Tutorial 7b: Correlation

Carlo Knotz

Table of contents

Introduction

The last two tutorials showed you how you can study relationships between two categorical variables (with the χ^2 test) and between one binomial variable and one metric or linear variable (with the t-test). What is left: How do you analyze relationships between *two continuous variables*?

The correct test is of course *Pearson's correlation coefficient* (as you know from Kellstedt and Whitten 2018). In this tutorial, you will learn how you can implement this in `R`.

The research question for this tutorial is again about religiosity, but we will now look at the effect of *age*. In other words, we will see if there is a correlation between how old a person is and how religious she feels.

And, as before, we will start directly with the full *ESS* dataset (round 7, 2014).

> **Tip**
>
> *Hvis du ønsker å lese en norsk tekst **i tillegg**: "Lær deg R", Kapittel 5.3.3*

Hypotheses

There is a debate among sociologists about whether societies are becoming less religious (or "secularized"). Some argue that this indeed the case, while others are a bit more sceptical.[1] Those who support the idea that religion becomes less important in people's lives point to *generational replacement* as a mechanism: Older generations are typically more religious, but this is less true for younger generations. As older generations die and newer are born, society overall becomes less religious.[2]

If the generational-replacement hypothesis is true, then we should be able to see that younger people are less religious than older people — or a *positive correlation* between religiosity and age: The higher one's age, the more religious they are. This is the hypothesis we will test here.

The corresponding null hypothesis is then: There is no relationship between age and religiosity, or the correlation between age and religiosity is zero.

[1] Hout, M. and Fischer, C. S. (2014). Explaining why more Americans have no religious preference: Political backlash and generational succession, 1987-2012. *Sociological Science*, 1:423–447; Marwell, G. and Demerath, N. (2003). "Secularization" by any other name. *American Sociological Review*, 68(2):314–316.

[2] See also Inglehart, R. (1971). The silent revolution in Europe: Intergenerational change in post-industrial societies. *American Political Science Review*, 65(4):991–1017.

Setup, data import and management

You should at this point know quite well how you import data from the *ESS* and prepare them for analysis.

Setup

As before, you need the `tidyverse` for your data management and visualization:

```r
library(tidyverse)
```

Data import and management

You import the dataset with `read_dta()` from `haven`:

```
ess7 <- haven::read_dta("ess7.dta")
```

Then you need to covert it to the regular format with `labelled::unlabelled()`:

```
ess7 <- labelled::unlabelled(ess7)
```

We will again work only with the data from Norway, and the following variables:

- `essround`, `cntry`, and `idno`;
- `agea`, the respondent's age in years;
- `rlgdgr`, the variable measuring how religious each respondent feels on a scale from 0 to 10;

This means, you apply the familiar "trimming" operations:

```
ess7 %>%
  filter(cntry=="NO") %>%
  select(essround,cntry,idno,agea,rlgdgr) -> ess7
```

Data cleaning

You probably also remember from the previous tutorial that the `rlgdgr`-variable is stored as a *factor*:

```
attributes(ess7$rlgdgr)
## $levels
##  [1] "Not at all religious" "1"                    "2"
##  [4] "3"                    "4"                    "5"
##  [7] "6"                    "7"                    "8"
## [10] "9"                    "Very religious"
##
## $label
## [1] "How religious are you"
##
## $class
## [1] "factor"
```

This means you have to convert it to numeric with `as.numeric()` while subtracting 1 to account for the divergence between labels and numerical values:

```
ess7$relig_num <- as.numeric(ess7$rlgdgr) - 1
```

And again, you can use this opportunity to give the new numeric version a name that is easier to type.

The situation is easier when we look at `agea`:

```
class(ess7$agea)
## [1] "numeric"
```

It is already stored as `numeric` — and therefore ready to go. Still, it makes sense to look at a quick summary to get a sense of how the variable is distributed:

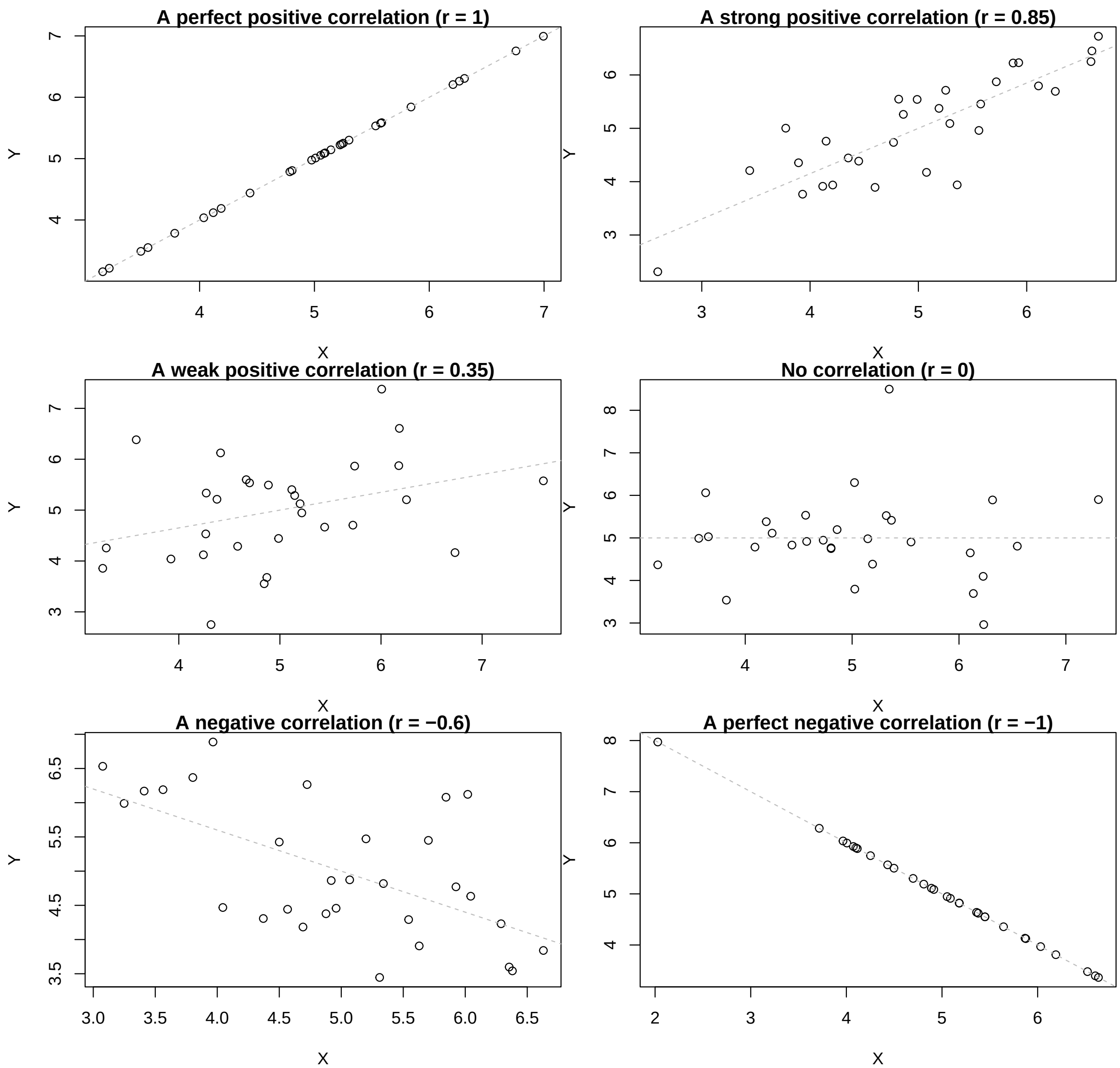
```
summary(ess7$agea)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.00   32.00   47.00   46.77   61.25  104.00
```

The lowest age recorded is 15 years, while the oldest is 104 years. The average age in the sample is 46.7, and 75% of the respondents are 61.25 years old or younger (see `3rd Qu.`).

Statistical analysis

We will jump right into the statistical analysis (but you can learn how to visualize the relationship between the two variables in the voluntary part below).

As mentioned in the Introduction, the appropriate measure for the strength of a relationship between two numeric variables is Pearson's correlation coefficient (r), which can range between -1 and 1. Negative values of r reflect *negative* relationships (more X, less Y) while positive values of r indicate *positive* relationships (more X, more Y). An r of 0 indicates that the two variables are not correlated—that they are unrelated to each other. This is also illustrated in the six graphs below.



