Regression-1

Vicki Hertzberg

February 8, 2017

Linear Regression

We start first by loading up the following packages: HistData, car, and stargazer.

```
#load up necessary packages
library(HistData)
library(car)
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2015). stargazer: Well-Formatted Regression and Summary Statistics
Tables.
```

```
##  R package version 5.2. http://CRAN.R-project.org/package=stargazer
```

Understanding Linear Regresson: The Univariate Case

We want to determine a straight line that determines the relationship between x, some independent (predictor) variable, and y, an outcome or dependent variable.

You will recall from high school algebra that the equation for a straight line is:

$$y = \alpha + \beta x$$

where α is the y-intercept (the value of y when x=0) and β is the slope of the line (the amount that y will increase by with every unit increase in x).

A classic example is the height data from Sir Francis Galton, a 19th century statistician. He collected height data on 905 children born to 205 parents. The dataset is part of the R package HistData, which contains Data Sets from the History of Statistics and Data Visualization. The variables are the height of the child (as an adult) and the mid-parent height, that is the average of the parents' heights.

If you go to the Environment tab, you will see the Global Environment. Click on “package:HistData” and you will see the Galton dataset.

Let’s see what is in there:

```
# see what is in the Galton dataset
summary(Galton)
```

```
##      parent          child      
##  Min.   :64.00   Min.   :61.70  
##  1st Qu.:67.50   1st Qu.:66.20  
##  Median :68.50   Median :68.20  
##  Mean   :68.31   Mean   :68.09  
##  3rd Qu.:69.50   3rd Qu.:70.20  
##  Max.   :73.00   Max.   :73.70
```

Let’s develop the model on ~90% of the dataset, then test it on the remaining ~10% of the data. In the datascience world, we call that first step “training” the model.

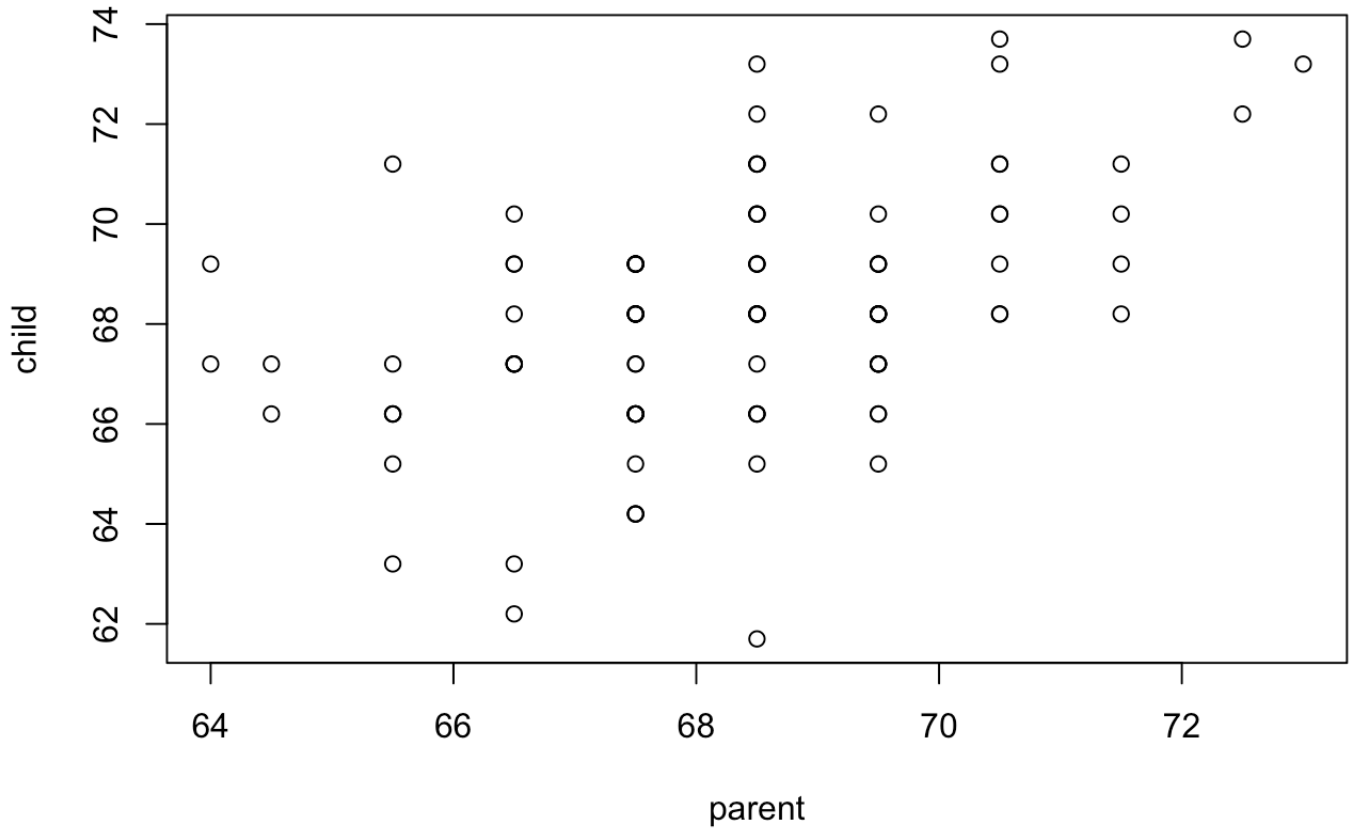
```
# divide the dataset into a training and a testing set based on a random uniform numb
er on fixed seed
set.seed(20170208)
Galton$group <- runif(length(Galton$parent), min = 0, max = 1)
summary(Galton)
```

```
##      parent          child           group        
##  Min.   :64.00   Min.   :61.70   Min.   :0.002463  
##  1st Qu.:67.50   1st Qu.:66.20   1st Qu.:0.245599  
##  Median :68.50   Median :68.20   Median :0.518798  
##  Mean   :68.31   Mean   :68.09   Mean   :0.501614  
##  3rd Qu.:69.50   3rd Qu.:70.20   3rd Qu.:0.743939  
##  Max.   :73.00   Max.   :73.70   Max.   :0.999420
```

```
Galton.train <- subset(Galton, group <= 0.90)
Galton.test <- subset(Galton, group > 0.90)
```

Now let’s graph our training set data.

```
#graph child on parent for testing dataset
plot(child ~ parent, data = Galton.test)
```



Let's do the regression now on the training set. Linear regression is performed with the R function "lm" and takes the form

object.name <- lm(y ~ x, data = data_set_name)

Let's do that now with the Galton data:

```
# linear regression of child height on mid-parent height in the training dataset
reg1 <- lm(child ~ parent, data = Galton.train)
summary(reg1)
```

```
##
## Call:
## lm(formula = child ~ parent, data = Galton.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.7923 -1.3502  0.0604  1.6498  5.9445
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 23.85366    2.99886   7.954  5.9e-15 ***
## parent       0.64736    0.04389  14.751  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.249 on 829 degrees of freedom
## Multiple R-squared:  0.2079, Adjusted R-squared:  0.2069
## F-statistic: 217.6 on 1 and 829 DF,  p-value: < 2.2e-16
```

Interpretation: for each increase in parent height of 1 inch, the child height increases by 0.65 inches.

Now the way that this is working is to estimate values for α and β such that when you plug in your given independent variables, you get predicted dependent variables that are close to the observed values. In statistics we optimize this closeness by minimizing the sum-of-squared-residuals, that is

$$\sum_{i=1}^{n}(Y_{obs.i} - Y_{pred.i})^2$$

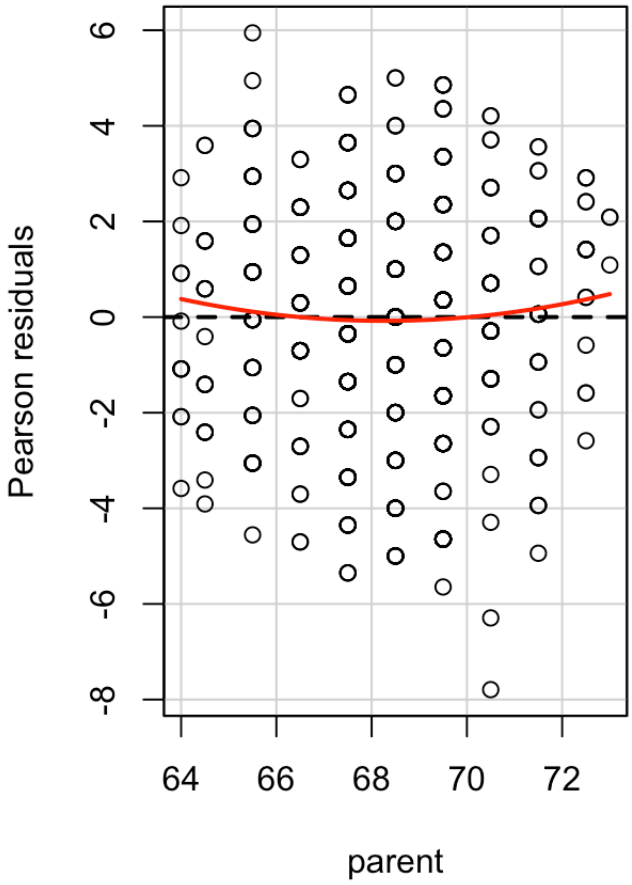
So let's look at how we did. First let's calculate the observed and predicted values in the training and testing datasets.

