Class10 - Assignment

FedericaFusi

3/12/2021

INSTRUCTIONS

For this assignment, we are going to apply some functions from the **forcats** package discussed in class while creating some new plot types: a **heatmap**, a **lollipop chart**, and a **Cleveland dot chart**.

Data

Data and the related codebooks can be found on the Tidytuesday github page provided here.

For this assignment, we are going to use only two datasets **tuition_cost** and **diversity_school**.

```
#devtools::install_github("thebioengineer/tidytuesdayR")

#tuesdata <- tidytuesdayR::tt_load('2020-03-10')

#tuition_cost <- tuesdata$tuition_cost
#diversity_school <- tuesdata$diversity_school
```

A few general requirements:

- Each plot should be done by using very few pipes (ideally one pipe!) - i.e., try to be as efficient and organized as possible when writing your code (which means you might need to 'clean up' your code once you are done to optimize it).
- Each plot should have appropriate title, subtitle, axis labels and titles, and legend titles and labels. Below you'll find additional instructions on how to add the data source as a caption.
- Colors should be used appropriately.

A STEP-BY-STEP GUIDE

Follow the various steps below to complete your assignment.

If you are not familiar with the new plot types, I encourage you to check out the **from Data to Viz** and the **R Graph Gallery** website. They both provide useful code examples and key information on how to get your graph right.

Step 0 - Create a customized theme

Before moving onto creating some plots, create your own customized theme to be applied to all of your plots. Since we are creating more than one plot, a customized theme will save us time and ensure that our plots have a consistent graphics. If you need a review, we talked about this in Class 7, slide 73.

At a minimum, make sure to:

- Pick one of ggplot2 theme as your basis
- Change the format of your plot title to be bold, size 16, and the format of your subtitle to be italic, size 14.
- Axis titles should be at least 14 and bold.
- Axis labels and all legend elements (title and labels) should be at least size 12.
- Set at least other two elements according to your preferences (e.g., try to change the font!)

HEATMAPS

Examples of heatmaps are provided on the data-to-viz website here and on R graph gallery here.

We want to create a heatmap illustrating the **average in-state tuition cost by state and school type**.

- Data come from the **tuition_cost** dataset
- **States** should be displayed on the y-axis in alphabetical order (e.g., Arizona should be the first state at the top)
- **School type** should be displayed on the x-axis based on in-state tuition cost (e.g., the school type with the lowest in-state tuition should appear first on the left and the school type with the highest in-state tuition should appear last on your right).
- The different "tiles" (or "squares") should be colored based on the **average in-state tuition cost** by state and school type.
- In a heatmap, colors represent numeric values in a sequential way (e.g., from smaller to larger). As a result, we want colors to run from lighter (smaller values) to darker (higher values). The minimum setting here is to use **scale_fill_gradient(low = "white", high = "blue")** where you manually set the ligher and darker color. The other option is to use a **sequential palette**.

Step 1 - Create your dataset

Start by cleaning up your data - Which variables do you need for your graph? Which ones do you have and which ones are missing and need to be created?

This represents the first part of your pipe.

Step 2 - Create the plot

The function to create a heatmap is **geom_tile**. See the hint below. You need to set your x and y in the aes as we usually do. You also need to set up the fill of your tiles!

```
geom_tile(aes(x = .......,
              y = .......,
              fill = ........))
```

Step 1 + Step 2 provide the skeleton of your graph. See what it looks like and make appropriate adjustment before moving forward.

Step 3 - Sequential paleteetess and colors

For this plot, make sure to use a palette. Palettes are predefined set of colors that you can apply to your graphs. There are SEVERAL palettes in R. Common ones include R Brewer, viridis, rcartocolor. A full list of palettes can be found here. Note that to use palettes you need to install the package and read the appropriate documentation to see how to use the palette within the scale_fill command.

Experiment with one palette of your choice! If you gest stuck, see the example below to help you out.

```
# Using a palette from rcartocolors package
# First install the palette package
install.package(rcartocolors)
library(rcartocolor)

# The documentation says that the function is _carto_c whereas _c stays for "continuous". This indicate

# We apply the palette to scale_fill since we are chaging the colors related to the argument "fill" in

# I chose the palette "BluGrn"

scale_fill_carto_c(palette = "BluGrn", direction = 1)

# Change direction = 1 to direction = -1 and see what happens! It is a pretty useful command for a heat
```

Step 4 - Finalize your plot

Once everything looks good, remember to:

1. Make sure to use factors to organize your graph
2. Add title, subtitles, axis labels, etc...
3. Add your theme to the graph

Pro-tip: You can set up titles and labels within the **labs** function. You can also add a **caption** to your title to add the data source. Try it out!

```
  labs(title = ........,
       subtitle = .......,
       x = .......,  # x-axis title
       y = .......,  # y-axis title
       fill = ........, #legend title
       caption = "Data come from the Chronicle of Higher Education")
```

LOLLIPOP CHARTS

Lollipop charts are a variation of barplots (see a bit about them here on the from-data-to-viz website), which are used to show the value of a set of categories.