A perfect positive correlation (r = 1)
A strong positive correlation (r = 0.85)
A weak positive correlation (r = 0.35)
No correlation (r = 0)
A negative correlation (r = −0.6)
A perfect negative correlation (r = −1)
X
Y

The `cor()` function

There are two ways to let `R` calculate the correlation coefficient. The first option is to only get the correlation coefficient by itself, without any additional statistical tests. To do this, you use the `cor()` function.

The `cor()` function is fairly easy to use:

1. You need to state the two variables that you want to correlate (as `x` and `y`);
2. You also need to tell `R` what to do with missing observations; you do this by specifying the `use` option. In this case, we tell `R` to use only non-missing observations (`complete.obs`). This is equivalent to using `na.rm` in the `mean()` or `sd()` functions.

```
cor(x = ess7$agea, y = ess7$relig_num,
    use = "complete.obs")
## [1] 0.219565
```

We get a correlation coefficient of 0.22. This means `agea` and `relig_num` are *weakly positively* correlated. This speaks for our hypothesis in general — but the relationship is not very strong.

Significance testing with `cor.test()`

Now that we know that there is a positive association between age and religiosity, the next question is: *is this association statistically significant?* In other words, is our finding only the result of random sampling variation, or is there really a systematic relationship between age and religiosity in the underlying population?

The `cor.test()` function allows you to run the significance test that is explained in Kellstedt and Whitten (2018, 183-184). To use this function, you just specify the two variables you want to use. It will take care of the `NA`s by itself.

> **!** There are different ways in which you can use this function:
>
> - By default, the function tests if the correlation coefficient is *different from 0* or not. In other words, it tests if the two variables are associated *in some way*, be it positive or negative. (This is also the procedure that is described in Kellstedt and Whitten).
> - You can also let the function test more specifically if the correlation coefficient is *larger than 0*—if there is a positive correlation. To do that, you set the `alternative` option to `"greater"`.
> - And you can test specifically if the correlation coefficient is *smaller than 0*—if there is a negative relationship. In that case, you set `alternative="less"`.

Testing if the true *r* is equal to 0 or not (default setting)

For example, to test if there is *any* relationship, positive or negative, between `agea` and `relig_num`, you would just run the following:

```
cor.test(x = ess7$agea, y = ess7$relig_num)
##
##  Pearson's product-moment correlation
##
## data:  ess7$agea and ess7$relig_num
## t = 8.5076, df = 1429, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1696758 0.2683316
## sample estimates:
##       cor
## 0.219565
```

If you take a quick look at the output, you see under `alternative` what `R` has been testing:

- The *alternative* hypothesis is that the true r **is not equal** to 0.
- The corresponding *null* hypothesis (which is not specifically mentioned in the output) is then that the true r **is equal** to 0.

And, since you have now learned about two other statistical tests and their interpretation, you should know what to look for next: the p-value! Based on this result, can we reject the null hypothesis (see also Kellstedt/Whitten 2018, Chapter 8)? As in the previous tutorial on the t-test, you can use `format.pval()` to translate the p-value into something that is a bit easier to interpret:

```
format.pval(2.2e-16, digits = 3, eps = 0.01)
```

```
[1] "<0.01"
```

As in the previous tutorial, the result is a very small number – smaller than 0.01.

You can read the other results as follows:

- `sample estimates:` shows you the sample correlation coefficient, which you already know from earlier;
- `95 percent confidence interval:` is obviously the 95% confidence interval for the correlation coefficient.

- `t` is the t statistic. As explained in Kellstedt and Whitten (2018, 183), we use the t-test in this case to test if the correlation coefficient is equal to 0 or not;
- `df` are the degrees of freedom (see Kellstedt and Whitten 2018, 183);

Testing if the the true *r* is greater than 0

But we actually started from the more specific hypothesis that there is a *positive* relationship between `age` and `relig_num`—that r should be greater than 0.

