Data Handling and Statistics - Computer Lab 2

Preparing your workfile

We add the basic libraries needed for this week's work:

```
library(tidyverse)    # for almost all data handling tasks
library(readxl)       # to import Excel data
library(ggplot2)      # to produce nice graphiscs
library(stargazer)    # to produce nice results tables
```

Introduction

For this Comuter lab we will work with the same datafile as we did for the 2nd Lecture. We will repeat some of the work done in the lecture with slight variations.

The example we are using here is taken from the CORE - Doing Economics resource. In particular we are using Project 8 which deals with international data on well-being. The data represent several waves of data from the European Value Study (EVS). A wave means that the same surevey is repeated at regular intervals (waves).

Aim of this lesson

In this lesson we will revise some hypothesis testing and basic (simple) regression analysis.

In terms of R skills you learn how to

- create summary tables using the `group_by`, `summarise` and `spread` commands
- create scatter plots using `ggplot` and `geom_point`
- conduct simple hypothesis tests on one or two sample means using `t.test`
- run simple regression models using `lm`

Importing Data

The data have been prepared as demonstrated in the Doing Economics Project 8, up to and including Walk-Through 8.3. Please have a look at this to understand the amount of data work required before an empirical analysis can begin. The datafile is saved as an R data structure (wb_data.Rdata) which is available from the Lecture 2 item on BB. Load this datafile into your work folder. Then start a new script file which you should save into the same folder (use a filename that does not contain any spaces!) and then ensure that you set, in your script file, the working directory to that folder by using the `setwd("PATH/TO/YOUR/FOLDER")`.

Details on the variables are avialable from here (Feb 2020: this page is temporarily unavailable, use the `wb_data_Des` object in the above datafile for basic variable info).

```
load("WBdata.Rdata")
str(wb_data)  # prints some basic info on variables
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    129515 obs. of  19 variables:
##  $ S002EVS    : chr  "1981-1984" "1981-1984" "1981-1984" "1981-1984" ...
##  $ S003       : chr  "Belgium" "Belgium" "Belgium" "Belgium" ...
```

```
##  $ S006       : num  1001 1002 1003 1004 1005 ...
##  $ A009       : num  3 5 2 5 5 5 5 5 4 4 ...
##  $ A170       : num  9 9 3 9 9 9 9 10 8 10 ...
##  $ C036       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ C037       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ C038       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ C039       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ C041       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ X001       : chr  "Male" "Male" "Male" "Female" ...
##  $ X003       : num  53 30 61 60 60 19 38 39 44 76 ...
##  $ X007       : chr  "Single/Never married" "Married" "Separated" "Married" ...
##  $ X011_01    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ X025A      : chr  NA NA NA NA ...
##  $ Education_1: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Education_2: chr  NA NA NA NA ...
##  $ X028       : chr  "Full time" "Full time" "Unemployed" "Housewife" ...
##  $ X047D      : num  NA NA NA NA NA NA NA NA NA NA ...
```

Checking your environment you will see two objects. Along the proper datafile (`wb_data`) you will find `wb_data_Des` which contains some information for each of the variables. It will help us to navigate the obscure variable names. Use the information in that object to understand what the variales `C038` and `C039` represent.

```
wb_data_Des[wb_data_Des$Names == "C038",]
```

Some initial data analysis and summary statistics

Let us investigate what the different categories of education status. Use the `count()` function. Look at the output to understand what it does or use `?count()` to call up the help.

```
wb_data %>% count(Education_1)
```

```
## # A tibble: 8 x 2
##   Education_1     n
##         <dbl> <int>
## 1           0  1541
## 2           1  4711
## 3           2  8149
## 4           3 20702
## 5           4  2872
## 6           5 11518
## 7           6   330
## 8          NA 79692
```

```
wb_data %>% count(Education_2)
```

```
## # A tibble: 8 x 2
##   Education_2                                                 n
##   <chr>                                                   <int>
## 1 <NA>                                                    79692
## 2 " (Upper) secondary education"                          20702
## 3 " First stage of tertiary education"                    11518
## 4 " Lower secondary or second stage of basic education"    8149
## 5 " Post-secondary non-tertiary education"                 2872
## 6 " Pre-primary education or none education"               1541
```

```
## 7 " Primary education or first stage of basic education"  4711
## 8 " Second stage of tertiary education"                    330
```

You can see that both variables `Education_1` and `Education_2` describe the same variable. The latter has short descriptions to the educational levels while the former represents these with numbers. You will notice that the ordering of `Education_2` is different to that in `Education_1`.

