Practical 2

March 31, 2025

1 Hands-On Exercises: Part 2

1. First, install and load the required packages: **Shiny**, **vroom** (for fast file reading), and **tidyverse** (for data analysis).

2. Download data from GitHub: data

3. Extract **injuries**, **products** and **population** dataset using **vroom** function.

4. Extract injuries data related to product **649, "toilets"**

5. Summarize **toilet-related injuries** by **location**, **body part**, and **diagnosis**. Apply the **weight** variable to estimate total injuries across the U.S.

6. Explore the injury patterns across **age** and **sex**. Furthermore, compare the number of people injured with the total population and calculate an injury rate.

7. Design a prototype for a Shiny app that:
- **Has one row** for the input (allowing for future expansion of inputs).
- **Includes one row** for three tables, each occupying 4 columns out of a 12-column layout.
- **Contains one row** for a plot displaying relevant data.

Variables to use:
- `prod_codes` for the product selection input.
- `selected()` for filtering data based on the chosen product code.
- `output$diag`, `output$body_part`, and `output$location` for rendering the three tables.
- `summary()` for computing aggregated data.
- `output$age_sex` for rendering the plot.

8. How would you refine the tables in a Shiny app to display only the most important information while keeping them easy to interpret?

Requirements:
- **Truncate the tables** to show only the most relevant data.
- Use **forcats functions** to:
1. Convert the variable to a factor.
2. Order the levels by frequency.
3. Group all levels beyond the top 5 into a single category.
- Implement this approach for `diag`, `body_part`, and `location` tables using the `count_top()` function.
- Ensure the tables remain user-friendly and focused.

9. How would you modify a Shiny app to allow users to toggle between viewing raw injury counts and population-standardized rates in a plot?

Requirements:
- Add a **selectInput()** to let users choose between `"rate"` and `"count"` for the Y-axis.
- Ensure the default selection is `"rate"` for better interpretability.
- Conditionally update the plot based on the user's selection:
- Display **raw injury counts** if `"count"` is selected.
- Display **population-standardized rates** if `"rate"` is selected.
- Maintain flexibility for adding more visualization options in the future.

10. How would you enhance a Shiny app to allow users to interactively explore narratives related to injury data?

Requirements:
- Add a **new row at the bottom** of the UI to display narratives.
- Include an **action button (`actionButton()`)** labeled `"Tell me a story"` to trigger the display of a new narrative.
- Use `textOutput()` to show the selected narrative.
- Implement `eventReactive()` to update the narrative **only when the button is clicked** or when the selected product changes.
- Ensure the narrative is sampled dynamically from the available dataset.

[ ]: