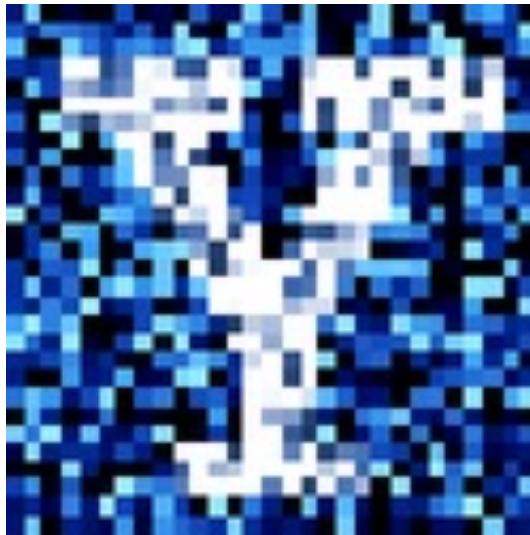


YData: Introduction to Data Science



Class 9: Data visualization

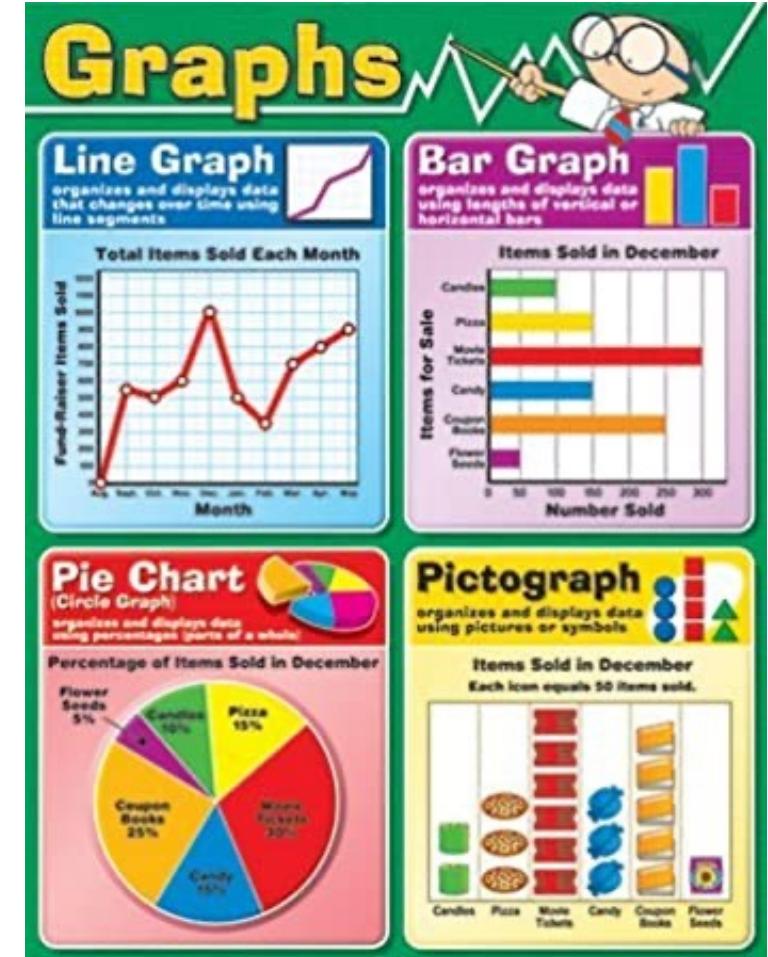
Overview

Quick review of pandas DataFrames and joining DataFrames

History of Data Visualization

Review and additional features of visualizing data with matplotlib

If there is time: visualizing data with seaborn



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

Quick review: pandas DataFrames

PLAYER	POSITION	TEAM	SALARY
	str	str	f64
"Paul Millsap"	"PF"	"Atlanta Hawks"	18.671659
"Al Horford"	"C"	"Atlanta Hawks"	12.0
"Tiago Splitter..."	"C"	"Atlanta Hawks"	9.75625
"Jeff Teague"	"PG"	"Atlanta Hawks"	8.0
"Kyle Korver"	"SG"	"Atlanta Hawks"	5.746479

Pandas DataFrames hold Table data

Selecting columns:

- `my_df[["col1", "col2"]]` # getting multiple columns using a list
- `my_df.drop(columns = ["co1", "col2"])` # removing columns

Extracting rows:

- `my_df.iloc[0]` # getting a row by number
- `my_df.loc["index_name"]` # getting a row by Index value
- `my_df [my_df["col_name"] == 7]` # getting rows using a Boolean mask
- `my_df.query("col_name == 7")` # getting rows using the query function

Quick review: pandas DataFrames

Sorting rows of a DataFrame

```
my_df.sort_values("col_name", ascending = False) # sort from largest to smallest
```

Adding a new:

- `my_df["new_col"] = values_array`

Renaming a column:

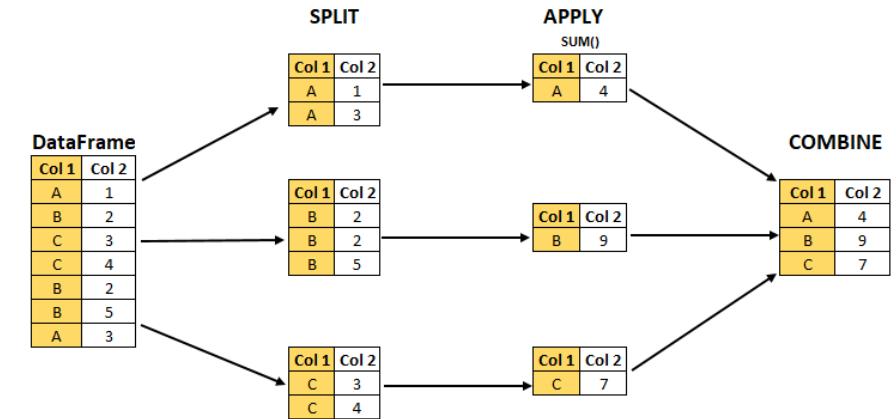
- `rename_dictionary = {"old_col_name": "new_col_name"}`
- `my_df.rename(columns = rename_dictionary)`

Creating aggregate statistics by group

We can get statistics separately by group:

- `dow.groupby("Year").agg("max")`

```
my_df.groupby("group_col_name").agg(  
    new_col1 = ('col_name', 'statistic_name1'),  
    new_col2 = ('col_name', 'statistic_name2'),  
    new_col3 = ('col_name', 'statistic_name3')  
)
```



```
nba_salaries.groupby("TEAM").agg(  
    max_salary = ("SALARY", "max"),  
    min_salary = ("SALARY", "min"),  
    first_player = ("PLAYER", "min")  
)
```

Review: Joining

Suppose we have two DataFrames (or Series) called `x_df` and `y_df`

- `x_df` have one column called `x_vals`
- `y_df` has one column called `y_vals`

Index `x_vals`

1	x1
2	x2
3	x3

DataFrame: x_df

Index `y_vals`

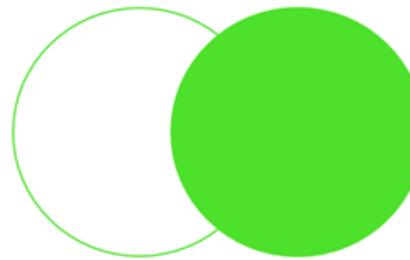
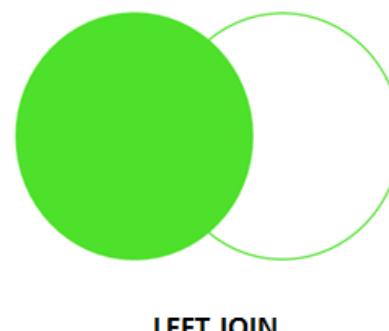
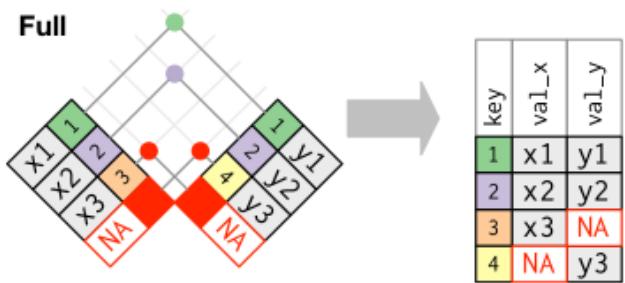
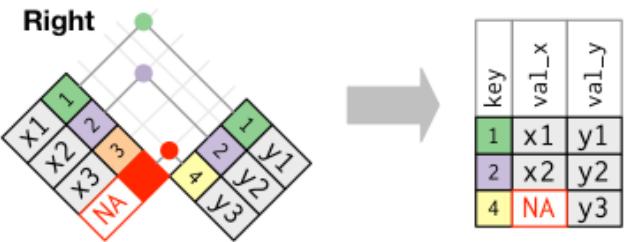
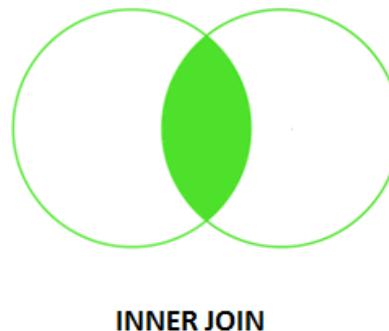
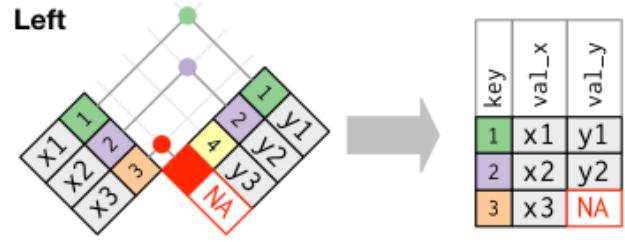
1	y1
2	y2
3	y3

DataFrame: y_df

We can join these two DataFrames into a single DataFrame by aligning rows with the same Index value using the general syntax: `x_df.join(y_df)`

- i.e., the new joined data frame will have two columns: `x_vals`, and `y_vals`

Review: Joining



`x_df.join(y_df, how = "left") # based on Index`

`x_df.merge(y_df, how = "inner", left_on = "x_col", right_on = "y_col") # based on columns`

Questions?



Let's do a quick review and a few warm-up exercise in Jupyter!

Data visualization!



A very brief history of data visualization...

Statistical Science
2008, Vol. 23, No. 4, 502–535
DOI: 10.1214/08-STS268
© Institute of Mathematical Statistics, 2008

The Golden Age of Statistical Graphics

Michael Friendly

Data visualization

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



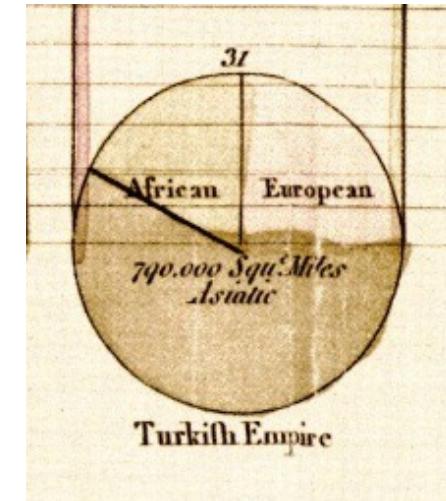
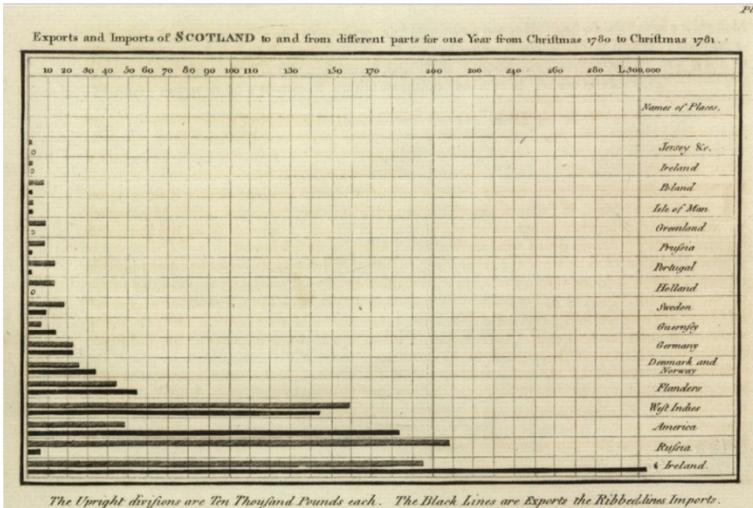
Whatever relates to extent and quantity may be represented by geometrical figures. Statistical projections which speak to the senses without fatiguing the mind, possess the advantage of fixing the attention on a great number of important facts.

—Alexander von Humboldt, 1811

A very brief history of data visualization

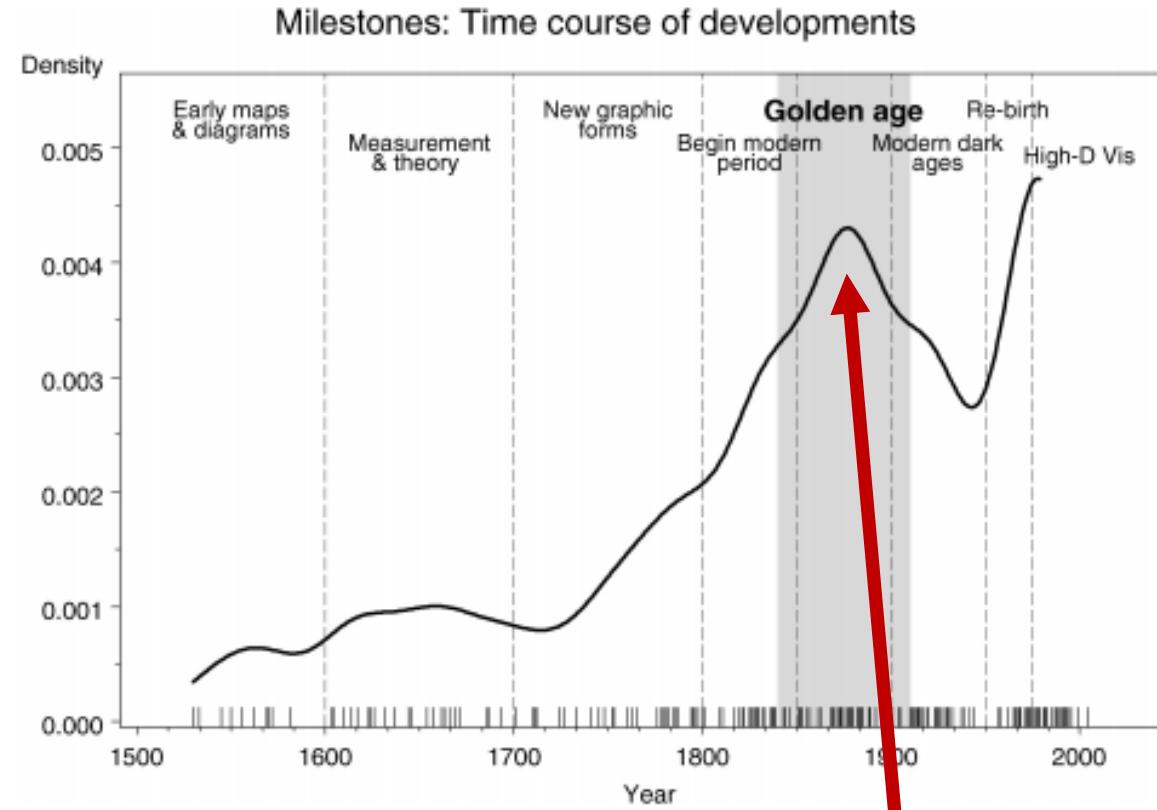
The age of modern statistical graphs began around the beginning of the 19th century

[William Playfair](#) (1759-1823) is credited with inventing the line graph, bar chart and pie chart



