YData: Introduction to Data Science

Class 12: Data visualization continued

Overview

Quick review and warm-up exercises:

- Line plots and histograms

Continuation of data visualization:

- Boxplots
- Scatter plots
- Bar plots and pie charts
- Creating subplots

If there is time

- Seaborn!

Announcement: Homework 5

Homework 5 has been posted!

It is due on Gradescope on Sunday February 26th at 11pm

- **Be sure to mark each question on Gradescope along with the page that has the answers!**

Announcement: class project

Start thinking about your final project!

The final project is a 6-10 page Jupyter notebook report where you analyze your own data to address a question that you find interesting

A draft of the project is due on April 7th

- So plenty of time, but good to start thinking about it now.

Data visualization!

Data visualization

Q: What are some reasons we visualize data rather than just reporting statistics?

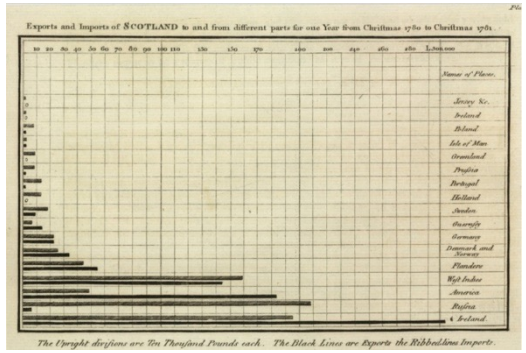
*Statistical projections which **speak to the senses without fatiguing the mind**, possess the advantage of fixing the attention on a great number of important facts.*

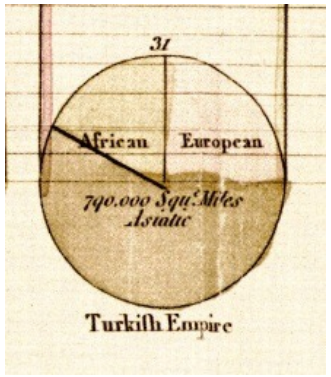
—Alexander von Humboldt, 1811

A very very brief history of data visualization

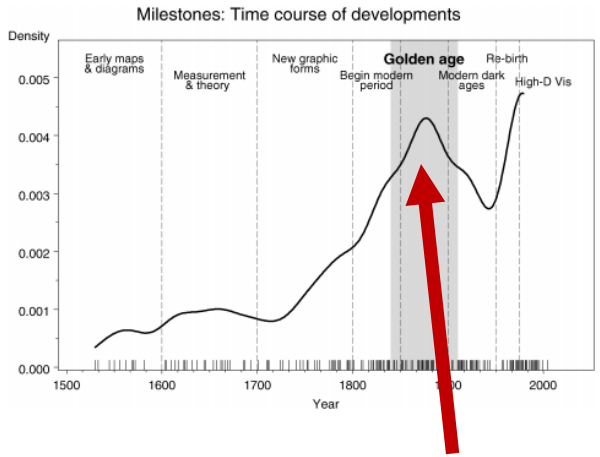
The age of modern statistical graphs began around the beginning of the 19th century

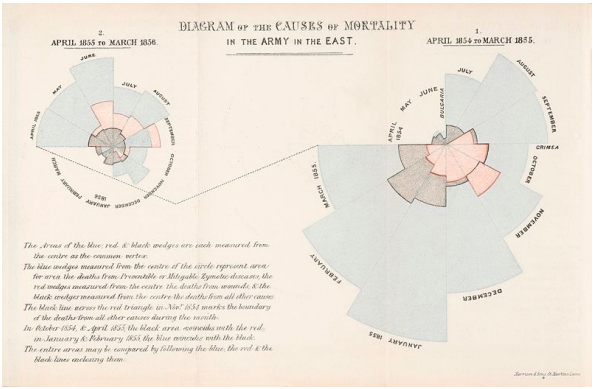
William Playfair (1759-1823)

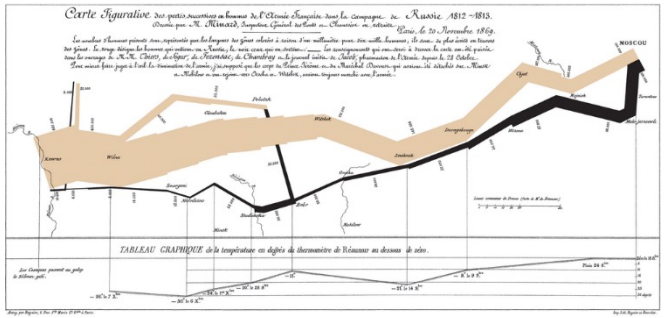


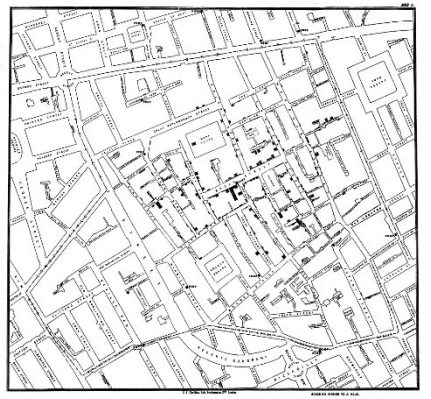


According to Friendly, statistical graphics researched its golden age between 1850-1900