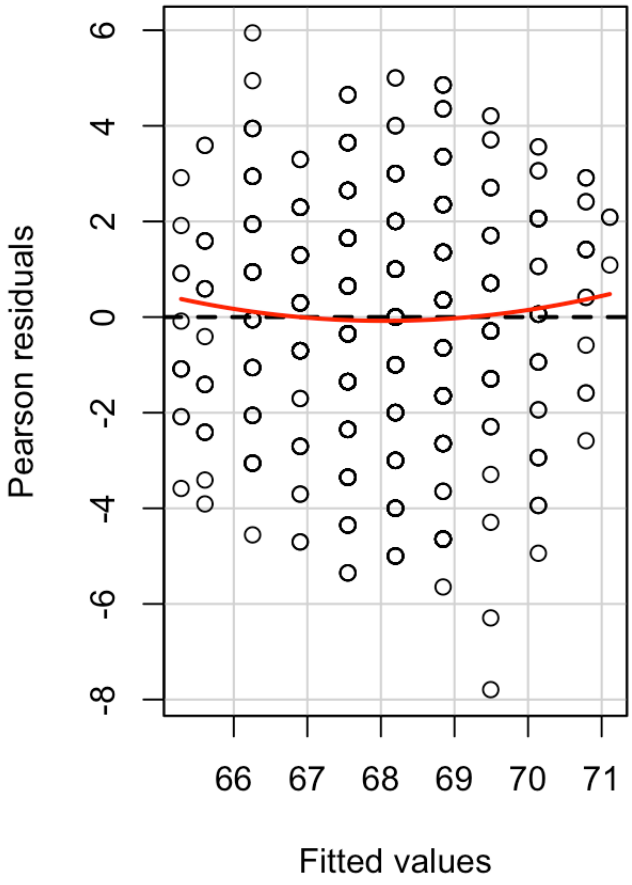
```
# get predicted values in the training and testing dataset
Galton.train$pred.child <- predict(reg1, newdata = Galton.train)
Galton.test$pred.child <- predict(reg1, newdata=Galton.test)

# calculate residuals in the training and testing dataset
Galton.train$resid <- Galton.train$child - Galton.train$pred.child
Galton.test$resid <- Galton.test$child - Galton.test$pred.child
```

Now that we have calculated these values, let's look at some simple plots. The Companion to Applied Regression (aka car) package, has some good functionality for this

```
library(car)
#get the residual plots
residualPlots(reg1)
```

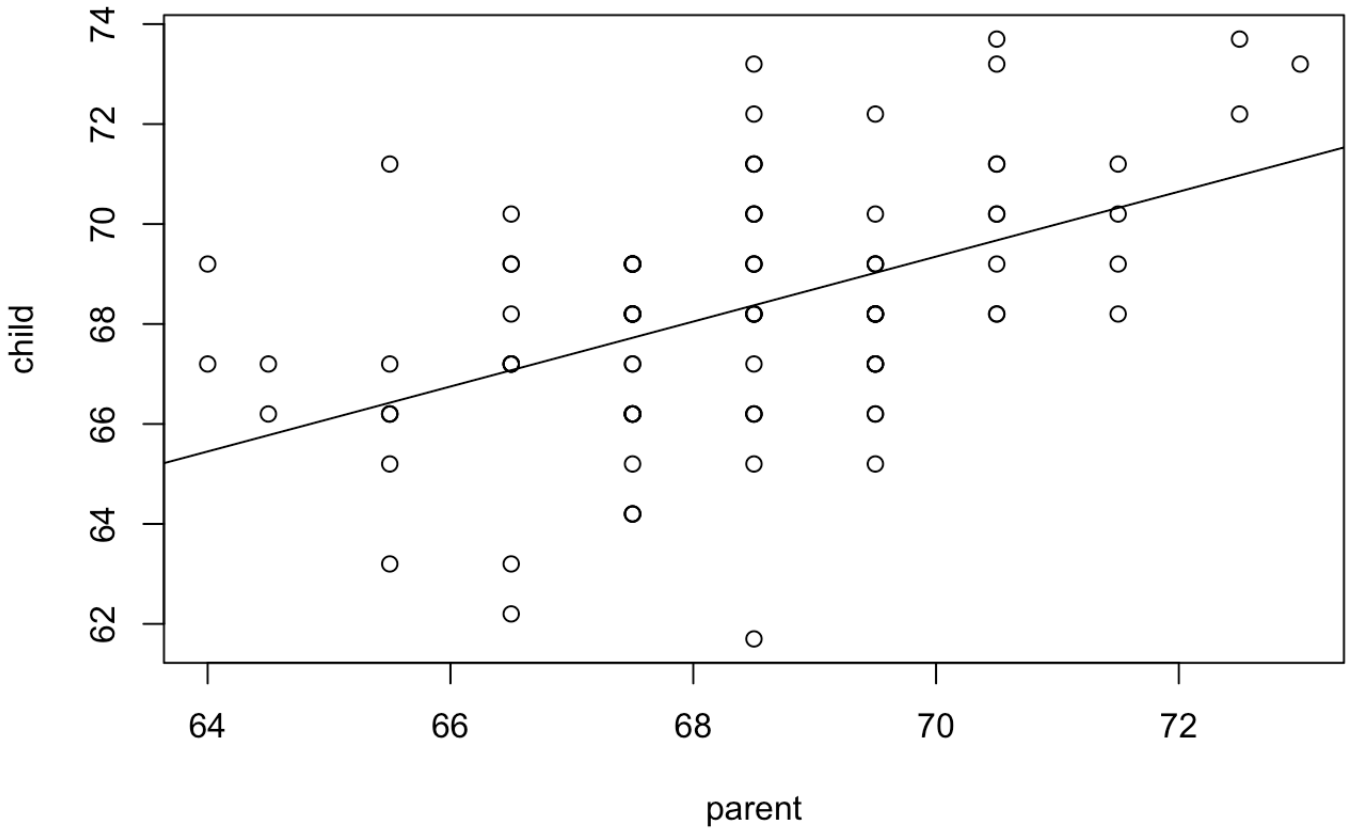





```
##            Test stat Pr(>|t|)
## parent         1.459    0.145
## Tukey test     1.459    0.144
```

