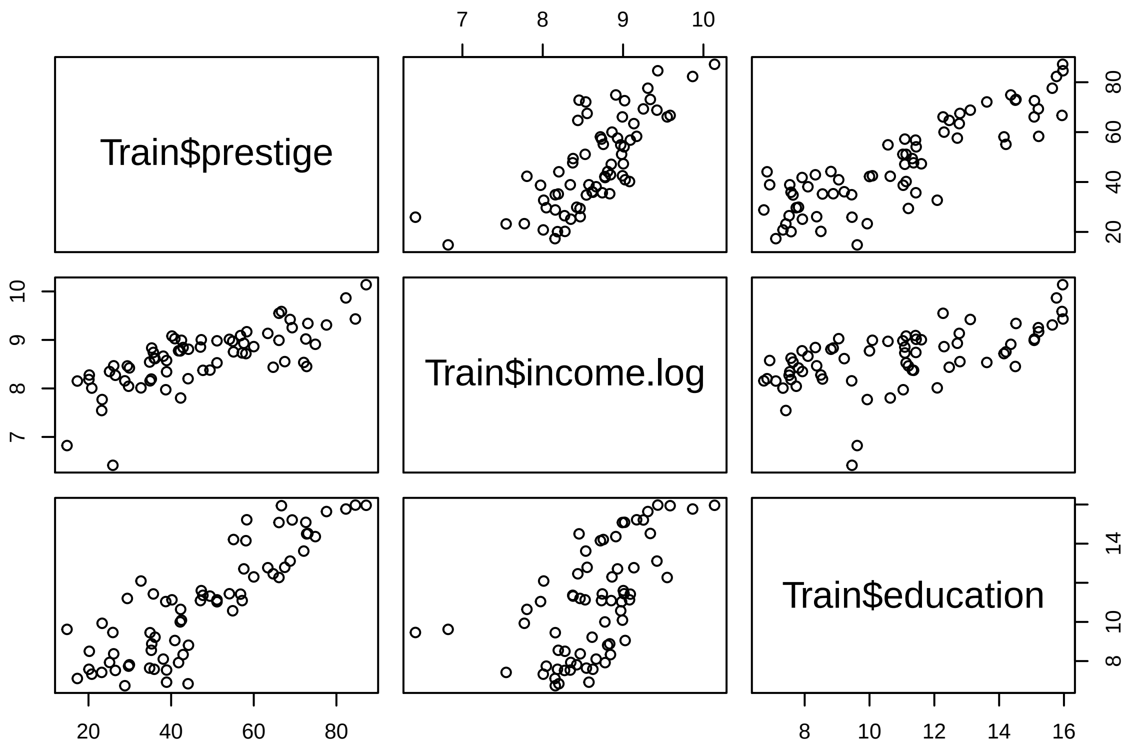
```
attach(Train)
pairs(Train$prestige ~ Train$income + Train$education)
```



Looks like the variable income has a nonlinear relationship, let's transfrom this value in the log() form. Note: it is always a good habit to check the bivariate relationship before runnning regression. Sometimes you need to thank about the funtional form of your variable: should it be in log(), squared or other formats?

3.a Variable transformation

```
Train$income.log = log(Train$income)
pairs(Train$prestige ~ Train$income.log + Train$education)
```




```r
#regression model
attach(Train)
lm <- lm(prestige ~ income.log + education)
```

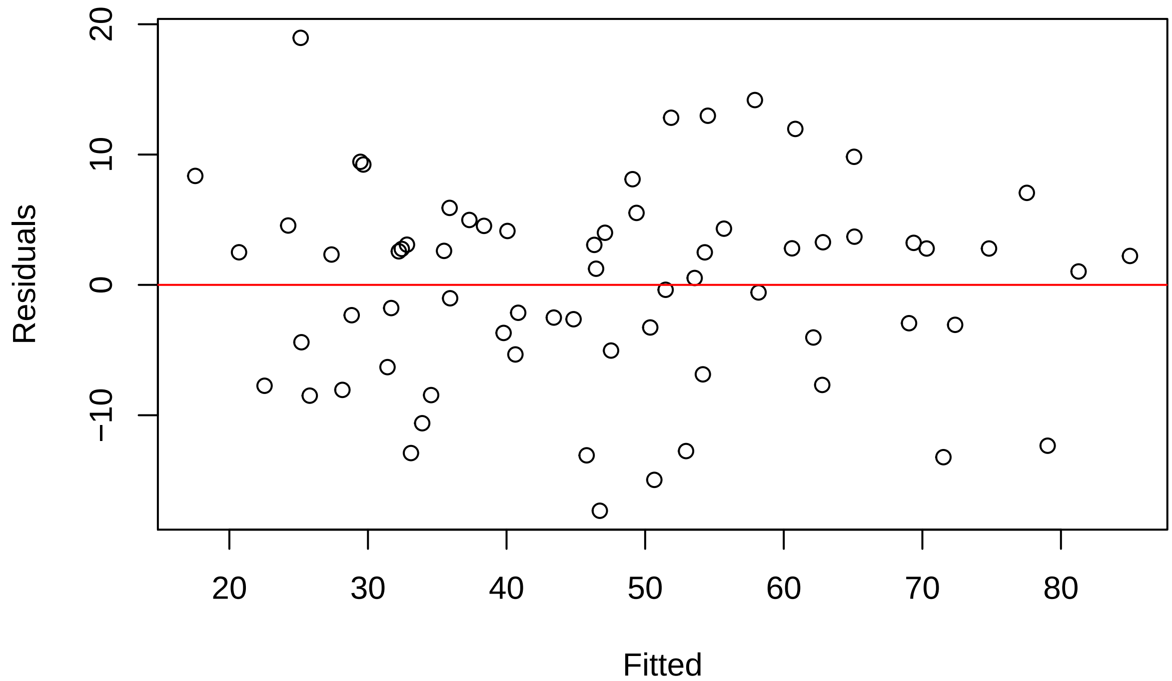
Looks much better!

