Visualization Practical Work

Data Visualization

Authors:
Víctor Morcuende Castell
Guillermo Nájera Lavid
Javier Rocamora García

UNIVERSIDAD
POLITÉCNICA
DE MADRID

January 2023

Visualization Practical Work

Practical assignment for the Data Visualization subject

Authors

Víctor Morcuende Castell

Guillermo Nájera Lavid

Javier Rocamora Garcia

EIT Digital Data Science Master

MADRID, January 2023

Contents

1 Problem characterization in the application domain

1.1 Context

Music therapy (MT) is the use of **music** to **improve** an **individual**'s stress, mood, and overall **mental health**. MT is also recognized as an **evidence-based practice**, using music as a **catalyst** for "happy" hormones such as oxytocin.

However, MT employs a wide range of **different genres**, varying from one organization to the next.

This **data-set aims to identify** what, if any, **correlations** exist **between** an **individual's music taste** and their **self-reported mental health**. Ideally, these findings could contribute to a more informed application of MT or simply provide interesting sights about the mind.

1.2 Data collection

The data collection for this data-set was **managed via Google Form**. **Respondents** were **not restricted** by **age** or **location**.

The form was posted in various Reddit forums, Discord servers, and social media platforms. Posters and "business cards" were also used to advertise the form in libraries, parks, and other public locations.

The form was relatively brief so that respondents would be more likely finish the survey. "Harder" questions (such as BPM) were left optional for the same reason.

1.3 Data interpretation

- **Background**: Respondents answer generic questions focused on **musical background** and listening habits.
- **Music genres**: Respondents rank how often they listen to **16 music genres**, where they can select:
  - **Never**
  - **Rarely**
  - **Sometimes**
  - **Very frequently**

- **Mental health**: Respondents **rank Anxiety**, **Depression**, **Insomnia**, and **OCD** on a scale of 0 to 10, where:
  - **0** - If they **do not experience** this
  - **10** - If **they experience** this regularly, constantly/or to an extreme.

1.4 Questions proposed of the potential user of the tool

- **Is there a relationship between music genre and mental health issues?**
- **Evolution in the degree of mental illnesses by age**
- **Analyze the distribution of mental illnesses in music listeners**

2 Data Abstraction and Task Abstractions

Now that the **data and tasks** are **defined** in the **application domain**, the next step is to **do** the **data abstraction** and the **task abstraction**.

2.1 Data abstraction

Data abstractions tries to **describe the data** in ways that **help us decide** what **operations and encoding methods** are available and **appropriate**. The goal is to determine which is the data type that best fits our described problems.

So as for the **dataset type** we are using a **Table data-set** (contained in a **csv** file type), which contains different **abstract data types** such as:

- **Attributes**: referring to each of the **variables** measured in the **columns** of the data-set table. Taking into account that the attribute "**Timestamp**" was the only **attribute deleted** from the **data-set** from the beginning, the declared attributes are as follows:

  Age, Primary streaming service, Hours per day, While working, Instrumentalist, Composer, Fav genre, Exploratory, Foreign languages, BPM, Frequency [Classical], Frequency [Country], Frequency [EDM], Frequency [Folk], Frequency [Gospel], Frequency [Hip hop], Frequency [Jazz], Frequency [K pop], Frequency [Latin], Frequency [Lofi], Frequency [Metal], Frequency [Pop], Frequency [R&B], Frequency [Rap], Frequency [Rock], Frequency [Video game music], Anxiety, Depression, Insomnia, OCD, Music effects, Permissions

- **Items**: referring to each of the **rows** contained in the data-set.

2.2 Task abstraction

Transforming task descriptions from domain-specific language into abstract form allows us to compare them between different domains.

As for the **actions**, firstly one needs to understand our necessity to **consume** the **information presented in the data-set**, with the purpose of **answering the questions** that arise from the problems proposed at the beginning, thus being able to **understand** the different **correlations** between the different **attributes** of the data-set and being able to **generate an hypothesis** based on them. This is the **reason** why a **visualization will be implemented**.

As for the **targets** (referring to the entities where the previous described actions will be performed) we will be using **multiple attributes** from the data-set, trying to **answer** the

different problems and in this sense **looking for** the different **dependencies**, **correlations** or **similarities** that may exist between them.