To test if this is true, we need to run:

```
cor.test(x = ess7$agea, y = ess7$relig_num,
         alternative = "greater")
##
##  Pearson's product-moment correlation
##
## data:  ess7$agea and ess7$relig_num
## t = 8.5076, df = 1429, p-value < 2.2e-16
## alternative hypothesis: true correlation is greater than 0
## 95 percent confidence interval:
##  0.1777628 1.0000000
## sample estimates:
##      cor
## 0.219565
```

And if you take another quick look at the output, you see that `R` has been testing a different hypothesis now:

- The *alternative* hypothesis is now that the true r (in the population) is greater than 0.
- The corresponding *null* hypothesis is that the true r is equal to or smaller than 0.

As before, can we reject the null hypothesis here – and what would that mean substantively?

Testing if the true *r* is smaller than 0

Finally, let's see what happens if we test if the true r is smaller than 0. In this case, the hypotheses are as follows:

- The *alternative* hypothesis is that the true r is smaller than 0.
- The corresponding *null* hypothesis is that the true r is equal to or larger than 0.

To test this, we set `alternative="less"`:

```
cor.test(x = ess7$agea, y = ess7$relig_num,
         alternative = "less")
##
##  Pearson's product-moment correlation
##
## data:  ess7$agea and ess7$relig_num
## t = 8.5076, df = 1429, p-value = 1
## alternative hypothesis: true correlation is less than 0
## 95 percent confidence interval:
##  -1.0000000  0.2605762
## sample estimates:
##      cor
## 0.219565
```

Do you see the difference to the earlier results? Can you figure out what this means?

Conclusion

This was the last of the three tutorials on bivariate statistical tests and how you implement them in `R`. You now now how to look for relationships between different types of variables:

- Two categorical variables with the χ^2 test.
- One binominal and one continuous variable with the difference-of-means t-test.
- Two continuous variables with the Pearson correlation coefficient (r).

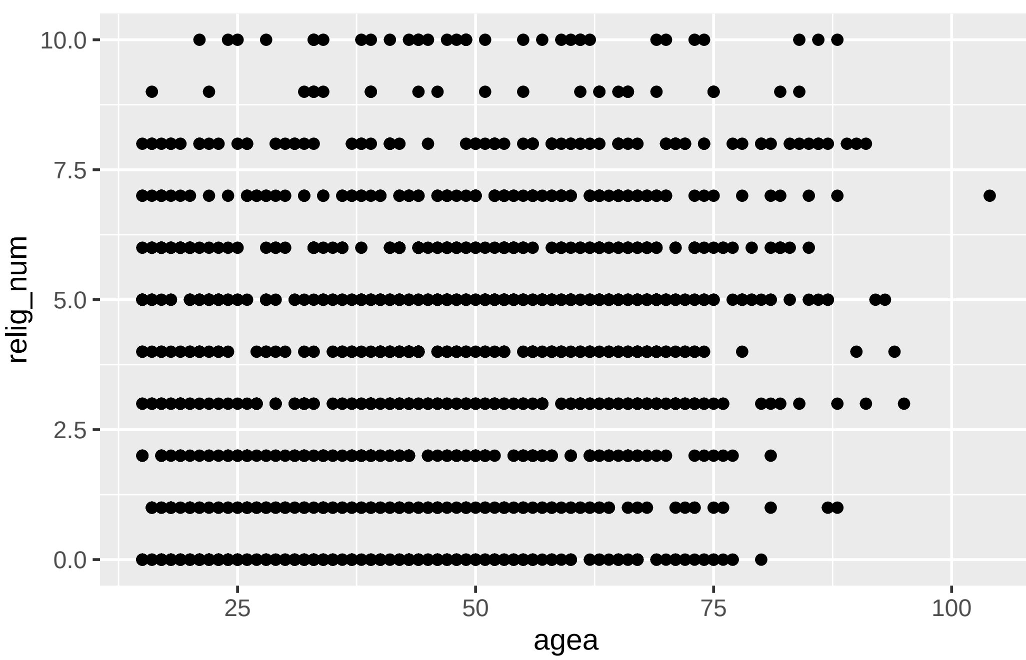
As before, there are de-bugging exercises you can do to practice more. You know the drill by now…this time: the correlation between body height and body weight.

(Voluntary): Visualization

You may be wondering: The previous two tutorials included a part on descriptive statistics before showing how to run the formal statistical tests, but not this one — why?! Perhaps you also remember that it is fairly easy to visualize the relationship between two continuous variables with a scatter plot (i.e., with `geom_point()`).

