Challenge-2

Elise Wong

2023-08-21

I. Exploring music preferences

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: The term "CSV" stands for "Comma-Separated Values". CSV files are popular for storing tabular data as they provide a simple and standardised format that can be easily read and processed by most software applications.

Question 1.2: load the `tidyverse` package to work with `.csv` files in R.

Solution:

```
library("tidyverse")
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ─
─
## ✔ dplyr     1.1.0     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.3     ✔ tibble    3.1.8
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ─
─
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force a
ll conflicts to become errors
```

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

Solution:

```
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 messag
e.
```

```
## # 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
```

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

Solution:

```
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 messag
e.
```

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:

```
?read_csv()
```

```
knitr::include_graphics("read_csv()_screenshot.png")
```

More information about the read_csv() command.

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

Solution: The ‘skip’ argument determines the number of lines to skip before reading data, if any. If ‘comment’ is supplied any commented lines are ignored after skipping.

Question 1.7: Display the contents of the data-set.

Solution:

```
playlist_data
```

```
## # 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
```

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:

```r
# I would first ensure that the working directory is correctly set to the location
and/or folder holding the aforementioned CSV file. Then, load the 'tidyverse' packa
ge to work with .csv files in R.
library("tidyverse")

# To import the file into R,
read_csv("sales_data.csv")

# To store the data into a variable named 'sales_data',
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:

```r
head(playlist_data)
```

```
## # A tibble: 6 × 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:

```
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:

```
ncol(playlist_data)
```

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

```
# Therefore, there are 7 columns in the dataset.
```

Question 2.4: What is the total count of DJs?

Solution:

```
nrow(playlist_data)
```

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

```
# There are 26 DJs recorded in the dataset.
```

Question 2.5: Display all the location of all the DJs.

Solution:

```
playlist_data$Location
```

```
##  [1] "City X" "City Y" "City Z" "City X" "City Y" "City Z" "City X" "City Y"
##  [9] "City Z" "City X" "City Y" "City Z" "City X" "City Y" "City Z" "City X"
## [17] "City Y" "City Z" "City X" "City Y" "City Z" "City X" "City Y" "City Z"
## [25] "City X" "City Y"
```

```
unique(playlist_data$Location)
```

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

```
# To view the corresponding locations to each DJ,
library("dplyr")
print(playlist_data %>% select(1,6))
```

```
## # A tibble: 26 × 2
##    DJ_Name Location
##    <chr>   <chr>
##  1 DJ A    City X
##  2 DJ B    City Y
##  3 DJ C    City Z
##  4 DJ D    City X
##  5 DJ E    City Y
##  6 DJ F    City Z
##  7 DJ G    City X
##  8 DJ H    City Y
##  9 DJ I    City Z
## 10 DJ J    City X
## # ℹ 16 more rows
```

Question 2.6: Display the age of the DJs.

Solution:

```
playlist_data$Age
```

```
##  [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
```

```
# To view the corresponding ages to each DJ,
library("dplyr")
print(playlist_data %>% select(1,5))
```