4. Model assumptions check

4.a Constant Variance

We can check through the residual VS fitted value plot

```r
plot(fitted(lm), residuals(lm),  xlab="Fitted", ylab="Residuals")
abline(h=0, col="red") # draws a horizontal red line at y = 0
```



Alternatively, we can run a former test.

```
# Evaluate homoscedasticity
# non-constant error variance test
ncvTest(lm)
```

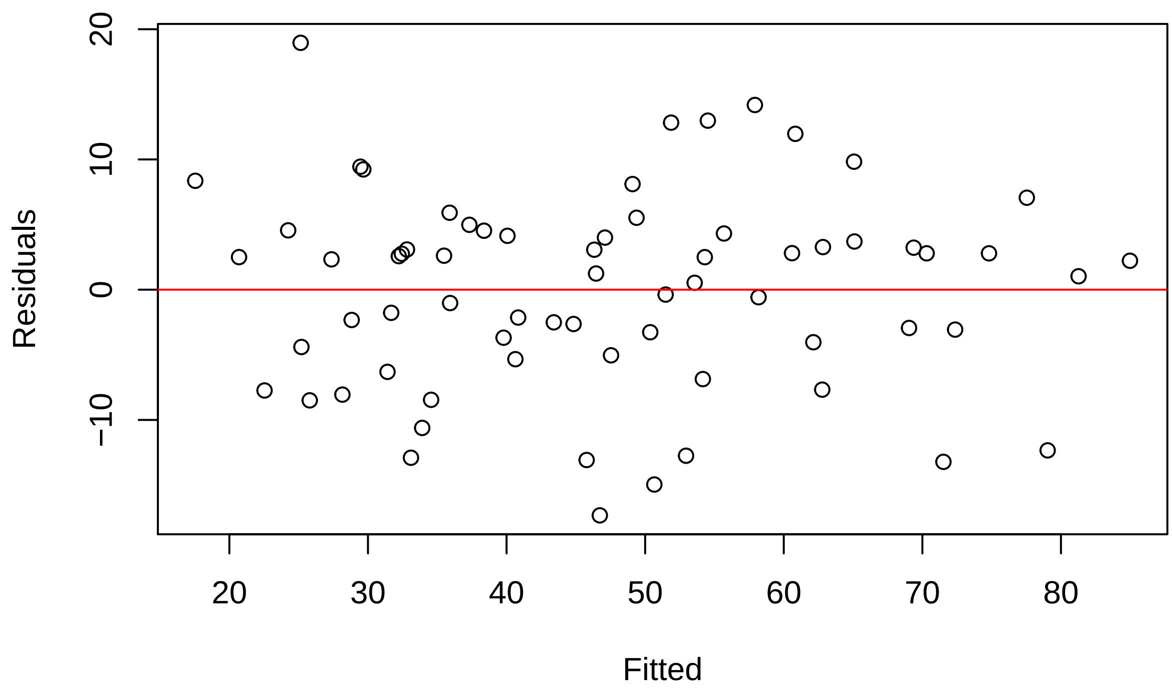
```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.05196521    Df = 1     p = 0.8196783
```

4.b Residual autocorrelation

In linear regression we require $\epsilon \overset{\text{i.i.d}}{\sim} (0, \sigma^2)$. This part check the iid assumption.

Again we can look at the residual VS fitted value plot. A formal test (Durbin-Watson test) is included in lmtest package.

```
plot(fitted(lm), residuals(lm),  xlab="Fitted", ylab="Residuals")
abline(h=0, col="red") # draws a horizontal red line at y = 0
```



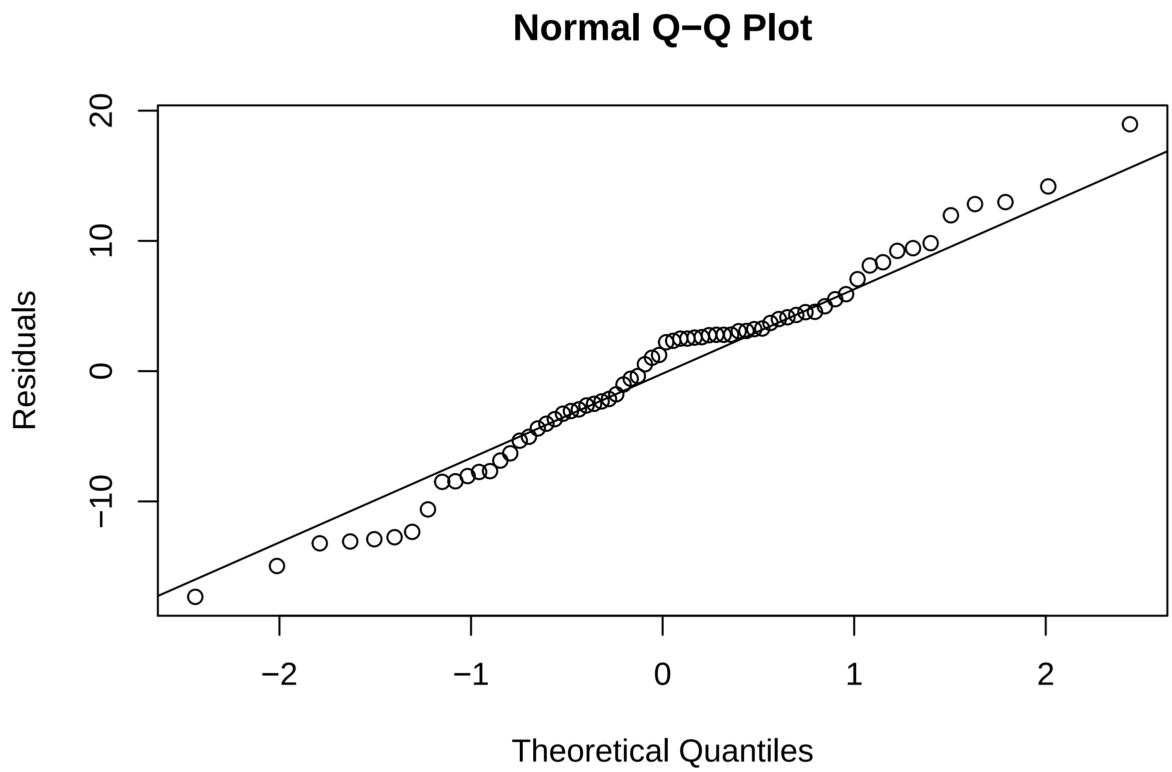

```r
# Test for Autocorrelated Errors
dwtest(lm)
```

```
##
##  Durbin-Watson test
##
## data:  lm
## DW = 1.7248, p-value = 0.1231
## alternative hypothesis: true autocorrelation is greater than 0
```

