

Challenge-2

Esther Kho. Yining

2024-01-23

Welcome! Hope you have watched the lecture videos and followed the instructions in code-along. Go through the steps described below, *carefully*. It is totally fine to get stuck - ASK FOR HELP; reach out to your friends, TAs, or the discussion forum on Canvas.

Here is what you have to do,

1. Pair with a neighbor and work
2. Download the `Challenge-2.Rmd` and `playlist_data.csv` files from Canvas
3. Move the downloaded files to the folder, “Week-2”
4. Set it as the working directory
5. Edit content wherever indicated
6. Remember to set `eval=TRUE` after completing the code to generate the output
7. Ensure that `echo=TRUE` so that the code is rendered in the final document
8. Inform the tutor/instructor upon completion
9. Submit the document on Canvas after they approve
10. Attendance will be marked only after submission
11. Once again, do not hesitate to reach out to the tutors/instructor, if you are stuck

I. Exploring music preferences

A. Background

Imagine that you have been hired as a data analyst by a radio station to analyze music preferences of their DJs. They have provided you with a dataset, `playlist_data.csv`, containing information about DJs, their preferred music genres, song titles, and ratings.

Using the data-set you are required to complete some tasks that are listed subsequently. All these tasks are based on the concepts taught in the video lectures. The questions may not be entirely covered in the lectures; To complete them, you are encouraged to use Google and the resources therein.

B.Tasks

Task-1 In the lecture, we used two data-sets, `starwars` and `anscombe's quartet` that were readily available with the packages, `tidyverse` and `Tmisc`, respectively. When we have to use custom-made data-sets or the ones like we downloaded from Canvas, we have to import it using the R commands before using them. All the questions below are related to this task.

Question 1.1: What does the term “CSV” in `playlist_data.csv` stand for, and why is it a popular format for storing tabular data?

Solution: CSV stands for ‘comma-separated values’. It is a popular format because it is a plain-text file, making it easy to import into a spreadsheet or other storage database for all software. **Question 1.2:** load the `tidyverse` package to work with `.csv` files in R.

Solution:

```
# Load the necessary package to work with CSV files in R.
```

Question 1.3: Import the data-set, `playlist_data.csv`

Solution:

```
# Import the "playlist_data.csv" dataset into R
library(readr)
read_csv("playlist_data.csv")
```

```
## Rows: 26 Columns: 7
## -- Column specification -----
## Delimiter: ","
## chr (4): DJ_Name, Music_Genre, Experience, Location
## dbl (3): Rating, Age, Plays_Per_Week
##
## i Use 'spec()' to retrieve the full column specification for this data.
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.

## # A tibble: 26 x 7
##   DJ_Name Music_Genre Rating Experience   Age Location Plays_Per_Week
##   <chr>   <chr>      <dbl> <chr>    <dbl> <chr>      <dbl>
## 1 DJ A     Pop          4.2 Advanced    28 City X         80
## 2 DJ B     Rock          3.8 Intermediate 24 City Y         60
## 3 DJ C     Electronic    4.5 Advanced    30 City Z        100
## 4 DJ D     Pop           4 Intermediate 22 City X         70
## 5 DJ E     Electronic    4.8 Advanced    27 City Y         90
## 6 DJ F     Rock          3.6 Intermediate 25 City Z         55
## 7 DJ G     Pop           4.3 Advanced    29 City X         85
## 8 DJ H     Electronic    4.1 Intermediate 23 City Y         75
## 9 DJ I     Rock          3.9 Advanced    31 City Z         70
## 10 DJ J    Pop           4.4 Intermediate 26 City X         95
## # i 16 more rows
```

Question 1.4: Assign the data-set to a variable, `playlist_data`

Solution:

```
# Assign the variable to a dataset
```

```
playlist_data <- read_csv("playlist_data.csv")
```

```
## Rows: 26 Columns: 7
## -- Column specification -----
## Delimiter: ","
## chr (4): DJ_Name, Music_Genre, Experience, Location
## dbl (3): Rating, Age, Plays_Per_Week
##
## i Use 'spec()' to retrieve the full column specification for this data.
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.
```

From now on, you can use the name of the variable to view the contents of the data-set

Question 1.5: Get more information about `read_csv()` command and provide a screenshot of the information displayed in the “Help” tab of the “Files” pane

Solution:

```
# More information about the R command, complete the code
```

```
?read_csv(playlist_data.csv)
```

Question 1.6: What does the `skip` argument in the `read_csv()` function do?

