Prep1S24

BilalTariq

2024-01-26

Reminder: Prep assignments are to be completed individually.

Reading

The associated reading for the week is Chapter 2, Chapter 3, and Section 8.2. This reading explores key aspects of visualizations, how to build them, and ethical issues around visualizations.

In addition to reading, I recommend you code along with the book examples. You'll see some of the problems below use data from the text. You can try out the code yourself - just be sure to load the mdsr package and any other packages referenced. You can get the code in R script files (basically, files of just R code, not like a .Rmd) from the book website.

Git Workflow Review

1. Before editing this file, verify you are working on the copy saved in *your* repo for the course.
2. Before editing this file, make an initial commit of the file to your repo to add your copy of the assignment.
3. Change your name at the top of the file and get started!
4. You should *save, knit, and commit* the .Qmd or .Rmd file each time you've finished a question, if not more often.
5. You should occasionally *push* the updated version of the file back onto GitHub. You don't need to do this with the .pdf till the end, unless you want to.

6. When you think you are done with the assignment, save the pdf as "*YourFirstInitialYourLastName_thisfilename*.pdf" before committing and pushing (this is generally good practice but also helps me in those times where I need to download all student homework files). For example, I would save this file as AWagaman_Prep1.pdf.

1 - Some basics

Chapter 2 describes Yao's taxonomy for graphics (which is very similar to what ggplot2 uses). The four basic elements are visual cues, coordinate systems, scale, and context.

part a - Position and Length

Solution:

part b - Polar Coordinate System

Solution:

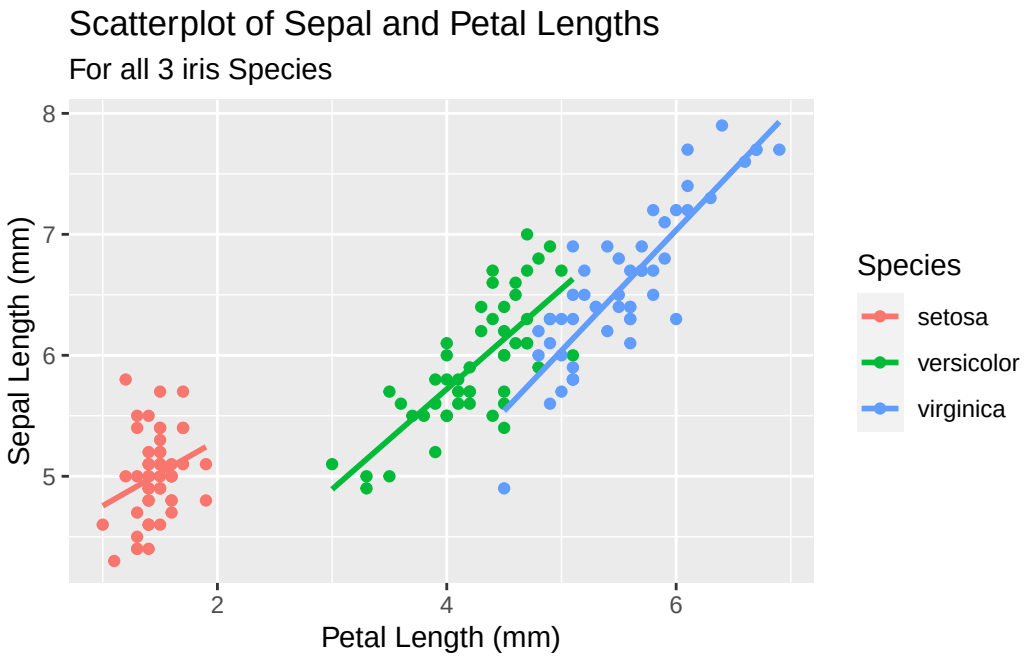
part c - Time

Solution:

part d - Consider the following plot made based on the iris data. What aspects of the plot contribute to its context?

```
data(iris)
ggplot(iris, aes(x = Petal.Length, y = Sepal.Length, color = Species))+
  geom_point()+
  geom_smooth(method = 'lm', se = FALSE)+
  labs(y = "Sepal Length (mm)",
       x = "Petal Length (mm)",
       title = "Scatterplot of Sepal and Petal Lengths",
       subtitle = "For all 3 iris Species")
```