4.c Residual normality

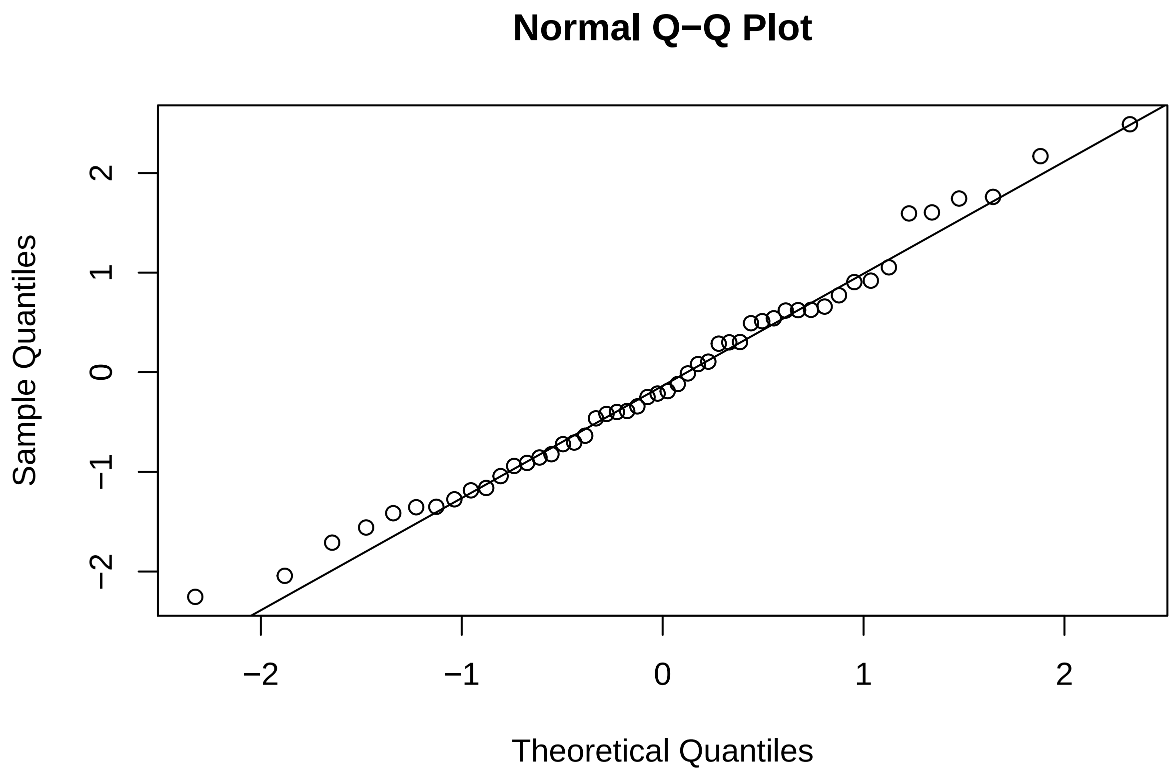
Normality assumption requires the error distributed as normal. We can check this through normal QQ plot.

```r
qqnorm(residuals(lm), ylab="Residuals") # Q-Q plot
qqline(residuals(lm)) # line through Q1 and Q3
```



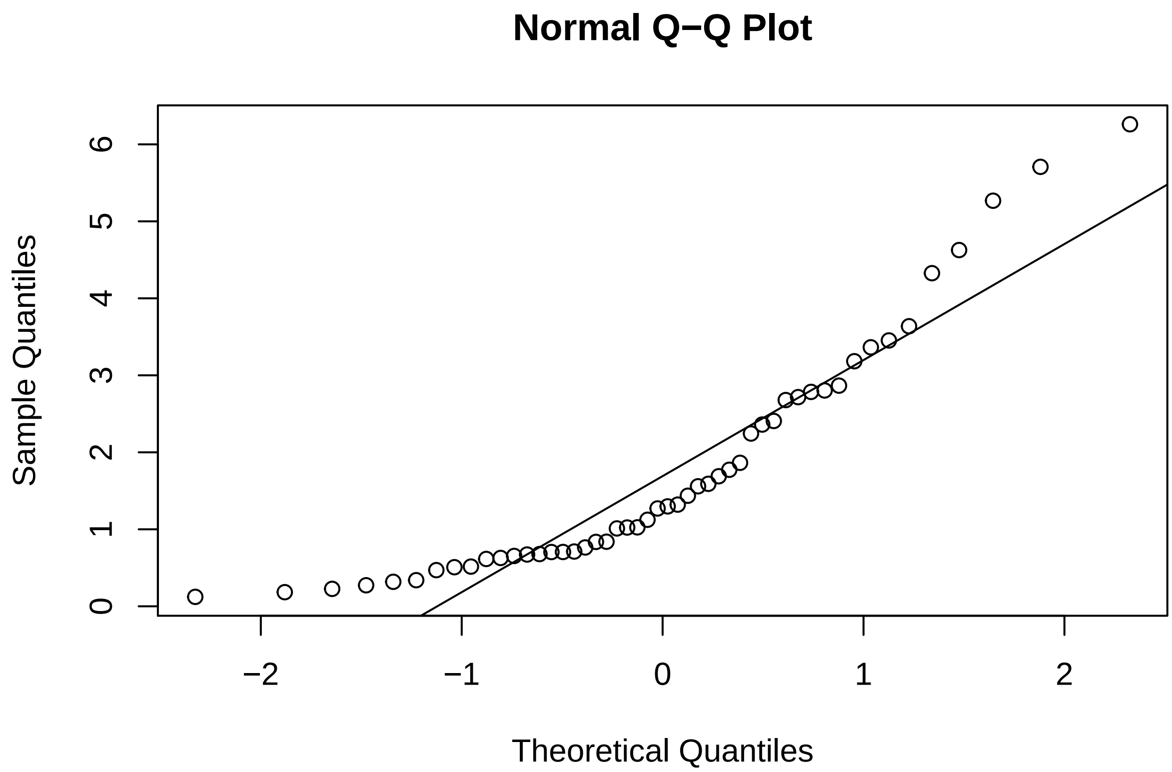
- Interpret QQ plot: To get an idea of the variation to be expected in a Q-Q plot, inspect the plots generated for a number of probability distributions. In the examples below, we use the standard normal, the lognormal, Student's t with one degree of freedom, and the uniform U(0; 1) distribution, respectively. Nine independent pseudo-random samples of size 50 are generated from each distribution. For each sample, a Q-Q plot with a quartile-line is produced.

```
for(i in 1:100) x = rnorm(50); qqnorm(x); qqline(x)
```