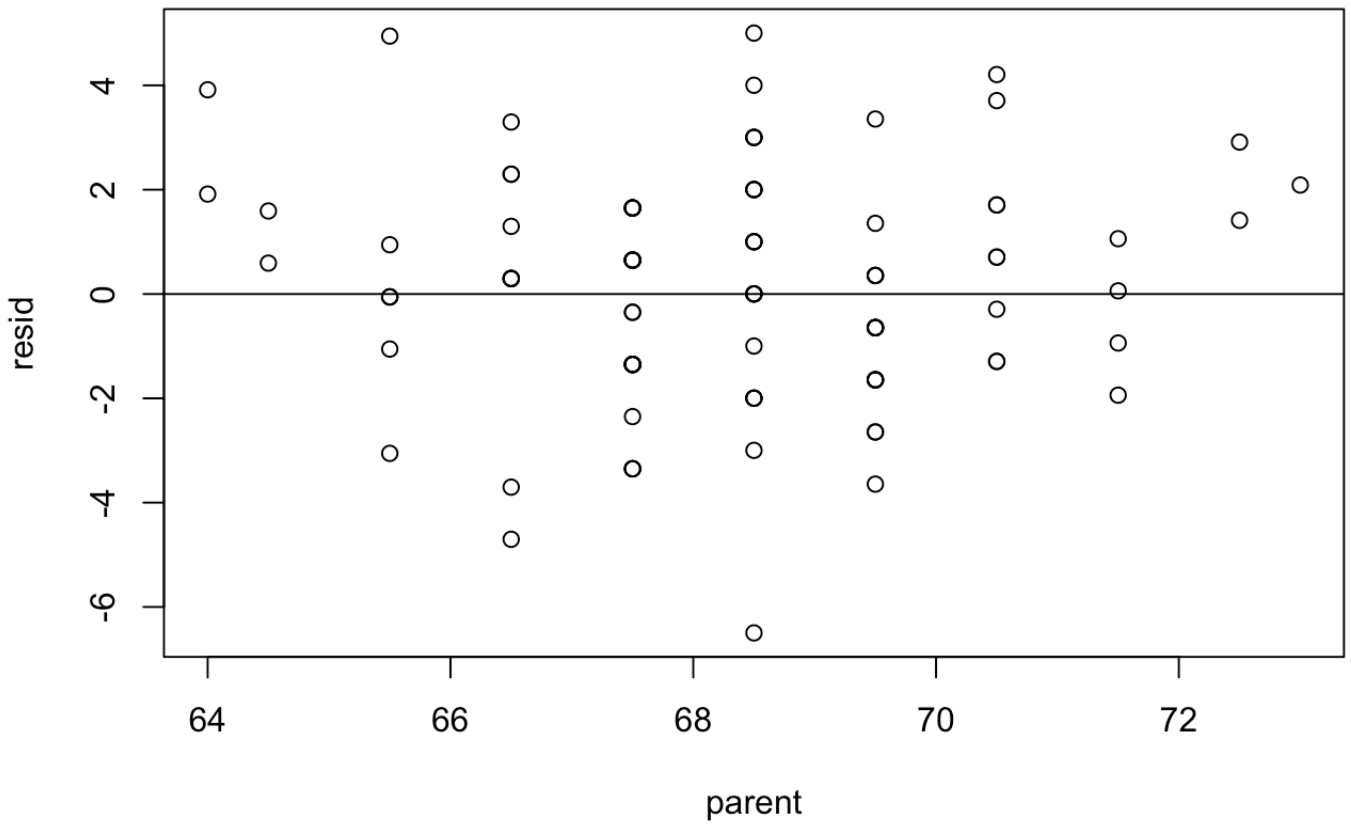
Let's look at how well we do in the testing dataset. First let's plot the data.

```
#plot test dataset
plot(child ~ parent, data = Galton.test)
#overlay the regression line
abline(a=23.85, b=0.65)
```

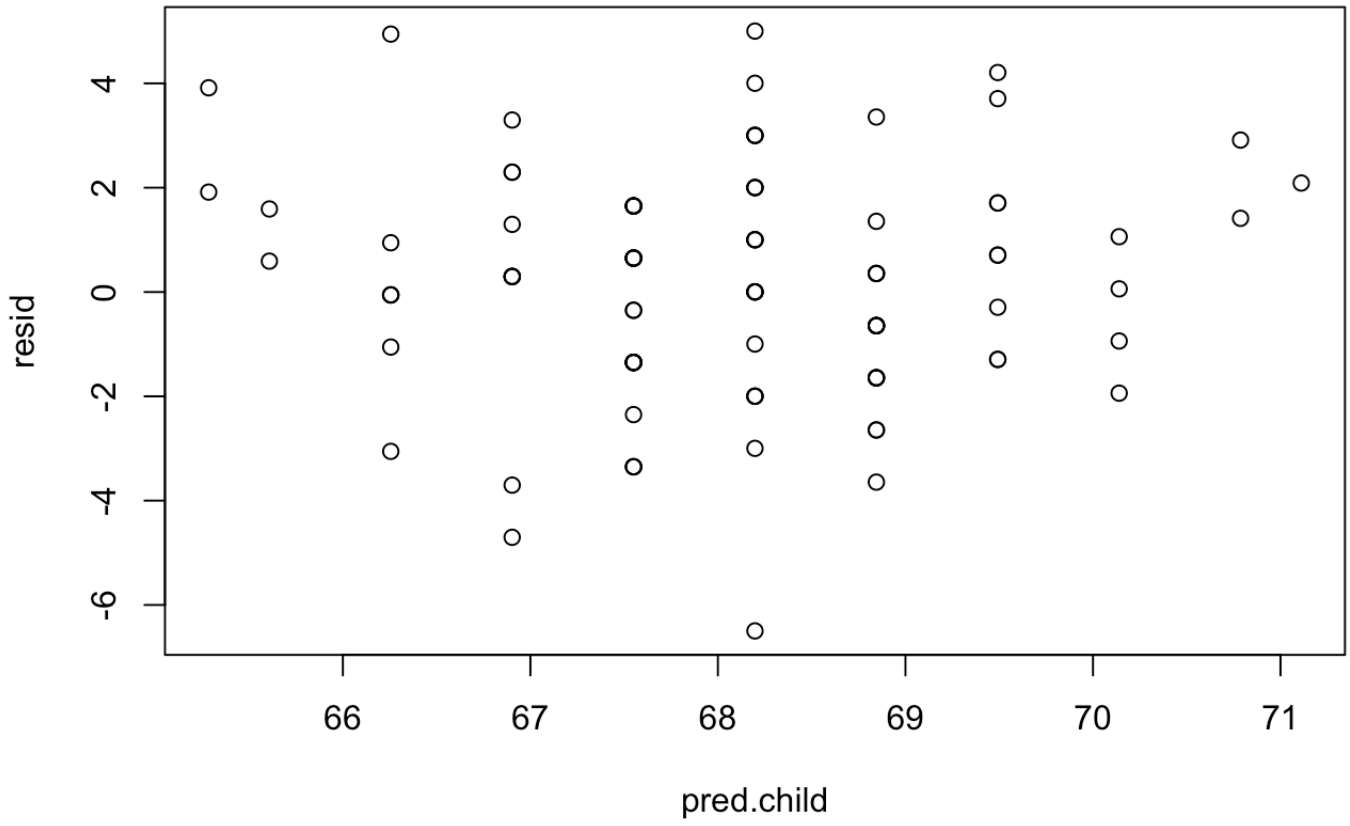


Now let's look at the residual plots:

```
#plot residual versus X
plot(resid~parent, data = Galton.test)
#overlay the zero line
abline(0,0)
```




```
#plot residual v predicted
plot(resid~pred.child, data = Galton.test)
```



One assumption in the statistical analysis of a linear regression are the hypothesis tests. We usually want to test a null hypothesis that the slope is 0, ie, there is no relationship between y and x. We express this mathematically as

$$H_0 : \beta = 0$$

We test a null hypothesis against an alternative hypothesis, ie, the slope is not equal to 0, ie, there is some (positive or negative) relationship between y and x. We express this mathematically as

$$H_A : \beta \neq 0$$

As you have seen in these data, the y-values do not line up as a perfect linear function of x, ie child height does not line up as a perfect linear function of parent height. There is an error that occurs for each observation. So we are really modeling a statistical model

