

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning

Making Plots

[Start Over](#)

Goals

The goals of this lab are:

- To refamiliarize yourself with Jupyter notebooks.
- To output a PDF document from a Jupyter notebook.
- To practice data cleaning skills, including selecting columns, identifying and removing unusual values, and maintaining a tidy dataset.
- To practice calculating summary statistics.
- To practice making data visualizations, such as histograms and scatterplots.

[Previous Topic](#)

[Next Topic](#)

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning

Making Plots

Start Over

Jupyter Notebook Shortcuts

Shortcuts in jupyter make coding in jupyter notebooks more efficient. Spend some time familiarizing yourself with the following:

Both Modes:

- `Shift + Enter` : run the current cell, select below
- `Ctrl + Enter` : run selected cells
- `Alt + Enter` : run the current cell, insert below

Command Mode (press `Esc` to activate):

- `Y` : change the cell type to Code
- `M` : change the cell type to Markdown
- `H` : **show all shortcuts**

Edit Mode (press `enter` to activate):

- `Shift + Control + Minus` : split cell at cursor
- `Command + Up` : go to cell start
- `Command + Down` : go to cell end

Note: these shortcuts are for jupyter and will not work within the shiny lab

[Previous Topic](#)

[Next Topic](#)

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning

Making Plots

Start Over

Setup

This lab will use both `numpy` and `python`. We will also be using a dataset scraped from the website Epicurious, which curates different recipes. Run this cell to load the packages and dataframe.

Python Code

 Start Over

 Run Code

```
1 import numpy as np
2 import pandas as pd
3
4 epicurious = pd.read_csv("data/epi_r.csv")
```

[Previous Topic](#)

[Next Topic](#)

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning

Making Plots

Start Over

Data Cleaning

1. Print the dimensions of the `epicurious` dataset. How many observations are there? How many variables are there?

Python Code [Start Over](#) [Solution](#) [Run Code](#) [Submit Answer](#)

```
1 |
2 |
3 |
```

2. View the head of the dataset. What kinds of variables do you see?

Python Code [Start Over](#) [Solution](#) [Run Code](#) [Submit Answer](#)

```
1 |
2 |
3 |
```

3. Since many of the variables are binary variables indicating different facts about the recipes, let's discard most of them for now and focus on the numeric variables. Identify which of the variables are numeric (if you get stuck, check out the data dictionary (<https://www.kaggle.com/datasets/hugodarwood/epirecipes>)). Save a version of the dataset with only those variables as `epicurious_num`. Also include the recipe titles to use later.

Python Code [Start Over](#) [Solution](#) [Run Code](#) [Submit Answer](#)

```
1 |
2 |
3 |
```

4. Compute a table of summary statistics for these variables, including the minimum, first quartile, median, third quartile, maximum, mean, and standard deviation. Look at these numbers, and think about any red flags you see (we will come back to this).

Python Code [Start Over](#) [Solution](#) [Run Code](#) [Submit Answer](#)

```
1 |
2 |
3 |
```

5. a. It is also helpful to discuss how many missing values the dataset has. Identify the percentage of missing values in each column.

Python Code [Start Over](#) [Run Code](#)

```
1 |
2 |
3 |
```

5. b. What percent of `protein` is missing?

- ☐ 0.00%
- ☐ 100.00%
- ☐ 20.7560%
- ☐ 20.5416%

Submit Answer

6. Is it appropriate to drop the missing data?

- ☐ Yes
- ☐ No

Submit Answer

7. Drop the rows with missing values and save the remaining values in a new data frame called `epicurious_num_nd`.

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning


Making Plots

Start Over

Python Code

 Start Over

 Solution

 Run Code

 Submit Answer

```
1 |  
2  
3
```

Previous Topic

Next Topic

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning

Making Plots

Start Over

Making Plots

In DATA 11800 you likely used `matplotlib` and/or `seaborn` to create visualizations. There are many different python packages out there to create interesting data visuals. The package we are going to explore in this lab is `plotnine`. Run the following code chunk.

