Lab 2: Practicing Data Visualization*

Getting started

Click the Github Assignment link on Canvas, then open it in Posit Cloud as a new project from a GitHub repository.

Warm up

Let's warm up with some simple exercises. Open the `lab2.qmd` Quarto file. Update the YAML of your Quarto file with your information, render, commit, and push your changes. Make sure to commit with a meaningful commit message. Then, go to your repo on GitHub and confirm that your changes are visible in your qmd file. If anything is missing, commit and push again.

Packages

We'll use the **tidyverse** package for data wrangling and visualization. We will also use **lubridate** for working with dates.

You can load them running the following chunk:

```
library(tidyverse)
library(lubridate)
```

Data

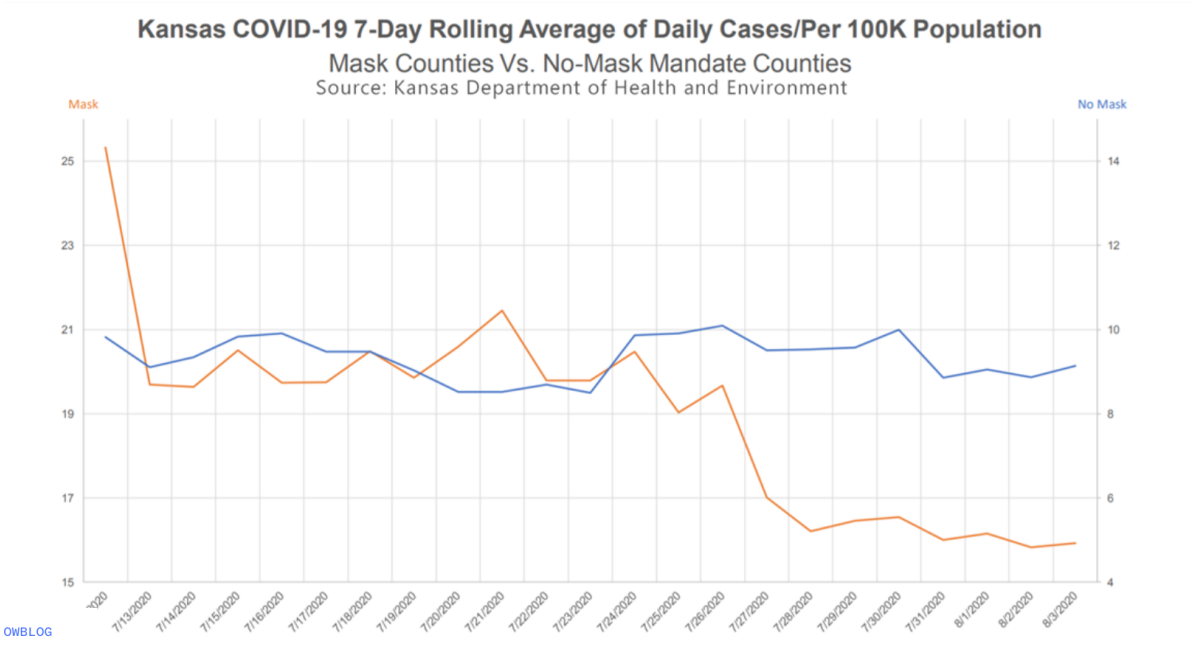
The first part of this lab does not require any data. The second part uses UK pay gap data collected by the UK government.

*Adapted in part from Data Science in a Box

Exercises

Part 1: Improving Data Visualization

The following visualization was shared on Twitter as "extraordinary misleading".



Think about what is misleading about this visualization and how you might go about fixing it.

1. Create a data frame that can be used to re-construct this visualization. You may need to guess some of the numbers, that's ok. You should first think about how many rows and columns you'll need and what you want to call your variables.
2. Make a visualization that more accurately (and honestly) tells the story.
3. What message is more clear in your visualization than it was in the original visualization?
4. What, if any, useful information do these data and your visualization tell us about mask wearing and COVID? It'll be difficult to set aside what you already know about mask wearing, but you should try to focus only on what this visualization tells. Feel free to also comment on whether that lines up with what you know about mask wearing.

> ❗ Remember to Commit!
>
> *Render, commit, and push your changes to GitHub with an appropriate commit message.*

Part 2: Visualizing the Pay Gap in the UK

You can find the data for this section in `data/paygap.csv`. These data come from https://gender-pay-gap.service.gov.uk/viewing/download. There are quite a few variables contained in this dataset; you can access the full codebook here; scroll down to "Data Dictionary".[1]

Each row represents a year-company.

Some (but not all!) important variables:

Variable	Description
`diff_mean_hourly_percent`	Mean% difference between male and female hourly pay (negative = women's mean hourly pay is higher)
`diff_median_hourly_percent`	Median% difference between male and female hourly pay (negative = women's median hourly pay is higher)
`employer_id`	Unique ID assigned to each employer that is consistent across every reporting year
`employer_size`	Number of employees employed by an employer
`due_date`	The date that the GPG data should have been submitted by. Format: dd/MM/yyyy HH:mm:ss

1. Import the data. We can consider the year part of `due_date` as the information about the year that the data covers. Create a new variable called just `year` that is the year value of `due_date` minus one.

   Hint:: The **lubridate** package has a function exactly for this purpose. Check out the cheatsheet here: https://rawgit.com/rstudio/cheatsheets/main/lubridate.pdf

2. How many observations are there per year? Do you notice anything strange here? Why might this be? **Hint:** Look at the help file of the `count()` function.

How many unique companies are in the dataset? **Hint**: R has a handy `unique()` function.

3. Is there any evidence that the pay gap decreased over the time period covered by this data set? Use at least one visualization to answer this question.

[1] Note: You will have to translate a bit between the codebook variable names and the data variable names, which have been turned from camel case to snake case.

4. Does there seem to be a relationship between the size of a company and the pay gap? Use at least one visualization to answer this question and explain how you arrived at your conclusion.

> ❗ Remember to Commit!
>
> *Render, commit, and push your changes to GitHub with an appropriate commit message.*

Final Steps

When you have made your final edits, remember to render, stage the changed files, then commit, and push to the repo.

When you are done with the lab (have made your last commit and pushed it to your repo), make sure to *close* the project by going to File »> Close Project. This will bring you back to the original RStudio instance.

Grading

Please put your names on your answer document and add the date. You won't get points for doing this, but you will *lose* points for not doing so.

For clarity's sake, you should label your code chunks (this helps make debugging easier). You will *lose* points for not naming chunks.

If an exercise asks a question, you should answer it in narrative form and not just rely on the code. You will be *penalized* otherwise.

Please suppress warnings and messages. You will *lose* points if you do not suppress warnings and messages.

If you are asked to make a visualization, make sure that it is presentable, professional in appearance, has a title, and has properly labeled axes. You will *lose* points if you do not do this.

The different components of this lab are worth the following points:

- Part 1
  - Q1: 2 pts
  - Q2: 4 pts
  - Q3: 2 pts
  - Q4: 2 pts

- Part 2
  - Q1: 2 pts
  - Q2: 3 pts
  - Q3: 4 pts
  - Q4: 4 pts
- Workflow (includes making sufficient commits and labeling chunks): 4 pts

Total: 27 pts