Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Master in Informatics Engineering

Data Visualization

Project: Shiny app

Autores:
Alexander Sandström
Karl Rosengren
Eliam Kourie

Index

1 Introduction

Since the start of the pandemic the world has found itself in a state of uncertainty and turmoil, with drastic measures and restrictions having a profound impact on the daily life of billions of people around the globe. During the pandemic, the interest in its different metrics has grown outside the sphere of researchers and decision makers, becoming a topic of discussion and interest among many ordinary groups of citizens. For that reason, the tools that determine how data is shown and interpreted has become more important than ever.

Our dataset contains 67 different variables related to Covid-19, and our goal has been to create a tool that can both encompass the grand scales of that dataset, allowing insight and discovery of data that span over two years of time and more than 200 countries. To achieve this we are using a choropleth map that covers the entire world, showing how countries compare on a dataset variable of the users choosing. To allow for more fine-grained comparison the map can be zoomed into the different continents and to inspect the variables more closely, we have included two graphs: a line plot and a scatter plot .

2 Problem characterizations

Due to the wide dimensions of the dataset it could be used to answer queries of an analyst comparing data between countries, searching for regional/national trends or finding correlation between variables in the dataset. An example use case could be to look for patterns with which the virus is spreading or the relationship between death toll and proportion of the population that are vaccinated.

2.1 Questions posed

To concretize the problem characterization we pose a set of questions that we would like to answer with the help of the available dataset.

1. How does the development of <quantitative variable> compare in the world over time?
2. How does <country> and <country> compare in the quantitative <variable>?
3. How does the quantitative <variable> correlate with the quantitative <variable> in <country>?

2.2 Detailed description of the dataset

Our data (*Coronavirus Pandemic (COVID-19)*, 2020) is retrieved from the web page *Our World in Data*[1] which is run by the organization *Global Change Data Lab*[2], based in the UK. The data contains 156.546 entries, which span 752 days and 232 countries. The data is aggregated from

[1] https://ourworldindata.org/
[2] https://global-change-data-lab.org/

a large number of actors[3] consisting of health departments as well as organizations such as the WHO[4] and the ECDC[5].

2.3 Choosing variables of interest in the dataset

The dataset provides a lot of variables, many of which are not relevant to the questions that we aim to be able to answer a user wielding our application. Using our questions as delimiters for which variables that are relevant, we have taken out a list the variables that can be chosen by the user:

- **iso_code:** Standard defining codes for the names of countries, dependent territories, and special areas of geographical interest, for example “AFG” which stands for “Afghanistan”**.**
- **continent:** The continent of the entry, for example “Africa”.
- **location:** The country of the entry, for example “Afghanistan”.
- **date:** The date of the entry, for example “2020-01-01”.
- **total_cases_per_million:** The total number of cases per million up to one date in one country.
- **new_cases_smoothed_per_million:** A 7-day rolling average of the number of new cases per million in one country.
- **total_deaths_per_million:** The total number of deaths per million up to one date in one country.
- **new_deaths_smoothed_per_million:** A 7-day rolling average of the number of new cases per million in one country.
- **reproduction_rate:** A 7-day rolling average of the expected number of cases directly generated by one case in a population where all individuals are susceptible to infection in one country.
- **people_fully_vaccinated_per_hundred:** The number of fully vaccinated people per hundred in one country.
- **total_boosters_per_hundred:** The number of boostered people per hundred in one country.

3 Data and task abstractions

Trying to answer the questions posed earlier in the report, the application should help the user **consume** the dataset. This makes it implicit that the intention with the application is not to **produce** new data but rather to use the tools for **presenting** or to give the user insight through **discovery** of the dataset. This implies giving the user a large amount of freedom, providing the tools with which the user can roam and then visualize the dataset without needing to deal with the raw dataset directly.

[3] https://github.com/CSSEGISandData/COVID-19
[4] *https://www.who.int/*
[5] *https://www.ecdc.europa.eu/en*

In order for our visualization to be effective we have to pair what we think is the most important variable to the most effective channel. Having such a wide range of variables our strategy was to move out of the specific domain and instead look at the variables as their more general data type. We were able to distill our questions into three categories of variables: temporal, positional and quantitative values. In the following sections we will explore our options in visualizing these different data types.

3.1 Temporal

The ‘date’ variable contains, as mentioned in the dataset description, 752 different days which start at 2020-01-01 and end at the last time the data was downloaded from the source, 2022-01-22. As our attribute ‘date’ can be used as a key to access the other attributes for a specific date and country, our dataset is to be considered as a time-series.

Time-series are most oftenly plotted on the x-axis, showing a quantitative attribute (y-axis) varying over a range of time-values. This is an effective way to show how a value changes over time, especially if the marks of the y-axis are continuous and do not clutter too much.