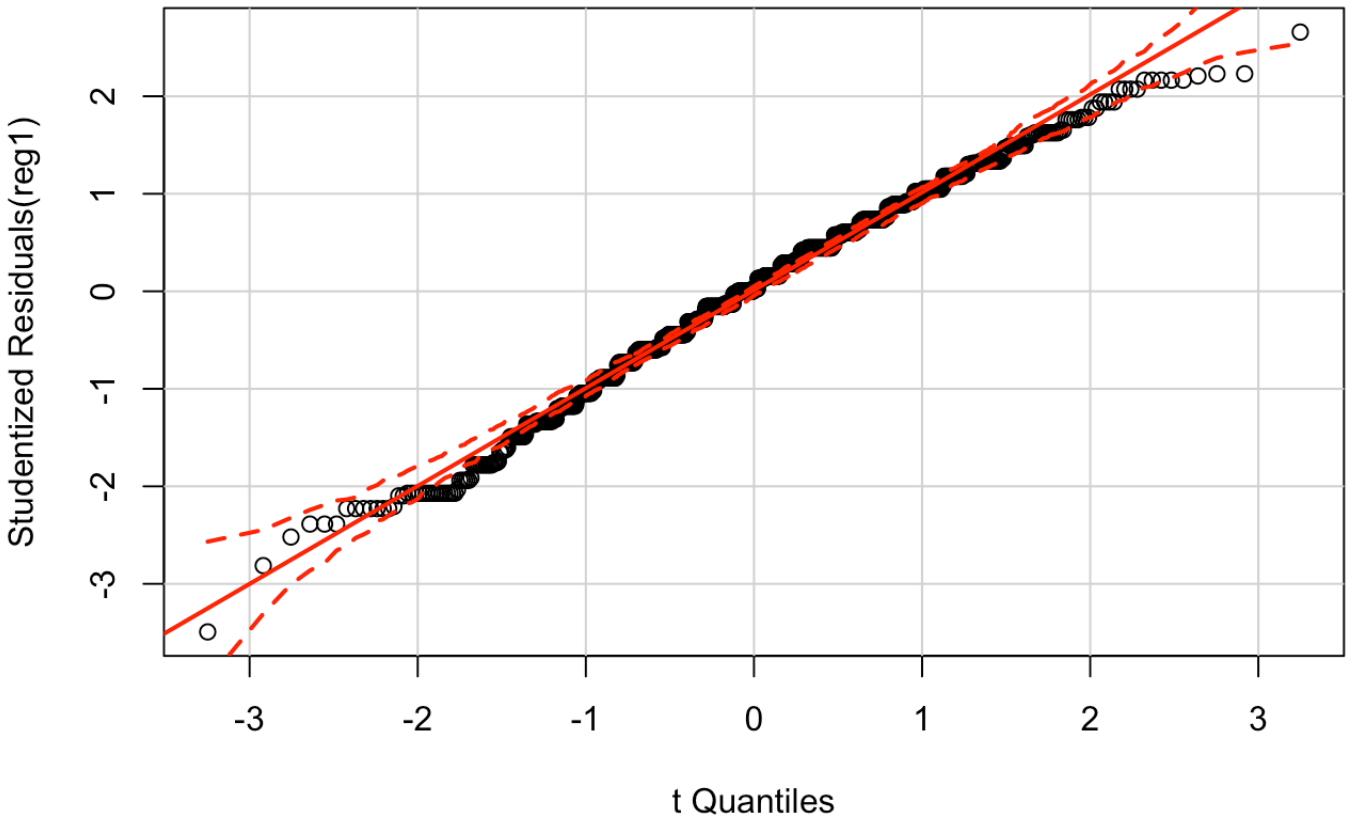
$$Y = \alpha + \beta x + \epsilon$$

Recall from above that we estimate α and β to minimize the sum of squared residuals.

To do a statistical test on the slopes we have to make some assumptions. One of the assumptions is that ϵ ~ N(0, σ^2), that is, that the errors between the observed and predicted values of Y take on a normal distribution with mean of 0 and variance of σ^2.

We can formally test this assumption, but we can also do a qq-plot which will give us a visual representation of the same. We get this plot as follows:

```
qqPlot(reg1)
```



In this plot we have taken each residual and divided by the estimated standard deviation to create studentized residuals. We then rank them and calculate the percentile for each studentized residual, then create this graph. Of course, the car package is doing all of this under the hood, so to speak.

To do the significance test, recall the summary of the linear regression model we had above.

```
##
## Call:
## lm(formula = child ~ parent, data = Galton.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.7923 -1.3502  0.0604  1.6498  5.9445
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 23.85366    2.99886   7.954  5.9e-15 ***
## parent       0.64736    0.04389  14.751  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.249 on 829 degrees of freedom
## Multiple R-squared:  0.2079, Adjusted R-squared:  0.2069
## F-statistic: 217.6 on 1 and 829 DF,  p-value: < 2.2e-16
```

Note that there are two estimated coefficients, so our estimated line is

$$child = 23.85 + 0.65 * parent$$

You will also notice that next to the estimate, there is a column for standard error, a column for t-value, and column for p-value. We divide the estimate by the standard error to compute the t-value, which then has 829 degrees of freedom. For the slope estimate, we calculate the the p-value is < 2e-16, ie, highly significantly different from 0.

At the bottom of summary you will notice a line for the F-statistics, which is the ratio of the mean-squared error of the model to the mean-squared error of the residuals. This has an F-distribution with 1 and 829 degrees of freedom, and takes on the value of 217.6 which has a p-value < 2e-16. Since this is a univariate regression you will see that if you take the value of the t-statistic for the slope and square it, that will give you the value of the F-statistic.

Isn't math fun?

Understanding Linear Regresson: The Multivariate Case

Suppose you have more than one independent variable that you think explains the dependent variable. That is instead of the simple univariate case of

$$y = \alpha + \beta x$$

You have the multivariate case of

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

As an example consider the PRESTIGE dataset (which comes with the car package). It consists of 102 observations and 6 variables as follows:

education: The average number of years of education for occupational incumbents.
income: The average income of occupational incumbents, in dollars.
women: The percentage of women in the occupation.
prestige: The average prestige rating for the occupation.
census: The code of the occupation used in the survey.
type: Professional and managerial(prof), white collar(wc), blue collar(bc), or missing(NA).

We want to determine how the prestige of an occupation is related to average income, education (average number of years for incumbents), and the percentage of women in the occupation.

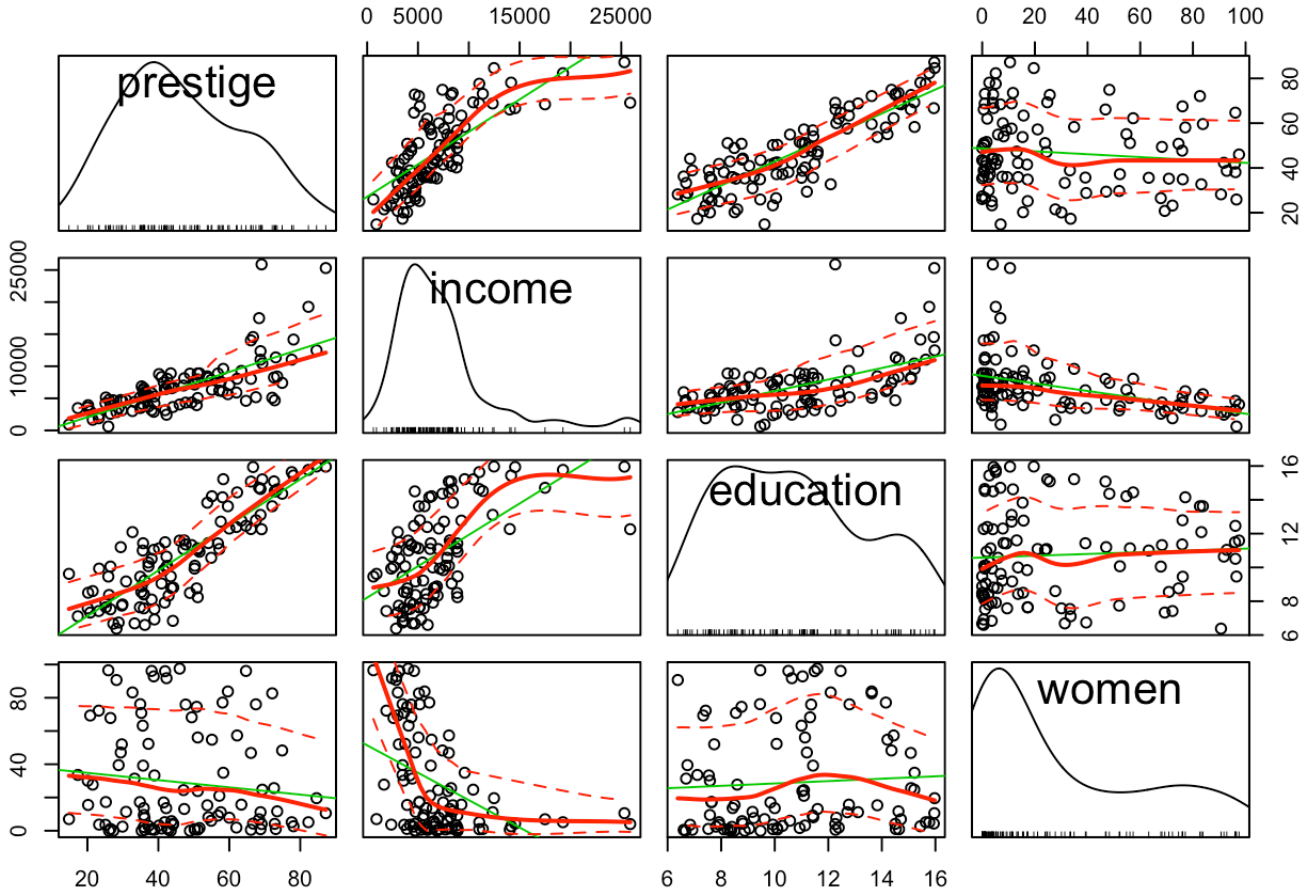
Let's explore first:

```
#explore the 5 m's for the Prestige dataset
summary(Prestige)
```

```
##    education          income          women           prestige
##  Min.   : 6.380   Min.   :  611   Min.   : 0.000   Min.   :14.80
##  1st Qu.: 8.445   1st Qu.: 4106   1st Qu.: 3.592   1st Qu.:35.23
##  Median :10.540   Median : 5930   Median :13.600   Median :43.60
##  Mean   :10.738   Mean   : 6798   Mean   :28.979   Mean   :46.83
##  3rd Qu.:12.648   3rd Qu.: 8187   3rd Qu.:52.203   3rd Qu.:59.27
##  Max.   :15.970   Max.   :25879   Max.   :97.510   Max.   :87.20
##      census       type
##  Min.   :1113   bc  :44
##  1st Qu.:3120   prof:31
##  Median :5135   wc  :23
##  Mean   :5402   NA's: 4
##  3rd Qu.:8312
##  Max.   :9517
```

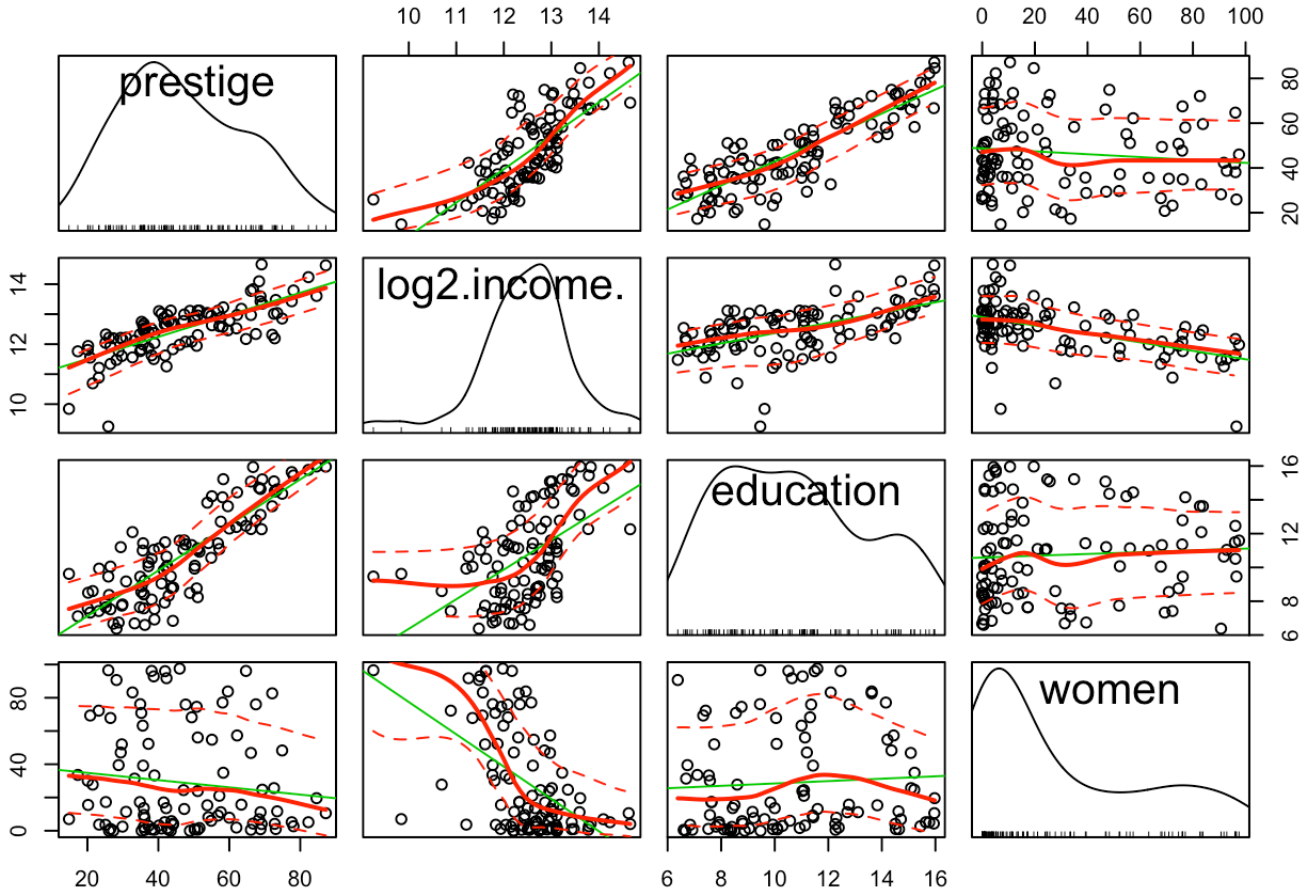
Now let's see some plots:

```
#produce a scatterplot matrix
scatterplotMatrix(~ prestige + income +education + women, span =0.7, data = Prestige)
```



Hmmm…the variable income looks kind of wonky, for lack of a better term. So let's see if a transformation will help:

```
scatterplotMatrix(~ prestige + log2(income) +education + women, span =0.7, data = Pre
stige)
```



Well, that looks a little better.

Now let's see what the regression of prestige on income (logged), education, and women looks like:

```
# let the regression rip
prestige.mod1 <- lm(prestige ~ education + log2(income) + women, data= Prestige)

#and see what we have
summary(prestige.mod1)
```

```
##
## Call:
## lm(formula = prestige ~ education + log2(income) + women, data = Prestige)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.364  -4.429  -0.101   4.316  19.179
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -110.9658    14.8429  -7.476 3.27e-11 ***
## education       3.7305     0.3544  10.527  < 2e-16 ***
## log2(income)    9.3147     1.3265   7.022 2.90e-10 ***
## women           0.0469     0.0299   1.568     0.12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.093 on 98 degrees of freedom
## Multiple R-squared:  0.8351, Adjusted R-squared:   0.83
## F-statistic: 165.4 on 3 and 98 DF,  p-value: < 2.2e-16
```

From this output we see that if education increases by 1 year, then average prestige rating will increase by 3.73 units, with income and women held constant.

For education we note that the p-value is < 2e-16.

So we interpret that to reject the null hypothesis H_0: $\beta_1 = 0$.

How do interpret the contributions of log2(income) and women?

Another point to remark: the multiple R-squared is 0.84, indicating that 84% of the variability in average prestige rating is due to these three independent variables.

Finally, the F-statistic of 165.4 is for testing the null hypothesis H_0: $\beta_1 = \beta_2 = \beta_3 = 0$. It is highly significant. (and that “squaring the t to get the F” trick only works for the univariate regression case…)

Since the p-value for women does not reach the traditional significance level of 0.05, we might consider removing it from our model. Let’s see what happens then…

```
#run the model with only two predictors
prestige.mod2 <- lm(prestige ~ education + log2(income), data= Prestige)

# look at the results
summary(prestige.mod2)
```

```
##
## Call:
## lm(formula = prestige ~ education + log2(income), data = Prestige)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.0346  -4.5657  -0.1857   4.0577  18.1270
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -95.1940    10.9979  -8.656 9.27e-14 ***
## education      4.0020     0.3115  12.846  < 2e-16 ***
## log2(income)   7.9278     0.9961   7.959 2.94e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.145 on 99 degrees of freedom
## Multiple R-squared:  0.831,  Adjusted R-squared:  0.8275
## F-statistic: 243.3 on 2 and 99 DF,  p-value: < 2.2e-16
```

Hmmmm….education and log2(income) remain highly significant, there is little reduction in R-squared, the model is still significant.