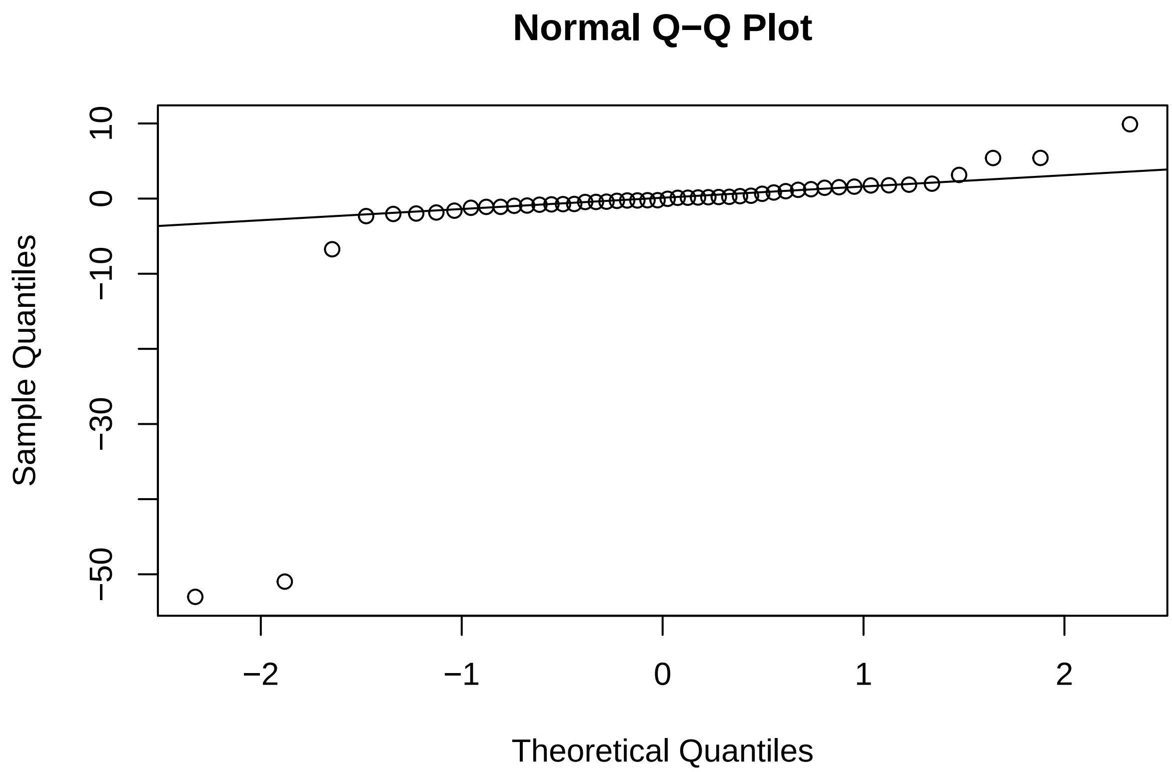

```r
# i.e., standard normal distribution (symmetric)

for(i in 1:100) x = rlnorm(50); qqnorm(x); qqline(x)
```



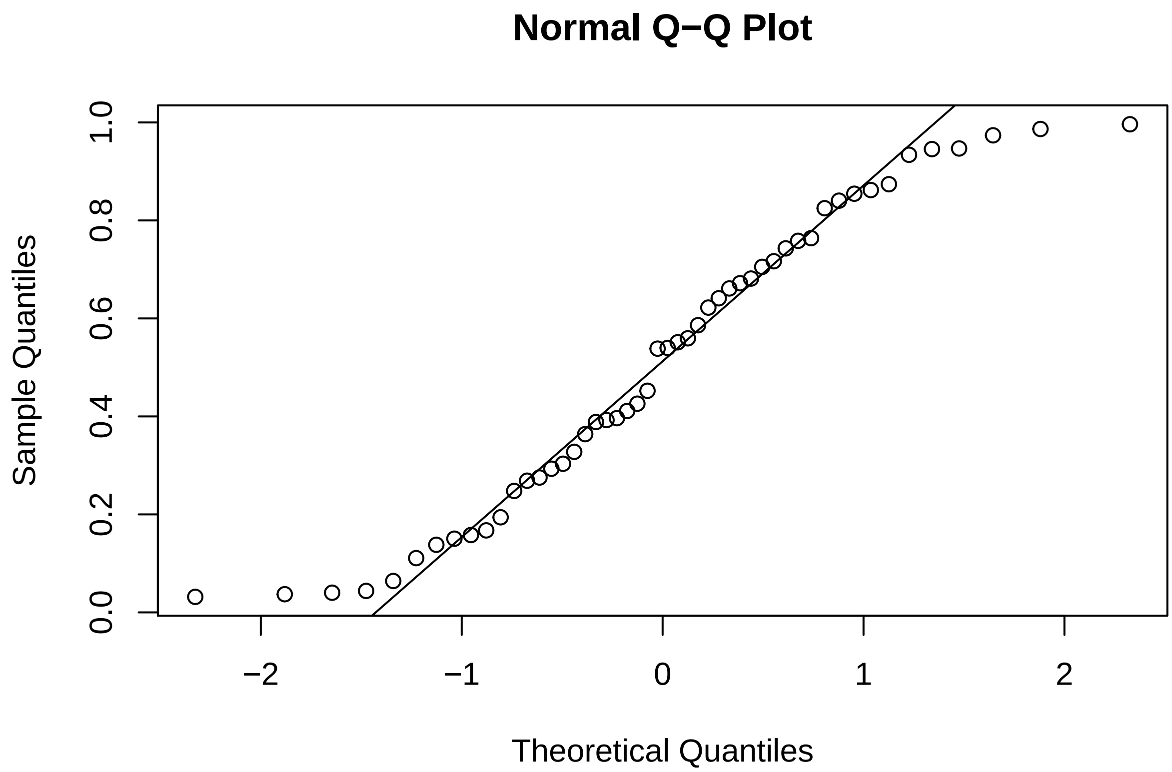

```
# lognormal distribution (long right tail, skew to right)

for(i in 1:100) x = rt(50,1); qqnorm(x); qqline(x)
```




```
# Student t-distribution with one df (heavy tails, platykurtic)

for(i in 1:100) x = runif(50); qqnorm(x); qqline(x)
```

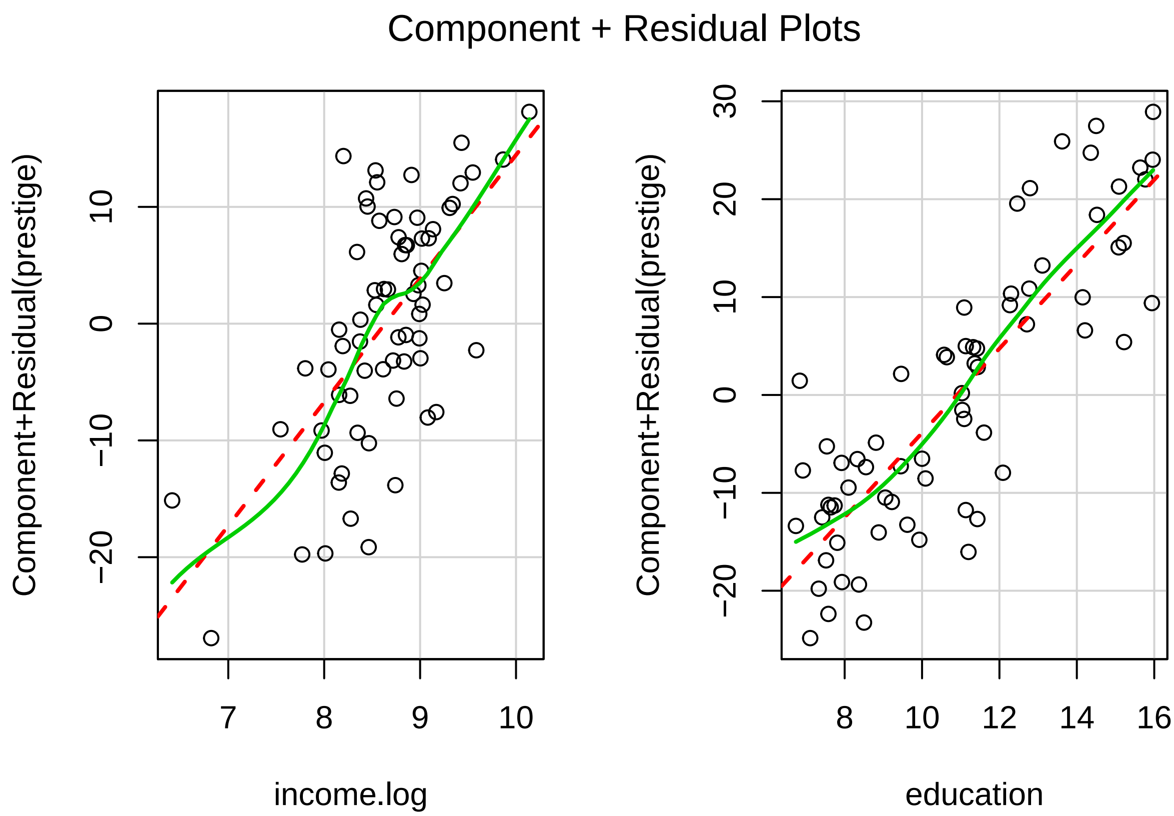



```r
# uniform (0,1) distribution (short tails, leptokurtic)
```