What is the variable type for `Education_2`? You may remember a command to do that or google. There are several ways to get that information.

In R it is best to deal with categorical variables like `Education_2` as factor variables rather than character variables. Change the variable type of `Education_2` to a factor variable.

```
wb_data$Education_2 <- as.factor(wb_data$Education_2)
wb_data %>% count(Education_2)
```

```
## Warning: Factor `Education_2` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## # A tibble: 8 x 2
##   Education_2                                              n
##   <fct>                                                <int>
## 1 " (Upper) secondary education"                       20702
## 2 " First stage of tertiary education"                 11518
## 3 " Lower secondary or second stage of basic education"  8149
## 4 " Post-secondary non-tertiary education"              2872
## 5 " Pre-primary education or none education"            1541
## 6 " Primary education or first stage of basic education"  4711
## 7 " Second stage of tertiary education"                  330
## 8 <NA>                                                 79692
```

As you can see in the above table, the different answer options for this variable are ordered according to the alphabet. That is the logical default in R. It would be nicer to see the order in terms of how much education they represent. We can re-order the outcomes. Google "r reorder factor levels" to find out how to achieve this. Note, "NA" is not a factor level (they indicate missing observations) and you can ignore it in the re-ordering.

```
wb_data$Education_2 <- factor(wb_data$Education_2,XXXX)
```

If you have done it correctly, you should be able to get the following result

```
wb_data %>% count(Education_2)
```

```
## Warning: Factor `Education_2` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## # A tibble: 8 x 2
##   Education_2                                              n
##   <fct>                                                <int>
## 1 " Pre-primary education or none education"            1541
## 2 " Primary education or first stage of basic education"  4711
## 3 " Lower secondary or second stage of basic education"  8149
## 4 " (Upper) secondary education"                       20702
## 5 " Post-secondary non-tertiary education"              2872
## 6 " First stage of tertiary education"                 11518
## 7 " Second stage of tertiary education"                  330
## 8 <NA>                                                 79692
```

Let's pick two countries (Germany and Turkey) for a particular year (`S002EVS == "2008-2010"`) and find out how many respondents fall into the different education categories. To do this we will resort to the powerful piping technique delievered through the functionality of the `tidyverse`

```
table1a <- wb_data %>%
            filter(S002EVS == "2008-2010") %>%      # select year
            filter(S003 %in% c("Germany","Turkey")) %>% # select countries
            group_by(Education_2,S003) %>%          # groups by Education and Country
            summarise(n = n()) %>%                  # summarises each group by calculating obs
            spread(S003,n) %>%                      # put Countries across columns
            print()
```

```
## # A tibble: 7 x 3
## # Groups:   Education_2 [7]
##   Education_2                                                Germany Turkey
##   <fct>                                                        <int>  <int>
## 1 " Pre-primary education or none education"                      NA    354
## 2 " Primary education or first stage of basic education"          34    888
## 3 " Lower secondary or second stage of basic education"          194    213
## 4 " (Upper) secondary education"                                1026    353
## 5 " Post-secondary non-tertiary education"                        54     14
## 6 " First stage of tertiary education"                           369    181
## 7 " Second stage of tertiary education"                            6      7
```

You can see that the distribution of highest educational achievement varies significantly between these two countries. If you wonder what the `spread(S003,n) %>%` part of the above code does, re-run the code without that line to see the difference.