And this is true, in principle, but the data we are working here make a descriptive analysis a bit complicated. Consider what happens when we try to visualize the data with a scatterplot:

```
ess7 %>%
  ggplot(aes(x = agea, y = relig_num)) +
    geom_point()
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



You can see clearly that, well, you cannot really see anything clearly.

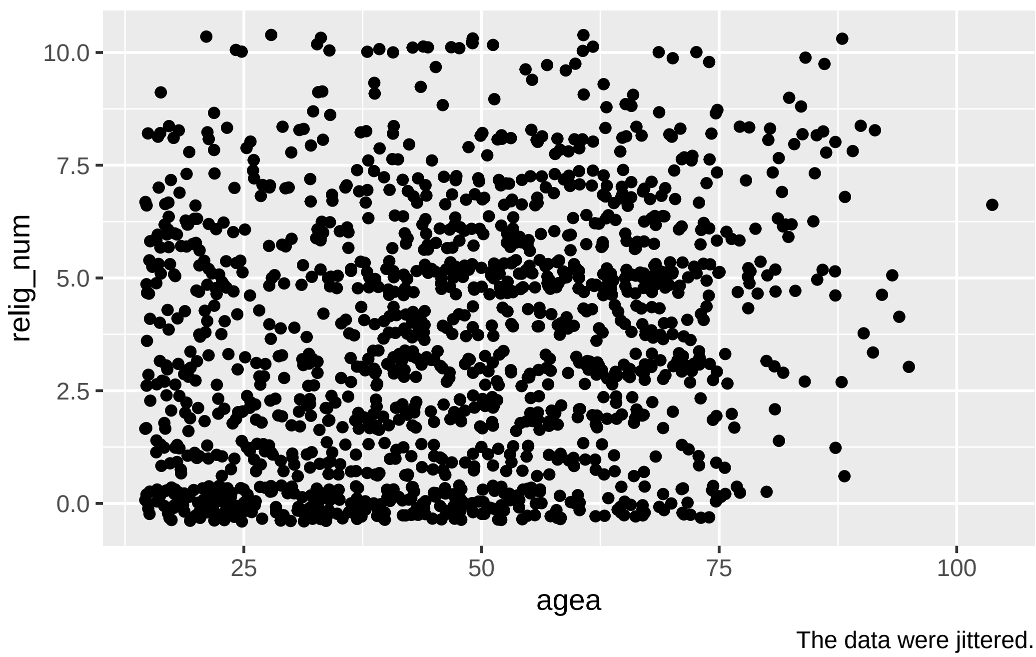
The problem: While `agea` is a true metric variable that can have any value between 0 and somewhere above 100, this is not the case for the `relig_num` variable. This variable can only have whole numbers between 0 and 10 (1,2,3,...) but not any numbers in between (e.g., 1.2234, 6.2562). This means that the observations on the `relig_num` variable are "binned" into whole numbers — and this is directly visible in the graph. But this makes it very difficult to see where in the graph most observations are, and if there is any association in the data.

`geom_jitter()`

One thing you can do in this case is to "jitter" the data. "Jittering" means that you gently shake the data out of their discrete categories. In technical terms, you add a bit of random noise to the data.

To "jitter" the data in the graph, you just replace `geom_point()` with `geom_jitter()`:

```
ess7 %>%
  ggplot(aes(x = agea, y= relig_num)) +
    geom_jitter() +
    labs(caption = "The data were jittered.")
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

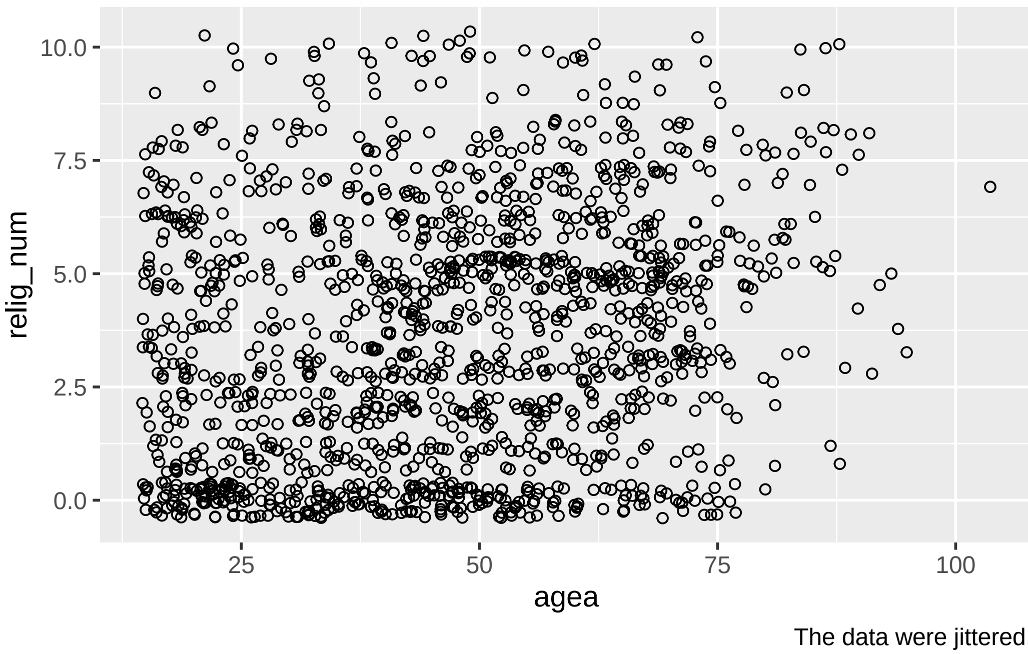


Now it is *a bit* easier to see where most of the observations are. Important: If you jitter your data, you need to tell your readers (e.g., by including a note with the `caption` option).

Still, the graph is hard to read and it is not clear what, if any, relationship there is between the data.

You can make it a bit easier to read still by changing the shape of the points with the `shape` option:

```
ess7 %>%
  ggplot(aes(x = agea, y = relig_num)) +
    geom_jitter(shape = 1) +
    labs(caption = "The data were jittered.")
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



`shape = 1` means: "Use hollow black rings".[3] But again, if there is any pattern in the data then it is still difficult to see.

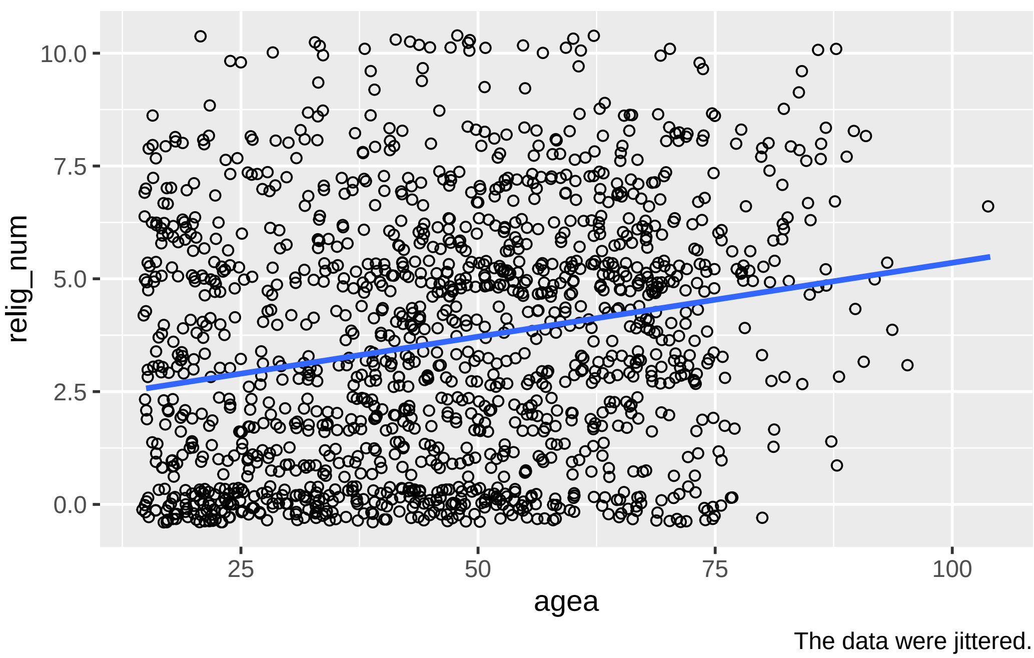
[3]See also http://www.sthda.com/english/wiki/ggplot2-point-shapes

geom_smooth()

What you can do now is to add a "fitted line" to the graph. This means that R estimates a linear (i.e., OLS) regression (which you will learn about in the next weeks) and plots the predicted or "fitted" values from this estimation to the graph. This can show you more clearly any patterns in the data.

To add such a fitted line, you use `geom_smooth()`. Importantly, you also need to specify that you want this line to be based on a linear model (`method = "lm"`).[4] In this case, we also turn off the confidence intervals (`se = FALSE`), but you can of course show these if you like!