Solution: The ‘skip’ argument in the `read_csv()` function allows the user to state the number of lines to skip at the start of the file while reading the data

Question 1.7: Display the contents of the data-set

Solution:

```
# Type the name of the variable, to see what it contains
```

```
playlist_data
```

```
## # A tibble: 26 x 7
##   DJ_Name Music_Genre Rating Experience   Age Location Plays_Per_Week
##   <chr>    <chr>      <dbl> <chr>      <dbl> <chr>      <dbl>
## 1 DJ A      Pop          4.2 Advanced    28 City X          80
## 2 DJ B      Rock          3.8 Intermediate 24 City Y          60
## 3 DJ C      Electronic    4.5 Advanced    30 City Z         100
## 4 DJ D      Pop           4 Intermediate 22 City X          70
## 5 DJ E      Electronic    4.8 Advanced    27 City Y          90
## 6 DJ F      Rock          3.6 Intermediate 25 City Z          55
## 7 DJ G      Pop           4.3 Advanced    29 City X          85
## 8 DJ H      Electronic    4.1 Intermediate 23 City Y          75
## 9 DJ I      Rock          3.9 Advanced    31 City Z          70
## 10 DJ J     Pop           4.4 Intermediate 26 City X          95
## # i 16 more rows
```

Question 1.8: Assume you have a CSV file named `sales_data.csv` containing information about sales transactions. How would you use the `read_csv()` function to import this file into R and store it in a variable named `sales_data`?

Solution:

Source

```

R 4.3.2 · ~/Documents/Y2S2/NM2207/Week 2/Challenge-2/
✓ lubridate 1.9.3 ✓ tidyverse 1.3.0
✓ purrr 1.0.2

— Conflicts — tidyverse_conflicts() —
* dplyr::filter() masks stats::filter()
* dplyr::lag() masks stats::lag()
i Use the conflicted package to force all conflicts to become errors
> library(ggplot2)
> library(shiny)
> library(Tmisc)
> # Import the "playlist_data.csv" dataset into R
> read_csv("playlist_data.csv")
Rows: 26 Columns: 7
— Column specification —
Delimiter: ","
chr (4): DJ_Name, Music_Genre, Experience, Location
dbl (3): Rating, Age, Plays_Per_Week

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
# A tibble: 26 × 7
  DJ_Name Music_Genre Rating Experience Age Location Plays_Per_Week
  <chr>   <chr>       <dbl> <chr>   <dbl> <chr>      <dbl>
1 DJ A    Pop           4.2 Advanced 28 City X      80
2 DJ B    Rock           3.8 Intermediate 24 City Y      60
3 DJ C    Electronic     4.5 Advanced 30 City Z      100
4 DJ D    Pop            4 Intermediate 22 City X      70
5 DJ E    Electronic     4.8 Advanced 27 City Y      90
6 DJ F    Rock           3.6 Intermediate 25 City Z      55
7 DJ G    Pop            4.3 Advanced 29 City X      85
8 DJ H    Electronic     4.1 Intermediate 23 City Y      75
9 DJ I    Rock           3.9 Advanced 31 City Z      70
10 DJ J    Pop            4.4 Intermediate 26 City X      95

# i 16 more rows
# i Use `print(n = ...)` to see more rows
> # Assign the variable to a dataset
> my_data <- read_csv("playlist_data.csv")
Rows: 26 Columns: 7
— Column specification —
Delimiter: ","
chr (4): DJ_Name, Music_Genre, Experience, Location
dbl (3): Rating, Age, Plays_Per_Week

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
> # More information about the R command, complete the code
> ?read_csv(playlist_data.csv)
> |