Python Code [Start Over](#) [Run Code](#)

```
1 import plotnine as p9
2
3
```

Plotnine is based off `ggplot2` and has a similar aesthetic. Here is the basic idea: You, as the user, tell plotnine what data to use, how to map the variables to the different aesthetics (encoding channels) of the graph, and what type of graph you need—plotnine takes care of the rest!

Histograms

First, we start by providing the data and mapping the variables to the graph's aesthetics. This means that we are defining things like what's on the x-axis or what color the graph is, among many others.

Below is a sample line of code for investigating the average ratings of the different recipes. Note that the functions `ggplot()` and `aes()` come from plotnine— the data frame comes first, then the aesthetics of the graph are defined with `aes()`. You can view plotnine syntax here (<https://plotnine.readthedocs.io/en/stable/>).

Python Code [Start Over](#) [Run Code](#)

```
1 ((p9.ggplot(epicurious_num_nd, p9.aes(x = 'rating'))))
2
3
```

If you run this chunk, a big, nearly blank box appears—there is no actual graph, but we can see that the height variable is now located on the x-axis. There's nothing there because we haven't added a **geom** yet. A geom is a command representing the type of plot we want. To add a histogram, we use `+ geom_histogram()` from plotnine.

Python Code [Start Over](#) [Run Code](#)

```
1 ((p9.ggplot(epicurious_num_nd, p9.aes(x = 'rating'))
2 + p9.geom_histogram())
3
```

8. Do you see a warning message saying "Pick better value with binwidth"? This error is unique to `geom_histogram()`. To fix the warning, you can change the number of bins by adding a new argument, `bins = 10`, into the `geom_histogram()` function.

Python Code [Start Over](#) [Solution](#) [Run Code](#) [Submit Answer](#)

```
1 ((p9.ggplot(epicurious_num_nd, p9.aes(x = 'rating'))
2 + p9.geom_histogram(bins = __))
3
```

9. What happens to the histogram? Are there any differences? CHECK ALL THAT APPLY

- ☐ Nothing Happens
- ☐ The bins are wider
- ☐ The bins are narrower
- ☐ The trend changes
- ☐ The trend stays the same
- ☐ There is less blank space

[Submit Answer](#)

You can also avoid the message by specifying the `binwidth` instead of `bins`. There is a direct relationship between the bin width and the number of bins, so setting one also fixes the other. In general, increasing the number of bins leads to narrower wins, and decreasing the number of bins leads to wider bins.

10. Instead of creating a histogram with 10 bins, create a histogram where the bin width is 0.5.

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning

Making Plots

Start Over

Python Code

Start Over

Solution

Run Code

Submit Answer

```
1 ((p9.ggplot(epicurious_num_nd, p9.aes(x = 'rating'))
2 + p9.geom_histogram(binwidth = ____))
3
```

Notice that on the y-axis we are displaying the counts of the observations in each bin. We can change this to instead display the proportion by adding another argument to ggplot().

Python Code

Start Over

Run Code

```
1 ((p9.ggplot(epicurious_num_nd, p9.aes(x = 'rating', y=p9.after_stat('density'))
2 + p9.geom_histogram(binwidth = 0.5))
3
```

11. Look at the plotnine syntax and add a command that will add a smooth density estimate to the plot

Python Code

Start Over

Solution

Run Code

Submit Answer

```
1 ((p9.ggplot(epicurious_num_nd, p9.aes(x = 'rating', y=p9.after_stat('density'))
2 + p9.geom_histogram(binwidth = 0.5)
3 + ____))
```

12. Now, let's look at calories instead of rating

Python Code

Start Over

Solution

Run Code

Submit Answer

```
1 ((p9.ggplot(epicurious_num_nd, p9.aes(x = ____, y=p9.after_stat('density'))
2 + p9.geom_histogram(bins = 30)
3 + p9.geom_density()))
```

13. a. We identified this earlier and we can also see in the calories plot that the distributions are skewed by a handful of extreme values. Find the top ten largest values for calories, protein, fat, and sodium. Which recipes do they correspond to? Create a dataframe that will help you solve the next question

Python Code

Start Over

Hint

Run Code

```
1
2
3
```

13. b. Select all true statements using the dataframe created in 13.a.

