Earthquakes

Joris Vincent, Lucie Kattenbroek, Guillermo Aguilar

May 5, 2021

Contents

1 Research question

For this pipeline we're interested in *is there a relation between the depth and the magnitude of earthquakes? How often do earthquakes of different magnitudes happen?*

Exercise 1: Draft a pipeline

Think about what type of steps you'd need to do for this. How do you want to visualise this?
(The answer to the above exercise is below)

There are, as always, many ways to do this. In this pipeline, we're going to take the following approach. To investigate the relationship between magnitude and depth of an earthquake, we will make a scatter plot with magnitude on one axis and depth on the other. To look at the number of quakes per magnitude, we will present the answer per (rounded) magnitude: so we're only going to look at integer magnitudes by rounding the magnitude to the nearest integers. Then we will count the number of instances of quakes per magnitude, and present that as a bar graph.

2 Operationalization

Exercise 2: Finding data

What kind of dataset will we need for this? What would be the rows, what would be the variables? Are there some additional issues that we might run into?

Exercise 3: Get trade data analysis repository

Fork the gitlab repository earthquakes to your own gitlab account.

- This repository contains a subdirectory `data` with the `earthquakes.csv` there.
- It also contains an empty `earthquakes.R` script

Clone the repository your local machine, pull.

Exercise 4: Intuition

Before you even open any programming-related software, start by answering some basic questions about your data. Do you know the file type? Do you know how to read the data into R?

3 Exploratory data analysis

A basic data analysis pipeline roughly follows these steps:

1. **Setting up** the environment
2. **Reading** the data into the environment
3. **Exploring the data** contents to get a feel for what you can expect from this data
4. **Cleaning and tidying the data**, if necessary
5. **Munging** (changing the structure of the) data
6. **Analysing** the data to (visually) summarise pattern(s) in it

3.1 Reading the data

Exercise 5: Read in the data

Read the data into R. If you're happy with how it has loaded in, create a script, and put the loading of the dataset in the script. Commit the script to git. *Make sure to test your script in a clean instance of R before commmitting to git.*

3.2 Exploring data

Now you've stored the data, it's time to check whether everything went according to plan.

Exercise 6: Exploring the data

What does the dataset look like? Do you understand approximately what's in there? Do you understand what the columns mean? Does it look clean? Does it look tidy?

Exercise 7

Which columns do we need to answer our research question?

3.3 Analysing

The first thing we want to do, is look at depth and magnitude of earthquakes.

3.3.1 Visualise depth and magnitude

You've already learned how to make a scatterplot, in the previous pipeline. Think about what the data needs to look like, so that we can plot the depth and magnitude of these earthquakes. Are any transformations necessary?

Exercise 8: Make a plot

Make a scatter plot of the two desired variables of our data about earth quakes.

3.3.2 Interpreting the visualisation

Exercise 9: Assess the results

Look at your graph. Describe the features of the graph. Can you think of an explanation for the pattern that becomes clear? There's no right answer there - just your best guess, as the person analysing the data.

Exercise 10: Commit the visualisation

Now that you have a visualisation, and you have a sense of what it shows, it's time to add it to our repository. Add the code for your visualisation to your script. Make sure to add some labeling as well. Commit the script to your repository.

3.4 Quakes per magnitude

That's the first part done! For the other part to our research question we want to look at the number of quakes of different magnitudes. We said that to look at the number of quakes per magnitude, we will present the answer per (rounded) magnitude: so we're only going to look at integer (whole number) magnitudes by rounding the magnitude to the nearest integers. Then we will count the number of instances of quakes per magnitude, and present that as a bar graph.

The way we will build this is that we'll first create a simple summary counting all quakes. Then we will have to run this summary, grouping the quakes by magnitude. But in order to group by magnitude, we first have to round to the nearest integer.

3.4.1 A simple summary

Let's start with first summarising to one `n_quakes`, which contains the total number of earth quakes in the whole data frame. It's always easier to start with some simple analysis, to make sure we don't run into unexpected isuses.

Exercise 11: Summarising a data frame

Create a summary with one column called `n_quakes`, which has exactly one row that is the number of earthquakes in the data frame.

3.4.2 Rounding the magnitudes

Next, we have to actually round the magnitudes to a full integer. You can use the function `round()` to do so.

Exercise 12: Self-sufficient learning

Figure out what `round()` does, and how the syntax works.

Exercise 13: Rounding the magnitude

Add a column `magnitude_rounded`, which contains the magnitude for each quake, rounded to the nearest integer.

Exercise 14: Grouped summary

Re-run your summary, but this time count the number of quakes per (rounded) magnitude)

3.4.3 Visualising distribution of magnitudes

Now that we have the number of quakes per magnitude, let's visualise this.

Exercise 15: Conceptualise the visualisation

What kind of visual would you pick for think? Thinking in the Grammar of Graphics, what kind of geometry would be useful?

I'm picking a bar graph! This is because they're easy and clean and very multipurpose.

Exercise 16: Making a bar plot

Using a bar plot, visualise the data. You can do this either based on the summary dataframe that you created. Alternatively, you can use `stat = "count"` and base the figure directly on the raw data – in this case you do not have to specify a `y` aesthetic mapping.

3.4.4 Interpreting the visualisation

Exercise 17: Assess the results

Look at your graph. Describe the features of the graph. Can you think of an explanation for the pattern that becomes clear? There's no right answer there - just your best guess, as the person analysing the data.

Exercise 18: Commit

Once you're happy with your visualisation, and you know how to interpret it, commit it to your repository. *You've added labeling, right?*

3.5 Munging/cleaning

One thing that we haven't addressed, is that there are different types of earthquakes in this dataset. While most are various naturally ocurring earthquakes, a few are the product of testing (nuclear) explosions underground. Including those in our analysis might not be the best idea; both for the magnitudes, they are probably limited to a specific range, and for the depth that is probably also true.

To exclude them from our analysis, we will have to filter our dataset. The `filter()` function (also in the tidyverse), can help us with that. It takes in a dataframe, and returns a dataframe with only those rows for which a given expression is true:

```
filter(df, foo != "bar")
```

returns a copy of the dataframe `df`, but only those rows for which the value of `foo` is *not* the text `"bar"`. The rest (i.e., the rows for which `foo == bar`) have been removed from the copy of the dataframe.

Exercise 19

Using the `filter` function, remove the earthquakes that are caused by explosions. Re-create your figures without these earthquakes.

Done

In this pipeline, you've gotten some more practice with building a whole pipeline. You have seen that there are a lot of recurring steps, like loading in the data, that are just good to have lots of practice in. In addition, we've introduced the `filter()` function, which is another core feature of data analysis using the tidyverse. As always, we've built up this pipeline in iterative manner, committing the functional inbetween steps to our version history.