```

Environment History Connections Tutorial

Files Plots Packages Help Viewer Presentation

R: Read a delimited file (including CSV and TSV) into a tibble

read_delim(readr)

Read a delimited file (including CSV and TSV)

Description

`read_csv()` and `read_tsv()` are special cases of the more general `read_delim()` reading the most common types of flat file data, comma separated values and tab separated values respectively. `read_csv()` uses `,` for the field separator and `.` for the decimal separator, both of which are common in some European countries.

Usage

```

read_delim(
  file,
  delim = NULL,
  quote = "\"",
  escape_backslash = FALSE,
  escape_double = TRUE,
  col_names = TRUE,
  col_types = NULL,
  col_select = NULL,
  id = NULL,
  locale = default_locale(),
  na = c("", "NA"),
  quoted_na = TRUE,
  comment = "",
  trim_ws = FALSE,
  skip = 0,
  n_max = Inf,
  guess_max = min(1000, n_max),
  name_repair = "unique",
  num_threads = readr_threads(),
  progress = show_progress(),
  show_col_types = should_show_types(),
  skip_empty_rows = TRUE,
  lazy = should_read_lazy()
)

```

```
# No output is required for this code
# Only the list of commands that execute the task mentioned in the question are required

read_csv("sales_data.csv")

sales_data <- read_csv("sales_data.csv")
```

Task-2 After learning to import a data-set, let us explore the contents of the data-set through the following questions

Question 2.1: Display the first few rows of the data-set to get an overview of its structure

Solution:

```
# Type the name of the variable we assigned the data-set to
head(playlist_data)
```

```
## # A tibble: 6 x 7
##   DJ_Name Music_Genre Rating Experience      Age Location Plays_Per_Week
##   <chr>    <chr>      <dbl> <chr>      <dbl> <chr>      <dbl>
## 1 DJ A    Pop           4.2 Advanced    28 City X           80
## 2 DJ B    Rock           3.8 Intermediate 24 City Y           60
## 3 DJ C    Electronic     4.5 Advanced    30 City Z          100
## 4 DJ D    Pop            4 Intermediate 22 City X           70
## 5 DJ E    Electronic     4.8 Advanced    27 City Y           90
## 6 DJ F    Rock           3.6 Intermediate 25 City Z           55
```

Question 2.2: Display all the columns of the variable stacked one below another

Solution:

```
# Stack columns of playlist_data
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr      1.1.4      v purrr      1.0.2
## v forcats    1.0.0      v stringr    1.5.1
## v ggplot2    3.4.4      v tibble     3.2.1
## v lubridate  1.9.3      v tidyr      1.3.0
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
glimpse(playlist_data)
```

```
## Rows: 26
## Columns: 7
## $ DJ_Name      <chr> "DJ A", "DJ B", "DJ C", "DJ D", "DJ E", "DJ F", "DJ G", ~
## $ Music_Genre  <chr> "Pop", "Rock", "Electronic", "Pop", "Electronic", "Rock~
## $ Rating       <dbl> 4.2, 3.8, 4.5, 4.0, 4.8, 3.6, 4.3, 4.1, 3.9, 4.4, 4.6, ~
```

```
## $ Experience      <chr> "Advanced", "Intermediate", "Advanced", "Intermediate",~
## $ Age             <dbl> 28, 24, 30, 22, 27, 25, 29, 23, 31, 26, 32, 28, 29, 25,~
## $ Location        <chr> "City X", "City Y", "City Z", "City X", "City Y", "City~
## $ Plays_Per_Week <dbl> 80, 60, 100, 70, 90, 55, 85, 75, 70, 95, 110, 75, 60, 8~
```

Question 2.3: How many columns are there in the dataset?

Solution:

```
# Number of columns

num_columns <- ncol(playlist_data)

num_columns
```

```
## [1] 7
```

Question 2.4: What is the total count of DJs?