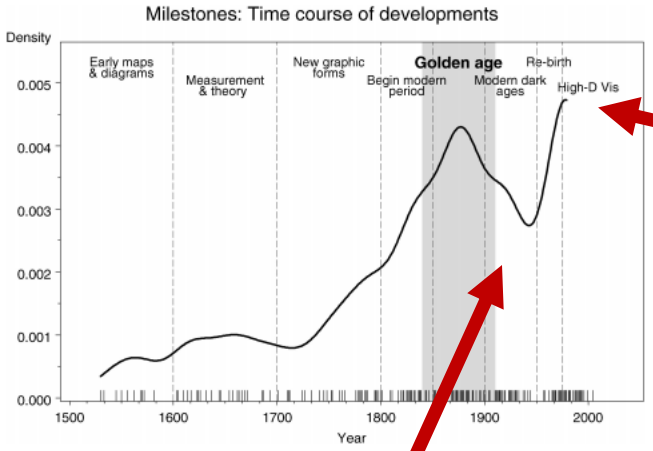


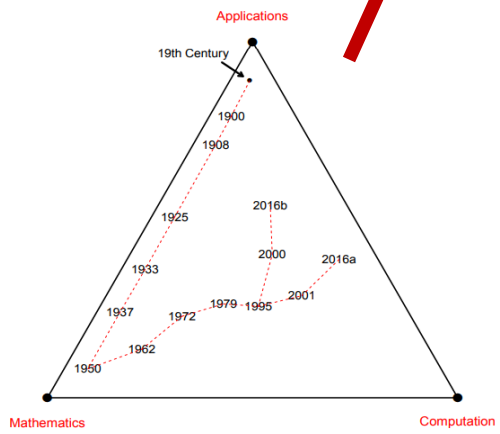


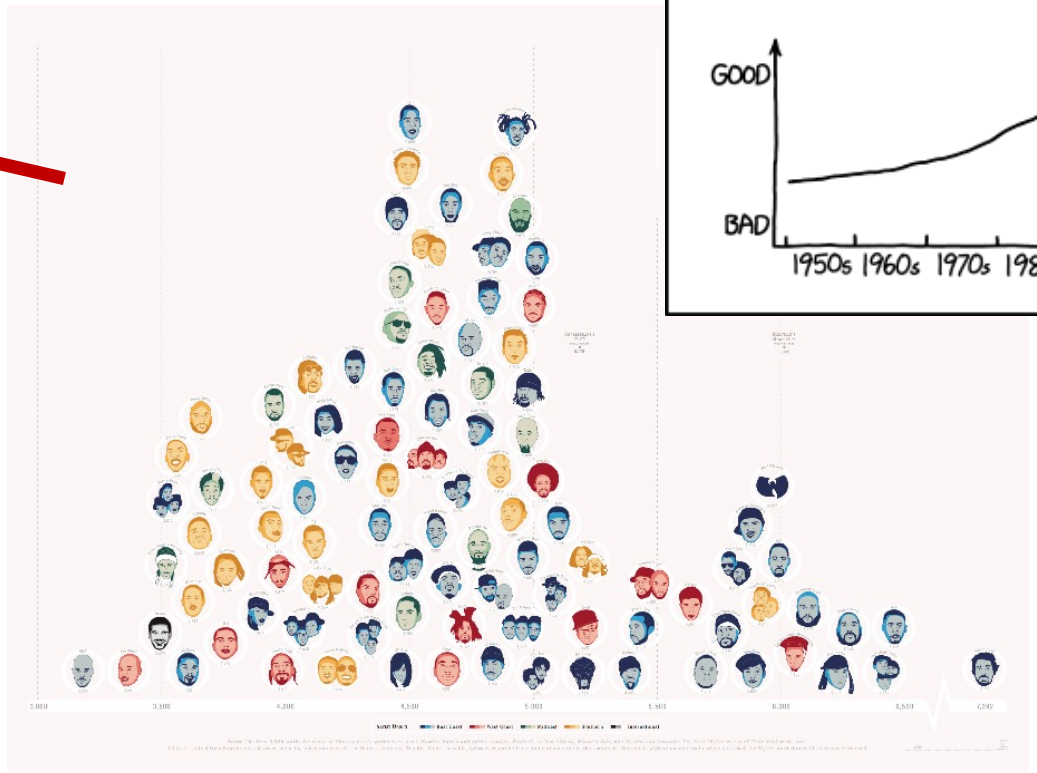


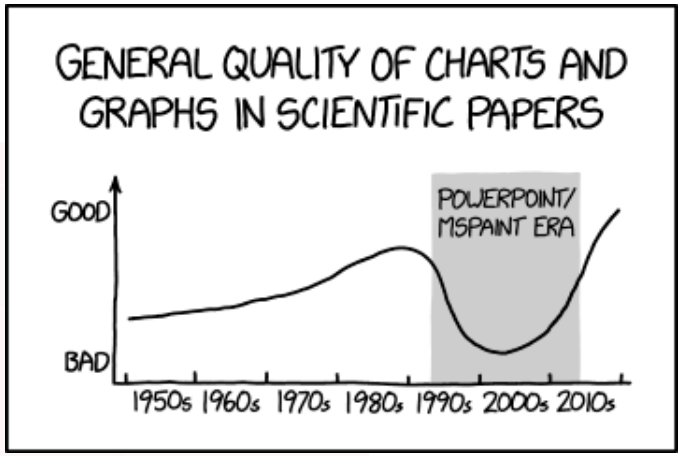
A very very brief history of data visualization

"Graphical dark ages" around 1950









Currently undergoing a "Graphical re-birth"

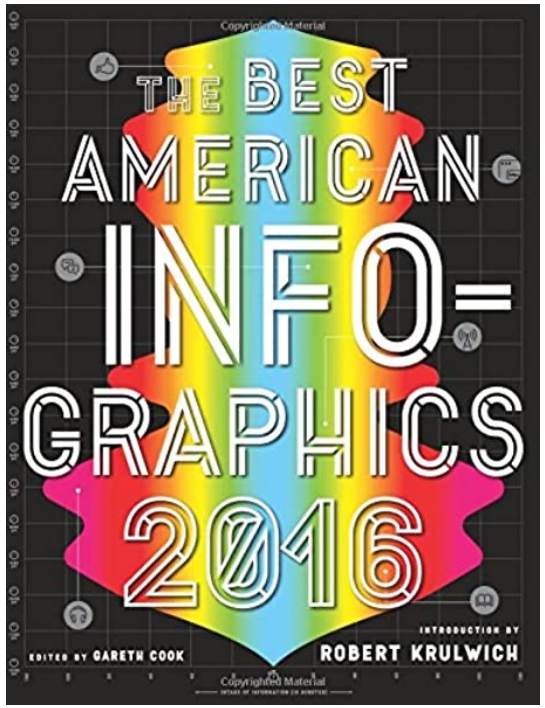
Computer Age Statistical Inference, Efron and Hastie

Coming up on homework 5: find an interesting data visualization…

Homework 5 : Find an interesting data visualization

- https://www.reddit.com/r/dataisbeautiful/
- https://flowingdata.com/

We will do a little show and tell in class



Visualizing data with matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

- Matplotlib makes easy things easy and hard things possible.

Note: there are two different "interfaces" to matplotlib, that use slightly different syntax

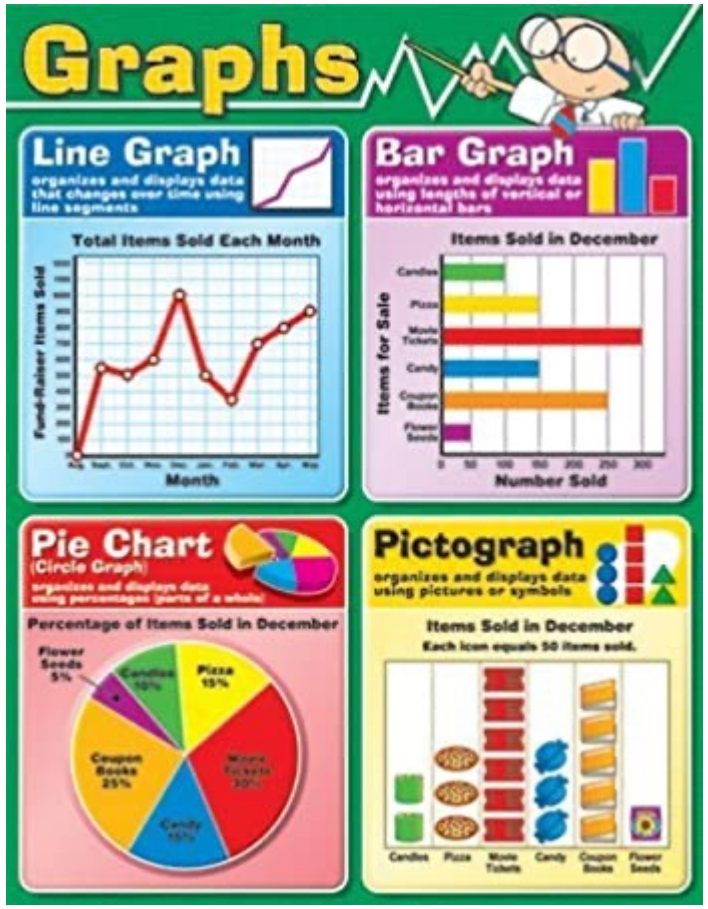
- An explicit "Axes" interface
- An implicit "pyplot" interface

```
import matplotlib.pyplot as plt
```

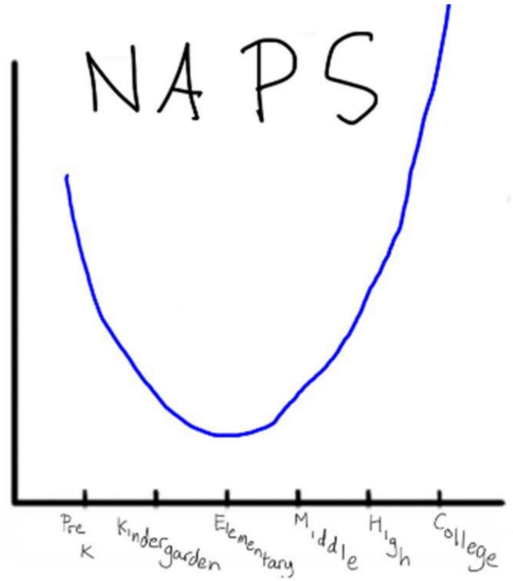
Types of plots

The type of plot you choose will depend on the type of data you have and what you want to emphasize

- i.e., There are different types of plots of categorical data, a single quantitative variable, two quantitative variables, etc.
- This will become even more apparent when we look at the seaborn package



Line graph



Line graph (line chart, or curve chart) displays information as a series of data points called "markers" connected by straight line segments.