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



Solution: It has a clear Title ("Scatterplot of Sepal and Petal Lengths") with an adequate subtitle ("For all 3 iris species"). It has axis labels of Petal Length and Sepal Length. Additionally, it has a legend on the side to identify the different lines.

2 - GDP and education

part a - Figure 3.3 in Section 3.1.1 shows a scatterplot that uses both location and label as aesthetics. Reproduce this figure.

Hint: you'll need to define 'g' based on code from earlier in Section 3.1.1. Also, make sure you load the packages in the ***`setup`*** *chunk!*

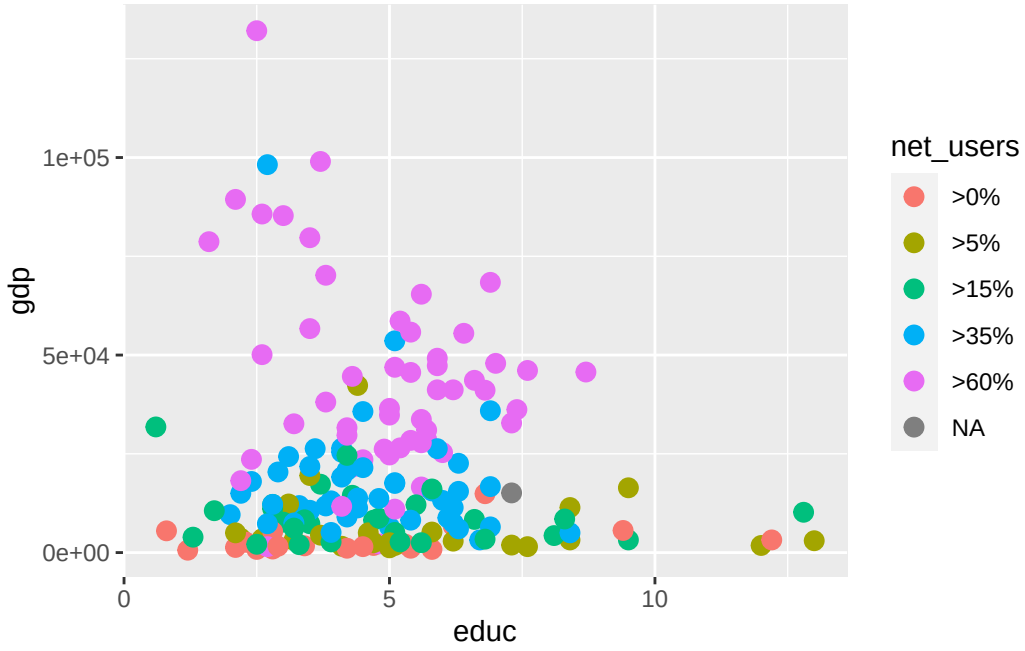
Solution:

```
data(CIACountries)

# define the plot object
g <- ggplot(data=CIACountries,aes(y=gdp,x=educ))

# print the plot
g + geom_point(aes(color=net_users),size=3)
```

```
Warning: Removed 64 rows containing missing values (`geom_point()`).
```



part b - Now, update the plot with more informative labels. Label the x-axis "% of GDP spent on education" and the y-axis "Gross Domestic Product (GDP)".

Hint: see Section 3.2.1 for an example of one way to label the axes.