Solution:

```
# Number of DJs

num_DJs <- length(unique(playlist_data$DJ_Name))

num_DJs
```

```
## [1] 26
```

Question 2.5: Display all the location of all the DJs

Solution:

```
# Location of DJs

unique_locations <- unique(playlist_data$Location[playlist_data$DJ_Name != ""])

unique_locations
```

```
## [1] "City X" "City Y" "City Z"
```

Question 2.6: Display the age of the DJs

Solution:

```
# Age of DJs

ages <- playlist_data$Age[playlist_data$DJ_Name != ""]

ages
```

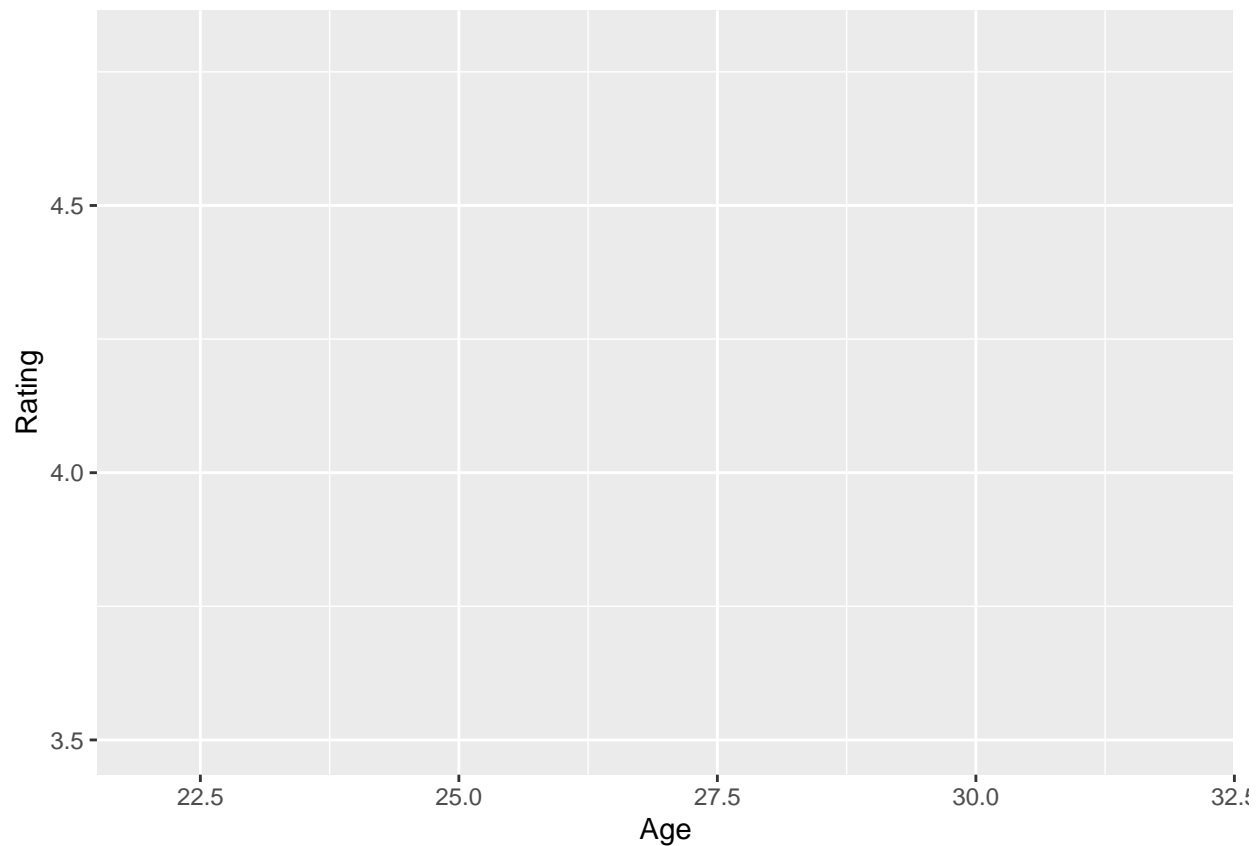
```
## [1] 28 24 30 22 27 25 29 23 31 26 32 28 29 25 31 26 27 24 29 23 28 24 30 22 27
## [26] 25
```

Task-3 Let us plot the data to get more insights about the DJs.