Beyond this way of showing a time-series, there has been an increase in the use of an interactive time slider, such as the one made popular by the bubble-chart tool implemented by Gapminder[6]. This allows traversing time while at the same time having two spatially positioned variables, which makes communicating different values of one time instant very effective.

A drawback of this method is that the user has to “memorize” what the previous state of the graph was when moving between time instants. Furthermore, if the user moves across time quickly then the effectiveness of the visualization will rely heavily upon the speed of which the graph is updated.

3.2 Location

Our locational data is stored in the variable ‘location’ as 232 different countries across the world, for example “Afghanistan”. This variable can be viewed both as a categorical value or as a position in a grid, where the grid is the world map.

Location is often shown as a categorical variable when it comes to comparing a quantitative attribute between different countries but with more than one variable separation is usually done using hue-colors. This is generally a good way to visualize locational data, usually offering good separability between the different locations. One drawback with this approach is that it has limited scalability when the amount of categories increases, as well as making the interpretation of the plot dependent on the user being able to distinguish between the colors.

[6] https://www.gapminder.org/tools/#$chart-type=bubbles&url=v1

Furthermore, our location variables could be used with a map, where a variable of the dataset could be mapped to a tile representing the location on the map, using either hue for categorical data or a saturation/luminance-scheme for quantitative data. This is usually referred to as a choropleth map. In comparison to assigning categorical values, the mapping approach scales better with more locations as users are quite familiar with the layout of a world map, although it might sacrifice the effectiveness of the second variable used with it.

3.3 Quantitative

Our quantitative data is stored in seven variables, as mentioned in the dataset description, which are categorized in several groups:

- We have a group of variables where the data is cumulative over the considered time:
    - **total_cases_per_million**
    - **total_deaths_per_million**
- Another group of variables is a 7-day rolling average per million:
    - **new_cases_smoothed_per_million**
    - **new_deaths_smoothed_per_million**
    - **reproduction_rate**
- The last group of variables is showing the number of a specific group (fully vaccinated / total boosters) per hundred.
    - **people_fully_vaccinated_per_hundred**
    - **total_boosters_per_hundred**

Since quantitative data could be infinitely large, it should be considered using a specific measure to make it comparable as in the first group of variables. Nevertheless, a variable could also be misleading taking only one specific day into account - which is why we decided to add the variables of the second group which are showing a rolling average of 7 days instead. In any case, representing mostly quantitative data the choropleth map has the drawback that the values are represented by ordinal colors, which is not considered very effective.

4 Interaction and visual encoding

Since we have questions that require different degrees of “zoom”, we have chosen to divide our application into one area where the user has a wider aspect ratio of the data and one where more specific queries can be answered. The more general area is composed of a choropleth map with a time-slider, showing a chosen variable as a color scale on each country-tile.

The second area is used to compare quantitative attributes more exactly, making it possible to compare the development of a variable in two countries using a line plot. The second tool in this area is a scatter plot which allows the user to explore the correlation of two variables in a chosen country.

4.1 Marks

The marks we've used in our visual representation of the dataset are motivated in the following subheadings.

4.1.1 Topological grid (map)

For question one a map seems most viable as it scales better than any other channel trying to show differences between more than 200 categories. However, this comes at a loss of effectiveness for the quantitative variable as it must be visualized either using area or color.

Visualizing topological data in tiles representing the country's borders is a very effective way to distinguish them since almost everyone is familiar with that kind of visual representation of countries. To show differences between the countries we will have to turn to a channel, preferably representing the quantitative variables we have on an ordinal scale represented by different colors depending if the variable chosen to represent was a negative or a positive one.

This choice seemed hard to motivate at first due to the lack of efficiency that the quantitative variable would get, but was in the end chosen as it enables showing many countries at the same time. This is especially important when exploring as the users are not limited to any geographical area in the world, while still being able to zoom in to a specific continent on the map and compare countries and variables in greater detail in the other plots.

4.1.2 Line plot

The line plot is to be used to describe changes of a numerical variable over time, comparing two countries as categorical variables. Since the selected quantitative variable differs over time as the pandemic evolved this was an excellent way to visualize the changes over time for only two countries. Since the line plot is also a very common way of visualizing data we chose it as it should be known amongst most of future users.

4.1.3 Scatter plot

We wanted the users to be able to easily compare data over two different quantitative variables chosen by themselves considering one country. Because of this, we chose to also implement a scatter plot that would plot two variables over time to identify patterns between the two or if there were no correlation over the certain period. Nevertheless, the scatterplot has the drawback that not more than two variables could be compared.

4.2 Channels

The channels we've used in our visual representation of the dataset are motivated in the following subheadings.