We can compare the results of these two analyses a little more cleanly using the stargazer package as follows:

```
# compare the results of the two regression models
stargazer(prestige.mod1, prestige.mod2, title="Comparison of 2 Regression outputs",
          type = "html", align=TRUE)
```

Comparison of 2 Regression outputs

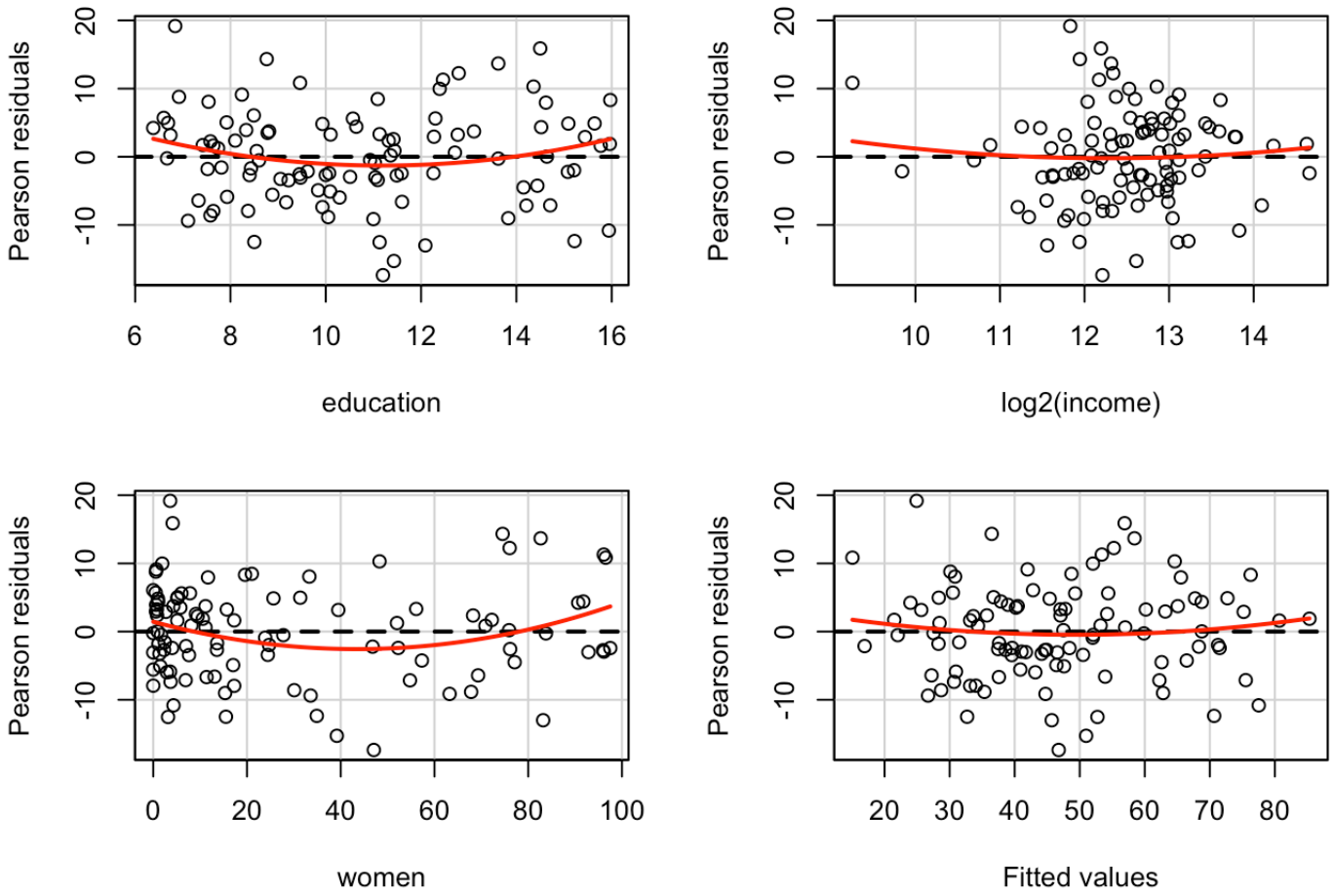
	Dependent variable:	
	prestige	
	(1)	(2)
education	3.731***	4.002***
	(0.354)	(0.312)
log2(income)	9.315***	7.928***
	(1.327)	(0.996)
women	0.047	
	(0.030)	
Constant	-110.966***	-95.194***
	(14.843)	(10.998)
Observations	102	102
R^2	0.835	0.831

Adjusted R^2	0.830	0.828
Residual Std. Error	7.093 (df = 98)	7.145 (df = 99)
F Statistic	165.428*** (df = 3; 98)	243.323*** (df = 2; 99)
Note:		*p<0.1;* ***p<0.05;*** p<0.01

We conclude that we lose little by eliminating a predictor variable. This is also consistent with a principle of parsimony, also known as the KISS principle, or “Keep It Simple, Silly”.

Let’s finish with some diagnostics. First the plots for the firs model with 3 independent variables:

```
# diagnostics for the first model with 3 independent variables
residualPlots(prestige.mod1)
```

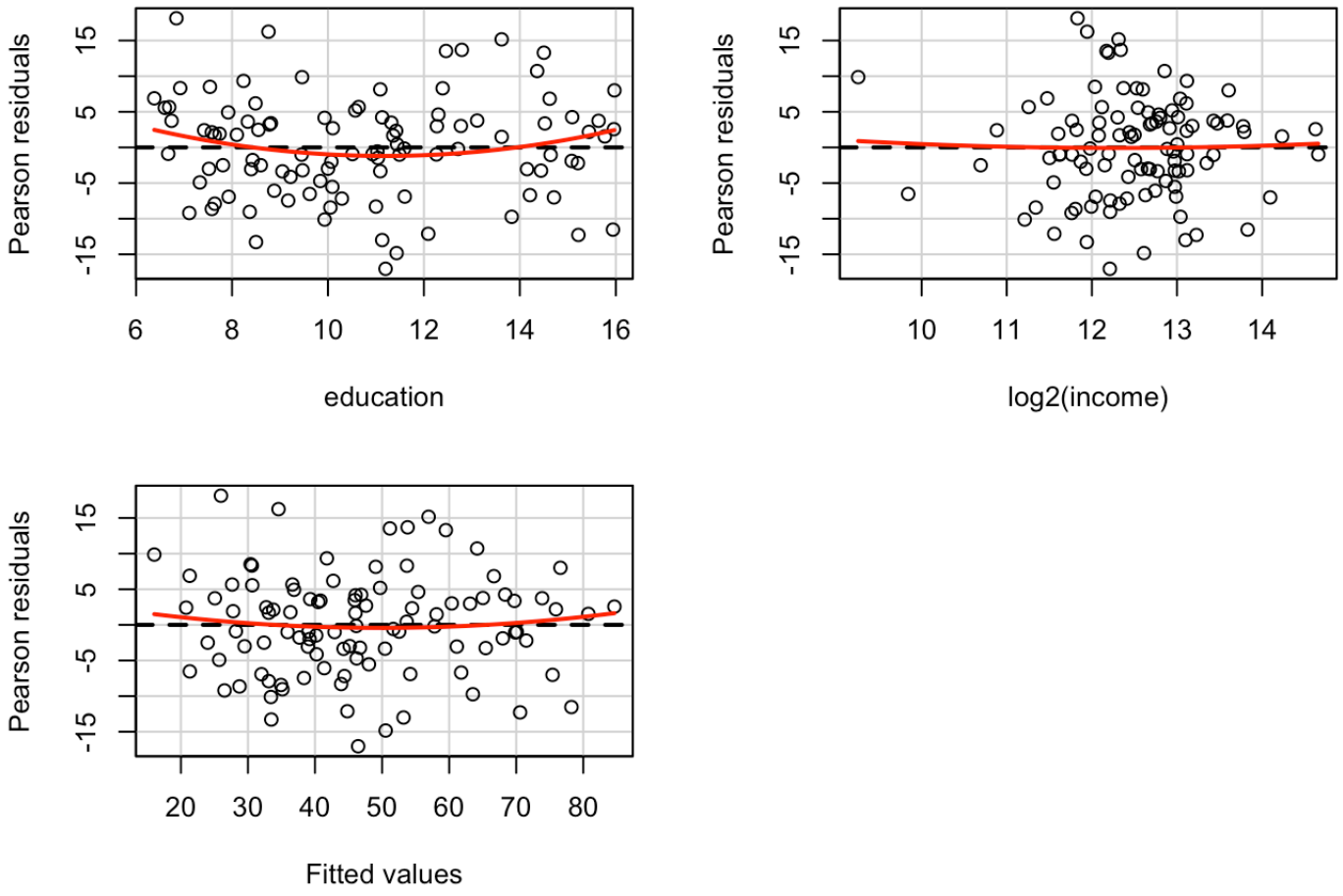



```
##                Test stat Pr(>|t|)
## education          1.756    0.082
## log2(income)       0.585    0.560
## women              2.277    0.025
## Tukey test         0.763    0.445
```

Notice that there is a non-zero trend for the variable women.

