YData: Introduction to Data Science

Class 10: Data visualization continued

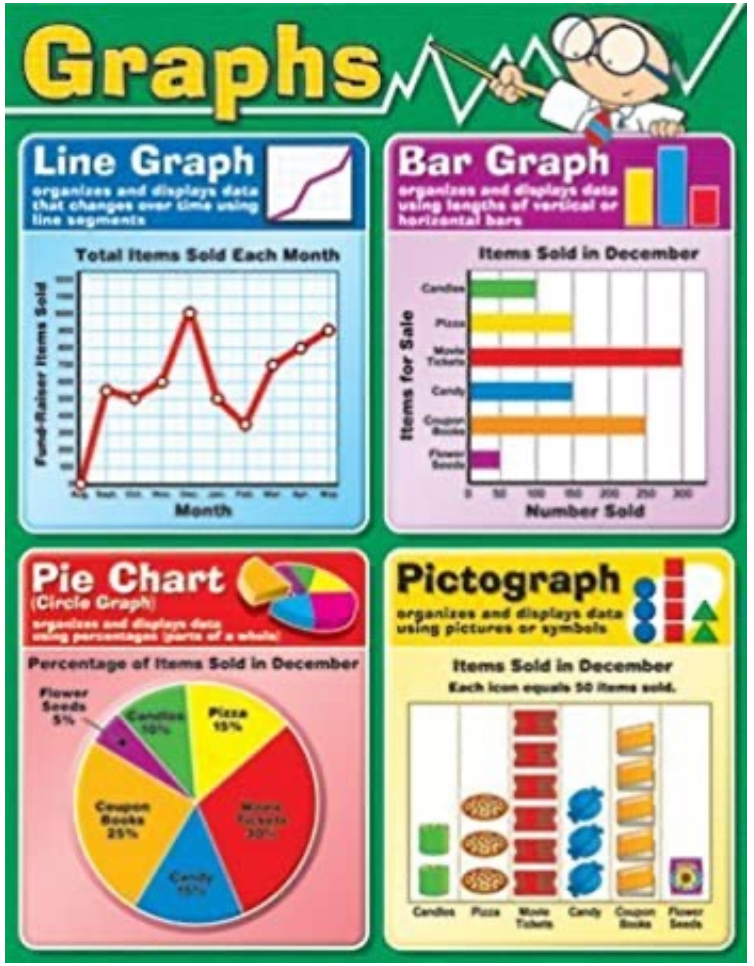
Overview

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: Interactive graphics



Announcement: Homework 4

Homework 4 has been posted!

It is due on Gradescope on Sunday February 18th at 11pm

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

Note: The homework is on the long side, but you already have the skills to do most of it, so please get started early

- It should be good practice/review of the key concepts covered so far

Data visualization

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

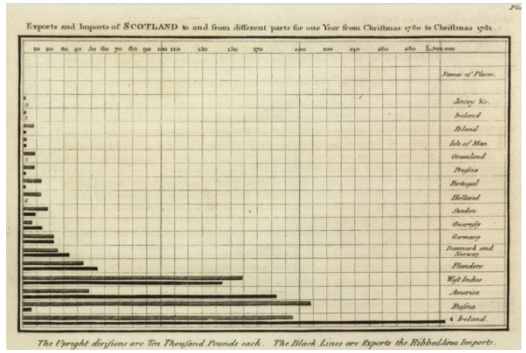
*Statistical projections which **speak to the senses without fatiguing the mind**, possess the advantage of fixing the attention on a great number of important facts.*

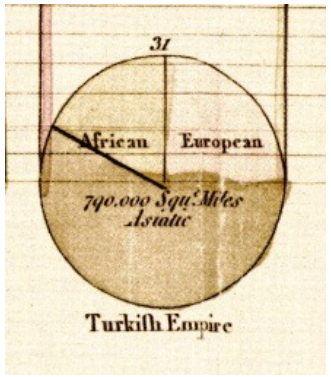
—Alexander von Humboldt, 1811

A very very brief history of data visualization

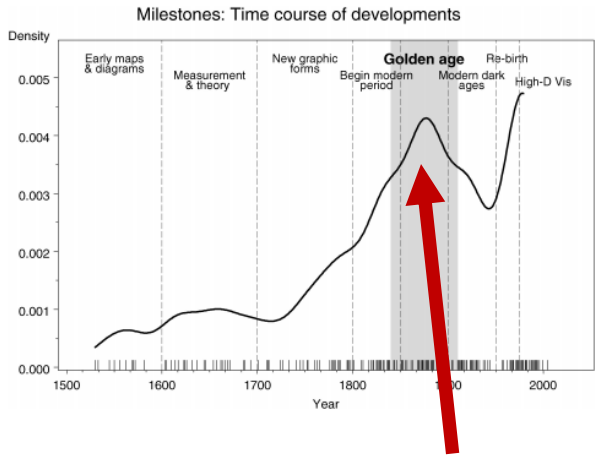
The age of modern statistical graphs began around the beginning of the 19th century

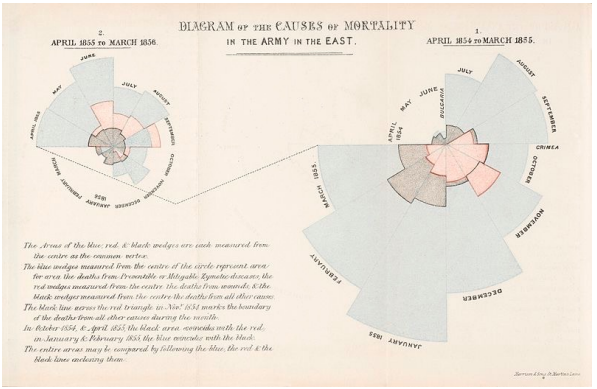
William Playfair (1759-1823)