3 Interaction and Visual Encoding

**Link for the shiny app developed:*

https://music-x-mental-health.shinyapps.io/Data-Visualization-ShinyApp/

Finally, now that the **data and task abstractions have been defined**, it is time to start the **design of the tool**. For that we will choose the **different idioms** (visualization and interaction) we think are suited and effective for the tasks and data.

First of all, we decided that, since our application has to solve **three different problems** in a **separated manner**, we would create **3 different tabPanels** as seen in **figure 3.1** for each of the questions, those being:

- **tabPanel 1**: this tabPanel will refer to the **first question** and so it will show the **relationship between mental illnesses and subject's favourite music genre**
- **tabPanel 2**: this tabPanel will refer to the **second question** and so it will show the **evolution of mental illnesses based on subject's age**
- **tabPanel 3**: this tabPanel will refer to the **last question** and so it will show the **distribution of Mental Illnesses in subjects**
- **Correlations**: we will keep this last tabPanel for testing the relationship between numerical variables

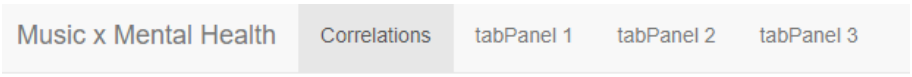

Figure 3.1: Creation of tabPanels in R with Shiny

The **names of these panels** will be **later substituted** by the **names of the idioms selected** for the different problems.

Now that we have our panels created, we will decide in each of them which **idiom** we will use to visualize and interact the different problems proposed:

- **Regarding first question**

  For this problem, the variables regarding **mental health issues** (**Anxiety**, **Depression**, **Insomnia**, **OCD**) and the variable regarding **favourite music genre** (**Fav genre**) will be needed.

  The variables regarding **mental health issues** are **numerical** and **refer to a ranking from 0-10** and the variable regarding **favourite music genre** is **categorical** and

refer to the different music genres available: Rock, Pop, Metal, Video game music, Hip Hop, EDM, Classical, Folk, R&B, K pop, Country, Jazz, Rap, Lofi, Gospel, Latin.

Taking into account our willing to **represent** a **numerical variable** (**mental health issue**) in function of a **categorical variable** (**fav genre**), we decided that the **best idiom** for representing this relationship was a **bar chart**, as seen in **figure 3.2**.

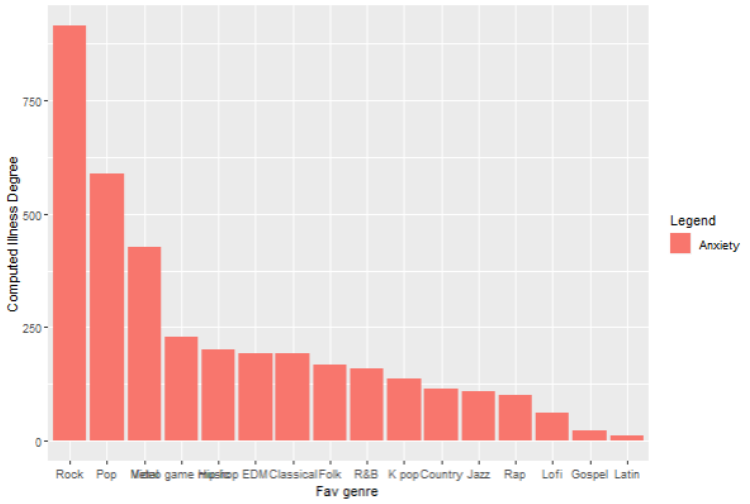

Figure 3.2: Simple Bar Chart for Question 1

Still, on the **user side**, since the relationship between **different mental illnesses** may be **wanted to be seen** at the **same time**, we finally decided to use a **Stacked Bar Chart** as the idiom, and allow the **user interaction** with it via a **Sidebar** in the **left side** with a **selectInput** that allows the user to **select the different mental illnesses** wanted to be shown in the chart. This chart can be seen in **figure 3.2**.