Question 3.1: Create a plot to visualize the relationship between DJs' ages and their ratings.

Solution:

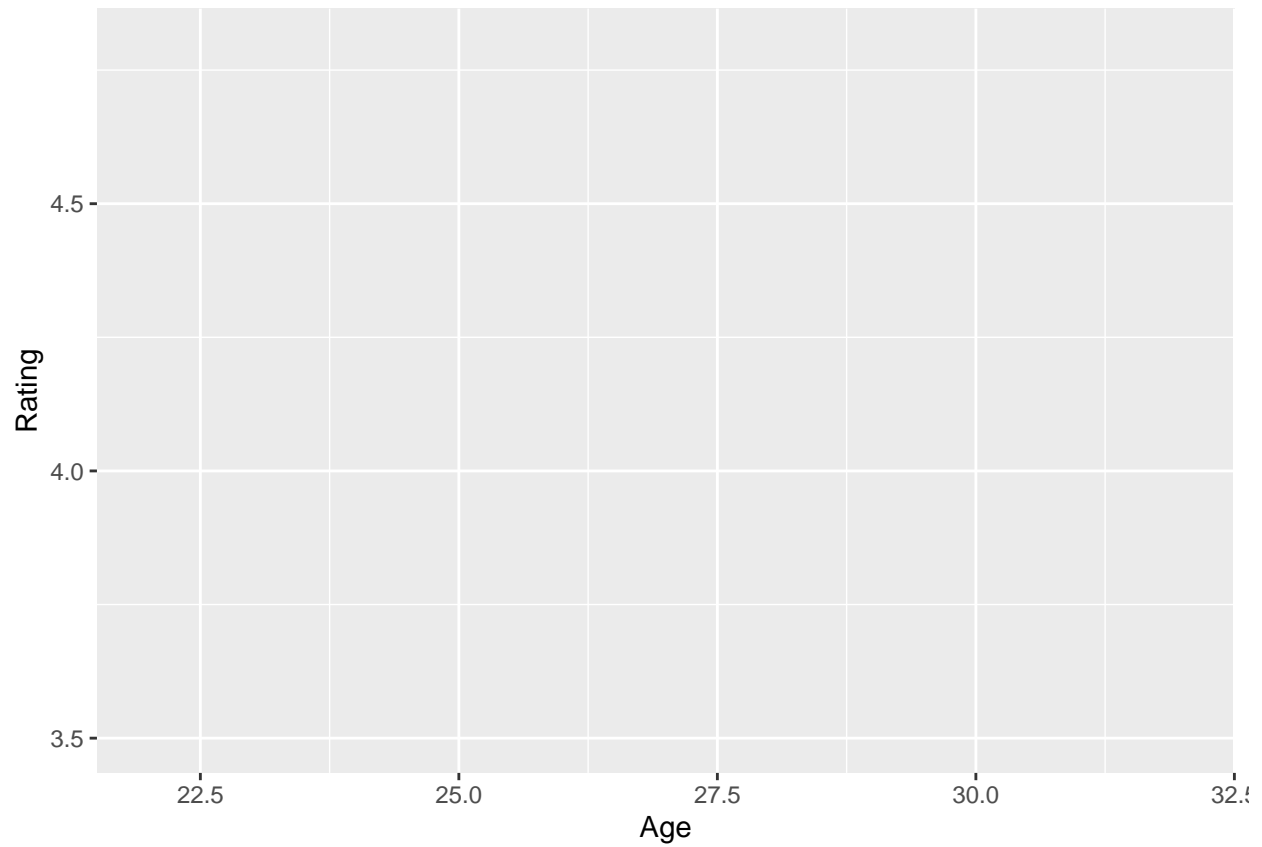
```
# complete the code to generate the plot  
ggplot(data=playlist_data,mapping=aes(x=Age,y=Rating))
```



Question 3.2: Label the x-axis as “Age” and the y-axis as “Rating.”