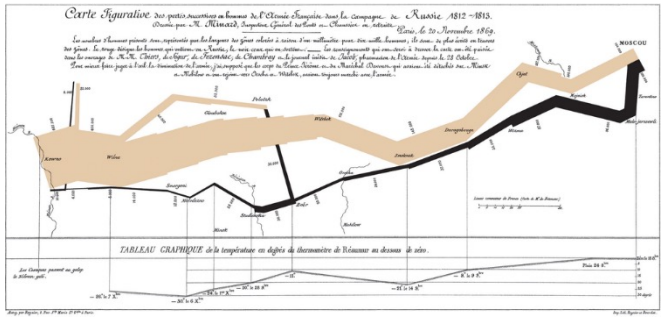


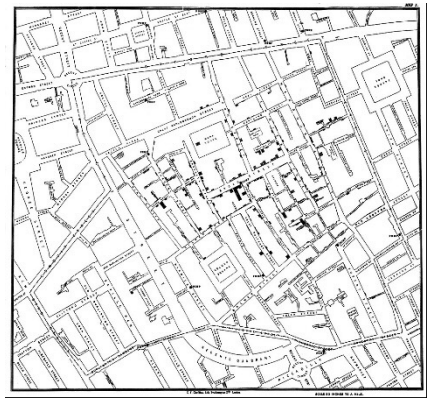


According to Friendly, statistical graphics researched its golden age between 1850-1900



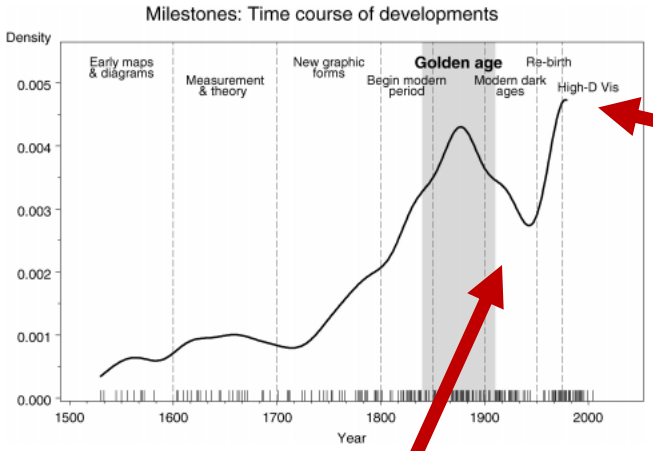


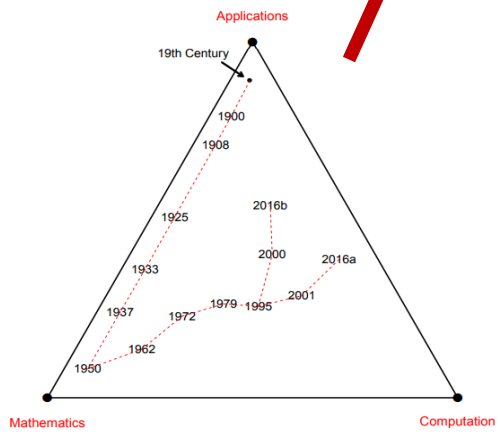


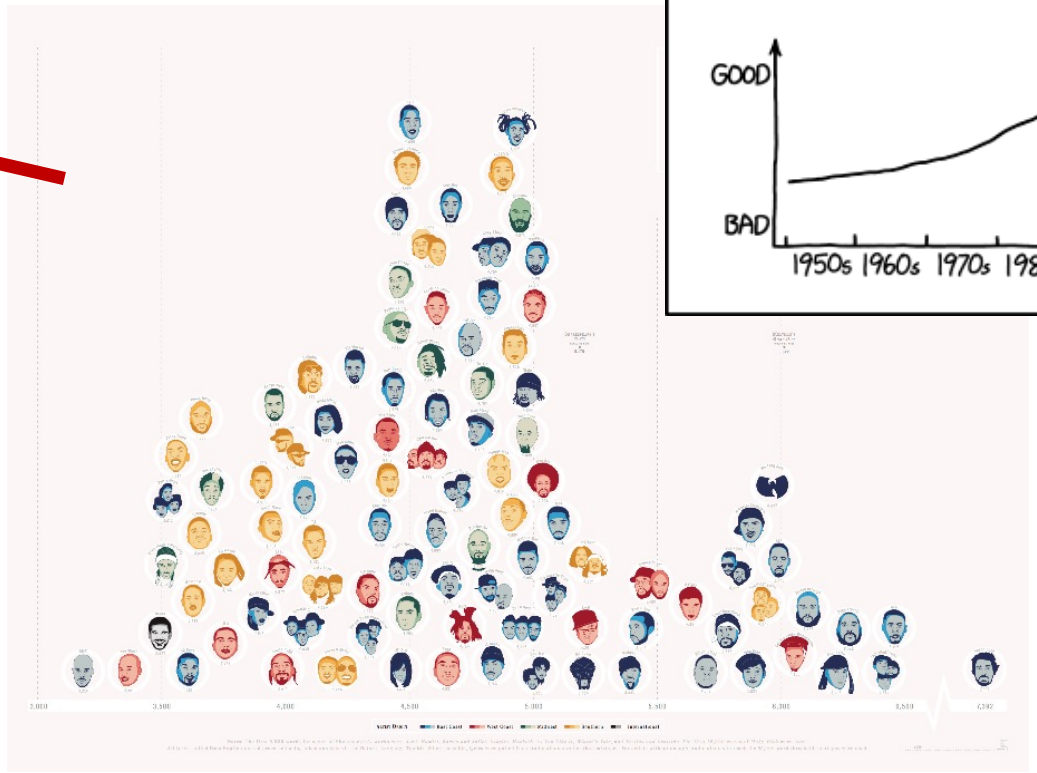


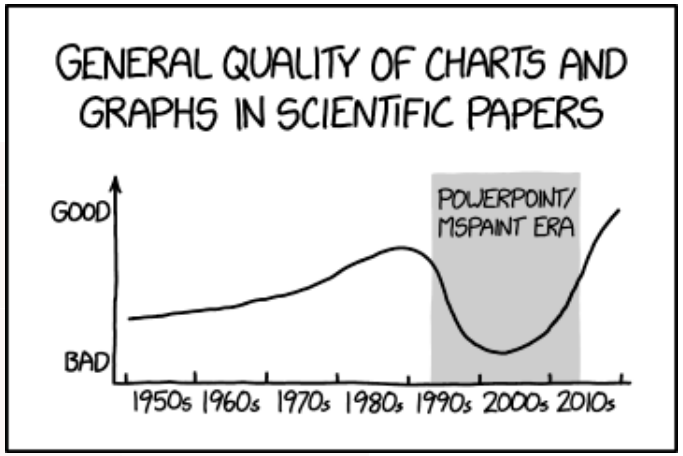
A very very brief history of data visualization

"Graphical dark ages" around 1950









Currently undergoing a "Graphical re-birth"