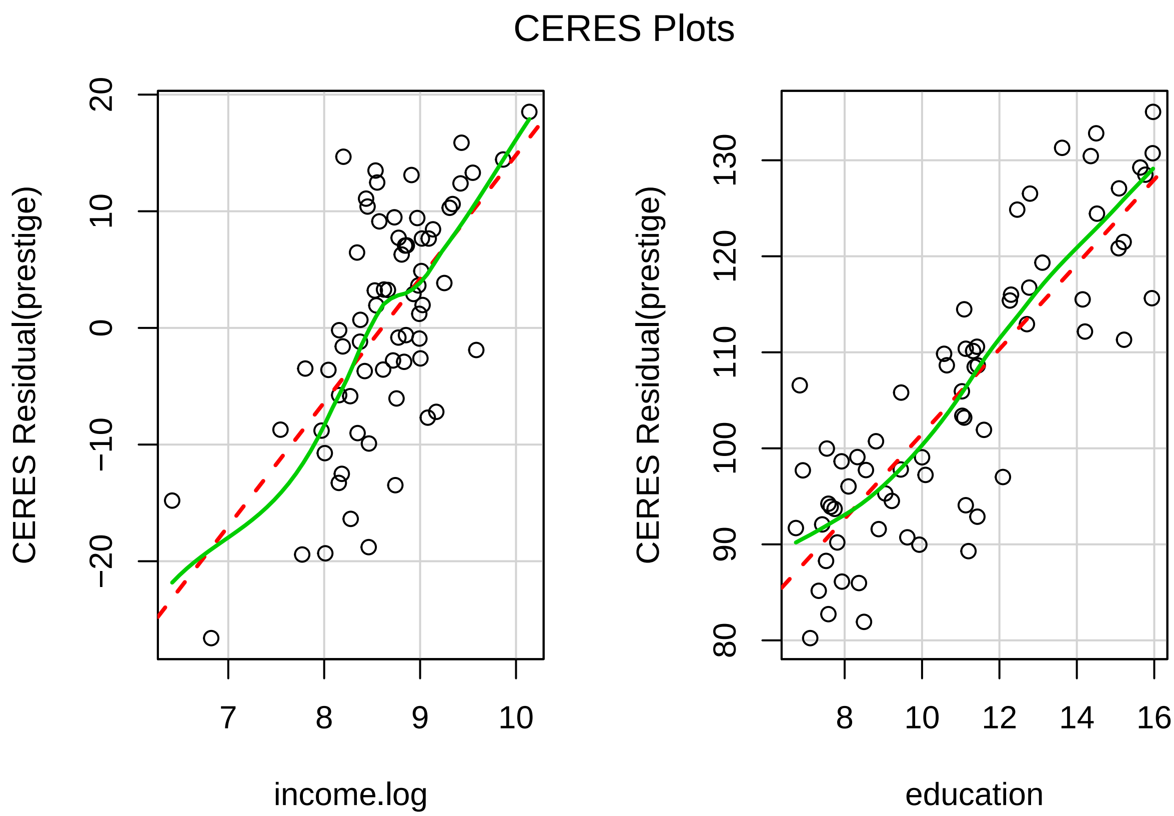
4.d Non-linearity.

Component residual plots, an extension of partial residual plots, are a good way to see if the predictors have a linear relationship to the dependent variable. A partial residual plot essentially attempts to model the residuals of one predictor against the dependent variable. A component residual plot adds a line indicating where the line of best fit lies. A significant difference between the residual line and the component line indicates that the predictor does not have a linear relationship with the dependent variable.

```r
# Evaluate Nonlinearity
# component + residual plot
crPlots(lm)
```




```r
# Ceres plots
ceresPlots(lm)
```



4.e Multicolinearility

We can use Variance Inflation Factor to detect potential multicollinearity problem. Generally speaking a VIF value larger than 4 is problematic.

```
a <- (vif(lm) >4)
cat('Multicolinearity is ', a, '\n')
```

```
## Multicolinearity is  FALSE FALSE
```

5. Influenatial Points

Linear regression is very sensitive to high leverage points. We need to be careful whether to delelte them or weight them.

5.a High leverage points

The leverage points are determined by the hat matrix H. Generally speaking the critical value is defined by 2p/N. p is the number of variables, and N is the number of observations.

```
hat <- influence(lm)$hat
cutoff <- 2*(length(lm$coefficients)-1)/length(Train) #cutoff points
cutoff
```

```
## [1] 0.5714286
```

```r
which( hat>cutoff)
```

```
## named integer(0)
```