- ☐ Pear-Cranberry Mincemeat Lattice Pie has the most calories
- ☐ Whole Branzino Roasted in Salt is included because of it's high fat content
- ☐ Grilled Lamb Chops with Porcini Mustard is a duplicate entry
- ☐ There are 10 recipes in this dataframe
- ☐ Of the rows in this dataframe, 6 have a rating of 5.0

Submit Answer

This cell will remove the rows containing those values, drop duplicate values, and save the remaining values in a new data frame called `epicurious_num_clean`.

Python Code

Start Over

Run Code

```
1 maxcal = epicurious_num_nd['calories'].nlargest(n=10).index
2 maxpro = epicurious_num_nd['protein'].nlargest(n=10).index
3 maxfat = epicurious_num_nd['fat'].nlargest(n=10).index
4 maxsod = epicurious_num_nd['sodium'].nlargest(n=10).index
5
6 maxvals = set().union(maxcal, maxpro, maxfat, maxsod)
7
8 epicurious_num_clean = epicurious_num_nd.drop(labels = maxvals, axis=0).drop_duplicates()
```

14. Adjust the code used to create the calorie graph to use the new filtered data.

Python Code

Start Over

Solution

Run Code

Submit Answer

```
1 ((p9.ggplot(____, p9.aes(x = 'calories', y=p9.after_stat('density'))
2 + p9.geom_histogram(bins = 30)
3 + p9.geom_density()))
```

Lab 1: Data Cleaning and Visualization

Avery's Questions/Notes:

Goals

Jupyter Notebook Shortcuts

Setup

Data Cleaning

Making Plots

Start Over

Scatterplots

15. Let's continue to the last type of graph we will review for this lab, the scatterplot. Use the plotnine syntax once again to look up the appropriate geom you need. Create a plot displaying the relationship between `calories` and `fat`. Notice that unlike the other plots we have created, such as histograms and densities, you have to supply an `x` and a `y` aesthetic. Remember that `x` is traditionally the explanatory variable and `y` is traditionally the response—which one makes sense to use as the explanatory variable here?

Python Code [Start Over](#) [Solution](#) [Run Code](#) [Submit Answer](#)

```
1 | p9.ggplot(epicurious_num_clean, p9.aes(x = '____', y = '____'))
2 | + ____
3 |
```

16. Select all true statements about the graph above

- ☐ Positive
- ☐ Negative
- ☐ Linear
- ☐ Non-linear
- ☐ Weak
- ☐ Strong

[Submit Answer](#)

17. You can change the color of the point by adding a `color` argument to `geom_point()`. Try adding your favorite color to the cell below.

Python Code [Start Over](#) [Run Code](#)

```
1 | (p9.ggplot(epicurious_num_clean, p9.aes(x = 'fat', y = 'calories'))
2 | + p9.geom_point(color = ____))
3 |
```

In addition to coloring the points a single color, color can also be used to encode data. The following chunk is an example of how to color the points by `rating`.

Python Code [Start Over](#) [Run Code](#)

```
1 | (p9.ggplot(epicurious_num_clean, p9.aes(x = 'fat', y = 'calories', color = 'rating'))
2 | + p9.geom_point())
3 |
```

18. Using the tools you've learned in this lab, create a plot of two or three variables that you find interesting. This is not checked and has no correct answer, use your imagination.

Python Code [Start Over](#) [Run Code](#)

```
1 |
2 |
3 |
```

[Previous Topic](#)