Solution:

```
# complete the code to generate the plot  
library(ggplot2)  
ggplot(data=playlist_data,mapping=aes(x=Age,y=Rating)) +  
  labs(x="Age",y="Rating")
```



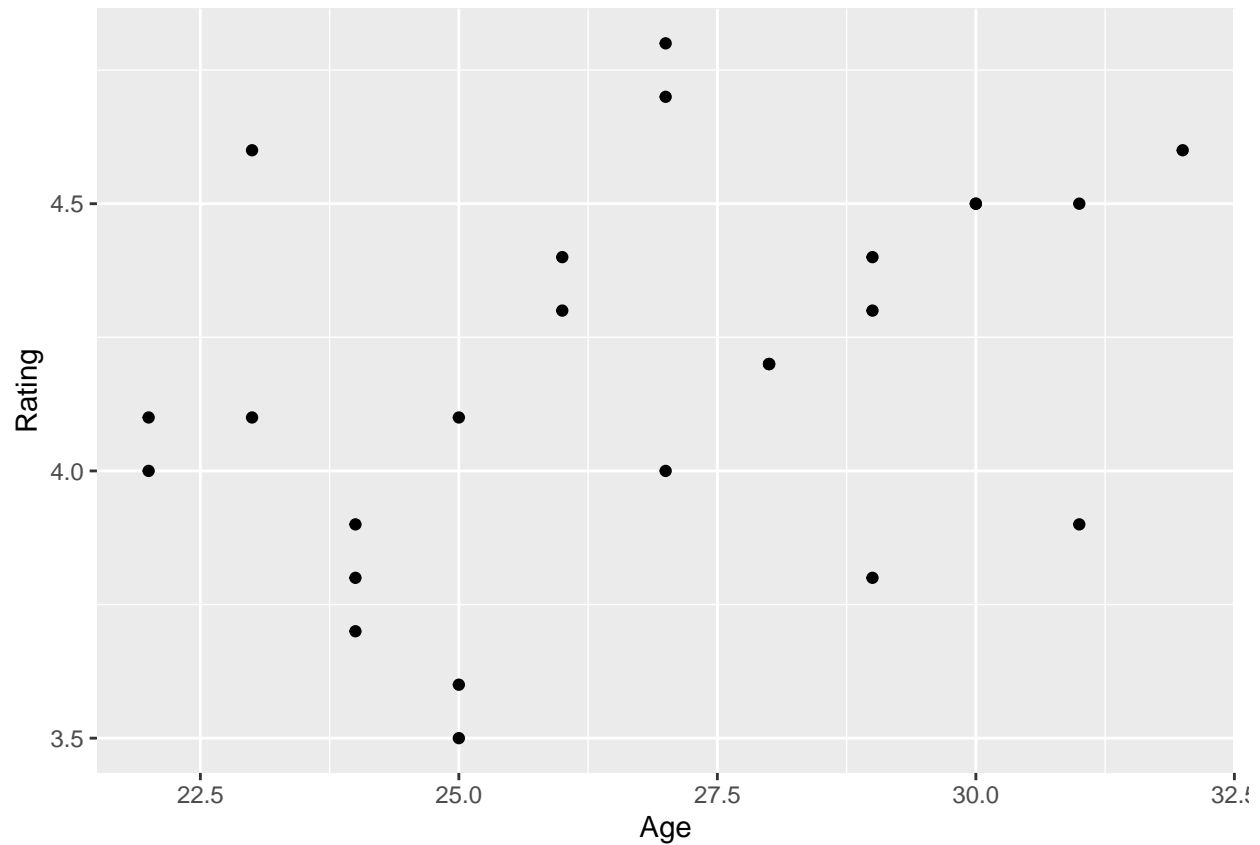
Question 3.3: Represent data using points