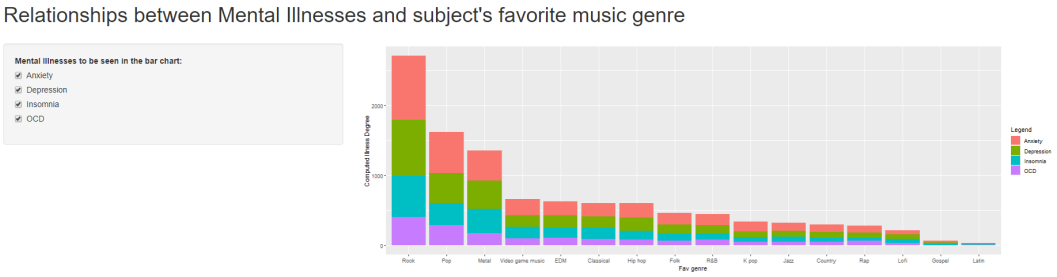

Figure 3.3: Stacked Bar Chart for Question 1

- **Regarding second question**

  For this question, the variable regarding the **age** (Age) and the variables regarding **mental health issues** (same as with the previous question) will be **needed**.

  The variable regarding the **age** is **numerical** and refer to the numerical value of the

age of the person.

Taking into account our will to **represent** a **numerical variable** (Age) in **function of** another **numerical variable** (mental health issue), we decided that the best **idiom** for representing this relationship was a **line chart**, as it can be seen in **figure 3.4**.

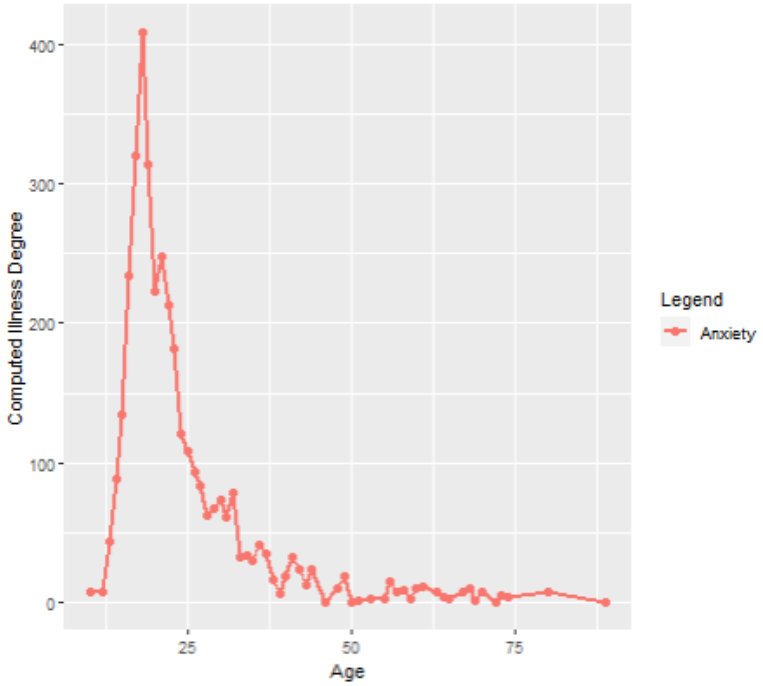

Figure 3.4: Simple Line Chart for Question 2

Still, on the **user side**, since the evolution of **different mental illnesses** may be wanted to be seen at the **same time**, we finally decided to use a **Line Chart** as the **idiom**, but applying the superimpose facet to join the different mental illnesses, and allow the **user interaction** with it via a **Sidebar** in the left side with a **selectInput** that **allows the user to select the different mental illnesses** wanted to be shown in the chart. Also the sideBar will allow the user to **see the data summarised in two ways**:

- **Sum**: the **y axis** corresponding to the **Computed Illness Degree** will be the **sum** of the **range of the mental illness degree selected** corresponding to the **age for each point**. This way we can see in **which age most of a mental illness is perceived**.
- **Mean**: the **y axis** corresponding to the **Computed Illness Degree** will be in the **range of 0 to 10** thanks to **performing** the **mean** of the values for each point. This way we can **visualize** how the **range of mental illnesses changes along the different ages**.

As it can be seen, when **more than one mental illness** is **added**, automatically a **black line** appears showing the **mean of the added mental illnesses** in the chart.

This chart can be seen in **figure 3.5**.