A very brief history of data visualization

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

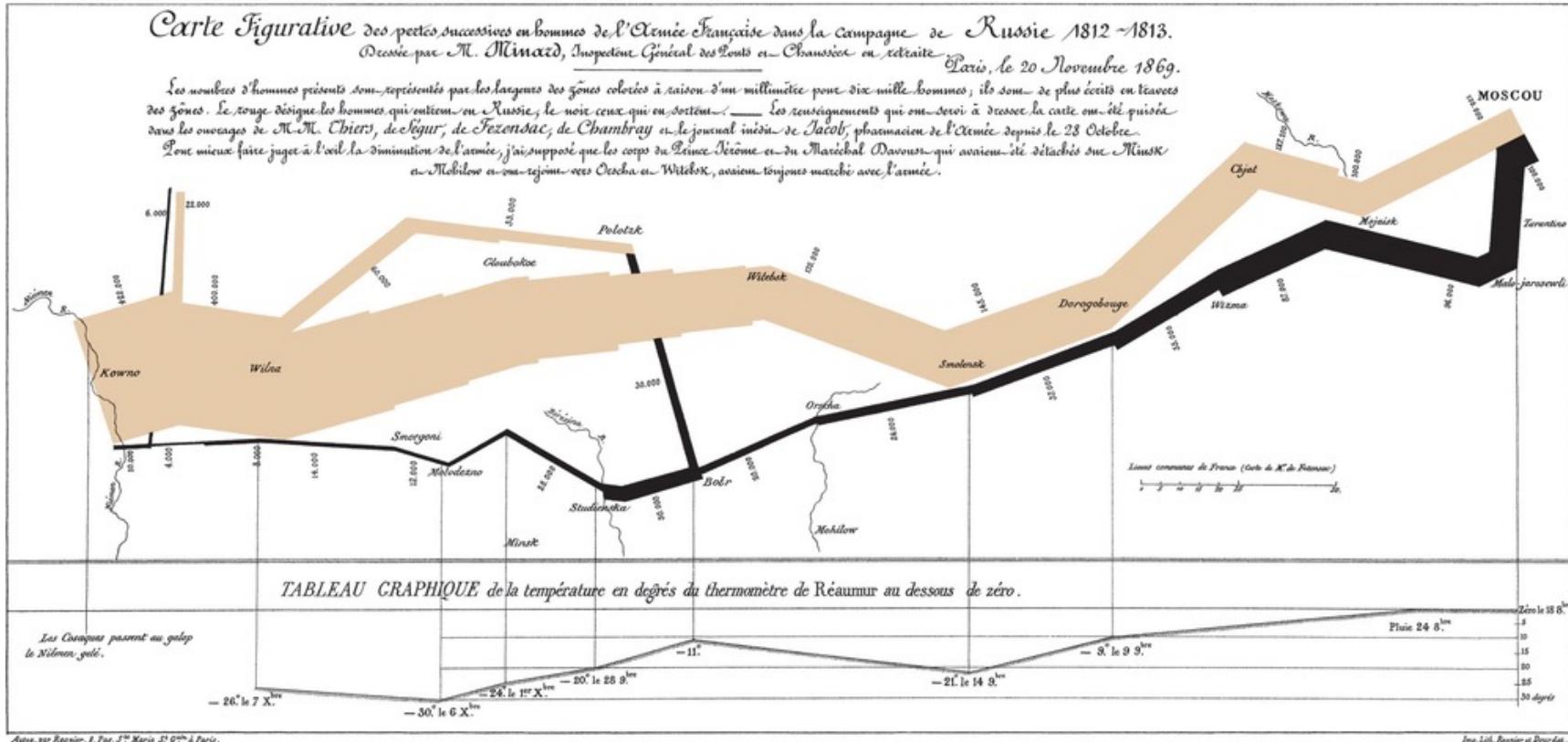


A very brief history of data visualization

Joseph Minard (1781-1870)