Solution:

```
# complete the code to generate the plot
```

```
library(ggplot2)
```

```
ggplot(data=playlist_data,mapping=aes(x=Age,y=Rating)) +  
labs(x="Age",y="Rating")+ geom_point()
```

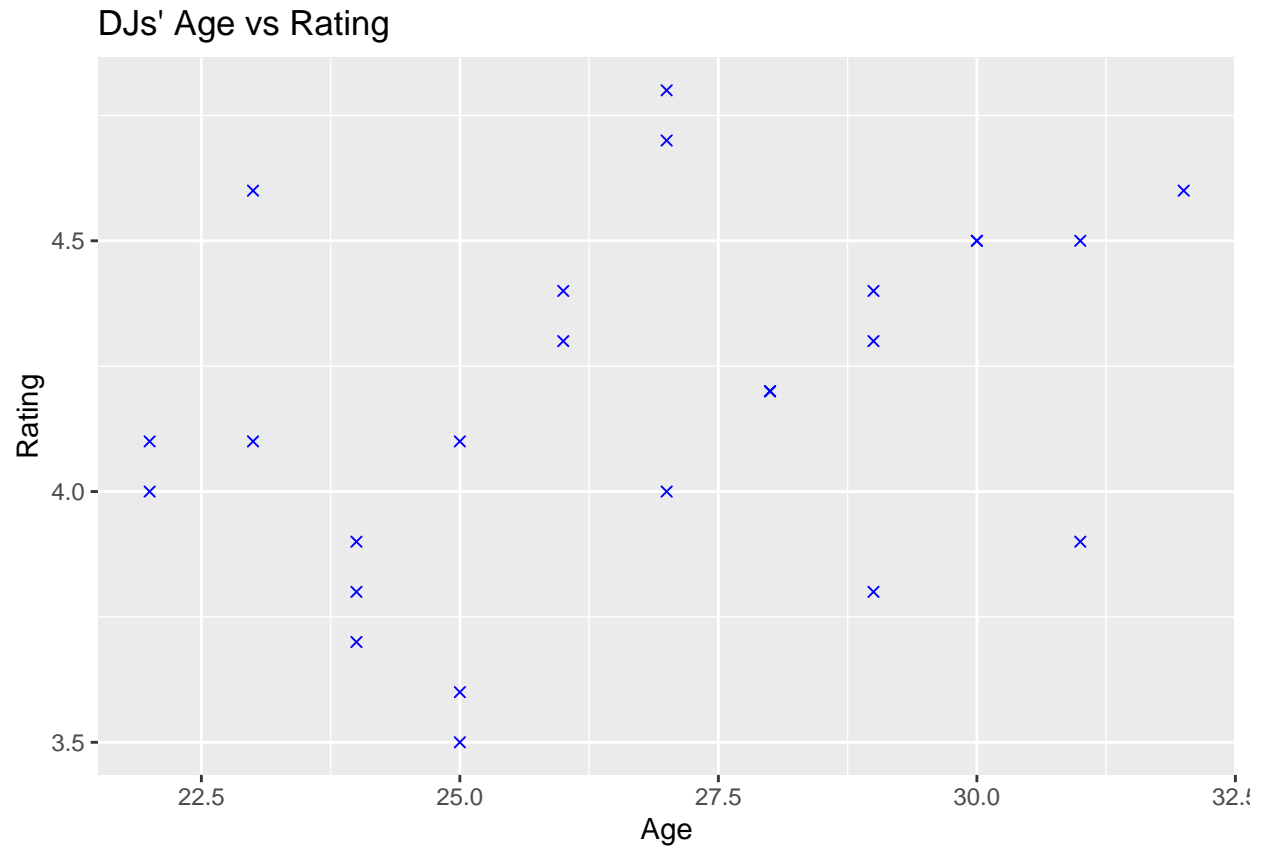



Question 3.4: Can you change the points represented by dots/small circles to any other shape of your liking?