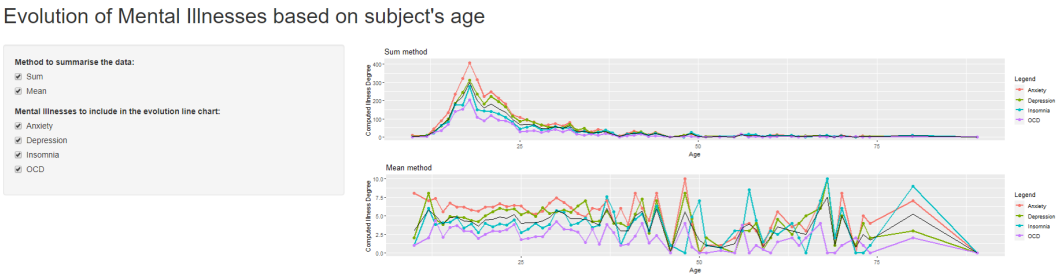

Figure 3.5: Line Chart applying the superimpose facet for Question 2

- **Regarding third question**

  For this question, we can take into account **any of the categorical variables** in the data-set (fav genre, primary streaming service etc.) as well as again the **variables regarding mental health issues** commented before.

  Taking into account the need to **represent several categorical variables** with respect to **several numerical variables**, we decided that the best **idiom** for representing this relationship was a **heatmap**.

  Still, on the **user side**, as a **different subject may be wanted to be chosen** as to see the **distribution of mental illnesses**, we created a **left panel** that **allows the selection of different categorical variables** for this purpose.

  This heatmap can be seen in **figure 3.6**

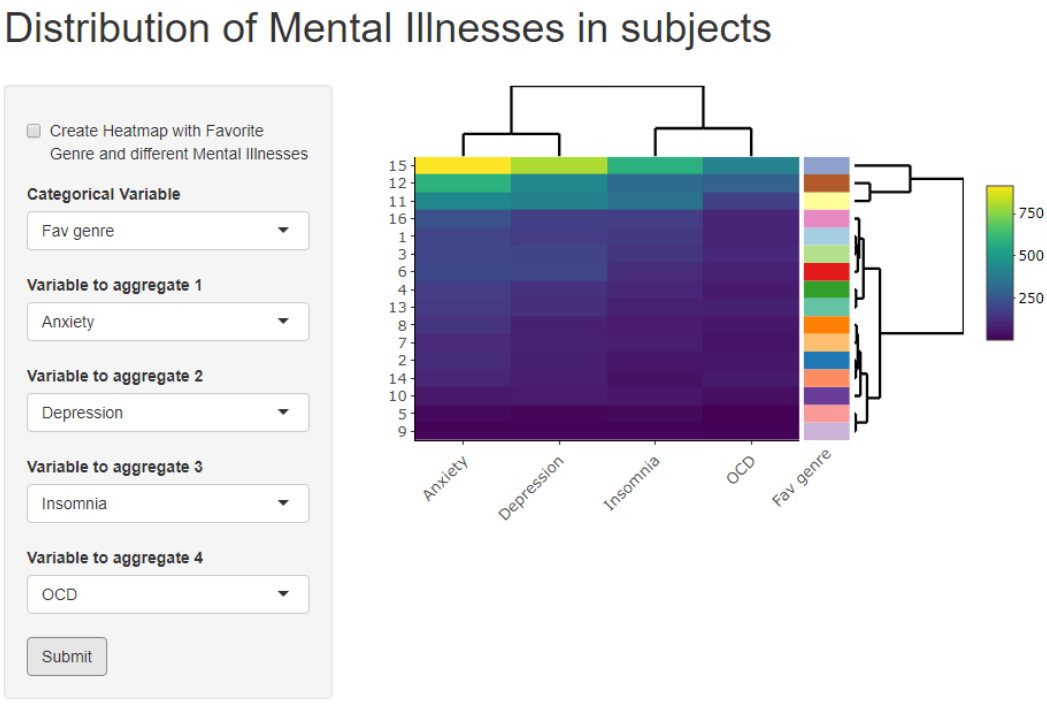

Figure 3.6: Heat Map for Question 3

4 Algorithmic implementation

Finally, the **previous design was implemented**. For that purpose we used the **R programming language** along the **suite R Studio**, which is an integrated development environment for the same. As it is known, R is a programming language for **statistical computing and graphics**.

As it can be seen in the **root folder** of the project, an **R project was created** with extension "**.Rproj**", and it contains all the **code and data** necessary for **running the application**. The most important file of this project would be then "**app.R**", which contains the **application code implementation**.