It is important to realise that most things can be achieved in different ways (i.e. there is not one correct way of doing things but many!). Here is an alternative way. First we create the subsample we are interested in (`temp_data`), and then we apply the `table` function

```
temp_data <- wb_data %>%
            filter(S002EVS == "2008-2010") %>%      # select year
            filter(S003 %in% c("Germany","Turkey")) # select countries

table1b <- table(temp_data$Education_2,temp_data$S003) %>% print()
```

```
##
##                                                    Germany Turkey
##    Pre-primary education or none education               0    354
##    Primary education or first stage of basic education  34    888
##    Lower secondary or second stage of basic education  194    213
##    (Upper) secondary education                       1026    353
##    Post-secondary non-tertiary education               54     14
##    First stage of tertiary education                  369    181
##    Second stage of tertiary education                   6      7
```

Sometime you actually want proportions rather than counts. The easiest way to achieve this is by using the already existing `table1b` and send that through the `prop.table` function. Recall that you can call `?prop.table` from the Console/Command Window to get some help on that function.

```
prop.table(table1b)
```

```
##
##                                                      Germany
##    Pre-primary education or none education       0.000000000
##    Primary education or first stage of basic education 0.009206607
##    Lower secondary or second stage of basic education  0.052531817
##    (Upper) secondary education                       0.277822908
##    Post-secondary non-tertiary education             0.014622258
```

```
##    First stage of tertiary education                    0.099918765
##    Second stage of tertiary education                   0.001624695
##
##                                                            Turkey
##    Pre-primary education or none education             0.095857027
##    Primary education or first stage of basic education 0.240454915
##    Lower secondary or second stage of basic education  0.057676686
##    (Upper) secondary education                         0.095586244
##    Post-secondary non-tertiary education               0.003790956
##    First stage of tertiary education                   0.049011644
##    Second stage of tertiary education                  0.001895478
```

Perhaps you can see that all the proportions sum up to 1. But what we really want is proportions that sum up to 1 by country. (You can leave out the two `options` lines. Look at the difference!)

```
options(digits=2)  # digits=7 is the default
prop.table(table1b,margin = 2)
```

```
##
##                                                    Germany Turkey
##    Pre-primary education or none education           0.0000 0.1761
##    Primary education or first stage of basic education  0.0202 0.4418
##    Lower secondary or second stage of basic education   0.1153 0.1060
##    (Upper) secondary education                       0.6096 0.1756
##    Post-secondary non-tertiary education             0.0321 0.0070
##    First stage of tertiary education                 0.2193 0.0900
##    Second stage of tertiary education                0.0036 0.0035
```

```
options(digits=7) # reset to the default
```

See what happens if you set `margin = 1`.

To parctice you should create a table which shows the proportion of respondents answering 1 to 5 on question `C039` (Work is a duty to society), comparing Great Britain with France for the last available wave (`S002EVS == "2008-2010"`).

You've got it right if you find that 36% of French respondents strongly disagree with the statement "Work is a duty to society". Recall that 1 represents strong agreement and 5 strong disagreement with that statement. What do you learn about the French?!

An additional exercise is to see whether the answers to the life satisfaction question (`A170`) have changed through time. Create a table with proportions of responses to `A170` for Great Britain across all waves.

```
temp_data <- wb_data %>%
            filter(XXXX == XXXX) # select GB

table3 <- XXXX(temp_data$XXXX,XXXX$S002EVS)
prop.table(XXXX, XXXX)
```