```
## # A tibble: 26 × 2
##    DJ_Name   Age
##    <chr>   <dbl>
##  1 DJ A       28
##  2 DJ B       24
##  3 DJ C       30
##  4 DJ D       22
##  5 DJ E       27
##  6 DJ F       25
##  7 DJ G       29
##  8 DJ H       23
##  9 DJ I       31
## 10 DJ J       26
## # ℹ 16 more rows
```

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.

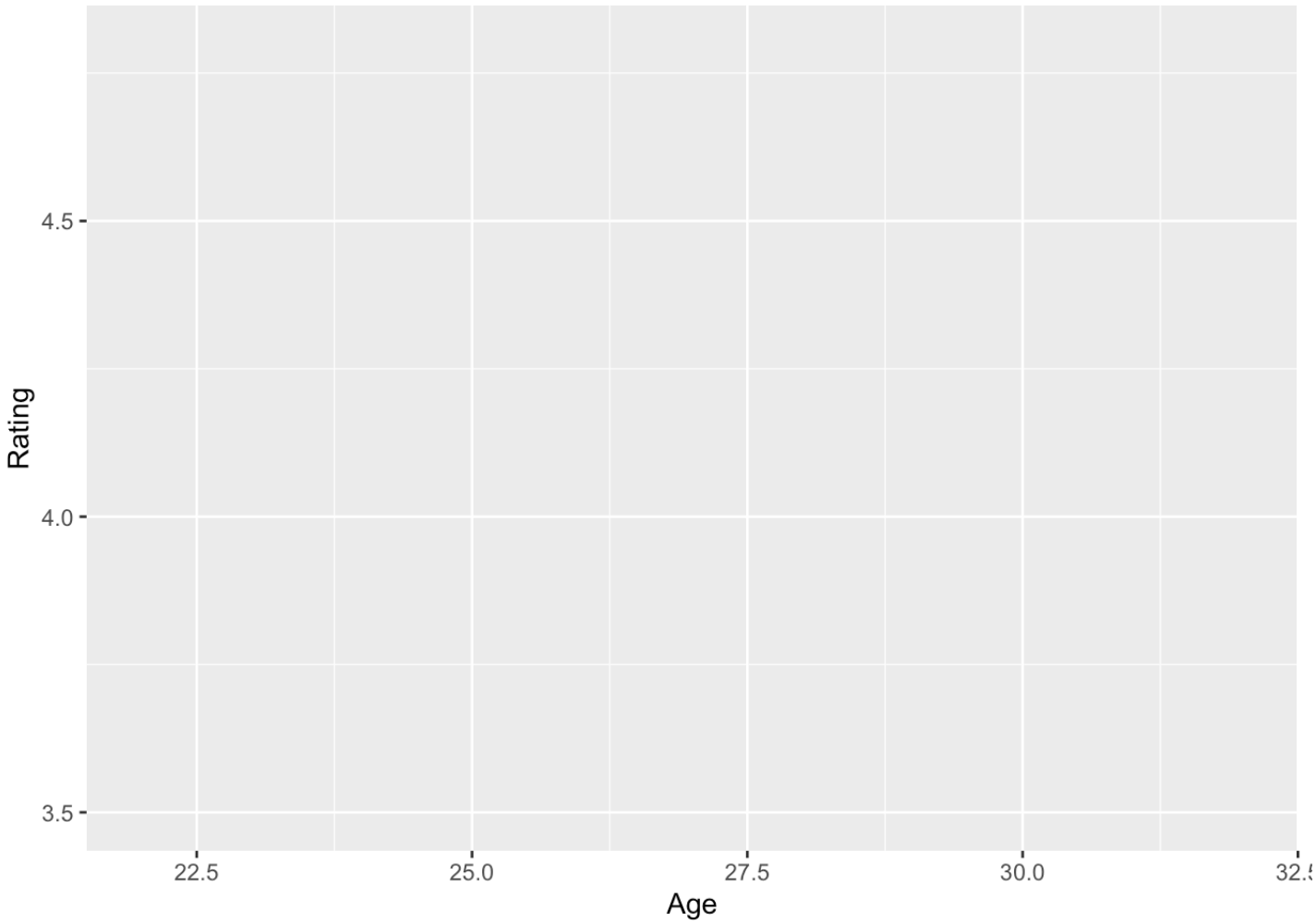
Solution:

```
library("ggplot2")
ggplot(data=playlist_data)
```

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

Solution:

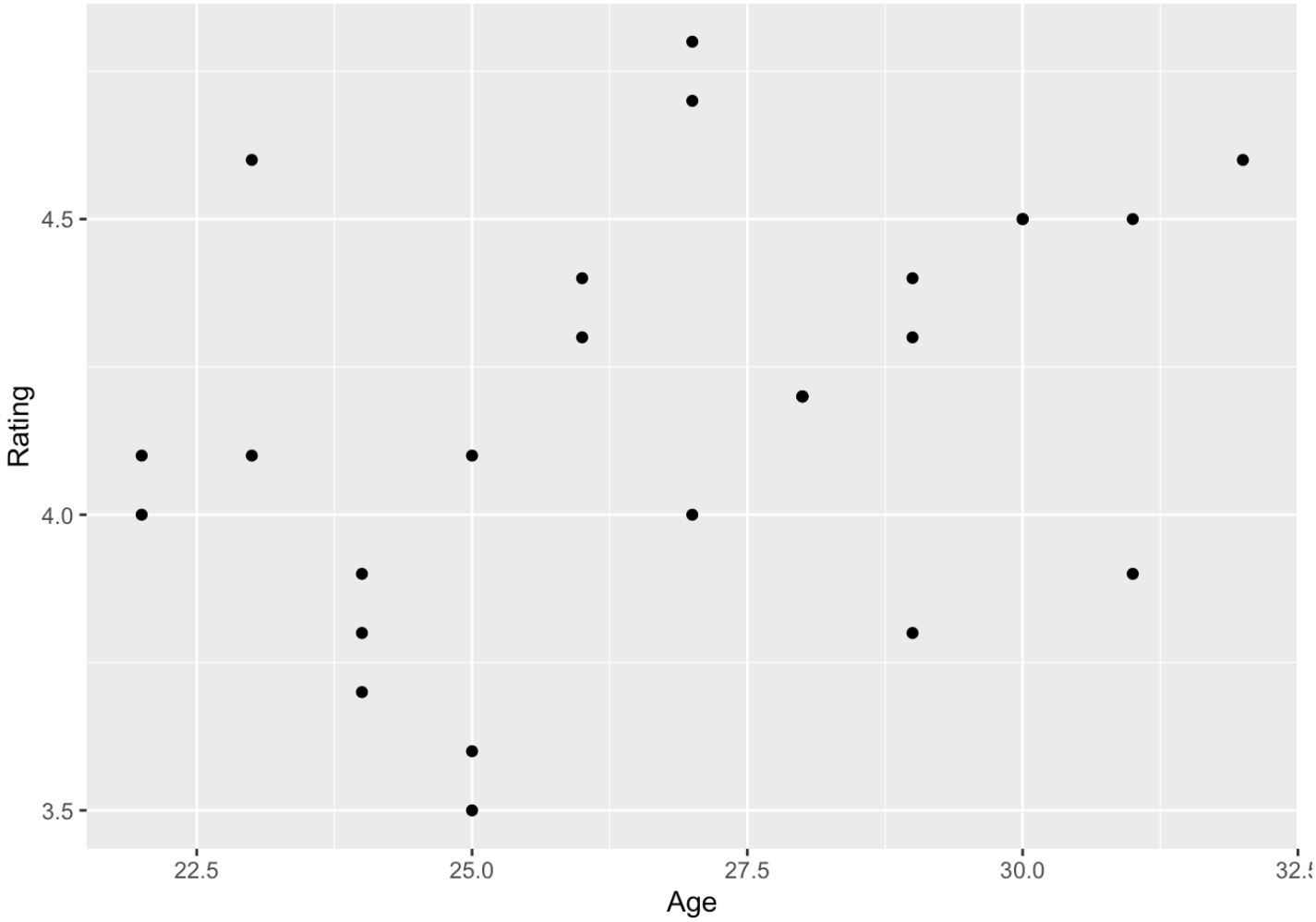
```
ggplot(data=playlist_data, mapping=aes(x=Age,y=Rating))
```



Question 3.3: Represent data using points.

Solution:

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

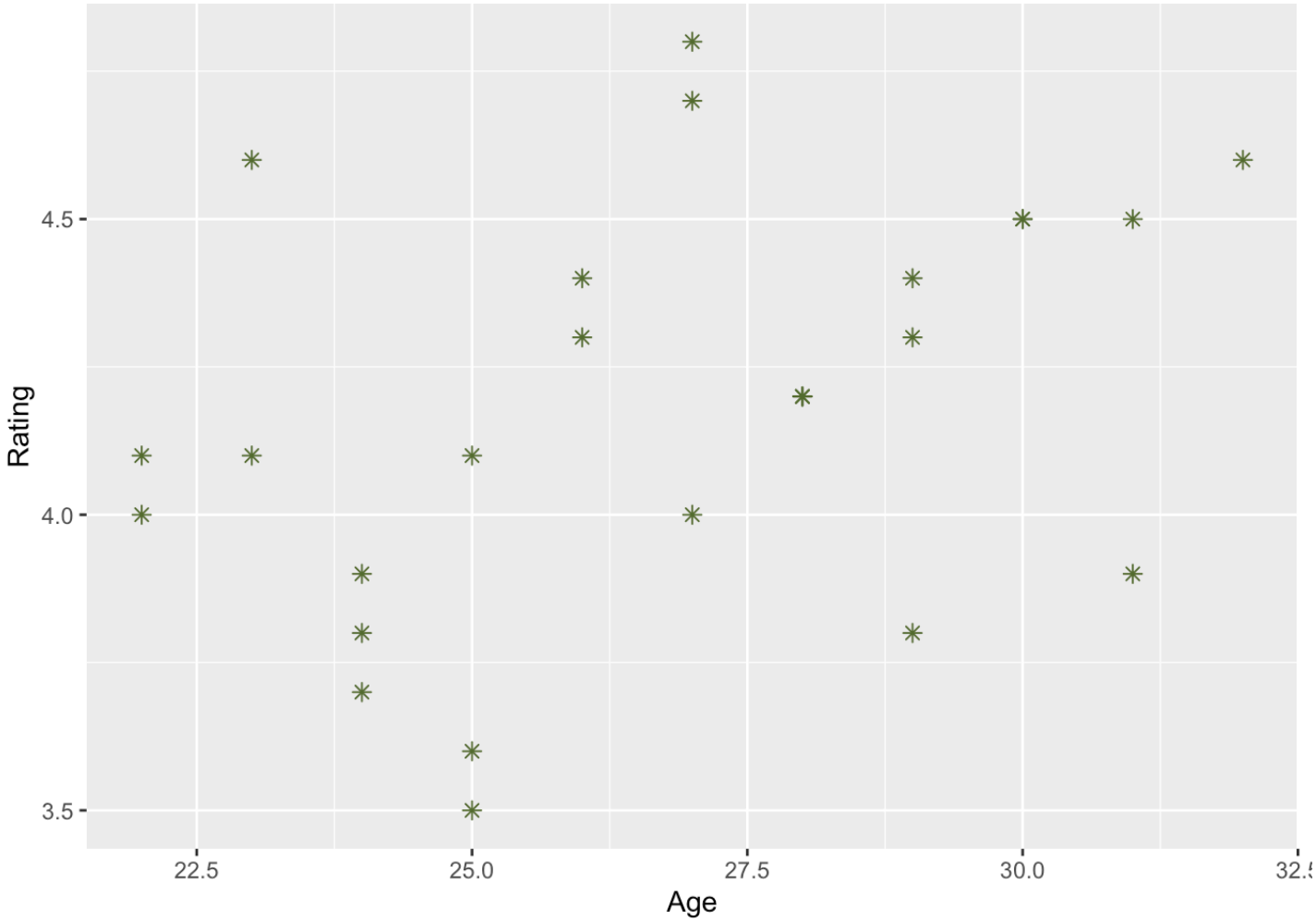


Question 3.4: Can you change the points represented by dots/small circles to any other shape of your liking?

Solution:

```
?geom_point()

ggplot(data=playlist_data, mapping=aes(x=Age,y=Rating)) +
geom_point(shape = "asterisk", color = "darkolivegreen", size = 2)
```

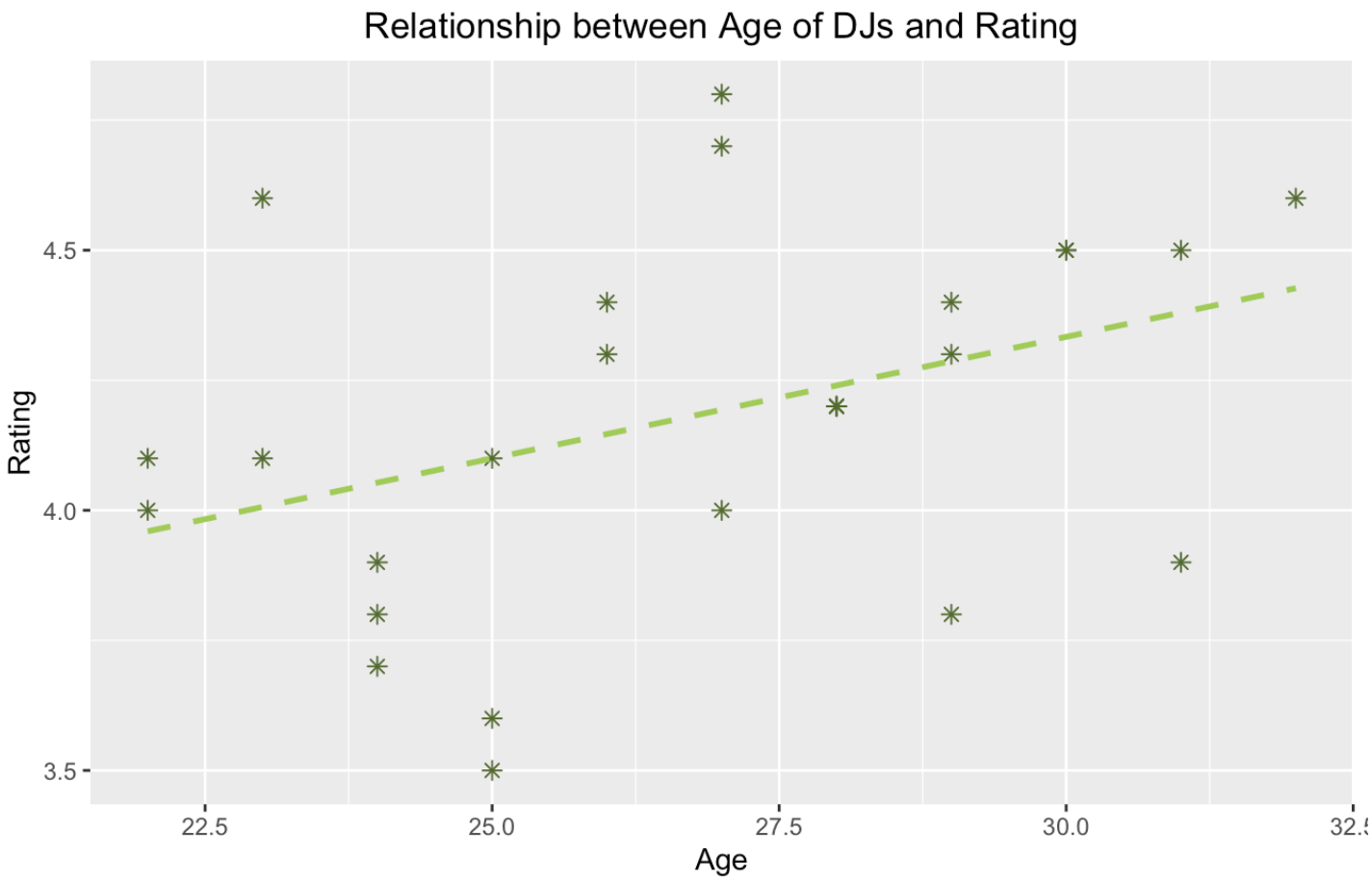


Question 3.5: Insert a suitable title and briefly provide your insights in the caption.

Solution:

```
ggplot(data=playlist_data, mapping=aes(x=Age,y=Rating)) +
geom_point(shape = "asterisk", color = "darkolivegreen", size = 2) +
geom_smooth(method=lm, se=FALSE, col='darkolivegreen3', linetype='dashed') +
labs(title="Relationship between Age of DJs and Rating",
     caption="When a best fit line is plotted, one may conclude that is a general p
ositive trend between Age and Rating of DJ. \n Nevertheless, there can be no clear
correlation or relationship deduced between the age of DJs and their \n respective
ratings from this set of data, due to the presence of many outliers in the scatter
plot.") +
theme(plot.title = element_text(hjust = 0.5),
      plot.caption = element_text(hjust = 0.5)
      )
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



When a best fit line is plotted, one may conclude that is a general positive trend between Age and Rating of DJ. Nevertheless, there can be no clear correlation or relationship deduced between the age of DJs and their respective ratings from this set of data, due to the presence of many outliers in the scatter plot.

Thank you for your time!