Moving forward, in this document we were commenting ("**app.R**") the first thing we can see is the different **dependencies required** and used for the app:

- library(**shiny**)
- library(**readr**)
- library(**caret**)
- library(**mltools**)
- library(**data.table**)
- library(**ggplot2**)
- library(**reshape2**)
- library(**plyr**)
- library(**tidyverse**)
- library(**hrbrthemes**)
- library(**viridis**)
- library(**plotly**)
- library(**heatmaply**)
- library(**dplyr**)
- library(**tidyr**)
- library(**rsconnect**)

One of the **most important dependencies** would be **Shiny**, which is an **R package** that makes it easy to **build interactive web applications** (apps) straight from R.

Our Shiny application has a **structure of components** that we **implemented first**, and it is as follows:

- **User interface object** (**ui**): it will control the layout and appearance of the app
- **Server function**: it will contain the instructions that the computer needs to build the app
- **Call to the shinyApp function**: creates Shiny app objects from an explicit UI/server pair

Then, moving on to the **interface**, we created several **elements**:

1. One of them was the **three different tabPanels** that we could see in the previous section in **figure 3.1**
2. Then, for each tabPanel the **structure was similar**, containing a **Sidebar** on the **left** and the **graphic plot** on the **right**
3. For the **Sidebars**, **different elements** were created, like **check-boxes**, **select inputs** and **submit buttons**

5 Deployment

The **application** can be **launched** in **two ways**:

- Launch **Shiny app link** (see **figure 5.1**):
  https://music-x-mental-health.shinyapps.io/Data-Visualization-ShinyApp/

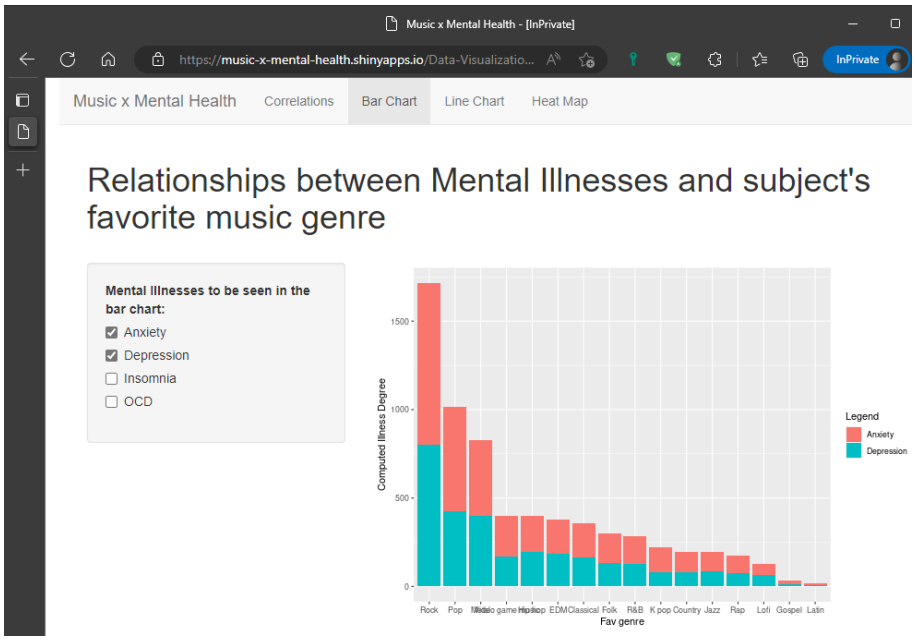

Figure 5.1: How to launch the application with the Shiny app link

- **Manually in R**:
  1. **Import** the **project** in the R Studio **or** just **open** the **file** '**Data-Visualization-ShinyApp.Rproj**' in the **project folder**
  2. **Install** the required **dependencies**:
     - library(shiny)
     - library(readr)
     - library(caret)
     - library(mltools)
     - library(data.table)

- library(ggplot2)
- library(reshape2)
- library(plyr)
- library(tidyverse)
- library(hrbrthemes)
- library(viridis)
- library(plotly)
- library(heatmaply)
- library(dplyr)
- library(tidyr)
- library(rsconnect)

3. **Press** the **button** '**Run App**' as seen in **figure 5.2** & **figure 5.3**.