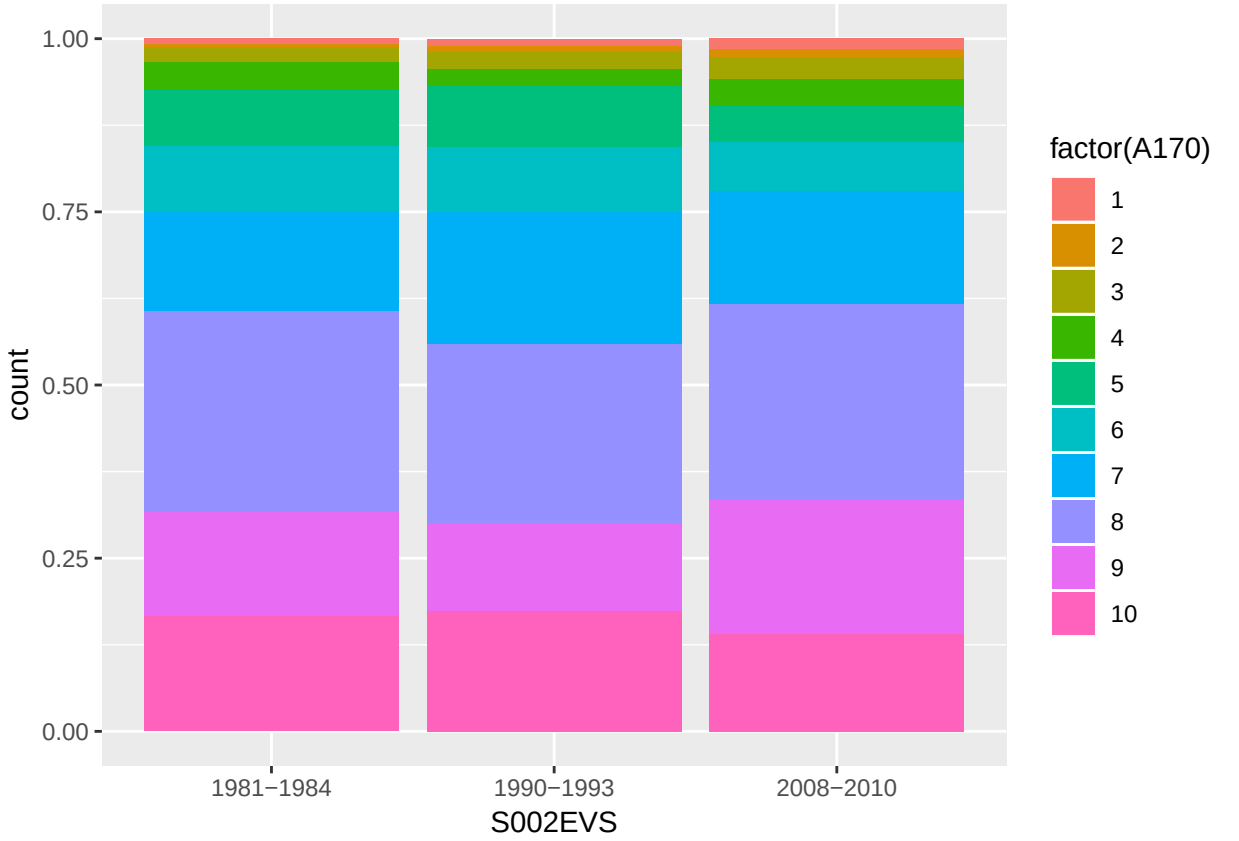
```
##
##        1981-1984   1990-1993   2008-2010
##   1  0.007805724 0.009595613 0.015045135
##   2  0.003469211 0.009595613 0.011033099
##   3  0.022549870 0.023303633 0.032096289
##   4  0.039895924 0.025359836 0.038114343
##   5  0.081526453 0.087731323 0.052156469
##   6  0.092801388 0.093899931 0.071213641
##   7  0.145706852 0.191226868 0.162487462
```

```
##    8  0.289679098 0.258396162 0.282848546
##    9  0.149176062 0.126113777 0.194583751
##    10 0.167389419 0.174777245 0.140421264
```

Have the responses to that question changed through time?

There are two ways how to make the result easier to look at. First you could put the command `options(digits=2)` before you print the table (see above). Try it and see what happens. Second, we could find a graphical representation.

```
ggplot(temp_data, aes(x=S002EVS, fill=factor(A170))) +
  geom_bar(position = 'fill')
```



To find this solution I googled "R ggplot geom_bar proportions". From this plot it is not obvious that there were significant changes in the distribution across time.

Hypothesis testing

Let's investigate whether there are differences in some of the responses between countries. In particular we shall test whether the proportion of respondents with any tertiary education (degree) is different in different countries.

To make this job as simple as possible we should first calculate a new variable in our dataset, which indicates whether a respondend has any tertiary education (`Education_2 %in% c(" First stage of tertiary education"," Second stage of tertiary education")`). There are several ways to do this, but here we use the `mutate` function in a pipe.

```
wb_data <- wb_data %>%
    mutate(grad = fct_recode(Education_2,
    "no degree"    = " Pre-primary education or none education",    # new level = old level
    "no degree"    = " Primary education or first stage of basic education",
    "no degree"    = " Lower secondary or second stage of basic education",
    "no degree"    = " (Upper) secondary education",
    "no degree"    = " Post-secondary non-tertiary education",
    "degree"  = " First stage of tertiary education",
    "degree" = " Second stage of tertiary education"))
```