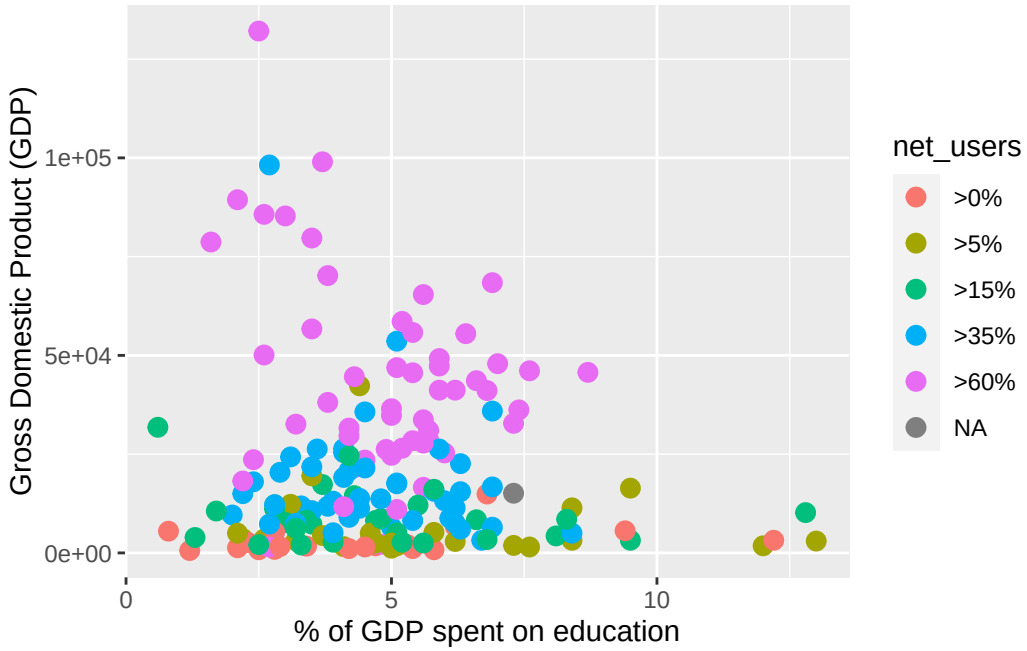
Solution:

```
data(CIACountries)

g <- ggplot(data=CIACountries,aes(y=gdp,x=educ))

g + geom_point(aes(color=net_users),size=3) +
  scale_y_continuous(name= "Gross Domestic Product (GDP)") +
  scale_x_continuous(name= "% of GDP spent on education")
```

```
Warning: Removed 64 rows containing missing values (`geom_point()`).
```



part c - Next, move the legend so that it's located on the top of the plot as opposed to the right of the plot.

Hint: see Section 3.1.4 for an example on how to change the legend position.

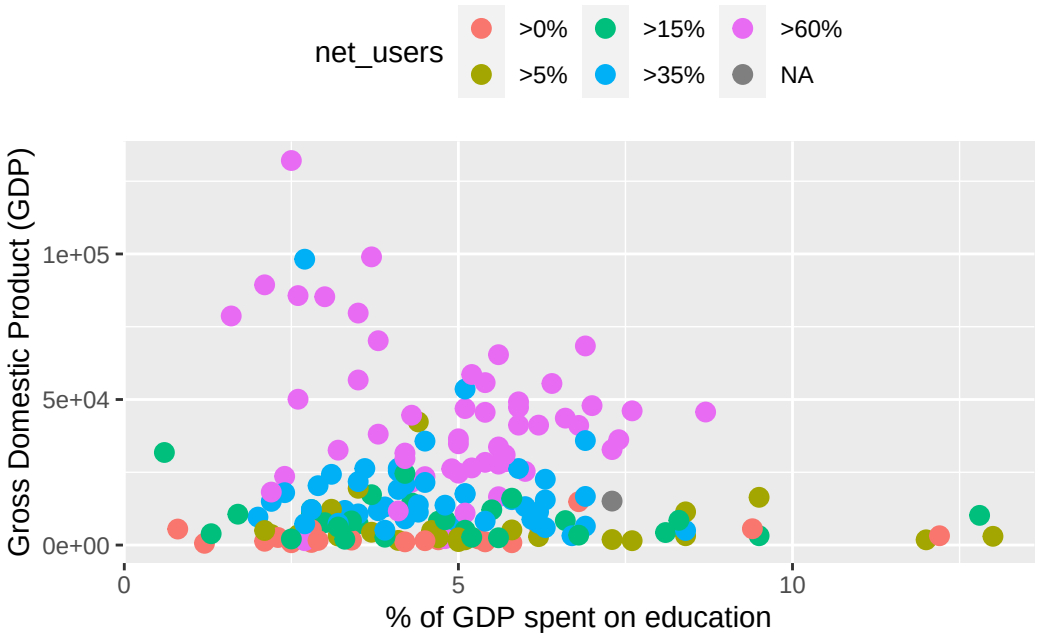
Solution:

```
data(CIACountries)

g <- ggplot(data=CIACountries,aes(y=gdp,x=educ))

g + geom_point(aes(color=net_users),size=3) +
  scale_y_continuous(name= "Gross Domestic Product (GDP)") +
  scale_x_continuous(name= "% of GDP spent on education") +
  theme(legend.position = "top")
```

```
Warning: Removed 64 rows containing missing values (`geom_point()`).
```