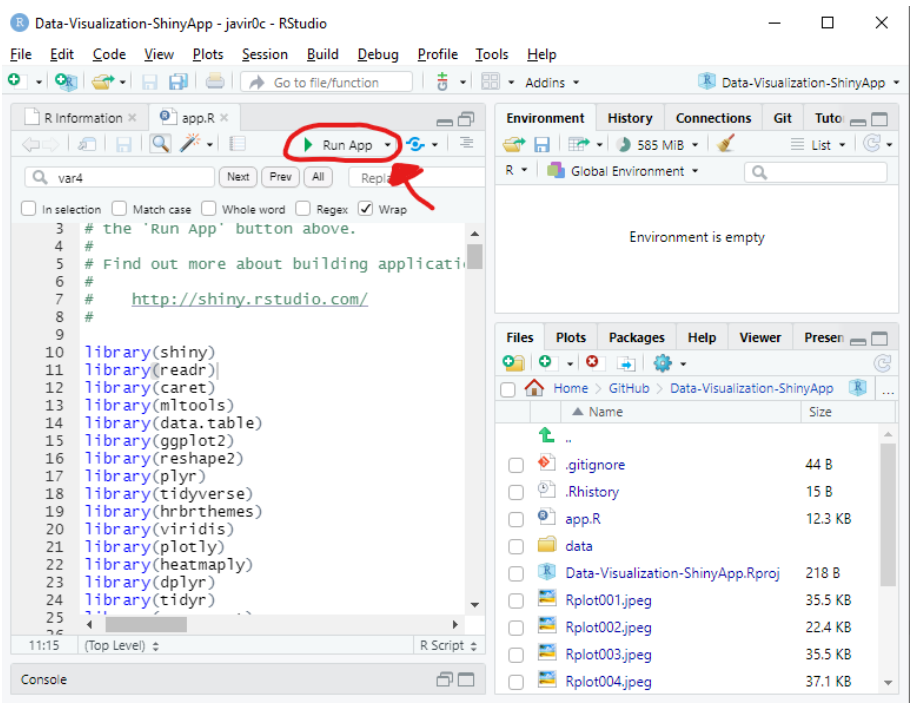

Figure 5.2: How to launch the application in R Studio

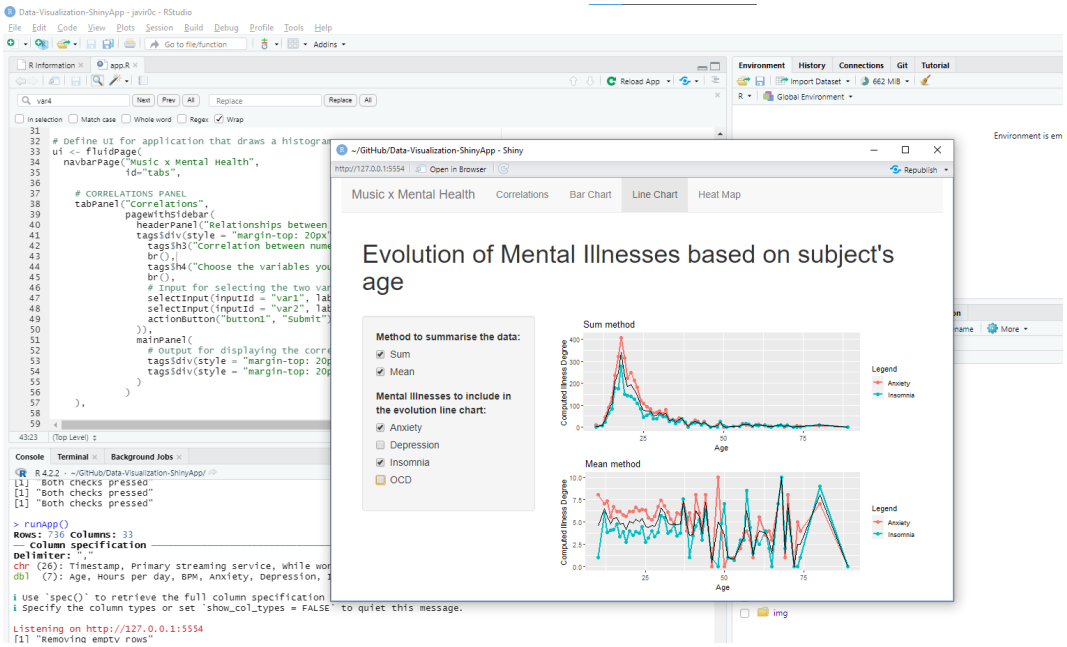

Figure 5.3: Application already launched in R Studio