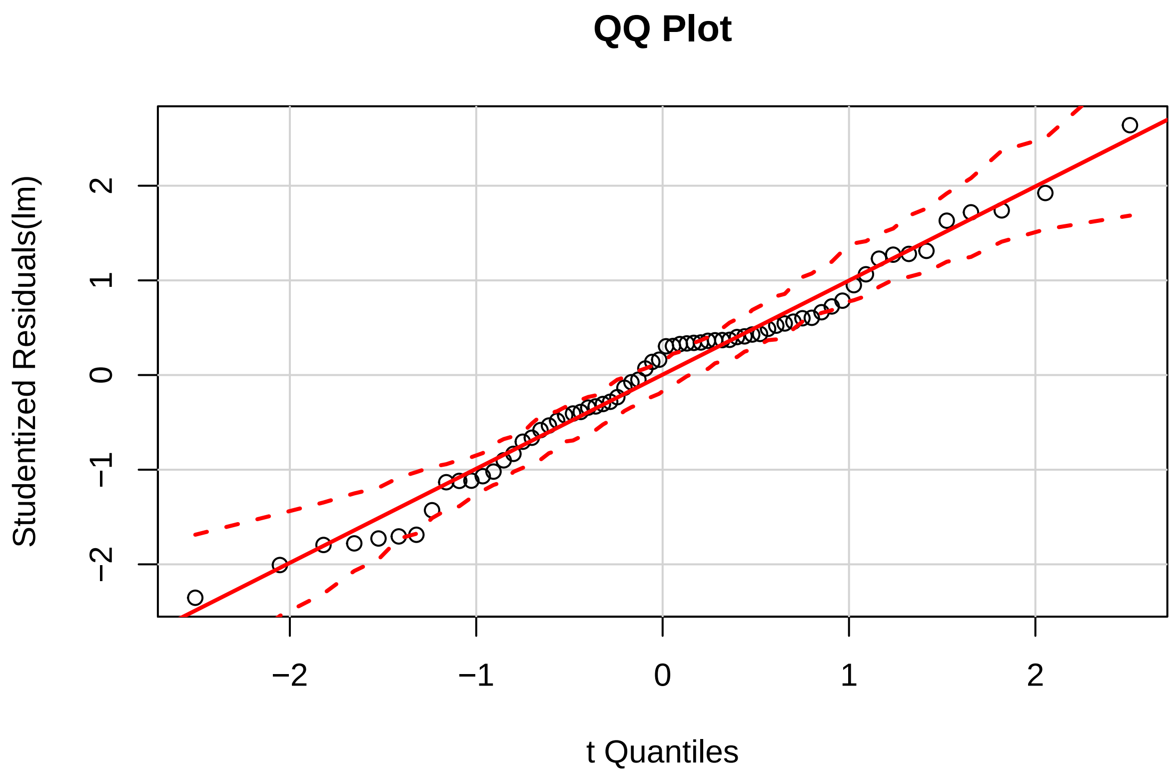
In our case there is no high leverage points.

5.b Outliers

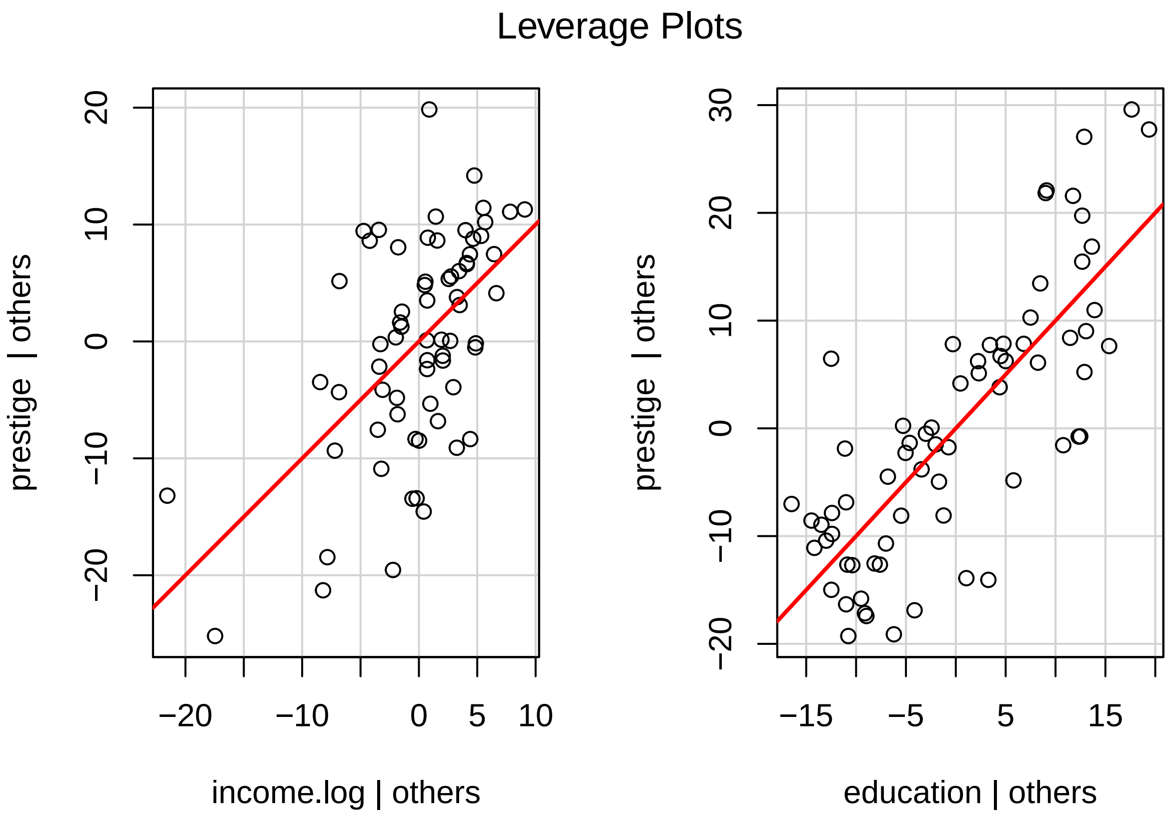
```r
# Assessing Outliers
outlierTest(lm) # Bonferonni p-value for most extreme obs
```

```
##
## No Studentized residuals with Bonferonni p < 0.05
## Largest |rstudent|:
##    rstudent unadjusted p-value Bonferonni p
## 42 2.639704           0.010411      0.70796
```

```r
qqPlot(lm, main="QQ Plot") #qq plot for studentized resid
```



```r
leveragePlots(lm) # leverage plots
```