And now the plots for the second model with 2 independent variables

```
# diagnostics for the second model with 2 independent variables
residualPlots(prestige.mod2)
```

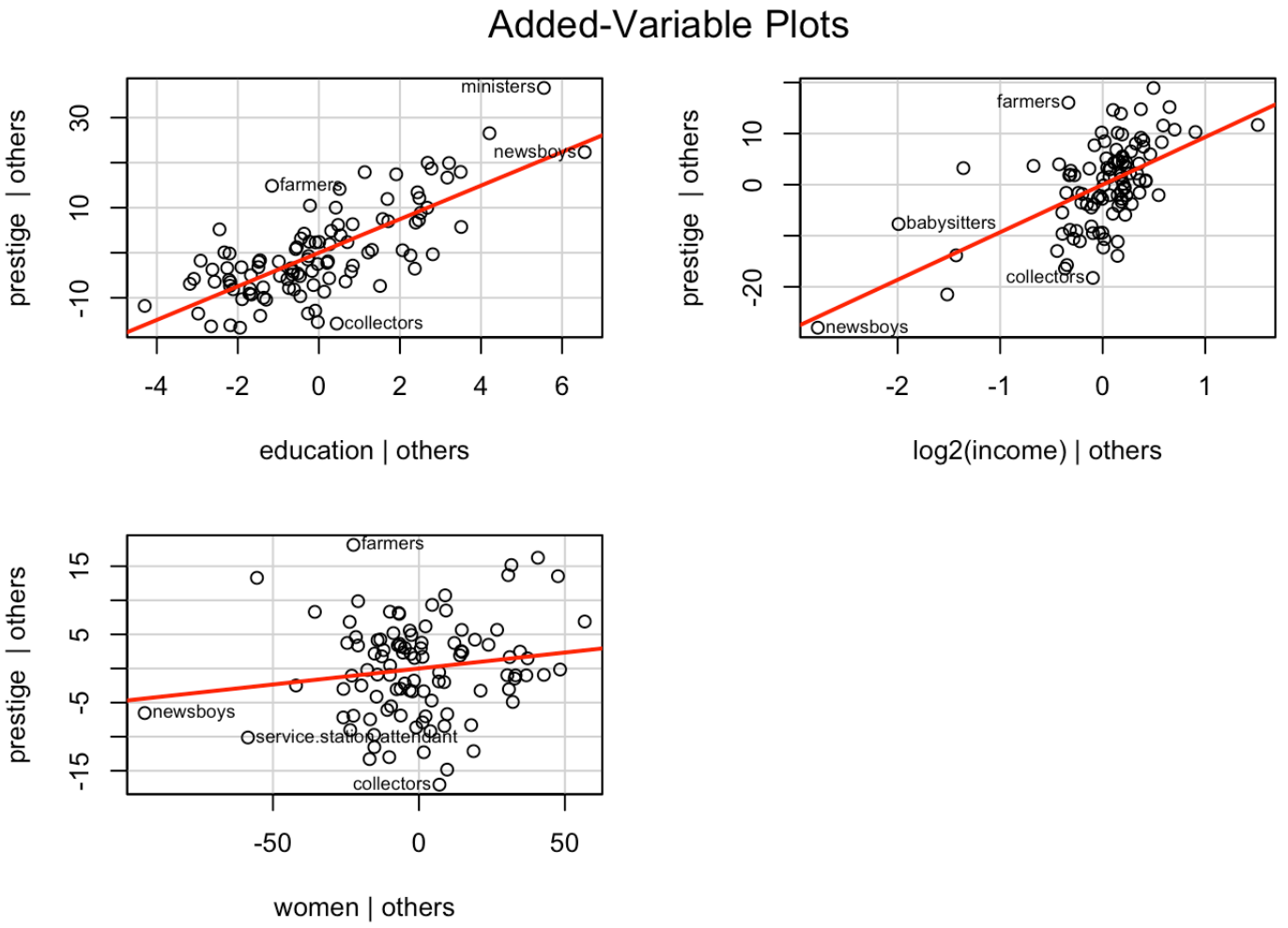



```
##              Test stat Pr(>|t|)
## education        1.615    0.109
## log2(income)     0.221    0.825
## Tukey test       0.653    0.514
```

Now the plots look really good, much better.

Another diagnostic tool is the added variable plot, that is the additional benefit of variable *i* given that all of the others are in. In this particular plot we can also identify the most influential observations.

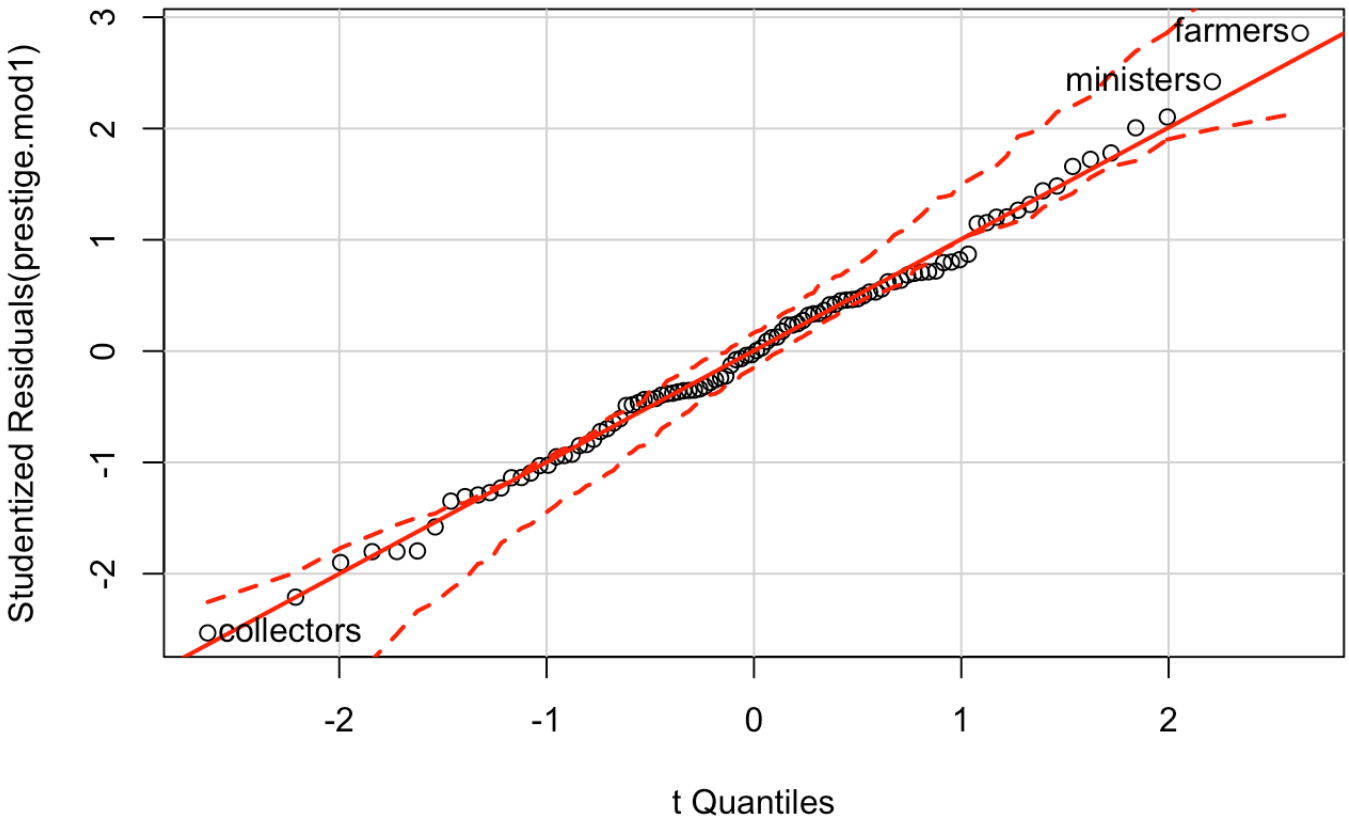
```
#added variable plots
avPlots(prestige.mod1, id.n=2, id.cex=0.7)
```




```
#id.n - identify n most influential observations
#id.cex - controls the size of the dot
```

Let's run the qq-plot:

```
# run the qq-plot
qqPlot(prestige.mod1, id.n=3)
```




```
## collectors  ministers    farmers
##          1        101        102
```

```
# here, id.n identifies the n observations with the largest residuals in absolute val
ue
```