Map of Napoleon's march on Russia

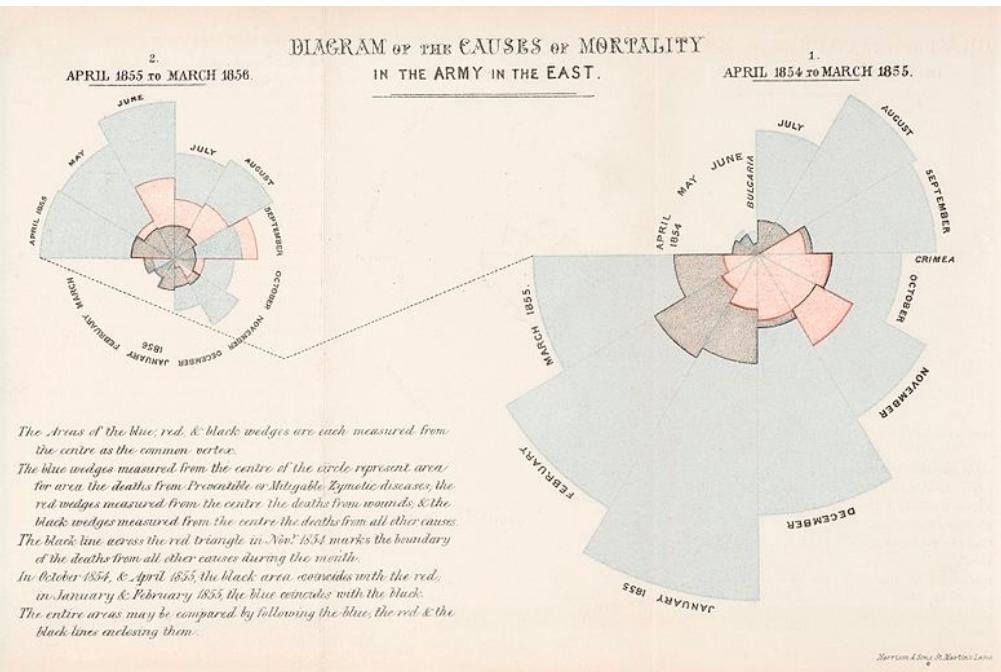
"It may well be the best statistical graphic ever drawn" – E. Tufte



A very brief history of data visualization

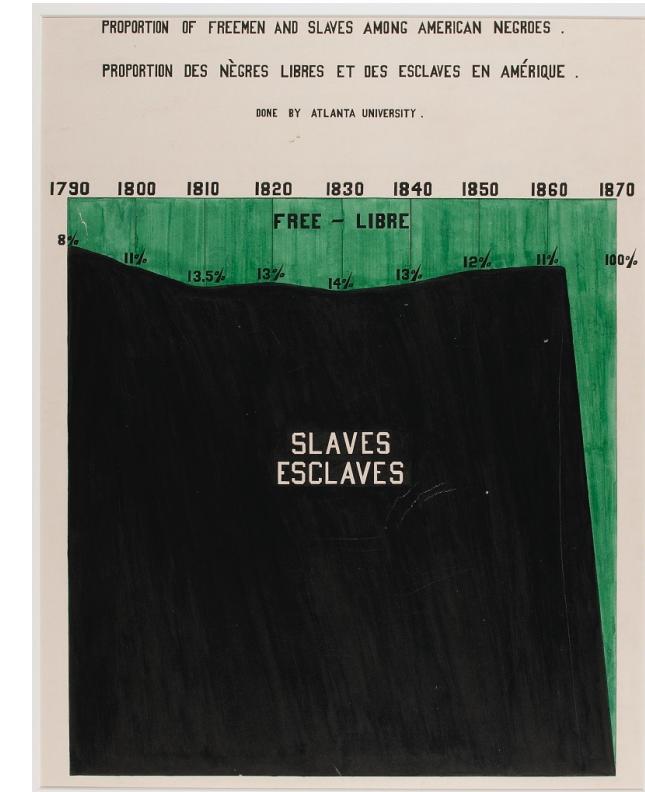
Florence Nightingale (1820-1910)

Causes of mortality in the army in the east



W.E.B. Du Bois (1868-1963)