4.2.1 Color saturation

In the map the quantitative variables can be visualized using a saturation color-scheme, starting from white and going up to a color that depends on the variable. For example, a negative color would be “total number of deaths”, which could be represented as white to dark-red, whilst a positive one like the number of fully vaccinated inhabitants could be a value of white to dark-green. The saturation scale is being chosen to avoid any color discrimination problems.

A map might not be the most effective way to represent quantitative data, whereas for example a plot of values could do that better. As stated in the marks section, the choice of the map was part of a bigger scheme where our dataset should be explorable. If the user would like he or she could ‘travel’ the world with our map to see differences both where and on what topic he or she would like. This way the map works as a gateway of exploration of the dataset in more depth and of easier availability of flexibility.

4.2.2 Hue for categorical variables

Hue is to be used for distinguishing between the two different countries shown in the line plot. Here we used two different colors but with different saturation to avoid discrimination.

5 Algorithmic implementation

In this chapter we introduce how we built the application and which libraries we used to implement our visualization tools.

5.1 Building the Application

5.1.1 Data import and preparation

In order to visualize the dataset we needed to import the downloaded data (CSV-file) first. The next step is to filter out the needed data and to initialize needed parameters, those parameters are for example the dates and countries.

5.1.2 User Interface

The first step of creating the user interfaces was to develop a simple list of variables that the user could choose from, later this list was developed to be connected to the final map. The list with its sidebar-panel was improved upon when we added more options for choosing variables as well as a separate panel for the scatter plot as well as the line graph.

The user interface is simple yet filled with options. Our objective with these choices was to make it as easy as possible for the user to understand the basic user interface as well as being able to navigate it easily, without compromising the complexity that the application allows.

5.1.3 Server

The server part of the project is used to update the user interface with every change that is made by the user. This updates the color scheme and the label of the variable as it is changed in the list. It is also what changes as the continent variable is changed to display a different part of the world on the map.

5.2 Libraries and issues solved

To implement the map we relied on a library called “highcharter”[7] from which we could produce the map as well as the scatter and line plots. We also used a Data manipulation library called Dplyr[8], which is a native R package to manipulate the dataset to fit the way we want to provide the data to our other functions and libraries if they do not fit in the way they originally were.

We used Shiny to create the application to have an already existing structured workflow for the user interface as well as building the application[9]. Shiny solved all structure-related problems that can occur when developing an application such as building the app on your local port, that was already solved in Shiny and we could focus on the dataset and what we wanted to visualize through it.

6 Conclusions and personal remarks

We as a group are happy about the result of the final application, although it does have some quirks that we didn’t have time to solve (described in the final section). The different views offer answers to a broad range of possible queries, allowing for the user to explore. We are especially happy with getting the choropleth map working as the implementation looks great and allows for a lot of freedom for the users to explore the dataset.

It was not the best experience developing in R and Shiny as there was many libraries but usually bad or no documentation on certain parameters of the different functions. It was quite laborious to fit the data into the different plots, sometimes employing quirky formatting in order to make it work - which felt unnecessarily hard.

6.1 Possible improvements to application

The main issue as we see it is that the application is running rather slow when changing the inputs of both the map and the scatter plot. We tried to make the data render reactively but in the end we could not make them render fast enough to feel like they weren’t buffering, which was disappointing. We also have a bug where for some dates the map does not render correctly, leaving it all white instead.

[7] https://www.highcharts.com/
[8] https://www.rdocumentation.org/packages/dplyr/versions/0.7.8
[9] https://shiny.rstudio.com/

An interactive map and user interface should not be slow as a lot of the users would deem this unusable. Furthermore, there is also a further development possible in the design of the UI, we all agree that with more time it would be possible to make this application look a lot better with better position on the detailed graphs, more colorful all around with a nice theme instead of the simple background.

References

(1) https://ourworldindata.org/
(2) https://global-change-data-lab.org/
(3) https://github.com/CSSEGISandData/COVID-19
(4) *https://www.who.int/*
(5) *https://www.ecdc.europa.eu/en*
(6) https://www.gapminder.org/tools/#$chart-type=bubbles&url=v1
(7) https://www.highcharts.com/
(8) https://www.rdocumentation.org/packages/dplyr/versions/0.7.8
(9) https://shiny.rstudio.com/

Anex

1 Instructions to run the program

1.1 Through the web

The app is published through shinyapps.io and can be reached with the following URL:
https://bengterik.shinyapps.io/covid/

We experienced rendering performance slightly worse in the web-application.

1.2 Run locally

The program and the data file are included in the zip-file provided in handin.zip. The application can be run through R-studio and in order for libraries to work properly it’s advisable to use the latest version of R-base. The required libraries will be installed automatically by the program if not found.

To launch the app follow these given steps:

1. Control that the data file resides in the same directory as the app.R file
2. Set the working directory to the directory where the files reside
3. Launch application