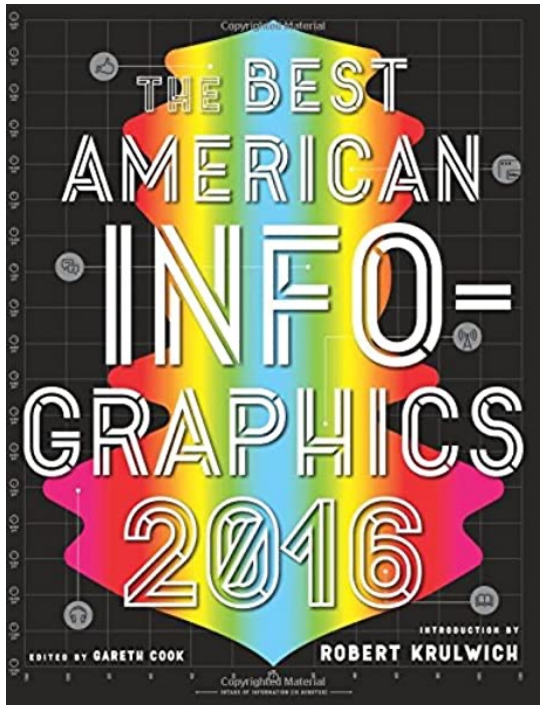
Computer Age Statistical Inference, Efron and Hastie

Coming up on homework 4: find an interesting data visualization…

Homework 4 : 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 now 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