We are using data from the diversity_school AND the tuition_cost dataset to show the percentage of a given group of students across institutions.

Step 0 - Tidy the data

The first step is to tidy the diversity_school dataset and merge it with the tuition dataset. Note that we want to keep **all but only** the observations in the tuition cost datasets.

Step 1 - Prepare your dataset

We are going to use a lollipop chart to show the **percentage of a group of students within each institution**. You can focus on the percentage (on the total enrollment) of women, Hispanics, Black, non-white students, and so on... depending on your interest;

We will include only * 4-year * private or public institutions * in Illinois * whose total_enrollment is higher than the state median.

As a first step, write the pipe to obtain the dataset that you need to draw your graph.

Step 2 - Lollipop graph

Lollipop graphs combine two geom functions: **geom_point** and **geom_segment**.

We used geom_point before to create a scatterplot. We can use it here in the same way: set your geom_point such that institution names will appear in the y-axis and percentage of on the x-axis. See what happens!

```
geom_point(aes(x = ......,
               y = ......))
```

Now we want to add segments to that go from zero to the dot. In geom_segment, **x** and **y** indicate the start of the segment while **xend** and **yend** the end of the segment. Note that for categorical variables the start and end point is the same and corresponds to the categorical variable itself.

```
geom_segment(aes(x = ......,
                 xend = ........,
                 y = .......,
                 yend = ......))
```

> Once you figured out the two codes above, combine them in one pipe to create the lollipop chart. I suggest putting geom_segment first and then geom_point (try it both ways to see the difference).

Step 3 - Finalize your graph

Make sure to

- order schools from highest percentage to lowest percentage by using factors.
- give a nice color to your dots and segments.
- change the size of your dots to make them larger.
- add titles, subtitles, labels, and your theme.

Pro-tip: If you wish, you can add labels to your dots to indicate the different percentages by using geom_text. It's pretty simple: geom_text wants to know where labels should be located by using the x and y axes as reference points. l**label** indicates the variable from which label names should be retrieved (e.g., the percentages). The **round** function is used to round your percentage at 2 decimal points.

```
geom_text(aes(x = ........,
              y = ........,
              label = round(........, 2)),
          size = ......) +
```

CLEVELAND DOT CHARTS

Cleveland dot charts are used to show the difference between two values within the same group - e.g., differences in enrollment from year 1 to year 2 or difference in enrollment between men and women, white and non-white students, and so on. See a few examples in the from-data-to-viz website or the r graph gallery webiste.

We want two show the gap in enrollment between two groups (e.g., men and women enrollment, Hispanic and non-Hispanic students and so on).

Step 0 - Change data format

The easiest way to create a Cleveland dot charts is to "mess" up your data a bit and put them into a long format such that your data looks like this:

school_name	group	perc
school1	group1	XX%
school1	group2	XX%
school2	group1	XX%
school2	group2	XX%
school3	group1	XX%
school4	group2	XX%

Think about the variables that you need for the graph. Keep only the relative columns and transform your data into a long dataset. We are still looking at the **same subset of schools** (in Illinois, 4-year degree, public or private, and whose enrollment is higher than the state median).

Step 2 - The plot

To create the plot, we need geom_point and geom_line. The process is pretty similar to what we did before.

When combine two different geom functions, it might be more practical to write down the main features of your graph in the ggplot function. In other words, it might be easier to set up your x and y axes in ggplot

and specify only distinctive features in the geom functions. Let's try it here. Use the hint below to create the skeleton of your plot

```
ggplot(aes(x = ..........,          # x axis
           y = ..........)) +       # y axis

# geom_line connects observations in your plot in order of the variable on the x axis.
# You need to specify how observations on the graph should be grouped.
# In this case, what is the unit of analysis
geom_line(aes(group = .......)) +

# geom_point creates the dots on your graph.
# x and y are already specify in ggplot
# You can run geom_point without further specifications.
geom_point()
```

Once this part is figured out, move onto the next two steps to complete your graph.

Step 3 - Colors and order

Make sure that your graph is well organized. In particular, the plot should look like an hourglass, where schools with a 50-50 percentage are in the middle and schools with a very high (or very low) percentage of one group are the extremes.

Use color to identify the two group in your graph - i.e., all dots indicating the percentage of one group should be of the same color. Include a proper legend with nice and explocative labels. Consider using **recode** to change the legend's labels.

Step 4 - Refine your graph

Add titles, subtitles, labels, and so on. Add your theme to the graph.