part d - Lastly, Section 3.1.2 discusses *scale*, and demonstrates how to display GDP on a logarithmic scale to better discern differences in GDP. Update the figure so GDP is on a log10 scale.

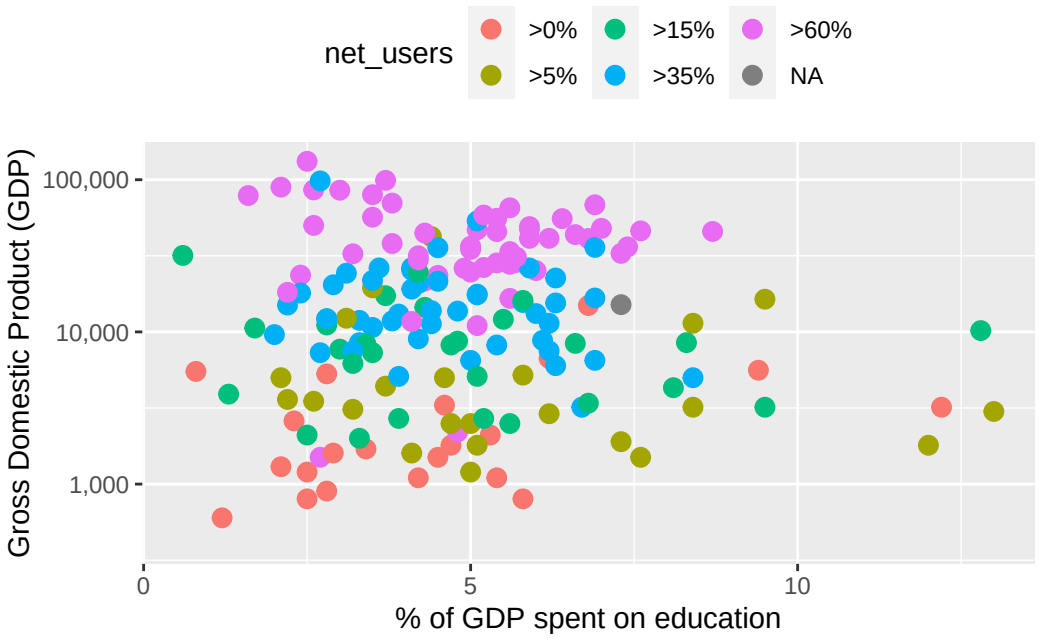
Solution:

```
data(CIACountries)

g <- ggplot(data=CIACountries,aes(y=gdp,x=educ))
```

```
g + geom_point(aes(color=net_users),size=3) +
  scale_y_continuous(name= "Gross Domestic Product (GDP)",trans = "log10",labels = scales:
  scale_x_continuous(name= "% of GDP spent on education") +
  theme(legend.position = "top")
```

```
Warning: Removed 64 rows containing missing values (`geom_point()`).
```



