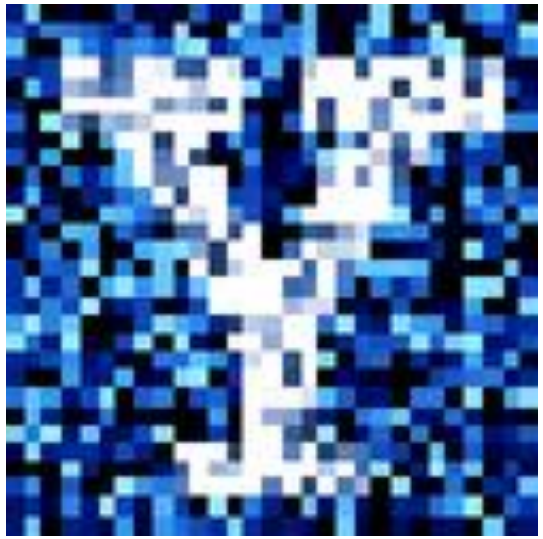
YData: Introduction to Data Science



Class 11: Data visualization continued

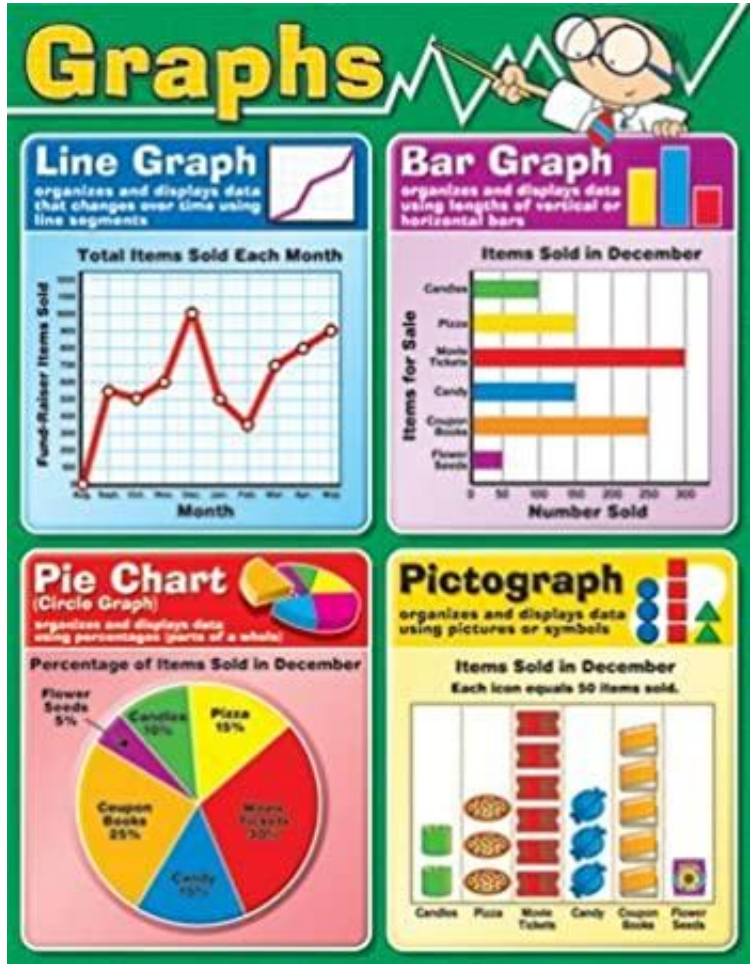
Overview

Brief recap of the history of Data Visualization

Review and additional features of visualizing data with matplotlib

Visualizing data with seaborn

If there is time: explore more on your own



Midterm exam

Homework 5 has been posted!

- A little longer, but should be good practice for midterm

Midterm: Thursday October 9th in person during regular class time

- Exam is on paper

A practice exam has been posted

Also, please post a practice question to Canvas discussions by 3pm on Friday

- If more than 50 students post reasonable questions, I will use one of the questions on the exam

Midterm exam “cheat sheet”

You are allowed an exam “cheat sheet”

One page, double sided, that contains **only code**

- No code comments allowed
- E.g., `list_of_strings = "Hello World".split(" ")`

Cheat sheet must be on a regular 8.5 x 11 piece of paper

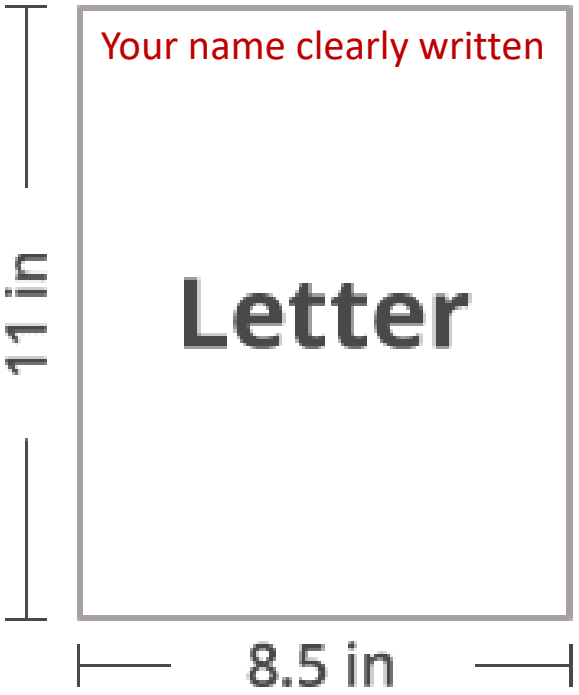
- Your name on the upper left of both sides of the paper

Strongly recommend making a typed list of all functions discussed in class and on the homework

- This will be useful beyond the exam

You must turn in your cheat sheet with the exam

- Failure to do so will result in a 20 point deduction



Data visualization

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

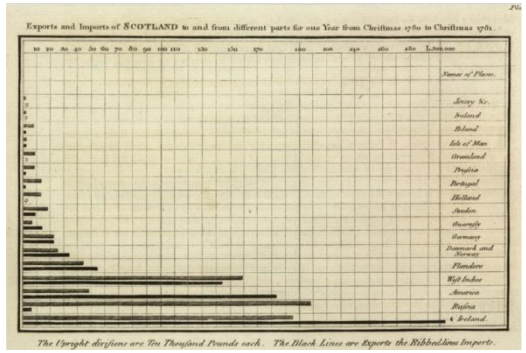
*Statistical projections which **speak to the senses without fatiguing the mind**, possess the advantage of fixing the attention on a great number of important facts.*

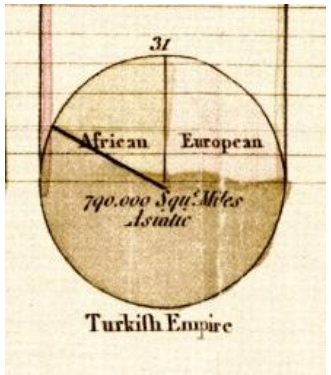
—Alexander von Humboldt, 1811

A very very brief history of data visualization

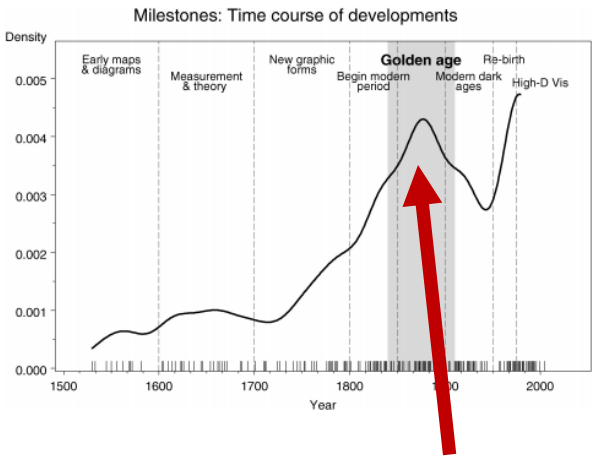
The age of modern statistical graphs began around the beginning of the 19th century

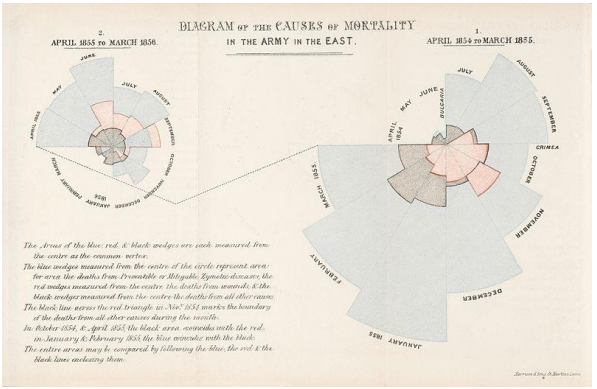
William Playfair (1759-1823)

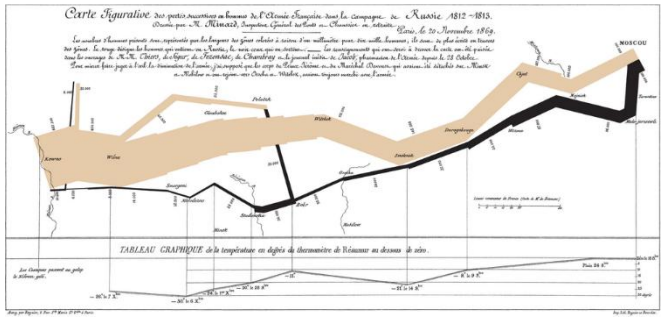


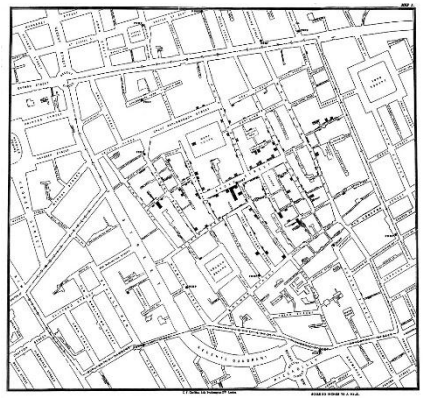


According to Friendly, statistical graphics researched its golden age between 1850-1900



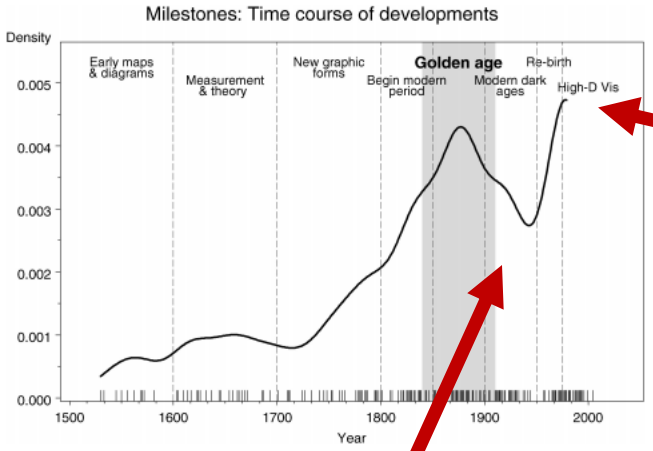


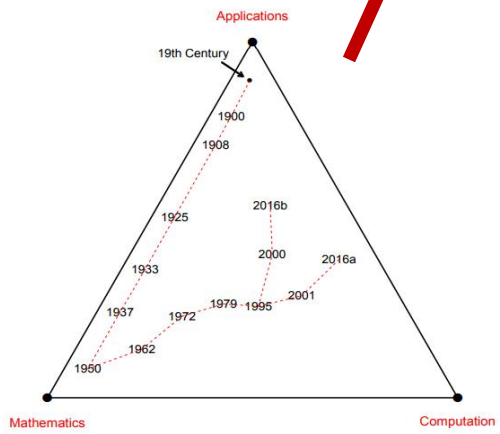


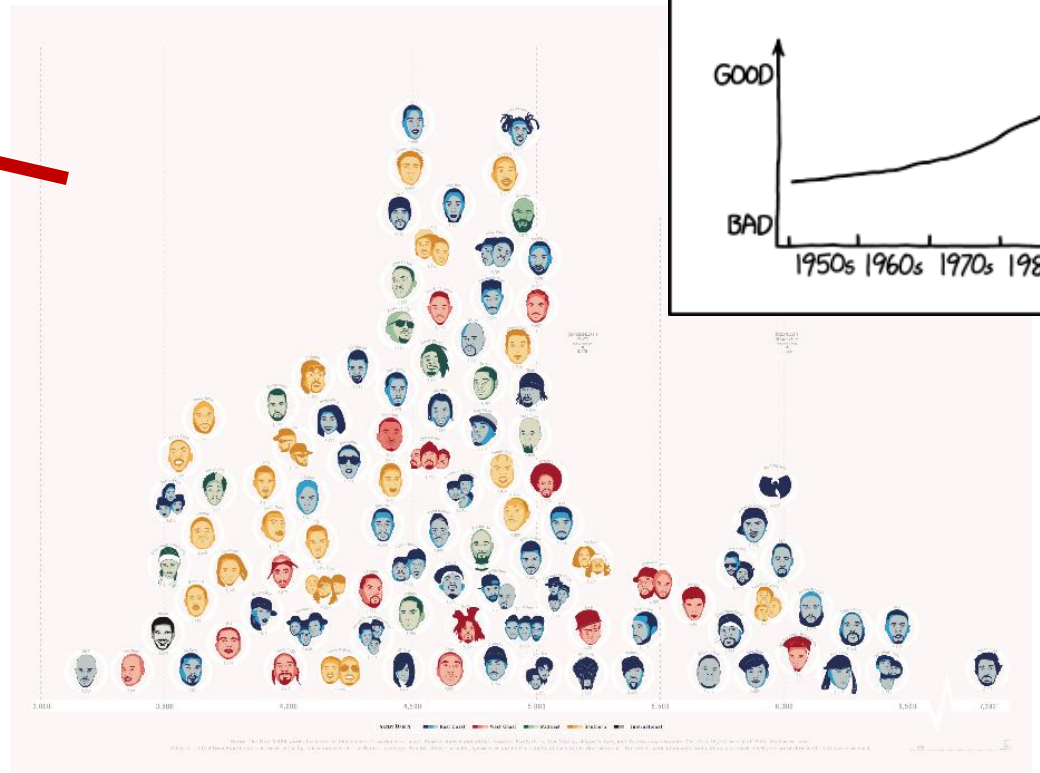


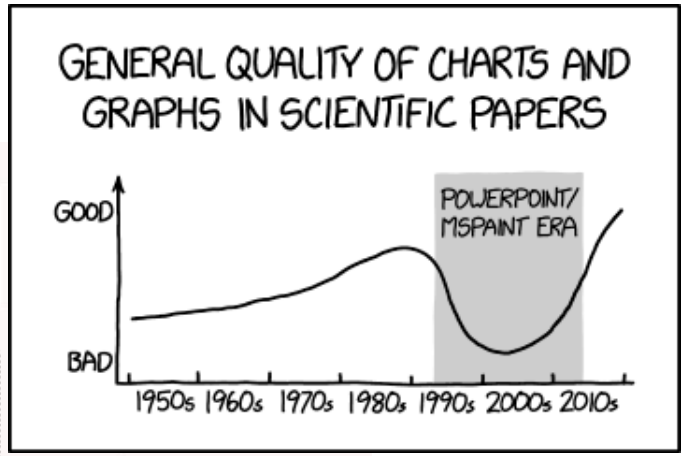
A very very brief history of data visualization

"Graphical dark ages" around 1950









Currently undergoing a "Graphical re-birth"

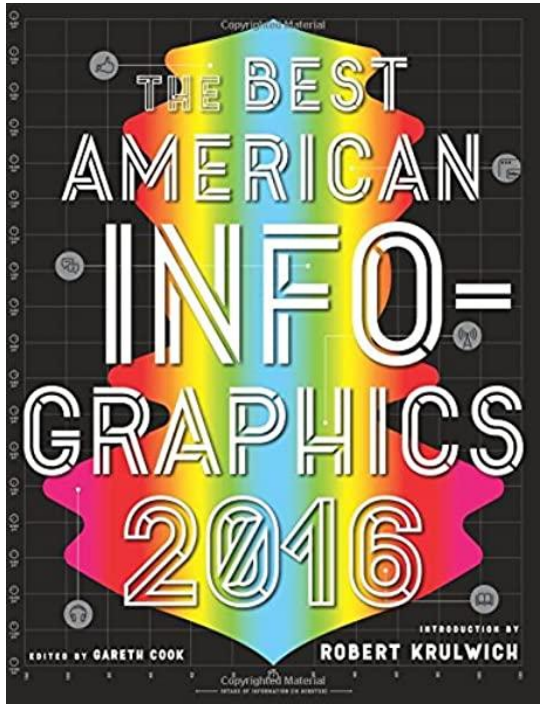
Computer Age Statistical Inference, Efron and Hastie

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



Review and continuation of data visualization

Review of visualizing data with matplotlib

We have already discussed creating a few data visualizations using matplotlib which is a comprehensive library for creating static, animated, and interactive visualizations.

Let’s continue to review and expand our use of this package

We will then discuss another visualization library called "seaborn" which makes it even easier to create beautiful looking graphics

Review of visualizing one quantitative variable

Q: How can we visualize one quantitative variable?

A: Histograms!

A: Box plots!

```
plt.hist(data1, edgecolor = "k", alpha = .5);
plt.hist(data2, edgecolor = "k", alpha = .5);

plt.boxplot( [data1, data2],  tick_labels = ["lab1", "lab2"])
```

Visualizing time series

Q: How can we visualize a time series?

A: Line plot

```
plt.plot(x1, y1, '-o', label = 'First line');
plt.plot(x2, y2, '-o’, label = ‘Second line’);
plt.legend();
```

Let's review this in Jupyter!

Review of visualizing two quantitative variables

Q: How can we visualize two quantitative variables?

A: Scatter plot!

```
plt.plot(x_array, y_array, '.');
```

We can also use the matplotlib `plt.scatter()` function…

Scatter plots

plt.scatter(x, y) 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!

Review/continuation of visualizing categorical data

Q: How can we visualize categorical data?

A: Bar plots and pie charts

```
import matplotlib.pyplot as plt
plt.pie(data, labels = label_names)
plt.bar(labels, data)

plt.xlabel("Drink type")
plt.ylabel("Number of drinks")
```

Subplots: pyplot interface

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

1 row

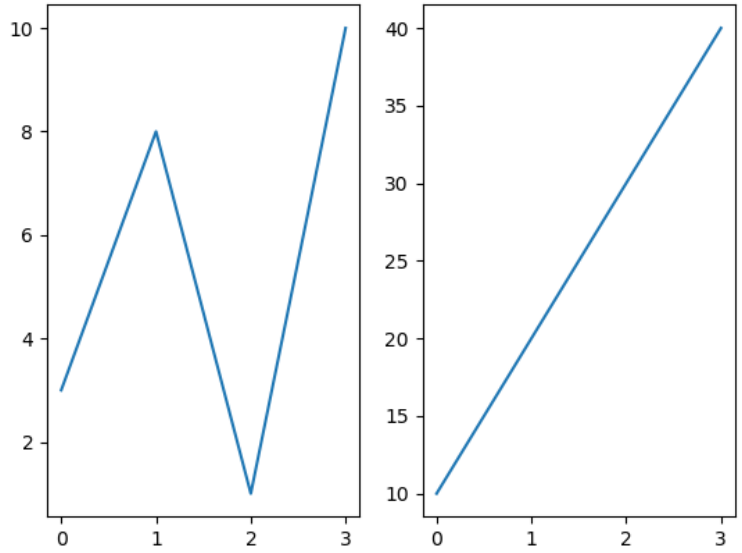
2 columns

```
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

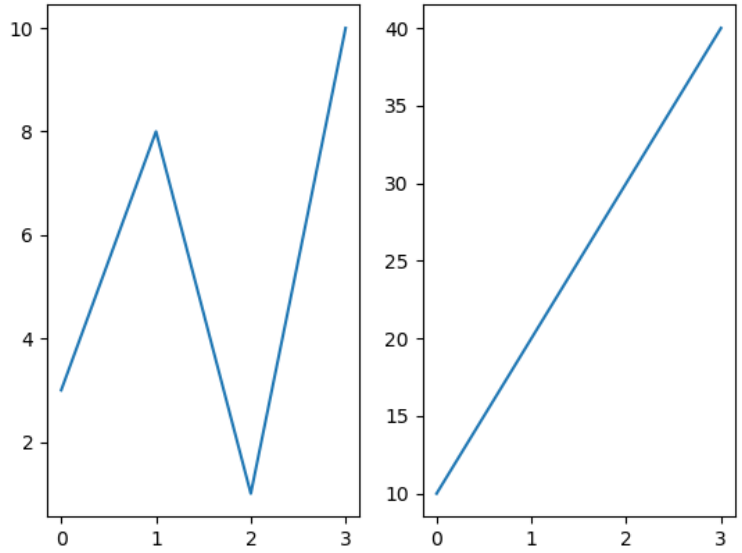


Subplots: axes interface

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
```

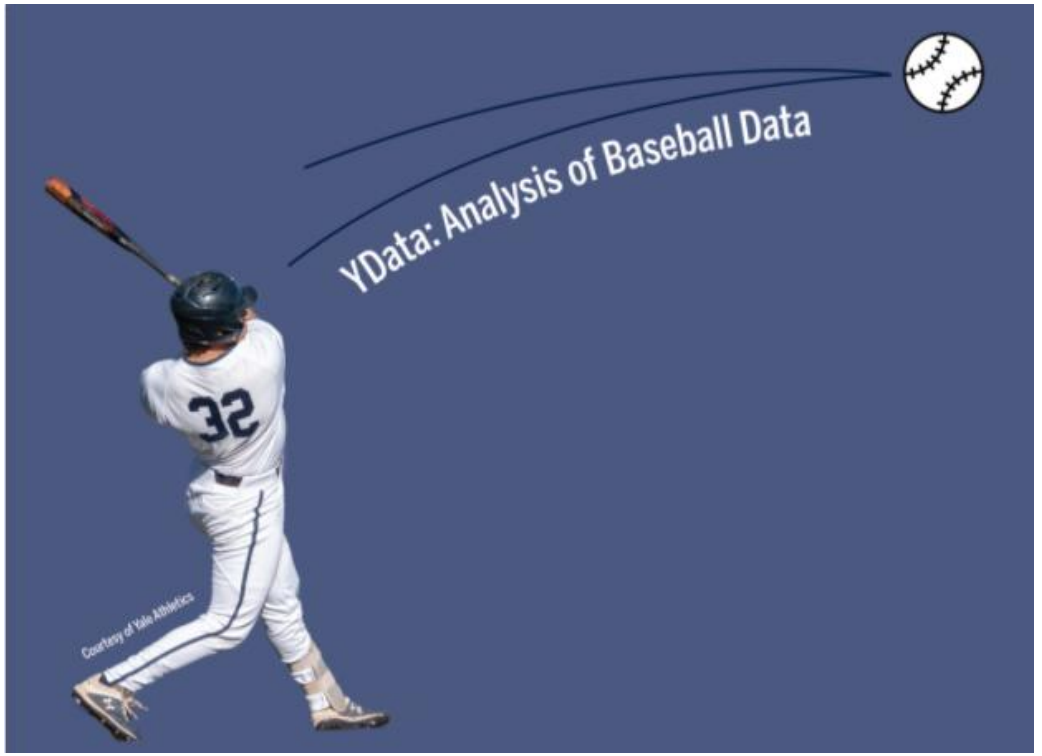


Let's explore this in Jupyter!

Using matplotlib as a canvas

We can also use matplotib as a canvas to create general figures

For example, in my Ydata baseball class, we drew a baseball diamond and illustrated where players were on base with red circles.



Let's briefly 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

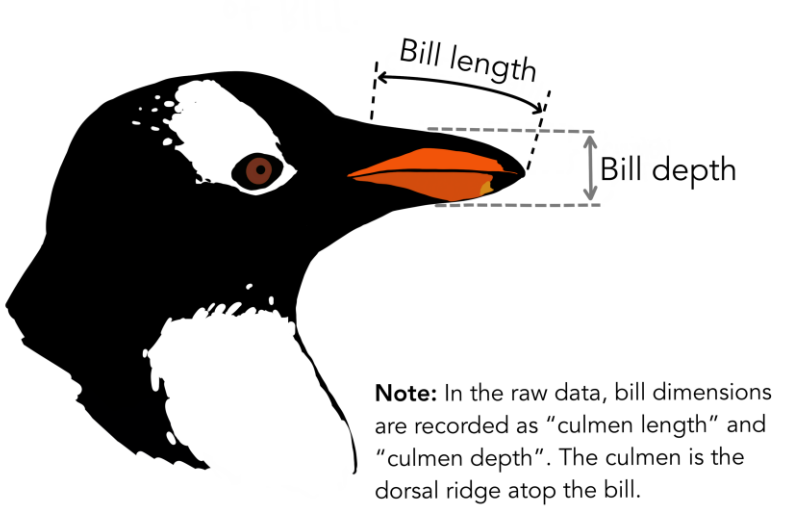
To make plots better looking we can set a theme

```
import seaborn as sns

sns.set_theme()
```

Inspiration: Palmer penguins

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



Seaborn figure level plots

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

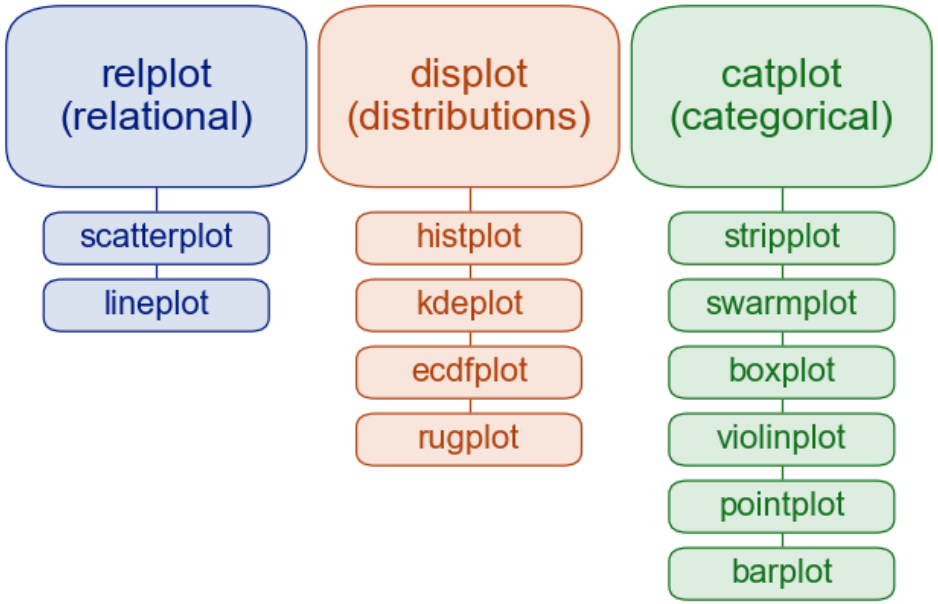
In particular, there are plots for:

1. Two quantitative variables
   - sns.relplot()

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

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

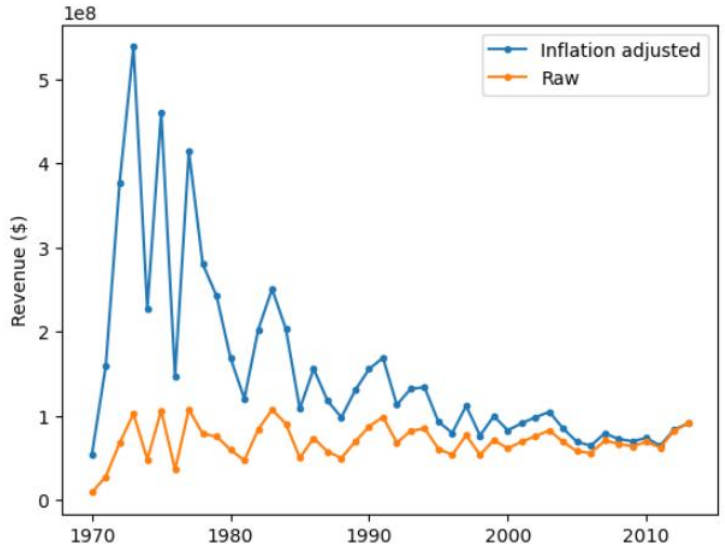
Figure level plots

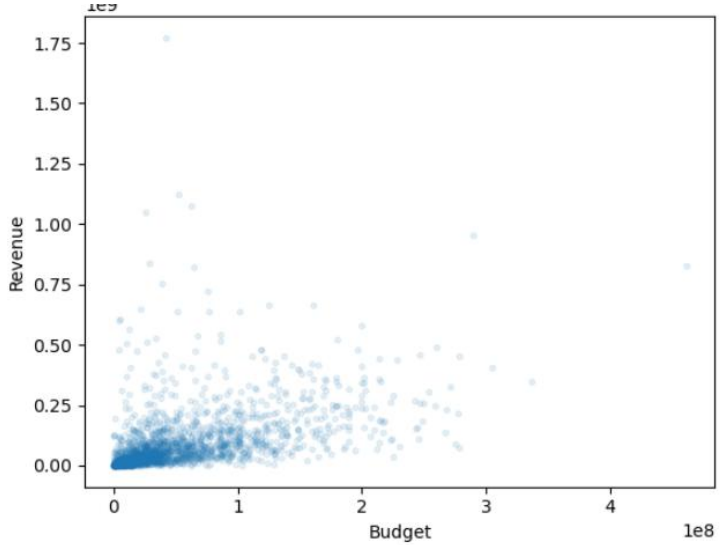


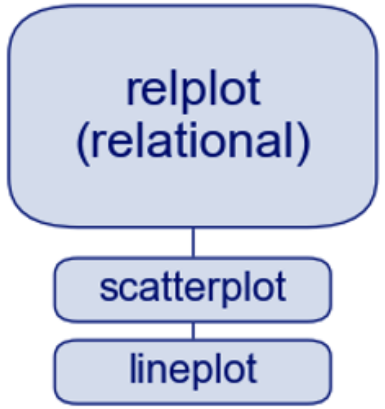
Plots for two quantitative variable

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

- Line plots and scatter plots!





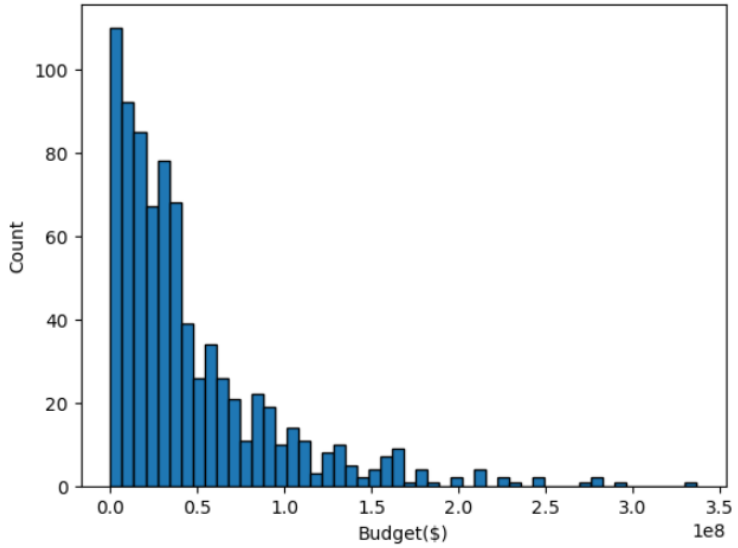


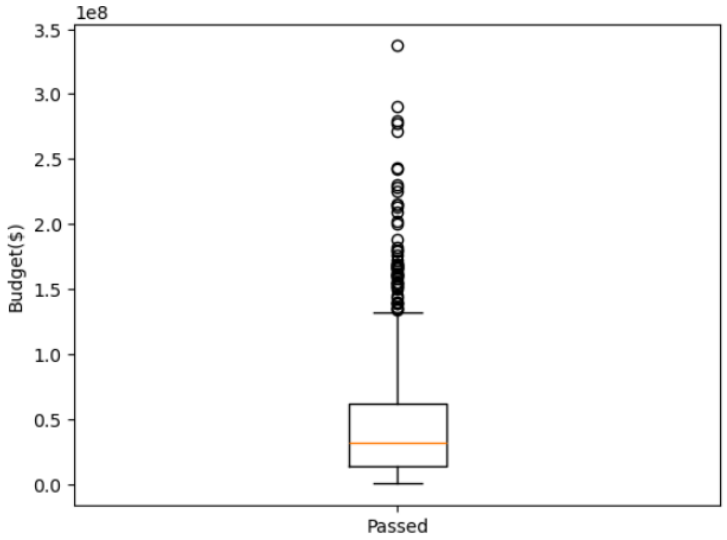
Let's explore this in Jupyter!

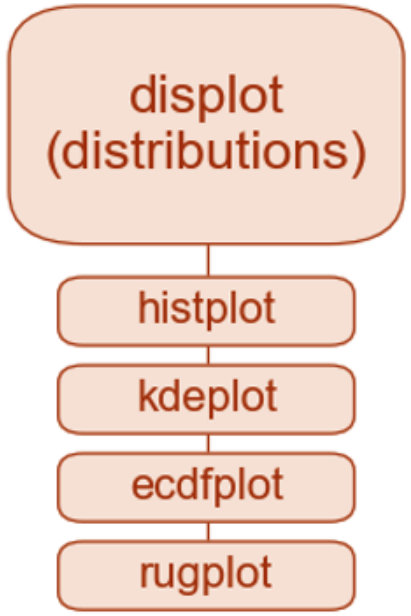
Plots for a single quantitative variable

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

- Histograms and boxplots





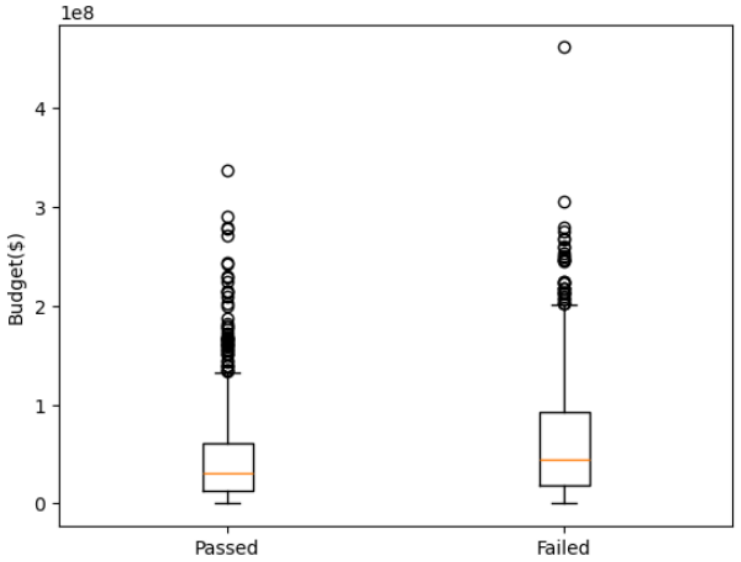


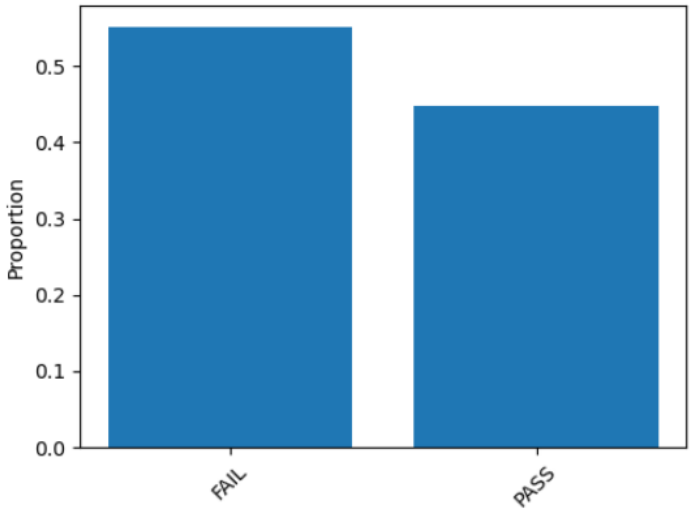
Let's explore this in Jupyter!

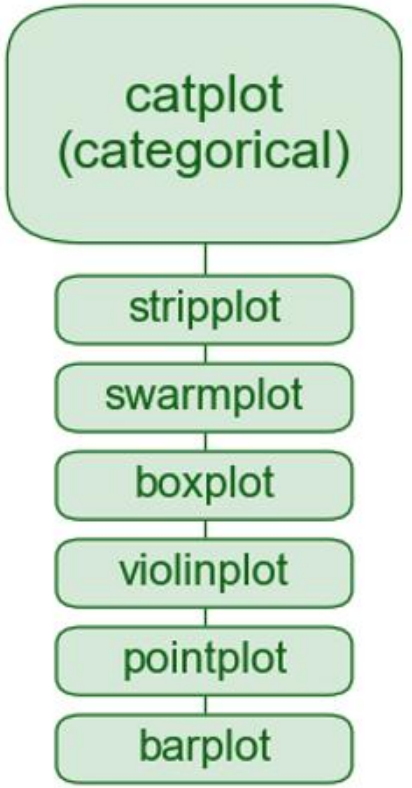
Plots for quantitative data comparing across different categorical levels

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!

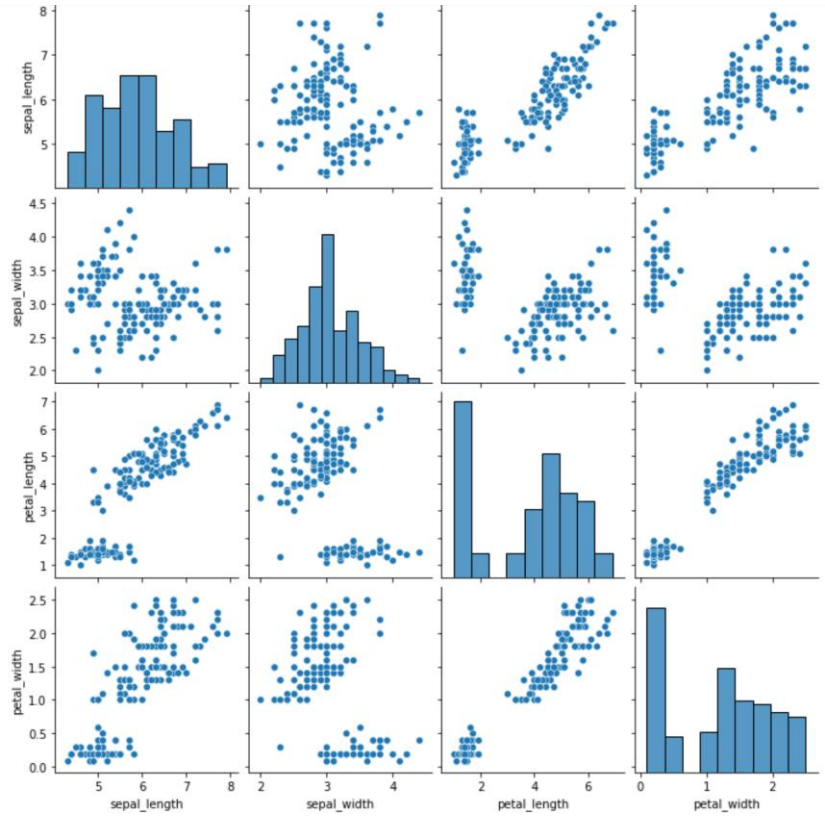
Pairs plot

A pairs plot, create scatter plots between all quantitative variables in a DataFrame

It can be one of the most useful ways to see relationships between multiple quantitative variabels

We can create pairs plots in seaborm using:

```
sns.pairplot(the_data)
```



Let's explore this in Jupyter!