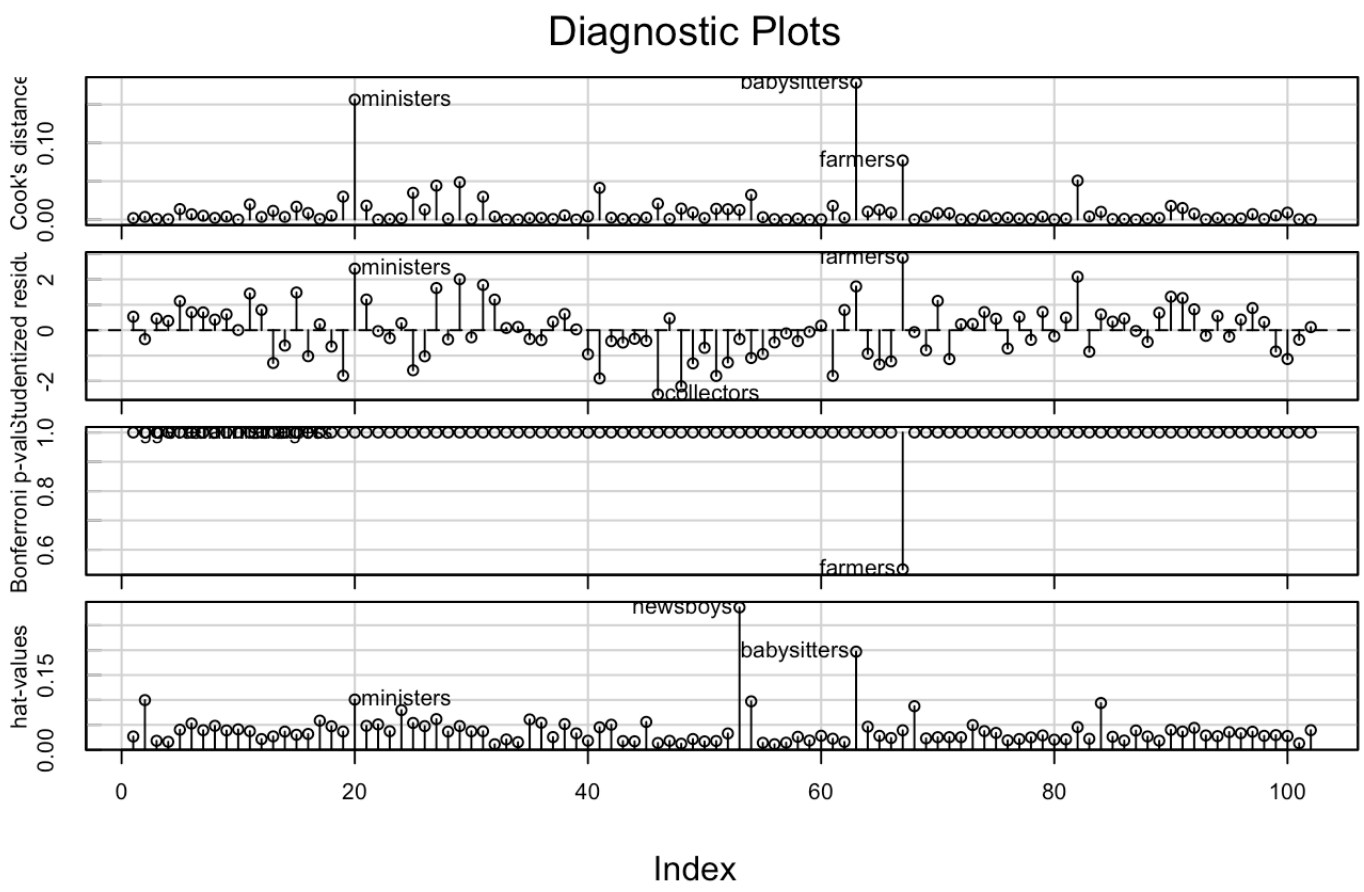
Are there any outliers?

```
#run Bonferroni test for outliers
outlierTest(prestige.mod1)
```

```
##
## No Studentized residuals with Bonferonni p < 0.05
## Largest |rstudent|:
##         rstudent unadjusted p-value Bonferonni p
## farmers 2.857462          0.0052259      0.53305
```

Are there any points that are of high influence?

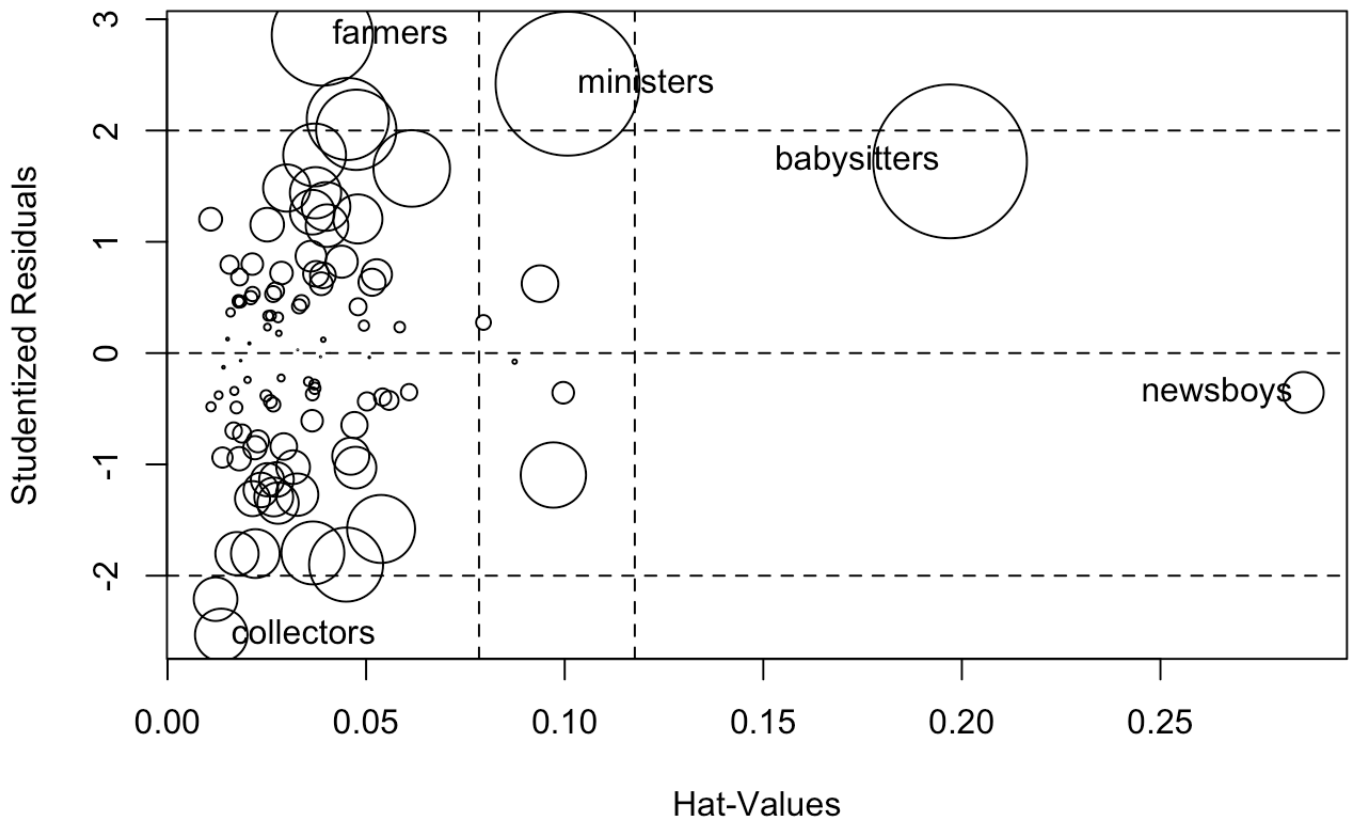
```
#identify highly influential points
influenceIndexPlot(prestige.mod1, id.n=3)
```



NB. If there are points that are a) outliers AND b) highly influential, these have potential to change the inference. You should consider removing them.

How do we make heads or tails out of the plots above? One way is with an influence plot.

```
#make influence plot
influencePlot(prestige.mod1, id.n=3)
```




```
##                StudRes        Hat      CookD
## ministers    2.4211475 0.10072159 0.15638036
## collectors  -2.5320314 0.01352116 0.02081914
## newsboys    -0.3534964 0.28595843 0.01262364
## babysitters  1.7227767 0.19703063 0.17848347
## farmers      2.8574618 0.03896096 0.07711591
```

Another diagnostic is to test for heteroskedasticity (i.e., the variance of the error term is not constant).

```
#test for heteroskedasticity
ncvTest(prestige.mod1)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.02841389    Df = 1     p = 0.8661394
```

We also want to look for multicollinearity, that is are some of our independent variables highly correlated. We do this by looking at the Variance Inflation Factor (VIF). A GVIF > 4 suggests collinearity.

```
vif(prestige.mod1)
```

```
##    education log2(income)        women
##     1.877097     2.572283     1.806431
```