3 - Demos with Iris

The *iris* data set contains 5 variables on 3 species of iris. There are 4 measurement variables and the Species variable. We can look at a data set to get a sense of its structure with *glimpse*, *str* or *summary* - all provide a quick overview of the data set. We can look at the first few observations with *head*. To make plots, one must know what plots are appropriate for the variables involved.

Note: for this problem, you aren't actually making any of the plots.

> part a - Use one of data set overview commands to look at iris. List whether each variable is numeric or categorical (quantitative vs. qualitative).

Solution: Numeric: Sepal length, Sepal width, Petal Length, Petal width
Categorical: Species

```
data(iris)
summary(iris)
```

```
 Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
Median :5.800   Median :3.000   Median :4.350   Median :1.300
Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
       Species
setosa    :50
versicolor:50
virginica :50
```

> part b - Suppose we want to examine the distribution of Petal.Width. What plot would you recommend for this?

Solution: A scatter plot that uses color to identify the different species

> part c - Suppose we want to examine the distribution of Species. What plot would you recommend for this?

Solution: A histogram

part d - We want to examine the relationship between Petal.Width and Petal.Length across Species. However, our client insists the graphic be printed in black and white. What plot would you recommend for this? Be sure to name the plot and describe how all three variables are represented/included.

Solution: We will be using a scatter plot with Petal Width on the x axis and Petal Length on the y axis. Furthermore, there will be a legend on the side which will differentiate between the 3 species by using different types of shading (i.e filled circle, half filled circle, dotted circle).

4 - Learning about R functions

part a - Consider Figure 3.8 in Section 3.1.5. What does `reorder(drg, mean_charge)` do? To figure this out, recreate the plot, but use `x = drg` instead of `x = reorder(drg, mean_charge)`. What happens?

Solution: We see that the reorder function sorts the x values with the lowest Statewide average Charges at the left and the highest Statewide average charges to the right.

```
data(MedicareCharges)
ChargesNJ <- MedicareCharges %>%
  ungroup() %>%
  filter(stateProvider == "NJ")

# create the plot object
p <- ggplot(
  data = ChargesNJ,
  aes(x = reorder(drg, mean_charge), y = mean_charge)
) +
  geom_col(fill = "gray") +
  ylab("Statewide Average Charges ($)") +
  xlab("Medical Procedure (DRG)") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size = rel(0.5)))

# print the plot
p
```

part b - Another way to learn about a function is via the help menu. You can search directly (search window in the Help pane) or type ?functionname in the console. Try this with *reorder*. From the help results, report the package that *reorder* is from and the name of the data set used in the help menu example.

Hint: The package is in curly brackets after the name of the function in the help menu.

Solution: It is from the stats package. The dataset used in the help example is InsectSprays.

Note: Normally, you'd look through the help information to learn about the function, it's arguments, outputs, etc. This problem is designed so you learn about the help menu, not to really focus on this function. That said, you should know what *reorder* does at the end of the problem. You can also access R documentation via the internet, rather than through R.

part c - Sometimes different packages have functions with shared names that don't do the same thing. We don't have a lot of packages loaded here, but we can demo

this with *filter*. Try looking up *filter*. What two packages are listed as having results for a *filter* function?

Hint: You can also find this where the packages are loaded in above. If there are conflicts, the most recently loaded package's function masks the function from previously loaded packages. You should see a notice of masking here in regards to filter (and lag).

Solution: The stats package and the dplyr package have a result for a filter function.

Note: If you are ever worried about a function conflict between packages, or want to use just one function (or data set) from a package without loading the entire rest of the package, you can reference the function this way - package::function. You will see this occasionally in our work this semester.