Let's check that this did what we wanted.

```
table4 <- wb_data %>% count(grad) %>% print()
```

```
## Warning: Factor `grad` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## # A tibble: 3 x 2
##   grad          n
##   <fct>     <int>
## 1 no degree 37975
## 2 degree    11848
## 3 <NA>      79692
```

Yes, great! Let's create a similar table but for two countries

```
temp_data <- wb_data %>%
            filter(S003 %in% c("France","Spain")) # select France and Spain

table(temp_data$grad,temp_data$S003)
```

```
##
##             France Spain
##   no degree    921   754
##   degree       420   154
```

Note that I created another object here, `temp_data`, from which I then create the table. In fact we had earlier created an object with the same name. Here we are over-writing the earlier object with this new one. I often do this if I create an object which I need for one thing but not any longer afterwards.

Now we can feed this information into the `prop.test` function. How this works is that we feed in the two counts of successes (degree observation) and then the number of observations. By default `prop.test` will test that the two proportions are equal.

```
prop.test(c(420,154), c(1341, 908), alternative = "two.sided")
```

```
##
##  2-sample test for equality of proportions with continuity
##  correction
##
## data:  c(420, 154) out of c(1341, 908)
## X-squared = 57.977, df = 1, p-value = 2.652e-14
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  0.1078577 0.1793334
## sample estimates:
##    prop 1    prop 2
## 0.3131991 0.1696035
```

In the test output you get the two sample propportions (0.31 for Italy and 0.17 for Spain) and we get a very small p-value (p-value = 3e-14). This means that it is extremely unlikely that we would have received such different sample proportions if the null hypothesis of equal proportions had been correct and hence we reject the null hypothesis.

Repeat this analysis by testing whether the proportions of respondents with degrees is different in Denmark and Sweden.

```
temp_data <- wb_data %>%
            XXXX(XXXX %in% c("Denmark",XXXX)) # select France and Spain

table(XXXX$grad,XXXX$XXXX)
```

You got the correct frequencies if you find 405 Danish respondents with degree.

```
prop.test(c(XXXX,XXXX), XXXX, alternative = XXXX)
```

You should find a p-value for the test of 0.337. How do you interpret this?

If the null hypothesis was true there was a 34% probability to get two sample proportions as different or more different as the ones we see. At 30% we judge that this is quite likely (larger than α) and hence we do not reject the null hypothesis.

Regression Analysis

Let us estimate a simple regression model for all data from Great Britain.

$$A170 = \alpha + \beta\ X011\_01$$

where `A170` refers to the Life Satisfaction variable and `X011_01` to the number of children. We looked already at the Life Satisfaction variable. Let's first have a look at the number of children variable in Great Britain

```
wb_data_GB <- wb_data %>%  filter(S003 == "Great Britain")
table6 <- table(wb_data_GB$X011_01)
prop.table(table6)
```

```
##
##           0           1           2           3           4           5
## 0.236710130 0.177532598 0.334002006 0.167502508 0.051153460 0.020060181
##           6           7           8
## 0.006018054 0.005015045 0.002006018
```

As you can see we have a lot of 0s here (23.7% of the observations), i.e. about a third of respondents have no children.

We will shortly see that running such a regression has a lot of problematic issues, but the computer doesn't know that and will happily estimate such a regression model.

Now we run a regresison

```
mod1 <- lm(A170~X011_01,data=wb_data_GB)
stargazer(mod1, type="text")
```

```
##
## ===============================================
##                         Dependent variable:
##                     ---------------------------
```