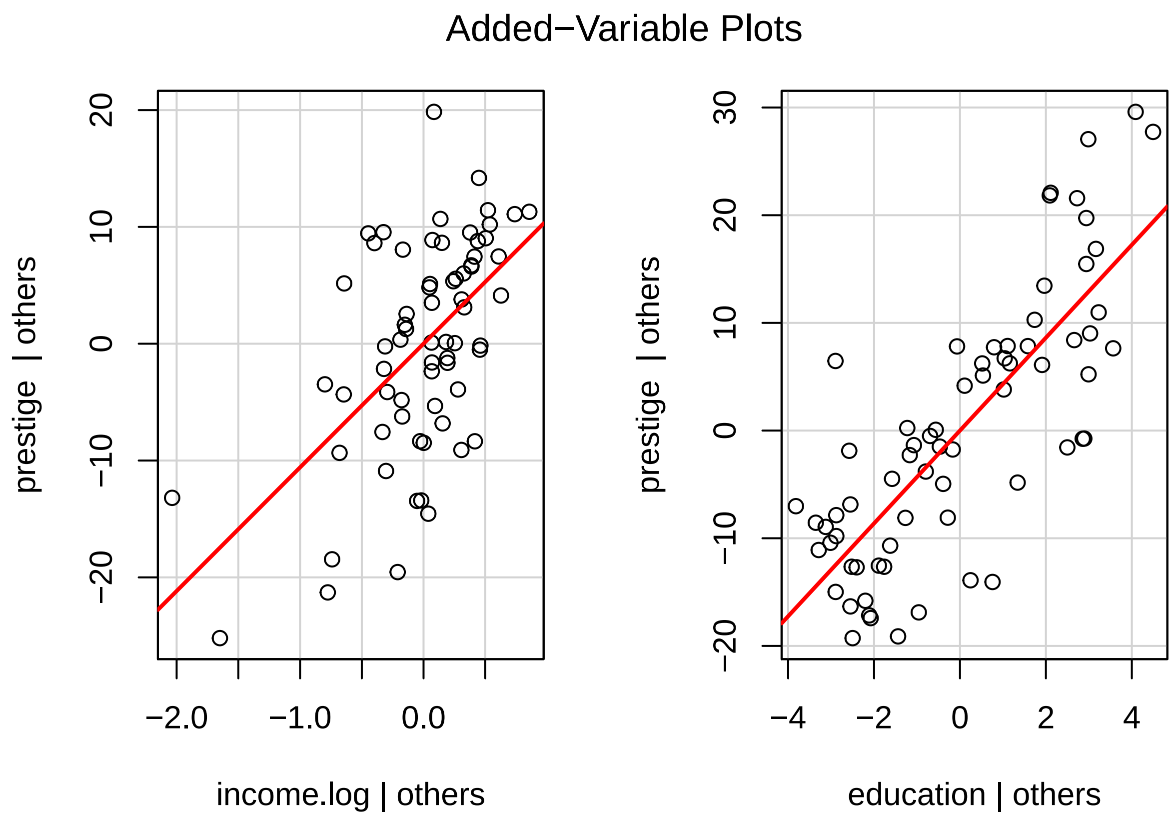


5.c Influential Points

Cook's distance measure is a combination of a residual effect and leverage, as shown by Equation 19 in Boomsma (2010). This combination leads to influence.

Influential Plots

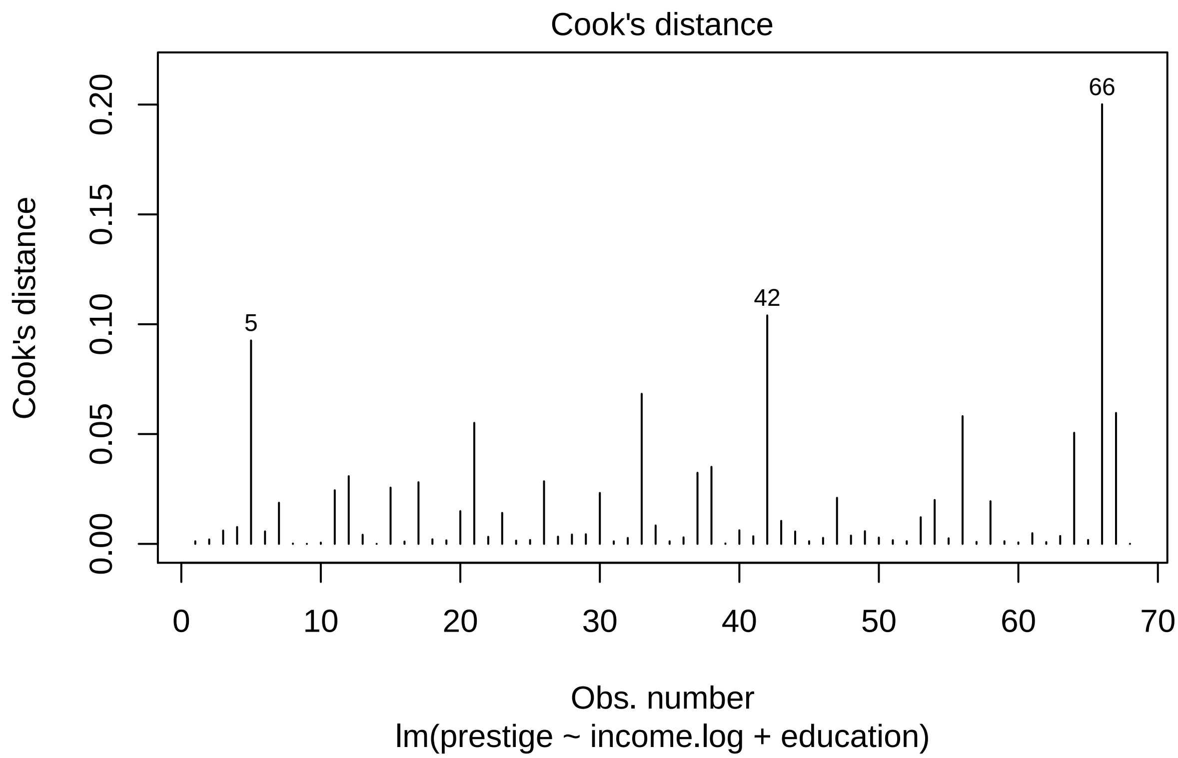
```r
# Influential Observations
# added variable plots
avPlots(lm)
```