We can create line graphs in matplotlib using:

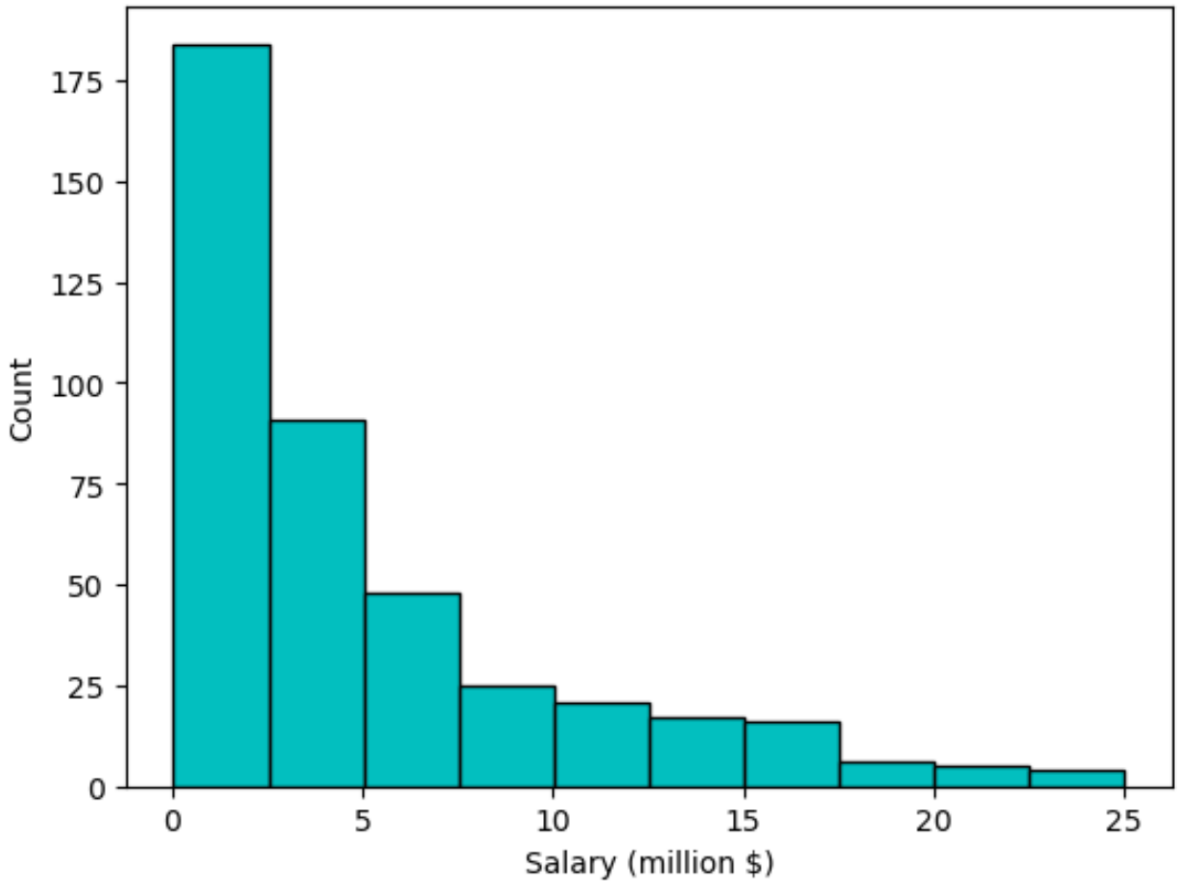
- plt.plot(x, y, '-o') # creates lines with circle markers

Make sure always label your axes:

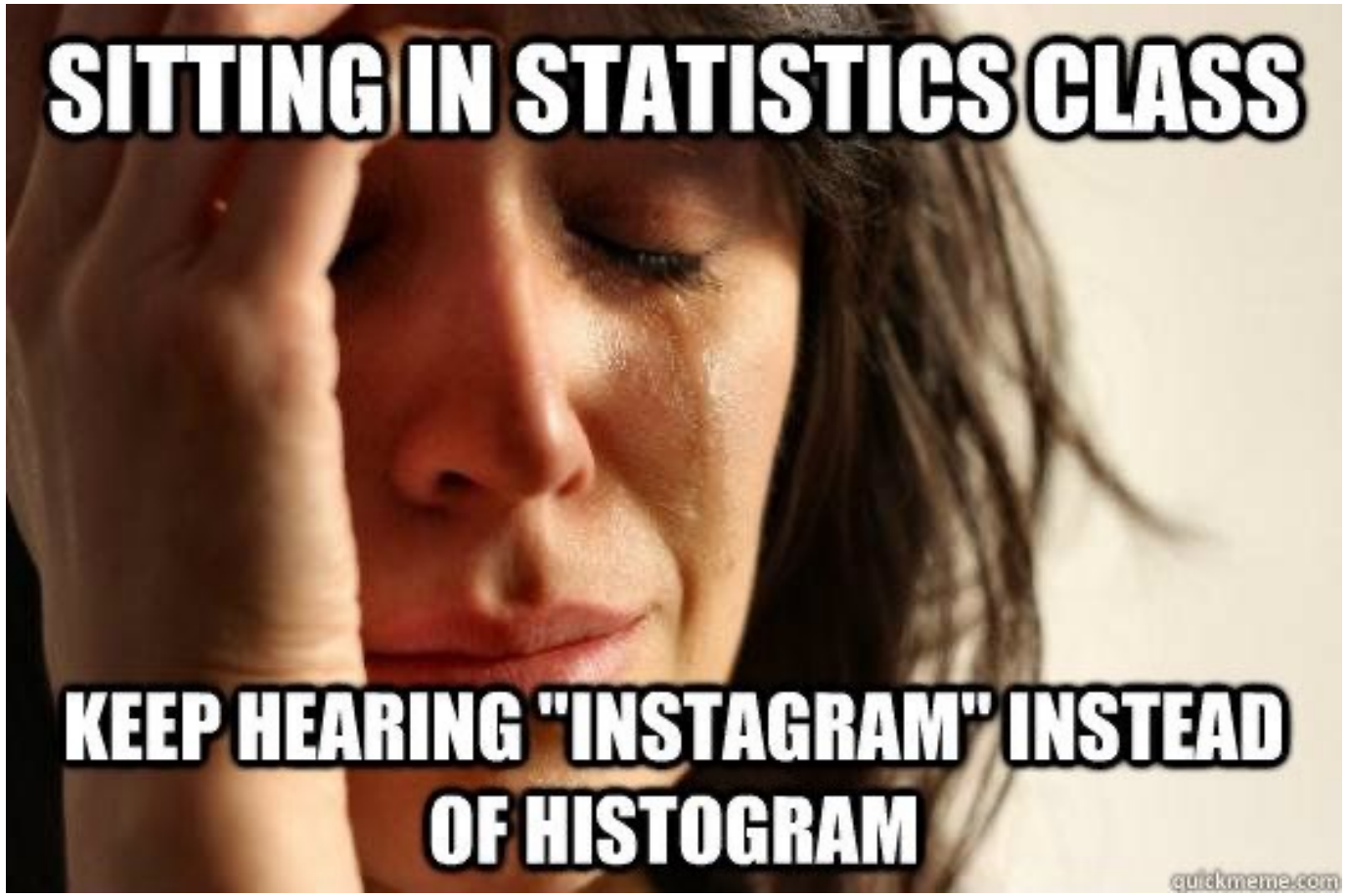
- plt.ylabel(“y label”)
- plt.xlabel(“x label”)
- plt.title(“my title”)
- plt.plot(x, y, label = “blah”)
- plt.legend()

Histograms – basketball players salaries

Life Expectancy	Frequency Count
(0 – 2.5]	184
(2.5 – 5]	91
(5 – 7.5]	48
(7.5 – 10]	25
(10 – 12.5]	21
(12.5 – 15]	17
(15 – 17.5]	16
(17.5 – 20]	6
(20 – 22.5]	5
(22.5 – 25]	4




```
plt.hist(data)
```



Let’s do some warm-up exercises Jupyter!

Five number summary

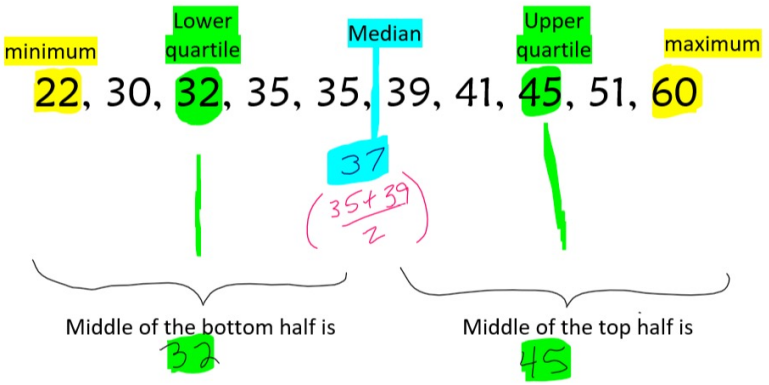
A **"Five Number Summary"** of quantitative data is defined as:

(minimum, Q_1, median, Q_3, maximum)

- Q_1 is the 25^{th} percentile - i.e., the value such that 25% of the data is less than this value
- Q_3 is the 75^{th} percentile - i.e., the value such that 75% of the data is less than this value

The Five number summary roughly splits the data into 4 equal parts

The **Interquartile range (IQR)** is defined as $Q_3 - Q_1$



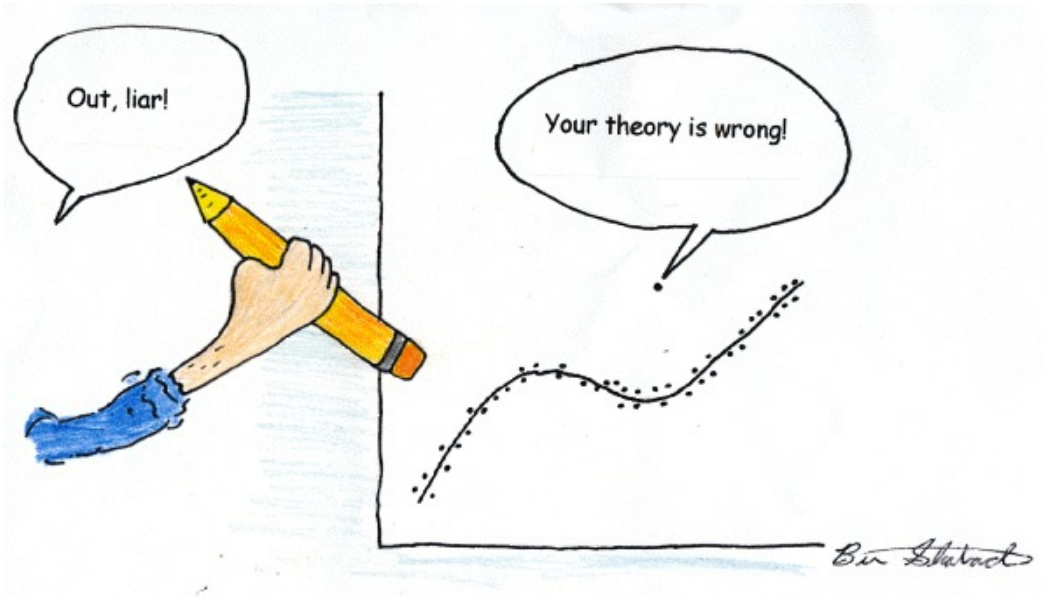
np.quantile(ndarray, [.25, .75])

Detecting of outliers

As a rule of thumb, we call a data value an **outlier** if it is:

Smaller than: $Q_1 - 1.5 * IQR$

Larger than: $Q_3 + 1.5 * IQR$



Boxplots

A **boxplot** is a graphical display of the 5 number summary and consists of:

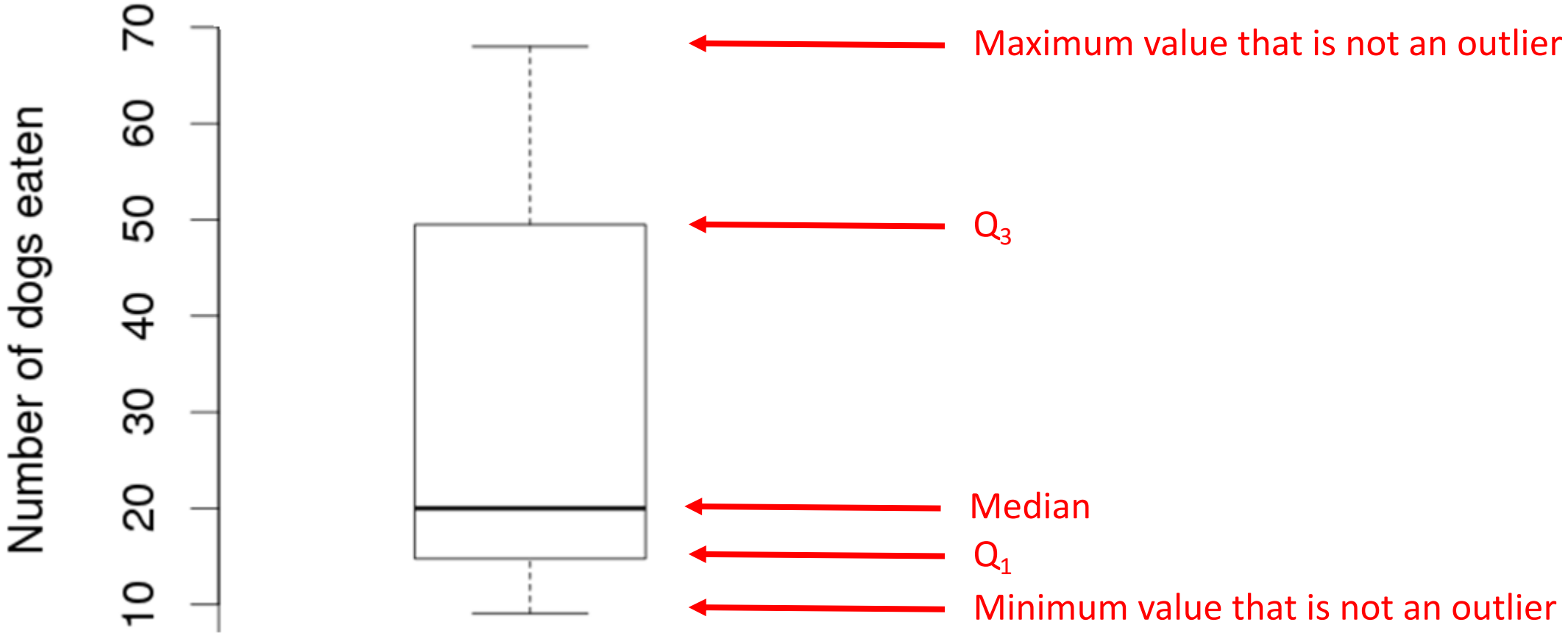
1. Drawing a box from Q_1 to Q_3

2. Dividing the box with a line (or dot) drawn at the median

3. Draw a line from each quartile to the most extreme data value that is not and outlier

4. Draw a dot/asterisk for each outlier data point.

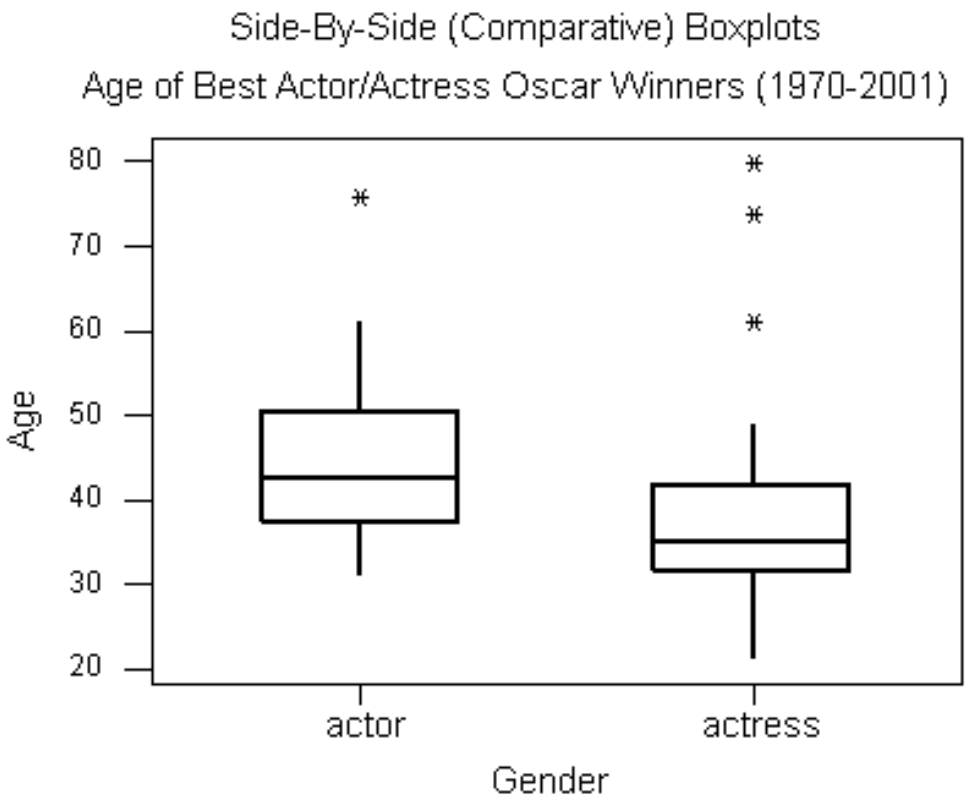
Box plot of the number of hot dogs eaten by the men's contest winners 1980 to 2010



Comparing quantitative variables across categories

Often one wants to compare quantitative variables across categories

Side-by-Side graphs are a way to visually compare quantitative variables across different categories.



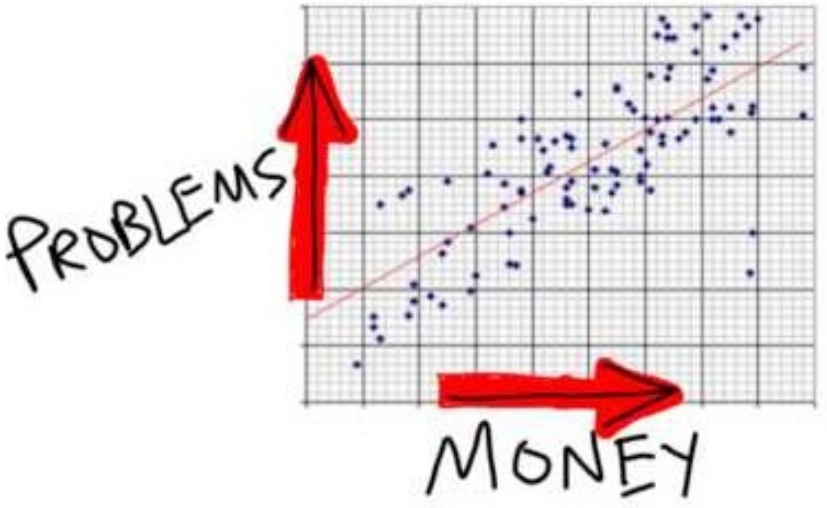
Let's explore this in Jupyter!

```
plt.boxplot(data)
```

Scatterplots

A **scatterplot** graphs the relationship between two variables

- Each axis represents the value of one variables
- Each point the plot shows the value for the two variables for a single data case



If there is an explanatory and response variable, then the explanatory variable is put on the x-axis and the response variable is put on the y-axis.

Scatterplots

There are two ways to create scatter plots in matplotlib:

1. Using plt.plot(x, y, '.')

2. Using plot.scatter(x, y)
   - This function has additional useful arguments such as:
     - s: specified the size of each point
     - color: specifies the color of each point
     - marker: specifies the shape of each point

Let's explore this in Jupyter!

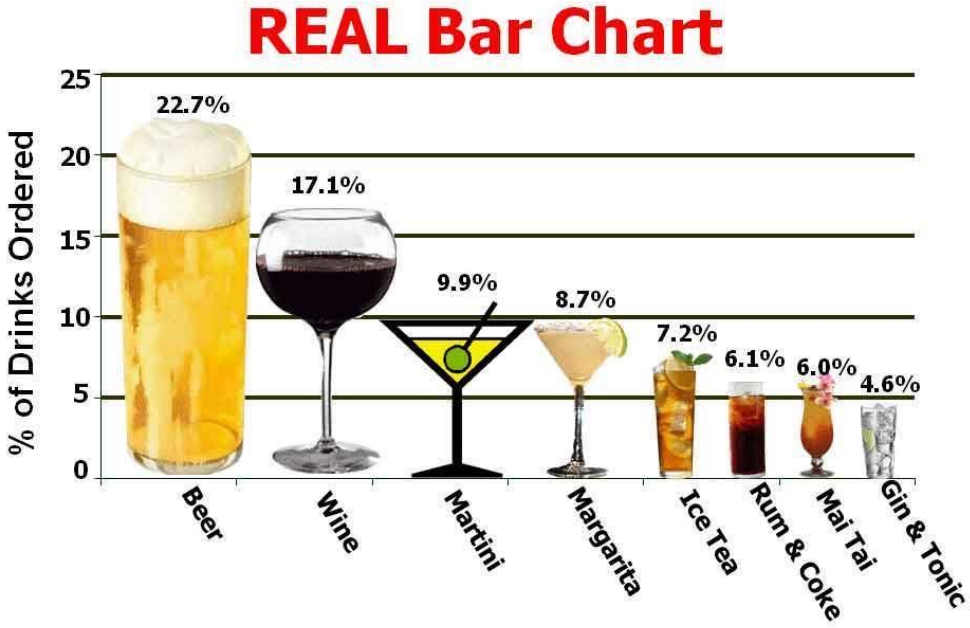
Bar plots and pie charts

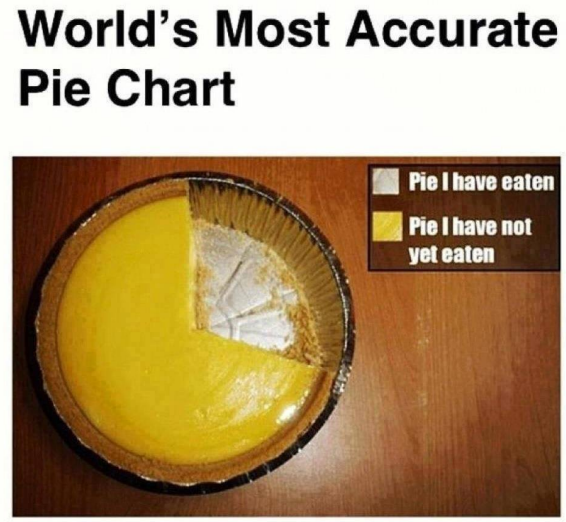
Bar plots and pie charts are used to plot ***categorical data***

Bar plots, the heights of the bars indicate the number of items in each category

In pie charts, the angle of each segment indicates the proportion of items in each category

Let's explore this in Jupyter!



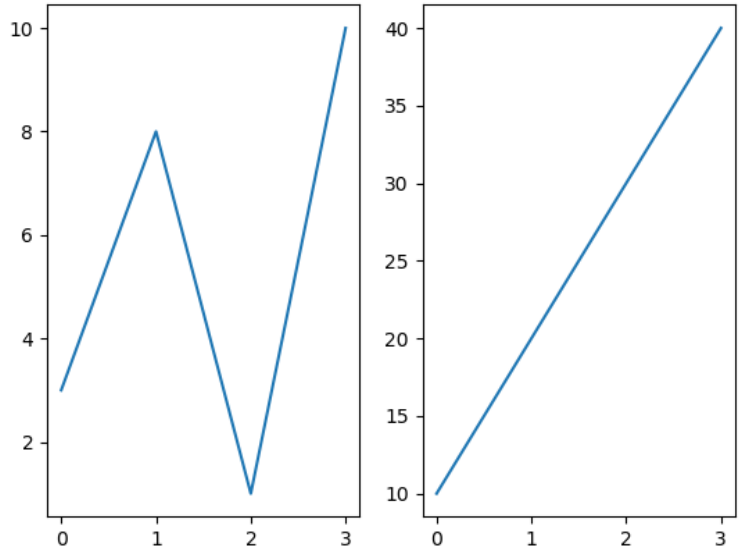


Subplots

There are two ways to create subplots in matplotlib

- 1. Use the pyplot interface
- 2. Use the axes interface

Let’s discuss the pyplot interface first and then we’ll discuss the axes interface



Subplots: pyplot interface

Matplotlib makes it easy to create multiple subplots within a larger figure

1 row

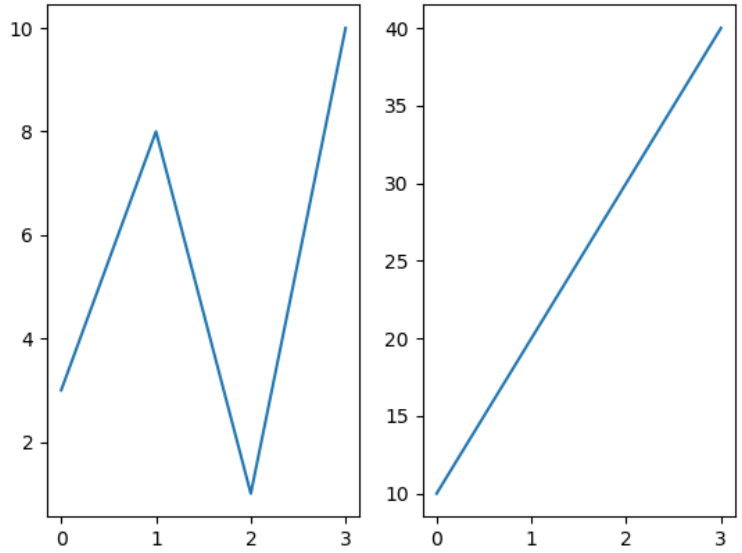
2 columns

```
plt.subplot(1, 2, 1);
plt.plot(x1, y1);
```

plot on the first subplot

```
plt.subplot(1, 2, 2);
plt.plot(x2, y2);
```

plot on the second subplot

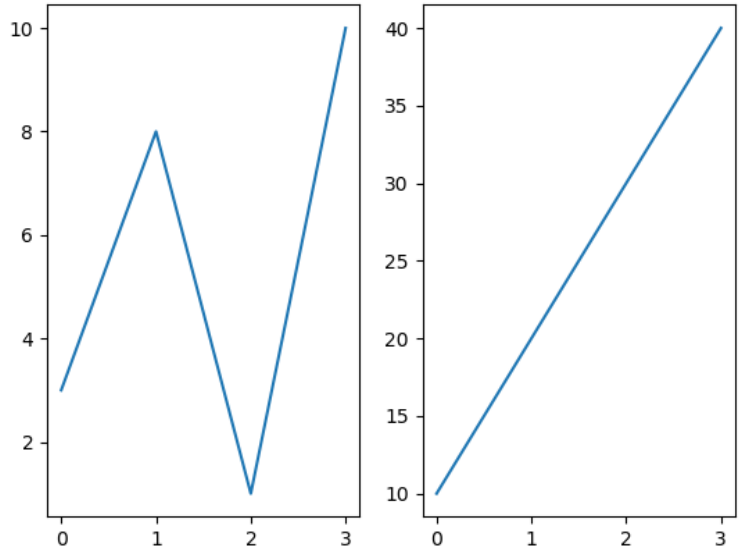


Subplots: pyplot axes interface

Matplotlib makes it easy to create multiple subplots within a larger figure

```
fig, ax = plt.subplots(1, 2);    # notice subplots
ax[0].plot(x1, y1);

ax[1].plot(x2, y2);
ax.set_ylabel("y label")    # notice set_ylabel
```



Let's explore this in Jupyter!

Seaborn

"Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics."

- i.e., it will create better looking plots that are easier to make

There are ways to create visualizations in seaborn:

1. **axes-level** functions that plot on a single axis
2. **figure-level** functions that plot across multiple axes

We will focus on figure level plots

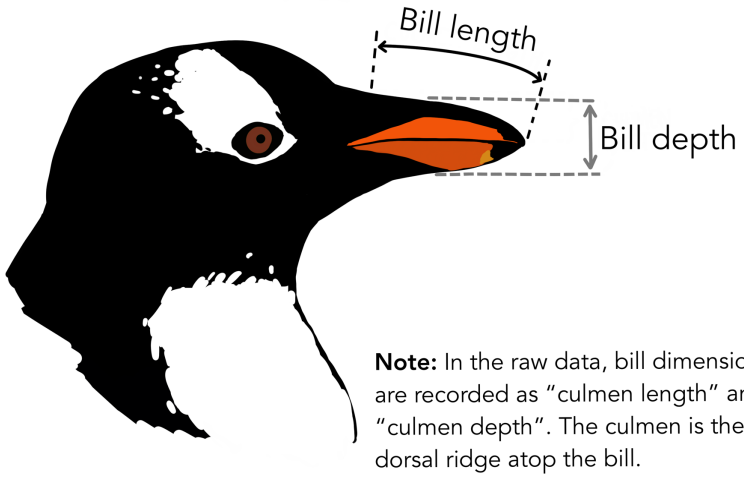
To make plots better looking we can set a theme

```
import seaborn as sns

sns.set_theme()
```

Inspiration: Palmer penguins

To explore seaborn, let’s look at some data on penguins!



Note: In the raw data, bill dimensions are recorded as “culmen length” and “culmen depth”. The culmen is the dorsal ridge atop the bill.

Seaborn figure level plots

Figure level plots are grouped based on the types of variables being plotted

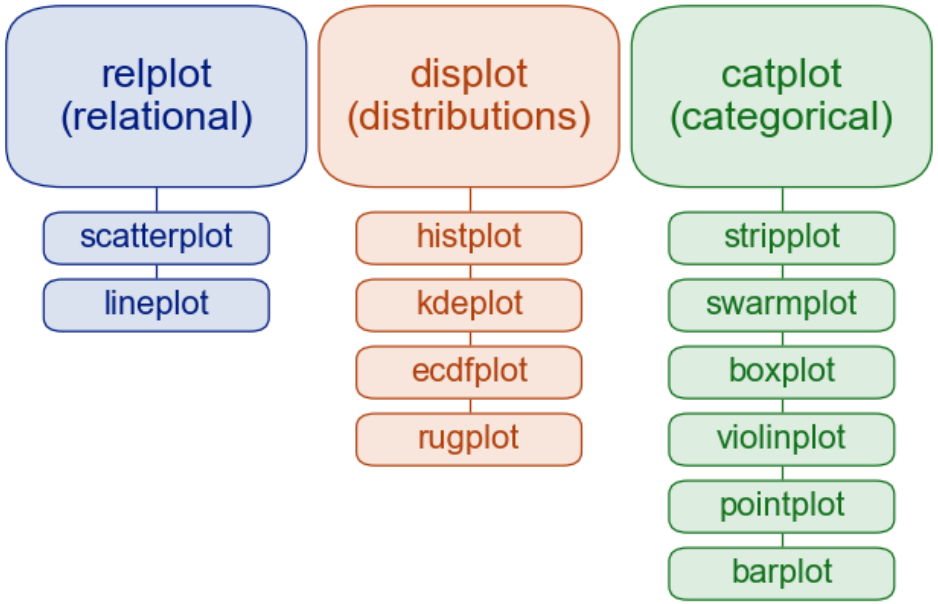
In particular, there are plots for:

1. Two quantitative variables
   - sns.relplot()

2. A single quantitative variable
   - sns.displot()

Quantitative variable compared across different categorical levels
   - sns.catplot()

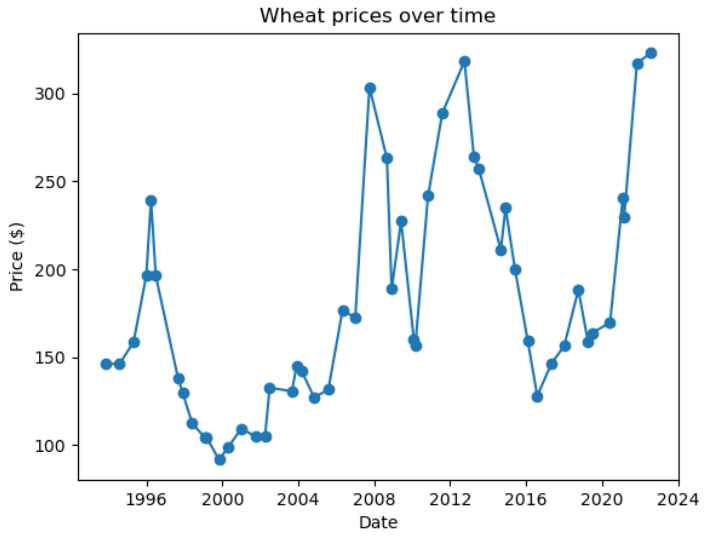
Figure level plots

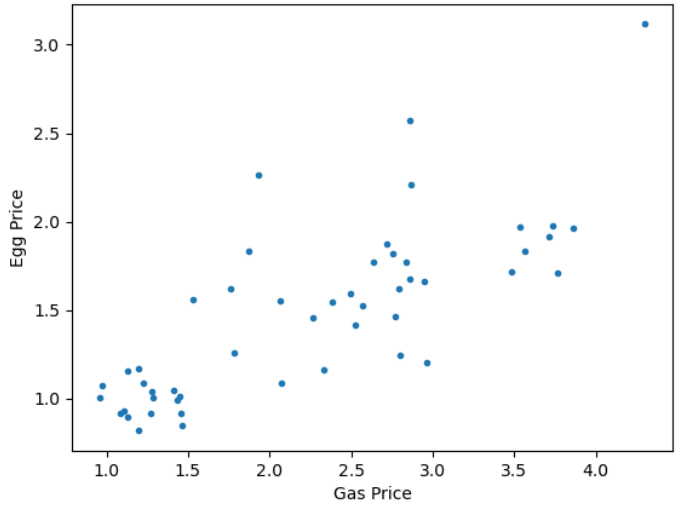


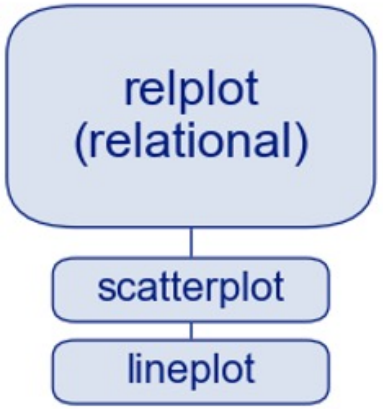
Plots for two quantitative variable

What types of plots have we seen for assessing the relationships between two quantitative variable?

- Line plots and scatter plots!





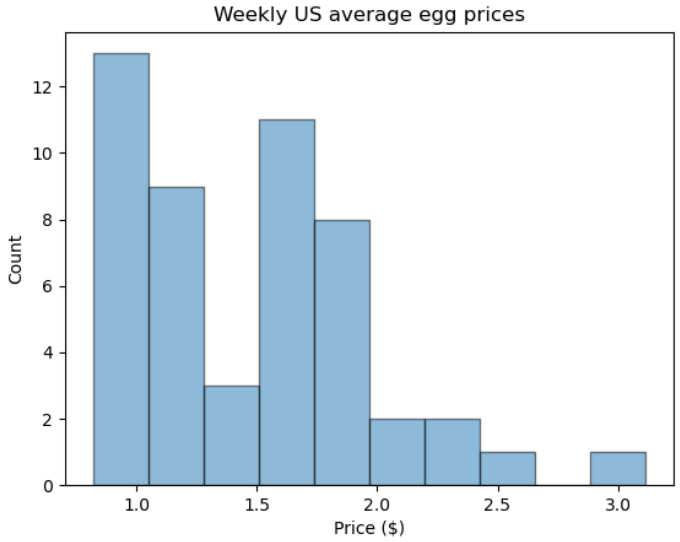


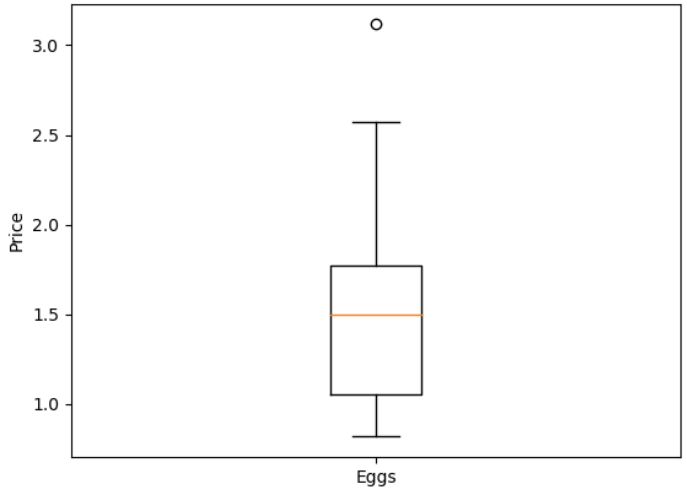
Let's explore this in Jupyter!

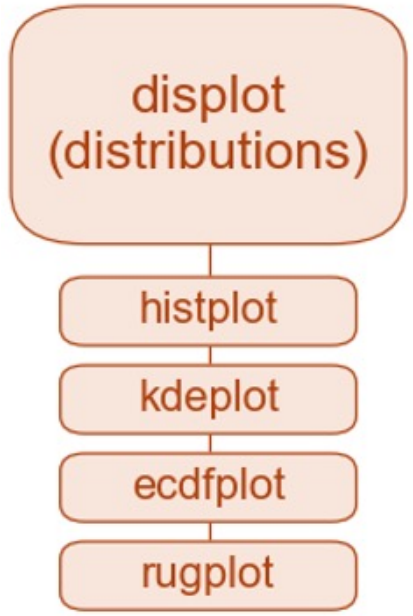
Plots for a single quantitative variable

What types of plots have we seen for plotting a single quantitative variable?

- Histograms and boxplots





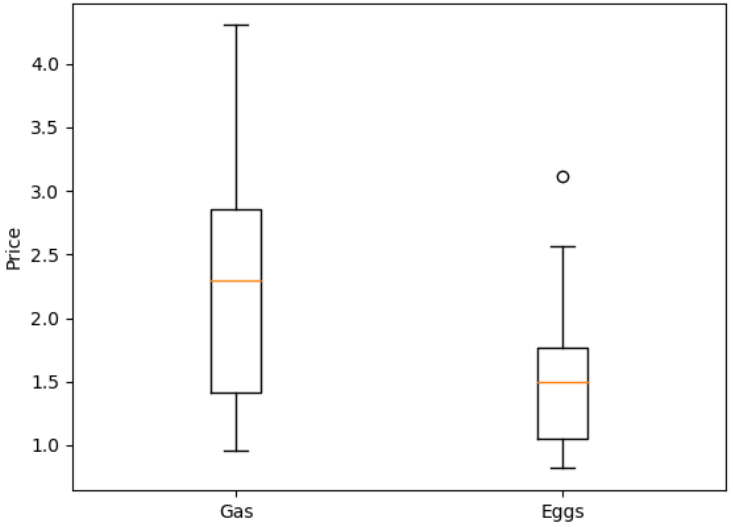


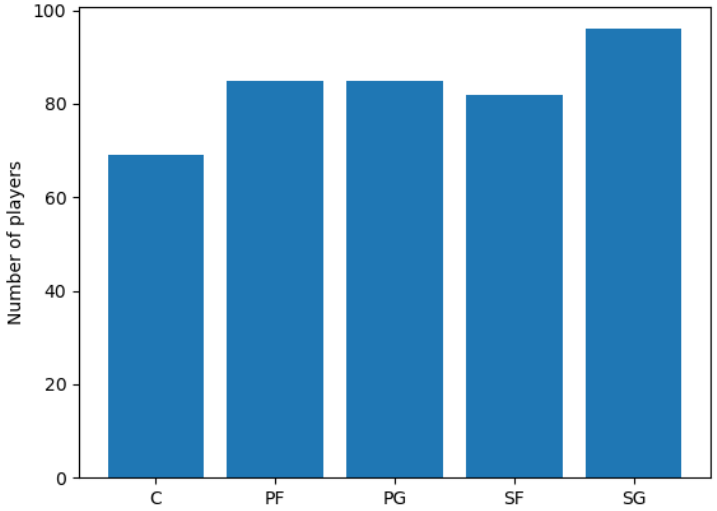
Let's explore this in Jupyter!

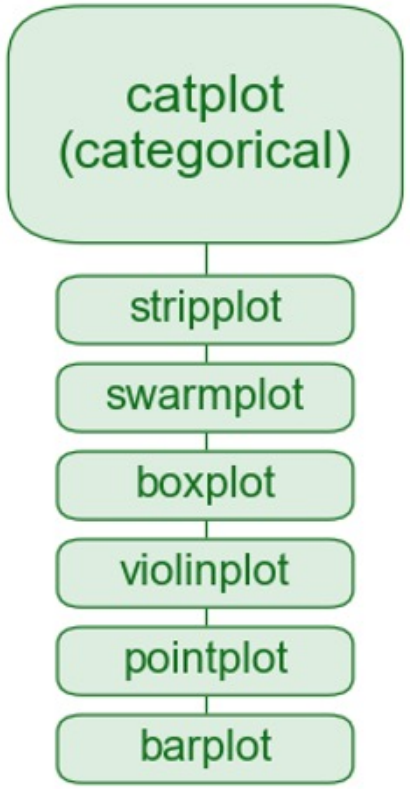
Plots for quantitative data comparing across different categorical levels

What types of plots have we seen comparing quantitative data at different levels of a categorical variable?

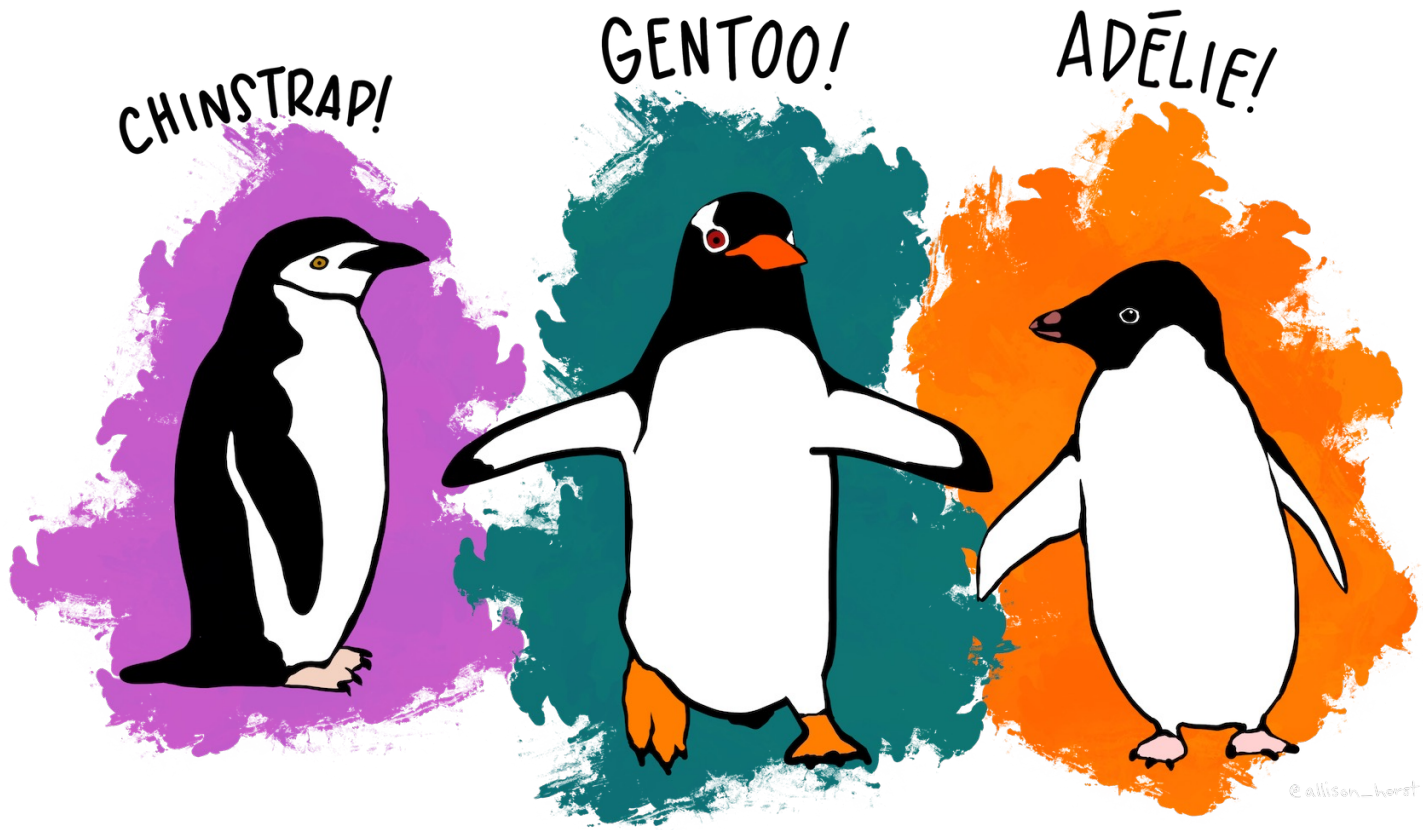
- Side-by-side boxplots, barplots (sort of)







Let's explore this in Jupyter!


CHINSTRAP!
GENTOO!
ADÉLIE!
@allison_horst