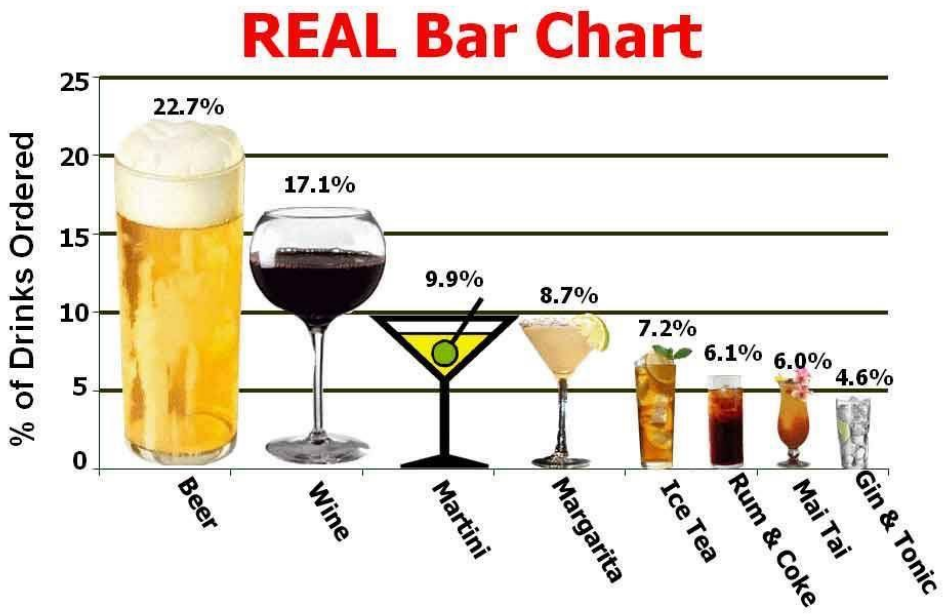
Review of visualizing data

```
import matplotlib.pyplot as plt

# categorical data
plt.pie(data, labels = label_names)
plt.bar(labels, data)

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

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



Percentiles

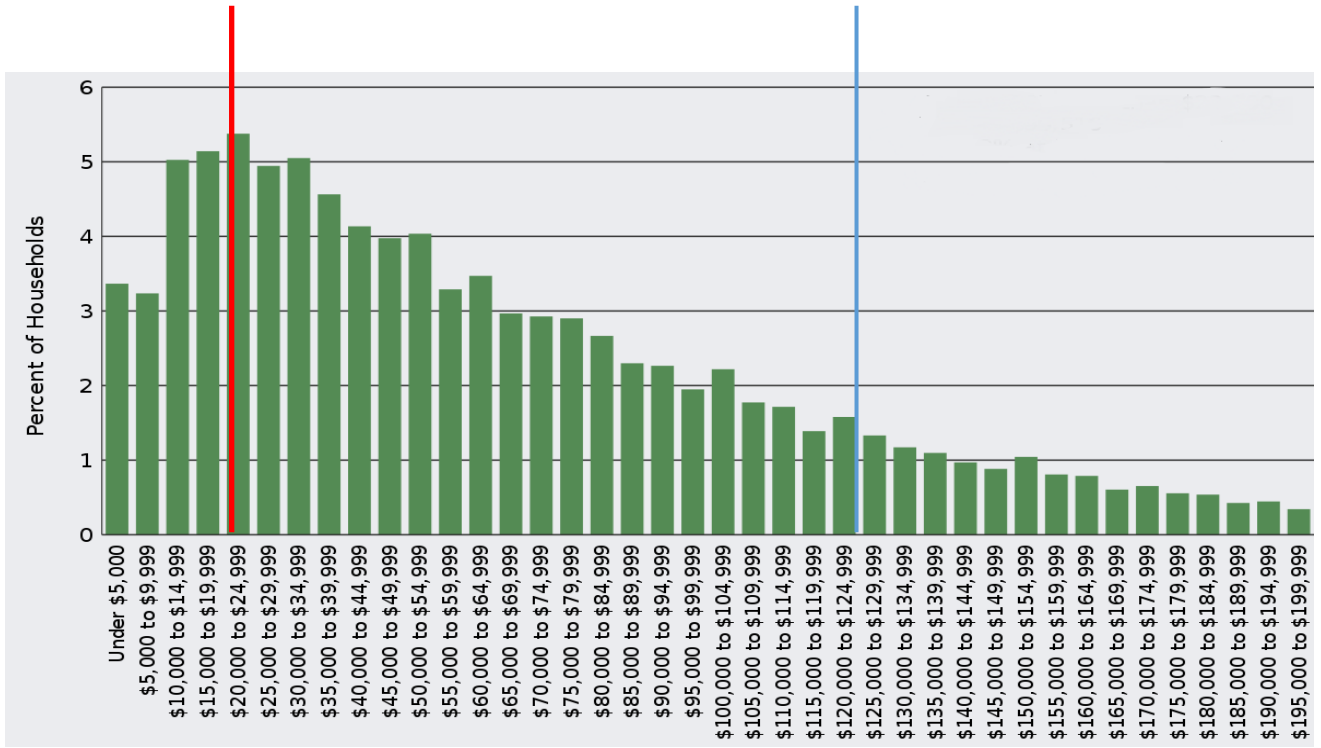
The **Pth percentile** is the value of a quantitative variable which is greater than P percent of the data

For the US income distribution what are the 20th and 80th percentiles?

np.quantile(*ndarray, q)*

20th percentile = $21,430

80th percentile = $112, 254



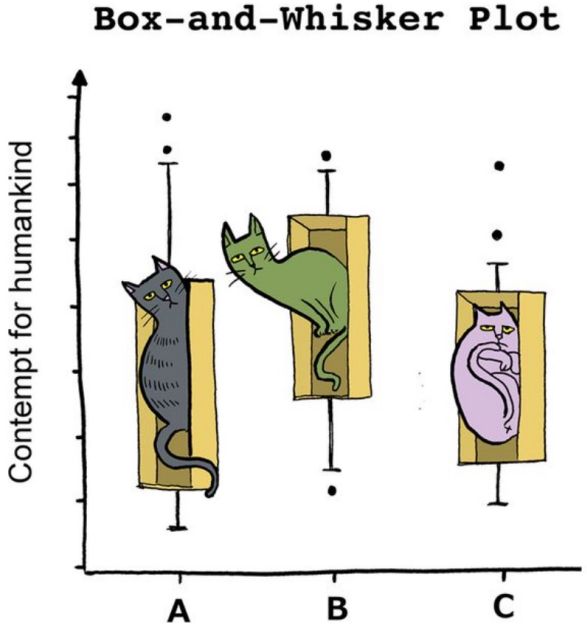
Five Number Summary

Five Number Summary = (minimum, Q_1, median, Q_3, maximum)

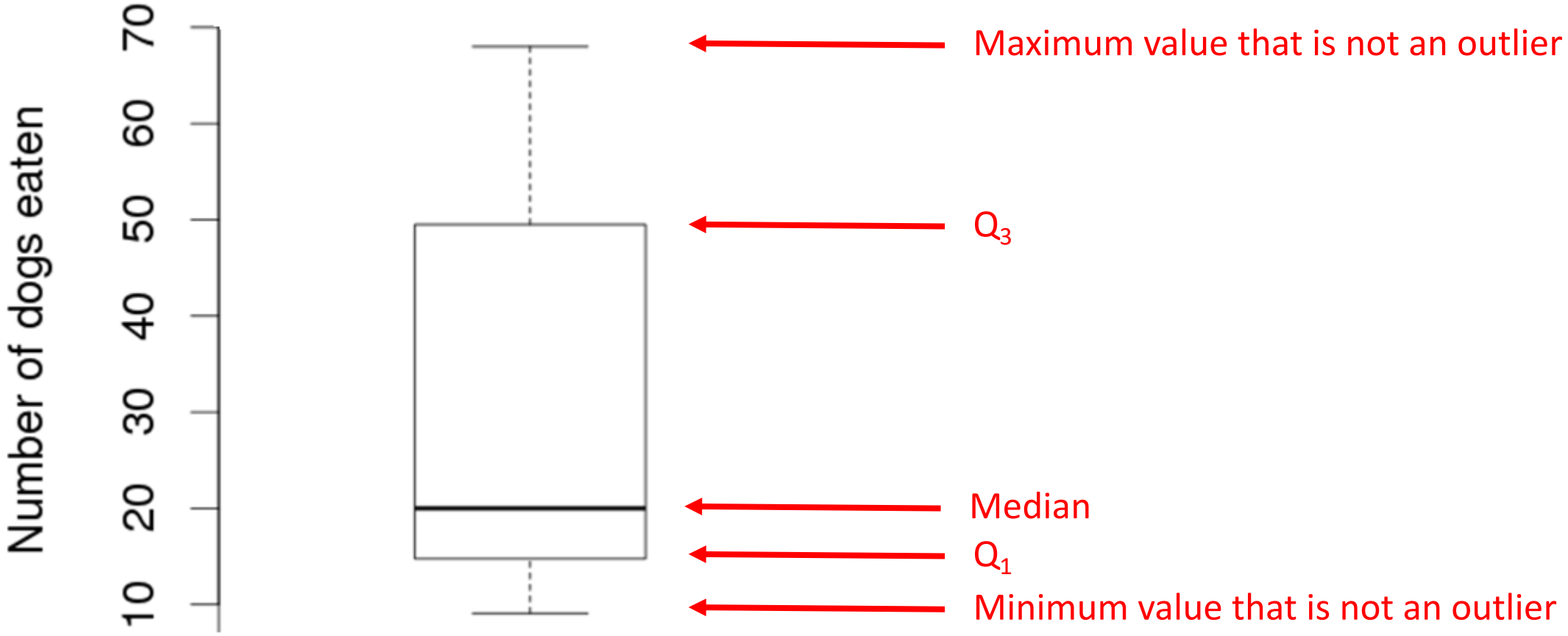
Q_1 = 25th percentile (also called 1st quartile)

Q_3 = 75th percentile (also called 3rd quartile)

Roughly divides the data into fourths



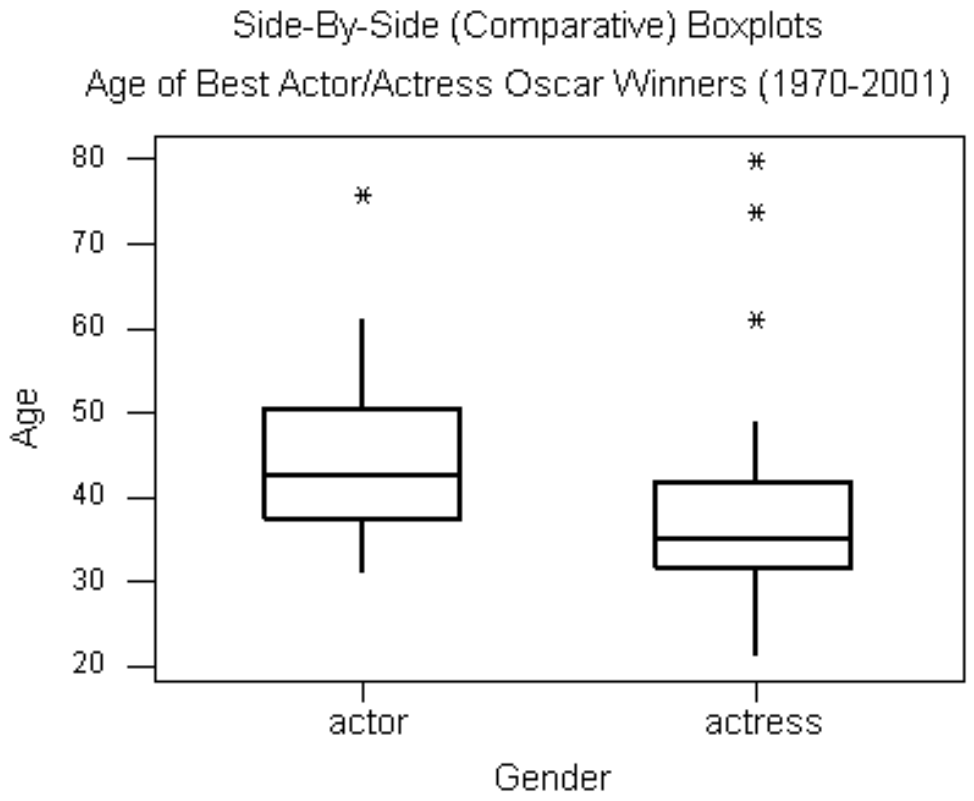
Box plot of the number of hot dogs eaten by the men's contest winners 1980 to 2010



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.



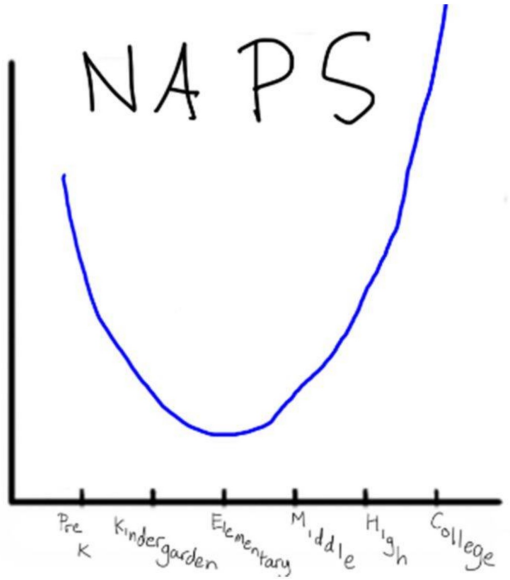