Percent of African Americans who were slaves

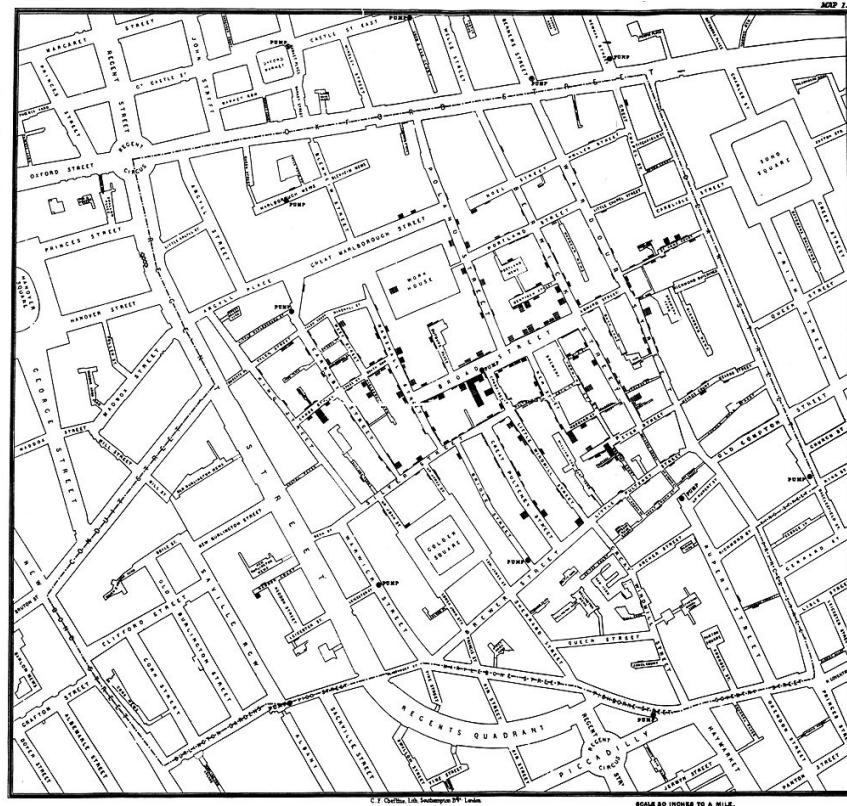


[See Dubois Visualization Challenge](#)

A very brief history of data visualization

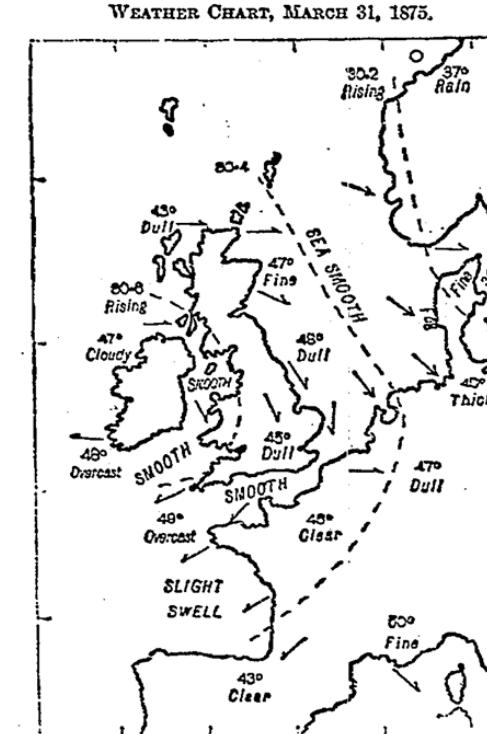
John Snow (1813-1858)