```
ess7 %>%
  ggplot(aes(x = agea, y = relig_num)) +
    geom_jitter(shape = 1) +
    geom_smooth(method = "lm", se = FALSE) +
    labs(caption = "The data were jittered.")
## `geom_smooth()` using formula = 'y ~ x'
## Warning: Removed 5 rows containing non-finite outside the scale range
## (`stat_smooth()`).
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

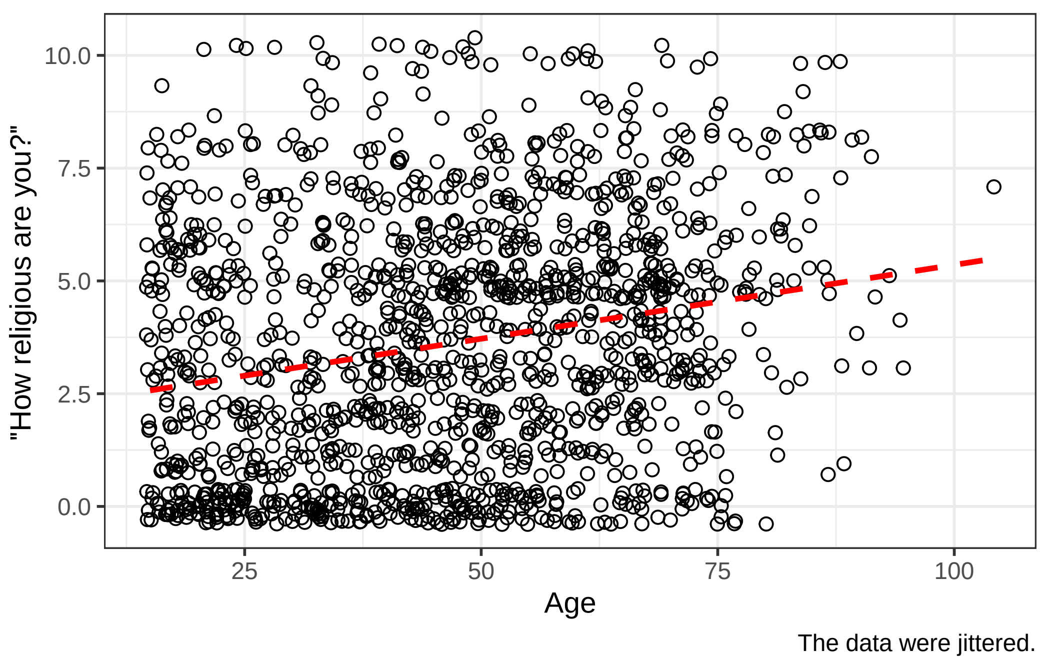


[4]Otherwise, R will use a fancy "smoothed" regression estimation procedure, which usually produces a very pretty result—but which is also difficult to understand. It is often better to keep it simple and just use the OLS method, which most people understand.

And now you can clearly see the *positive* association between the two variables: As age (`agea`) increases, religiosity (`relig_num`) increases as well!

With some more polishing, we get a decent publication-quality graph:

```
ess7 %>%
  ggplot(aes(x = agea, y = relig_num)) +
    geom_jitter(size = 2, shape = 1) +
    geom_smooth(method = "lm", se = FALSE, color = "red", linetype = "dashed") +
    labs(x = "Age", y = "''How religious are you?''",
         caption = "The data were jittered.") +
    theme_bw()
## `geom_smooth()` using formula = 'y ~ x'
## Warning: Removed 5 rows containing non-finite outside the scale range
## (`stat_smooth()`).
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Note that we made the points a bit larger (`size = 2`) and changed the color and type of the fitted line.