```
plt.boxplot( [data1, data2],  labels = ["lab1", "lab2"])
```

Let's explore this in Jupyter!

Visualizing time series

Q: How can we visualize a time series?

A: Line plot



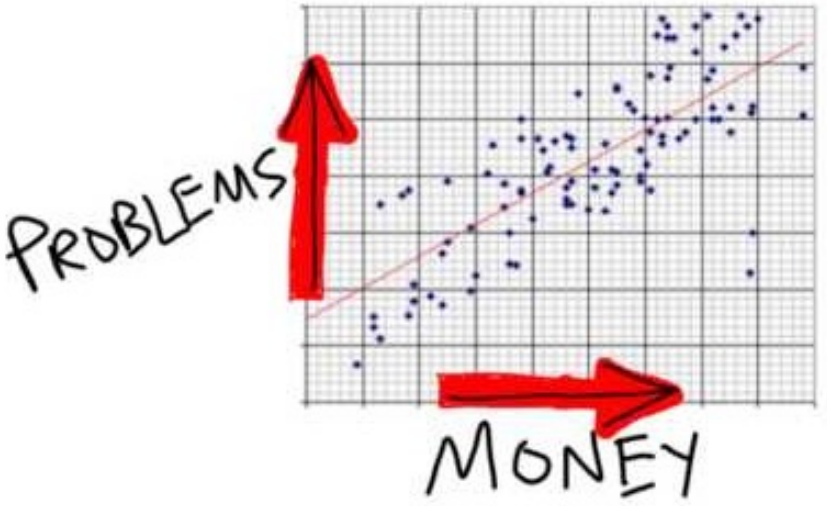

```
plt.plot(x1, y1, '-o', label = 'First line');
plt.plot(x2, y2, '-o', label = 'Second line');
plt.legend();
```

Let's explore 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!

Subplots: pyplot interface

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

1 row

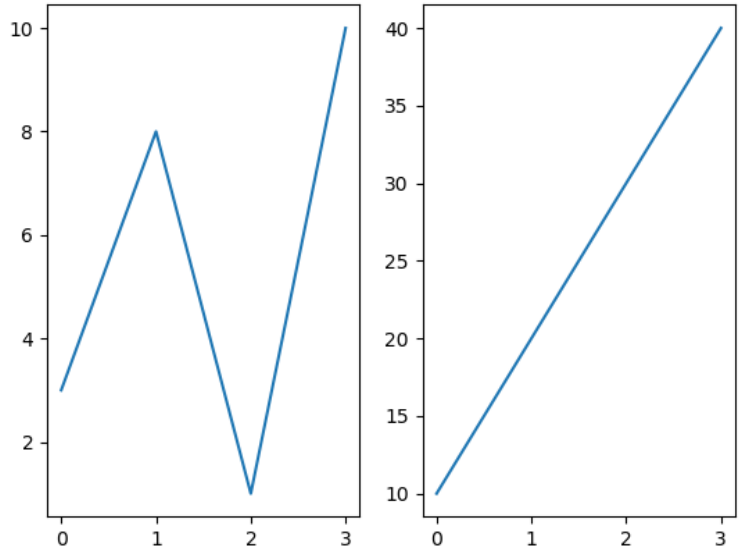
2 columns

```
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



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

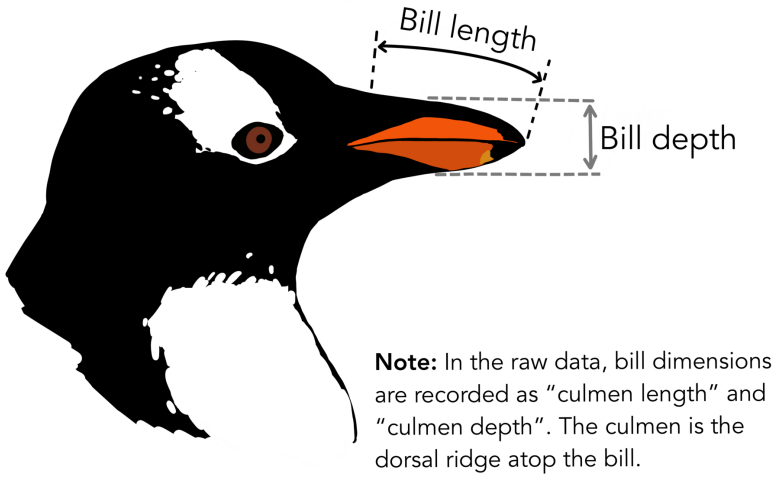
To make plots better looking we can set a theme

```
import seaborn as sns

sns.set_theme()
```

Inspiration: Palmer penguins

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



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

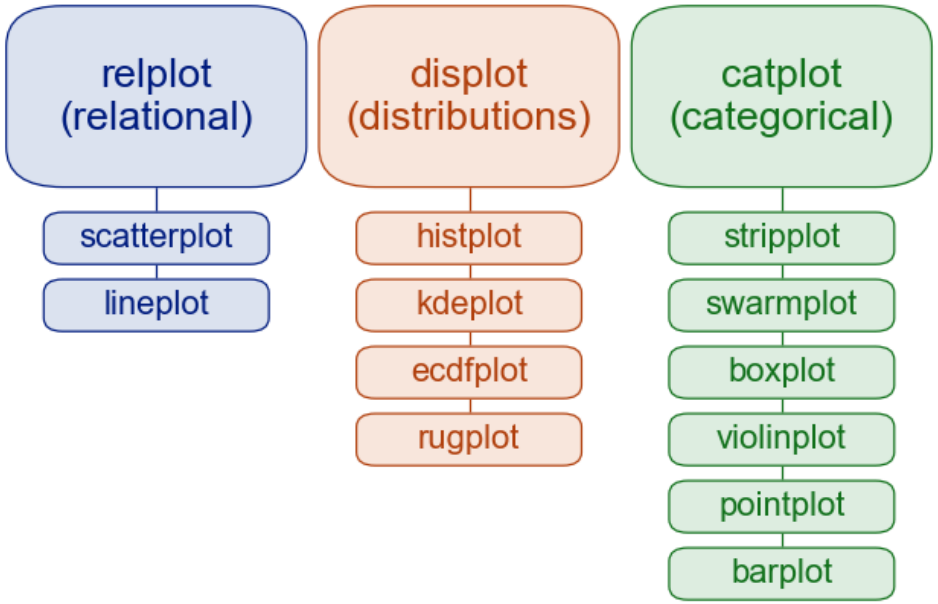
In particular, there are plots for:

1. Two quantitative variables
   - sns.relplot()

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

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

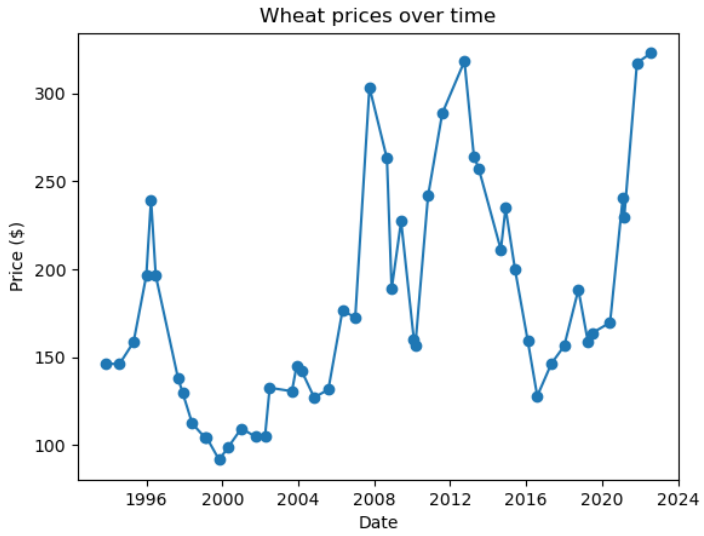
Figure level plots

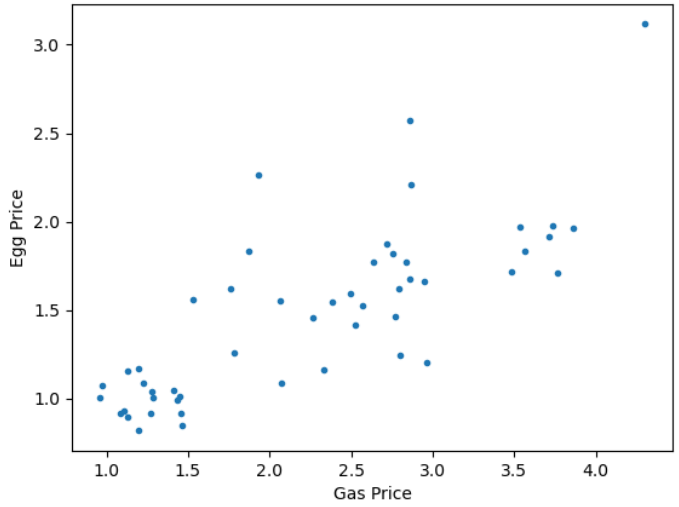


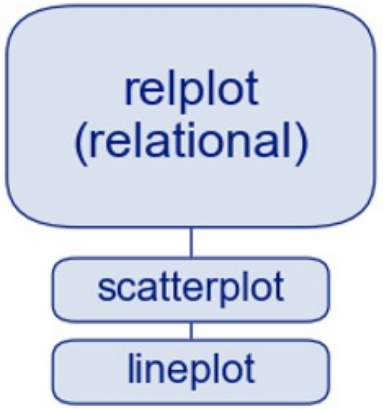
Plots for two quantitative variable

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

- Line plots and scatter plots!





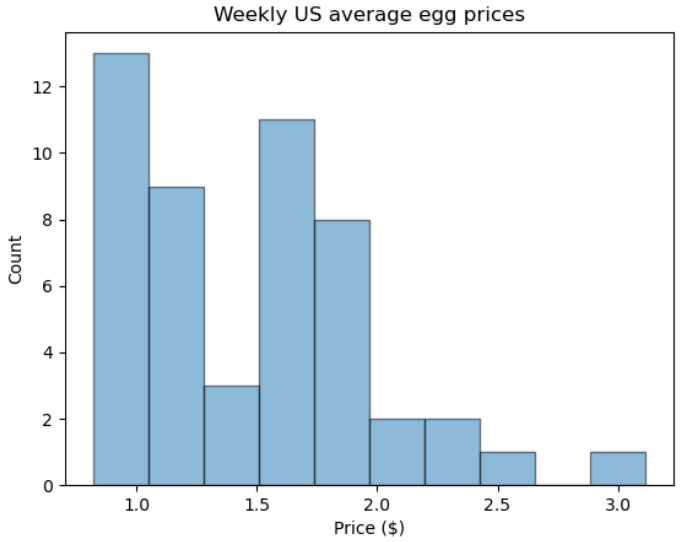


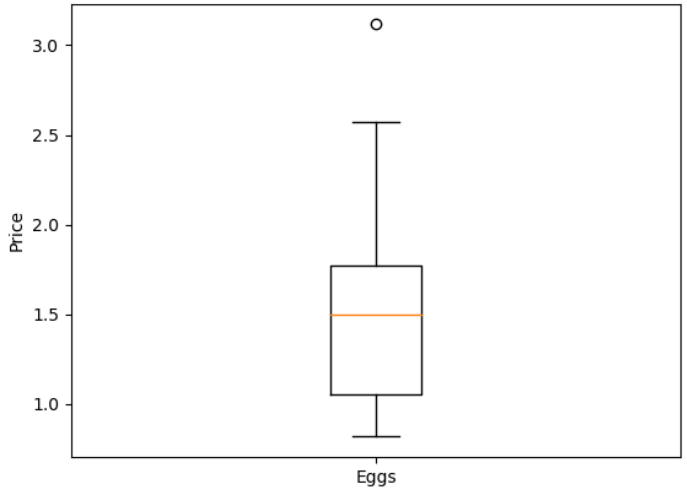
Let's explore this in Jupyter!

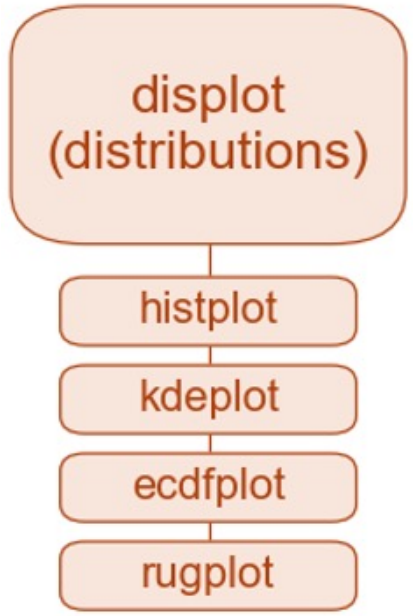
Plots for a single quantitative variable

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

- Histograms and boxplots





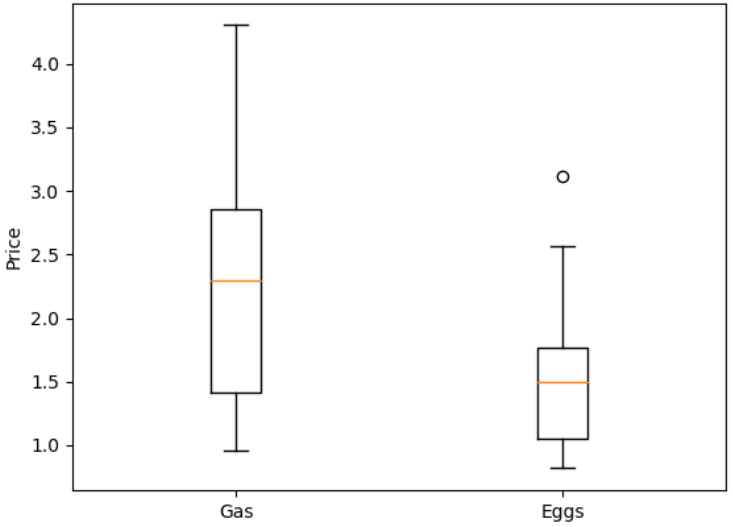


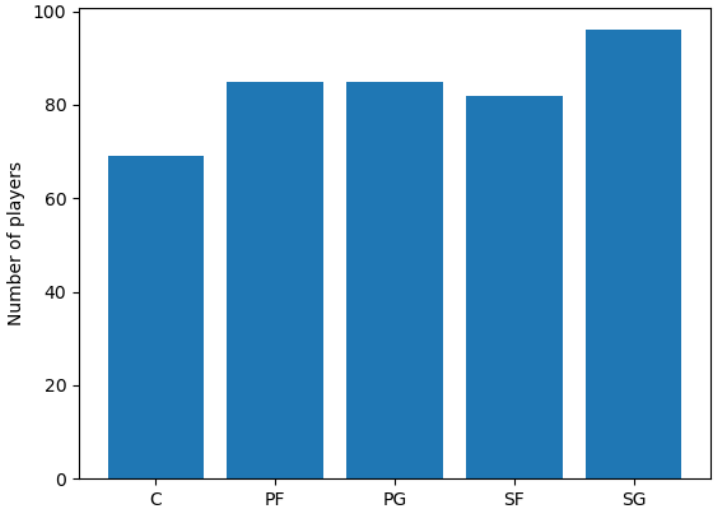
Let's explore this in Jupyter!

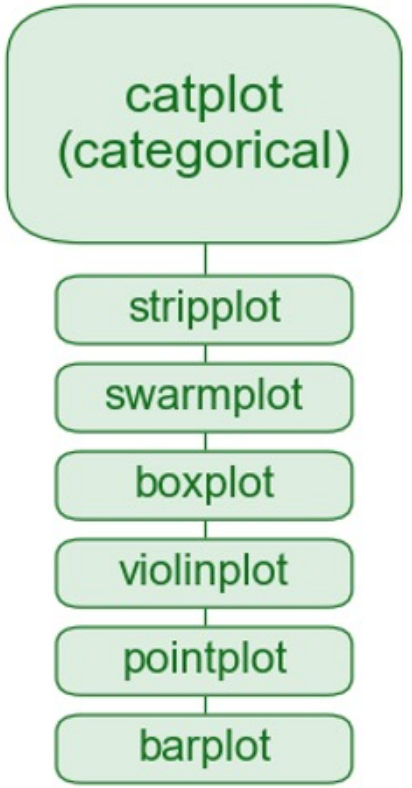
Plots for quantitative data comparing across different categorical levels

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!