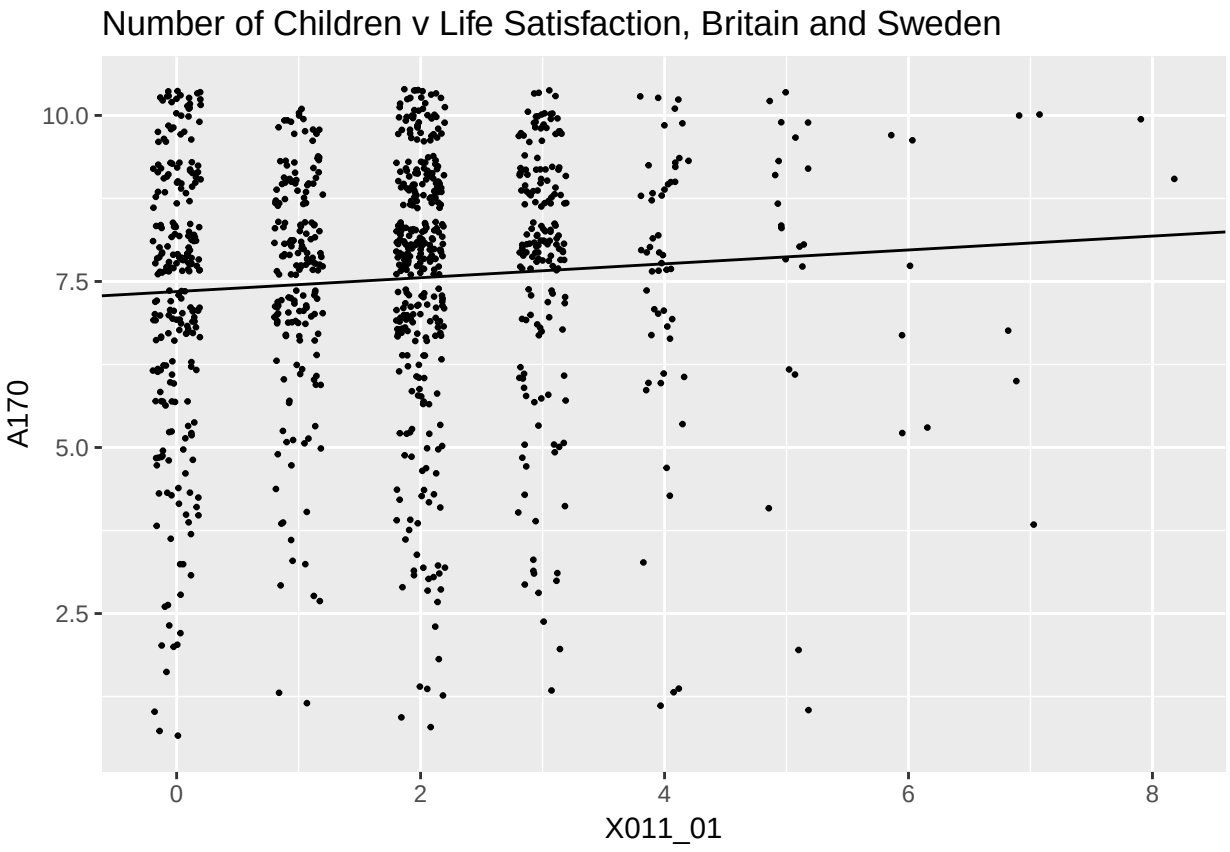
```
##                                 A170
## ------------------------------------------------
## X011_01                       0.104**
##                               (0.046)
##
## Constant                     7.348***
##                               (0.102)
##
## ------------------------------------------------
## Observations                    997
## R2                             0.005
## Adjusted R2                    0.004
## Residual Std. Error      1.996 (df = 995)
## F Statistic            5.192** (df = 1; 995)
## ================================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

Taken at face value this seems to suggest that "increasing" your number of children by one will, on average, increase your Life Satisfaction measure by 0.1.

Let's represent these data in a plot:

```
ggplot(wb_data_GB, aes(x=X011_01, y=A170)) +
    geom_jitter(width=0.2, size = 0.5) +    # Use jitter rather than point so we can see indiv obs
    geom_abline(intercept = mod1$coefficients[1], slope = mod1$coefficients[2])+
    ggtitle("Number of Children v Life Satisfaction, Britain and Sweden")
```

```
## Warning: Removed 2612 rows containing missing values (geom_point).
```



Note that we use `geom_jitter` rather than `geom_point`. This adds some random noise to the data so that we can see the individual observation (replace `geom_jitter(width=0.2)` with `geom_point()` to see the difference it makes). `geom_abline` adds a line. We specify the intercept and slope from our regression model (`mod1$coefficients[1]` and `mod1$coefficients[2]`). `ggtitle` adds the title to the graph.