ES 2 Lab 6 – Histograms

Before starting this lab, you should complete the Jupyter notebooks on summary statistics and histograms. You can copy them into your workspace with the commands:

```
cp /cluster/tufts/class/es/2/shared/summarystats.ipynb ./
cp /cluster/tufts/class/es/2/shared/histograms.ipynb ./
```

Introduction

Summary statistics such as mean and median are convenient, but often they fail to tell the true story of the data. In the lab, you'll plot some histograms of COVID cases by age, and examine how simple choices greatly influence the outcome.

The CDC publishes an anonymized dataset of every of positive case reported in the country, available at https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data/vbim-akqf. You can download it as a CSV file, but with over 20 million rows, it's a little too much for our purposes.

Instead, we've prepared a subset of this data with only every 100th patient and some of the data fields removed. Our version of the data looks like this:

```
date,sex,age,death
2020/04/07,F,5,N
2020/04/22,F,10,N
2020/04/21,F,4,N
2020/04/28,M,0,N
2020/05/04,F,6,U
```

The data fields are:

- `date`: The earliest known date of infection
- `sex`: M/F/O/U for male/female/other/unknown
- `age`: an integer representing the patient's age
- `death`: Whether the individual died of COVID, Y/N/U for yes/no/unknown

You can read the full details of the original data fields on the CDC web page.

Tasks

Your overall task is to draw two histograms: one with confirmed cases as a function of age, and one with deaths as a function of age.

1) Download the data from the course website. If you're using JupyterLab, you can just use the path `/cluster/tufts/class/es/2/shared/data/covid_cases.csv`.

2) First you'll need to read the CSV file and create a list containing the age of every individual. Write metacode describing how you will do this. Save your metacode; you'll need to turn it in with your final submission.

Unlike previous labs, we want you to develop the metacode yourself. Think of it as taking off the training wheels: breaking a problem into small chunks and writing metacode is to programming what balancing is to riding a bike. In fact, this skill is more important than the Python coding itself, because it will transfer to any programming language you might use in the future.

If you're not sure where to start, take a look at some previous labs and give it your best shot. We are more than happy to look over your metacode and give you feedback (provided you ask a few days before the deadline!).

3) Next, you'll need to create a list containing the ages of individuals who have died. Again, start by writing metacode for this. There are several clever ways to combine this task with the previous one, but it's also fine to just reopen the file and re-read each line.

4) Translate your metacode from the previous two steps into Python code. If your code is reading the data correctly, you should have 204032 total cases, with an average age of 41.1188. The dataset contains 1706 confirmed deaths, with an average age of 68.9777.

5) Write code which takes the two lists of ages and produces the two histograms.

6) Experiment with different bins. Some things to try:

- Every age gets its own bin (1 year per bin)
- Large bins (25 or 30 years)
- Bins of different sizes (see the links below for examples)

7) Decide on which bins you want to use for your final visualization. There are no definite right or wrong answers here, but every choice inevitably distorts or hides some of the data. Consider the different choices made by each of these visualizations:

- New Hampshire deaths by age: https://www.nh.gov/covid19/dashboard/overview.htm#dash
- Massachusetts cases by age: https://www.mass.gov/info-details/covid-19-response-reporting
- National cases/deaths by age: https://covid.cdc.gov/covid-data-tracker/#demographics

Challenge tasks

- Plot both histograms on the same chart.
- A basic rule of data visualization is that the amount of ink ought to be proportional to the quantity being portrayed. Matplotlib's default behavior for non-uniform histogram bins violates this rule (e.g., if I combine two equal bins together, the new bin is twice the original width, but also twice the height). Figure out how to make a non-uniform "histogram" which preserves the relationship between ink and data quantity.
- Given that you also have the illness date and patient sex, make some other plots that examine patterns in these.

Questions

- Give at least one advantage of using small histogram bins, and one advantage of large bins.
- Describe why you chose the histogram bins you did. Explain enough detail to convince a skeptical person that your choice is better than the alternatives, and be prepared to defend your answer in recitation!

- Using the dataset, estimate the total number of cases and total number of deaths in the US. (Remember that we picked every 100th line from the CDC data.) This doesn't match the statistics published on the CDC website; why do you think this is? If possible, justify your answer with observations about the data.

What to turn in

- Your metacode, either as a text file or a PDF
- Your code, as a plain Python file
- Your answers to the questions above, in a PDF file

Resources

Matplotlib documentation for the `hist()` function, with examples: https://matplotlib.org/stable/api/_as_gen/matplotlib.axes.Axes.hist.html#matplotlib.axes.Axes.hist