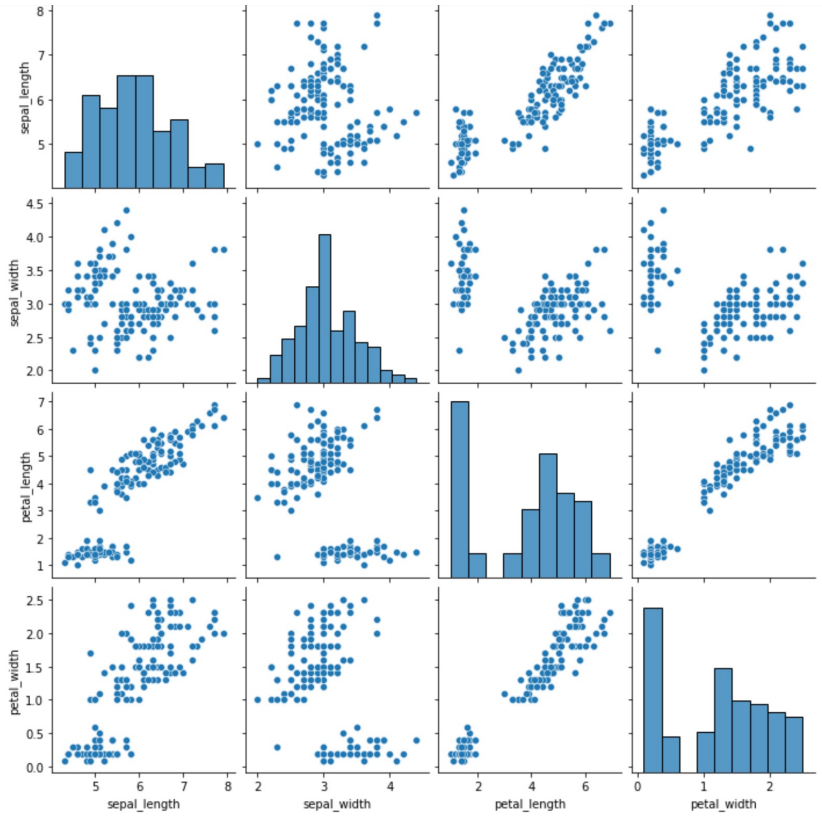
Pairs plot

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

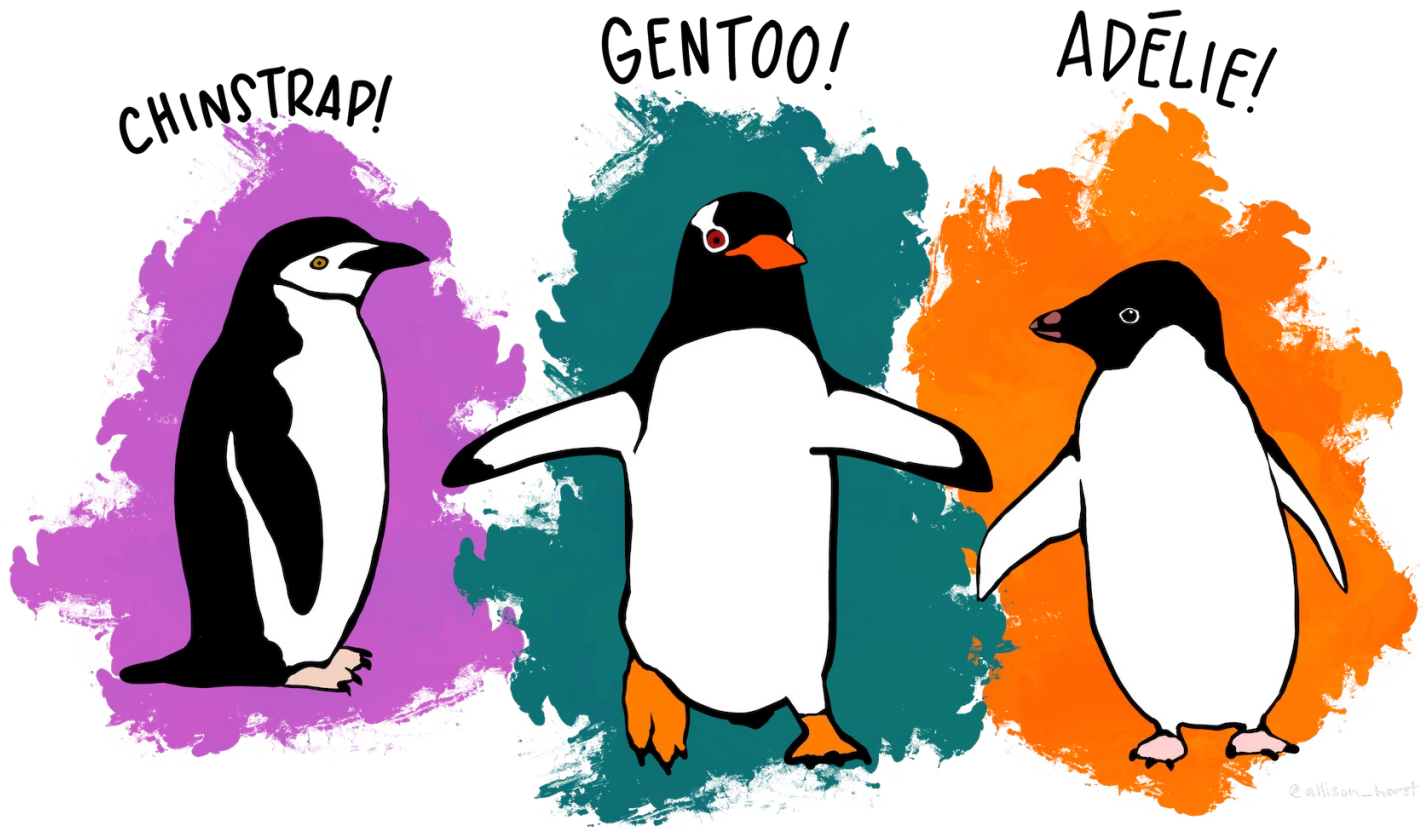
It can be one of the most useful ways to see relationships between multiple quantitative variabels

We can create pairs plots in seaborm using:

- sns.pairplot(the_data)



Let's explore this in Jupyter!


CHINSTRAP!
GENTOO!
ADÉLIE!
@allison_horst

Interactive visualizations for data exploration

Interactive visualizations are useful for exploring data to find trends

- They can be shared on the internet
- They can't be put in static pdfs
  - But can still be useful for your final project to find trends that you can display with static graphics

We will use plotly to create interactive graphics

- import plotly.express as px

Plotly interactive plots

Line plots

- fig = px.line(data_frame =, x =, y = , color = , hover_name = , line_shape =)

Scatter plots

- px.scatter(data_frame = , x = , y = , size = , color = , hover_name =)

Add axis labels

- fig.update_layout(xaxis_title="X", yaxis_title="Y")

Let's explore this in Jupyter!

Next class…

Interactive graphics continued and maps