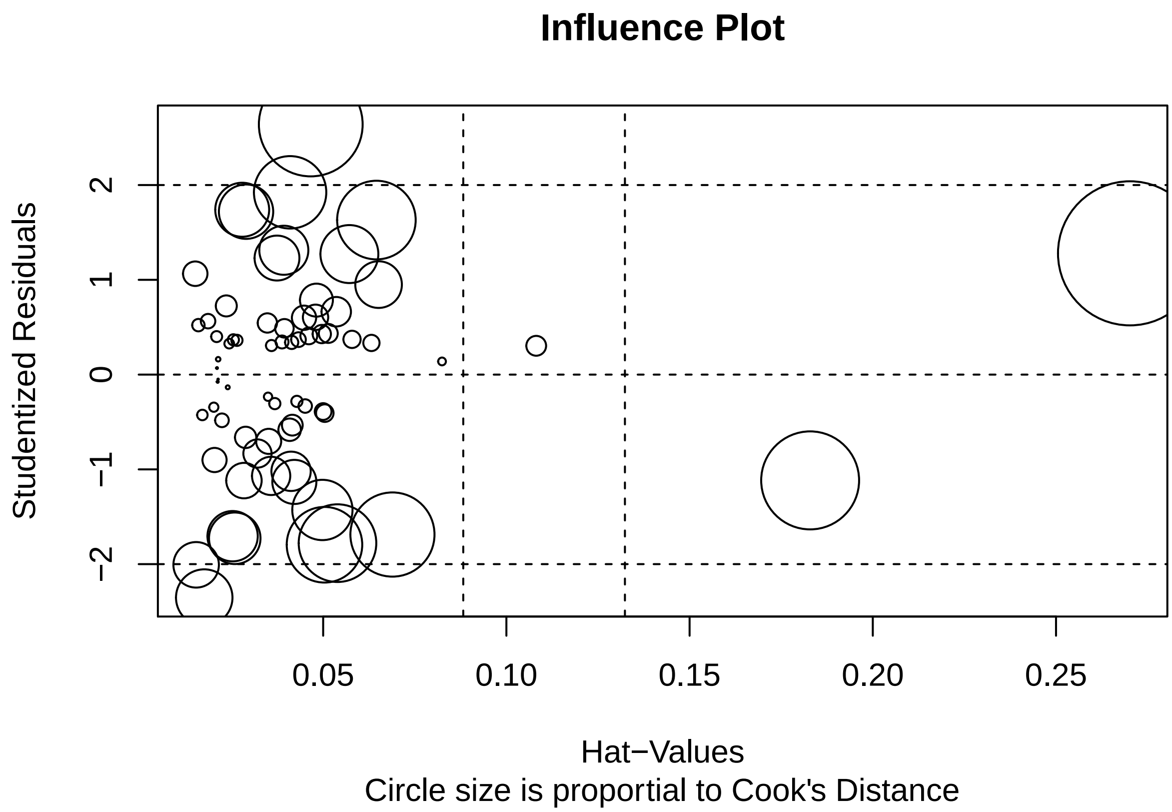

```r
# Cook's D plot
# identify D values > 4/(n-k-1)
cutoff <- 4/((nrow(Train)-length(lm$coefficients)-2))
cutoff
```

```
## [1] 0.06349206
```

```r
plot(lm, which=4, cook.levels=cutoff)
```




```
# Influence Plot
influencePlot(lm,   id.method="identify", main="Influence Plot", sub="Circle size is proportial to Cook
```

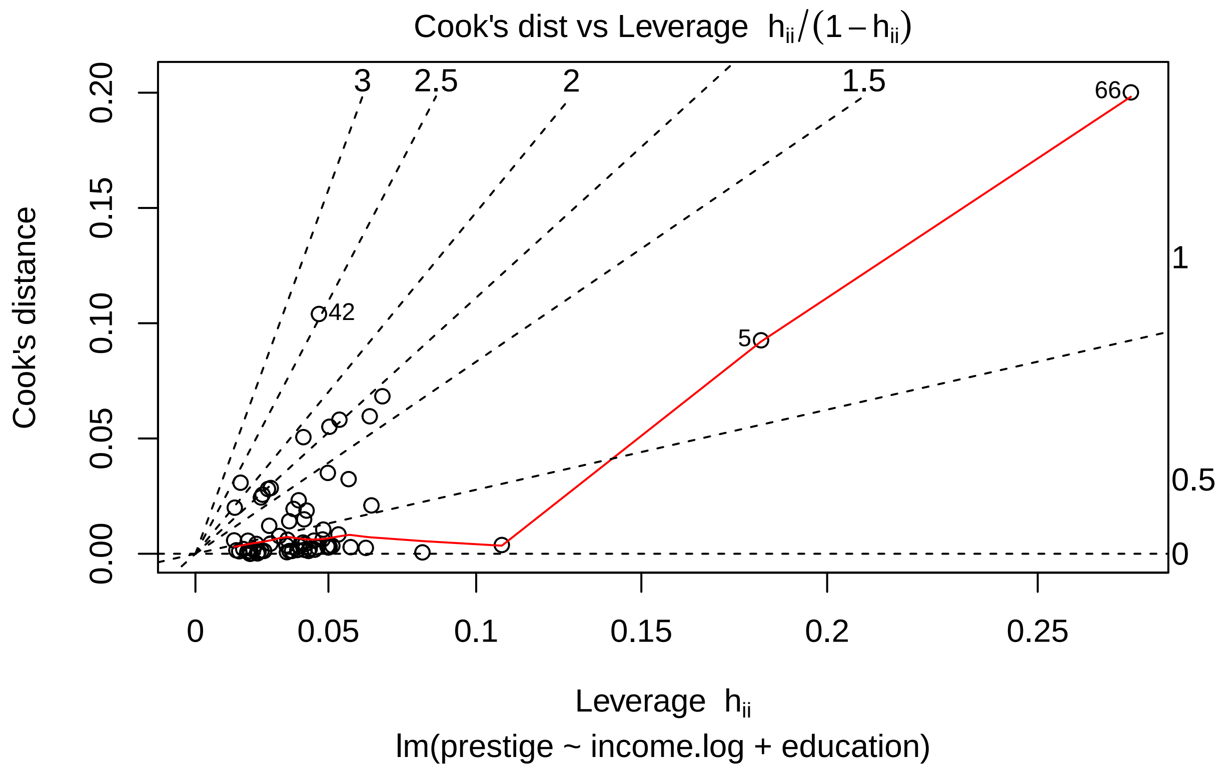


Cook's Distance We can also plot leverage points against Cook's distance.

```
# Cook's Distance
cook <- cooks.distance(lm)
a <- which.max(cook)
cat('The obs which maxes cooks D is', a, '\n')
```

```
## The obs which maxes cooks D is 66
```

```
plot(lm, which=6) # leverage against Cook's distance
```



Check regression while deleting the max cook obs.

```r
summary(lm)
```

```
##
## Call:
## lm(formula = prestige ~ income.log + education)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.329  -4.561   1.731   4.180  18.956
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -91.1493    14.2189  -6.410 1.88e-08 ***
## income.log   10.5854     1.8927   5.593 4.79e-07 ***
## education     4.3111     0.4138  10.418 1.71e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.685 on 65 degrees of freedom
## Multiple R-squared:  0.8293, Adjusted R-squared:  0.824
## F-statistic: 157.9 on 2 and 65 DF,  p-value: < 2.2e-16
```

```r
Train.sub <- subset(Train, cook<max(cook))
lm2 <- lm(prestige ~ income.log + education, data=Train.sub)
summary(lm2) # linear model estimates without Liby
```

```
##
## Call:
## lm(formula = prestige ~ income.log + education, data = Train.sub)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.873  -4.536   1.017   3.883  19.133
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -101.910     16.459  -6.192 4.75e-08 ***
## income.log    12.000      2.184   5.494 7.26e-07 ***
## education      4.162      0.428   9.723 3.15e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.647 on 64 degrees of freedom
## Multiple R-squared:  0.8301, Adjusted R-squared:  0.8247
## F-statistic: 156.3 on 2 and 64 DF,  p-value: < 2.2e-16
```