Solution:

```
# complete the code to generate the plot
```

```
library(ggplot2)
ggplot(data=playlist_data,mapping=aes(x=Age,y=Rating)) + geom_point(shape = 4, colour = "blue") + labs(x="Age", y="Rating")
```



Question 3.5: Insert a suitable title and briefly provide your insights in the caption

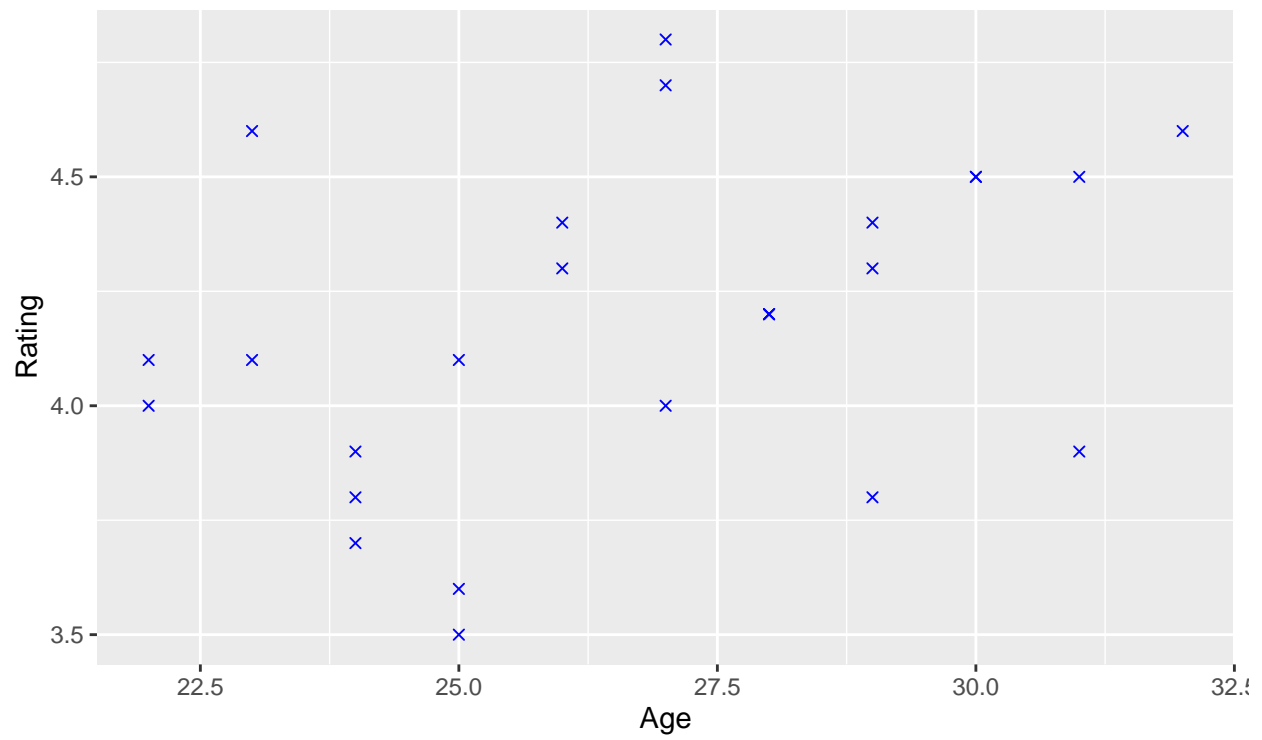
Solution:

```
# complete the code to generate the plot
```

```
library(ggplot2)
```

```
ggplot(data=playlist_data,mapping=aes(x=Age,y=Rating)) + geom_point(shape = 4, colour = "blue") + labs(title="there's no relationship between a DJ's age and their rating")
```

DJs' Age vs Rating



This plot suggests that there's no relationship between a DJ's age and their rating