Clusters of cholera cases in London epidemic of 1854



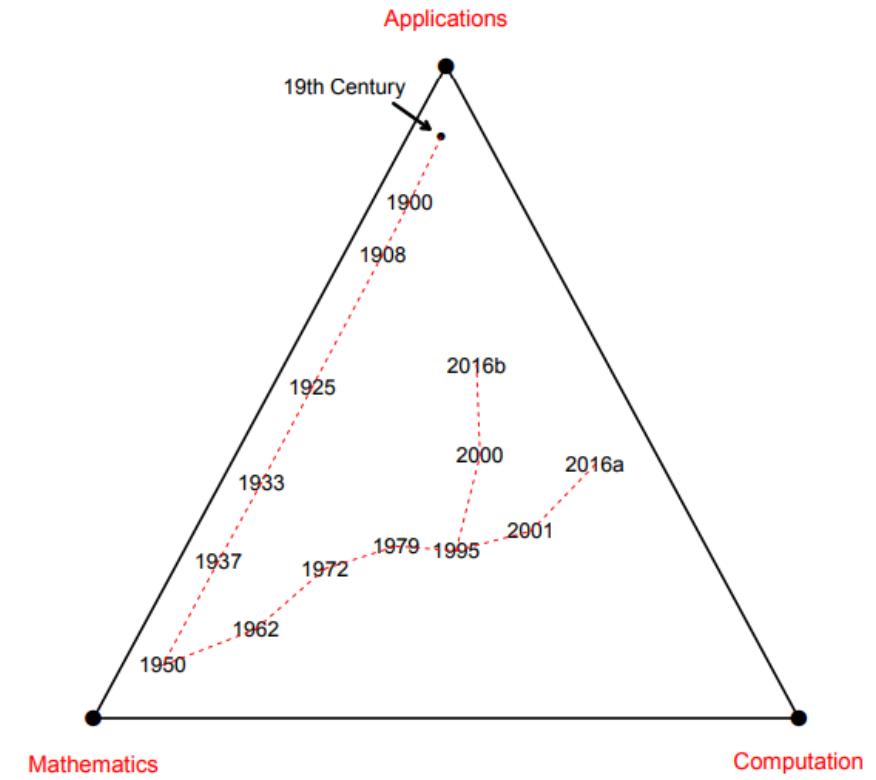
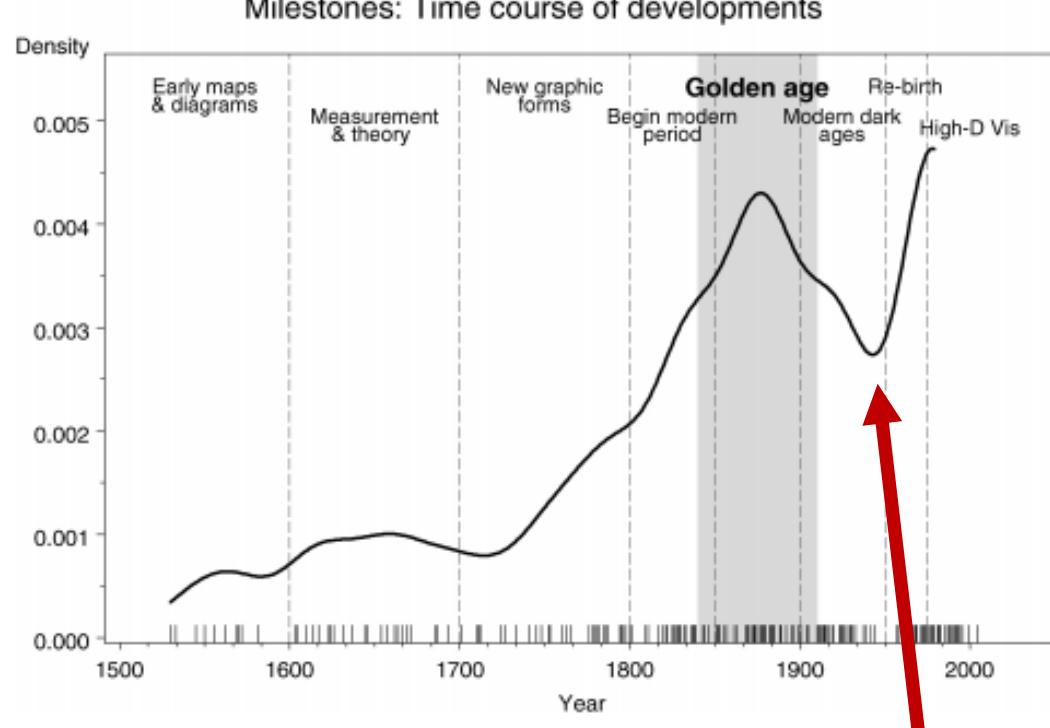
Francis Galton (1822-1911)

First weather map published in a newspaper (1875)



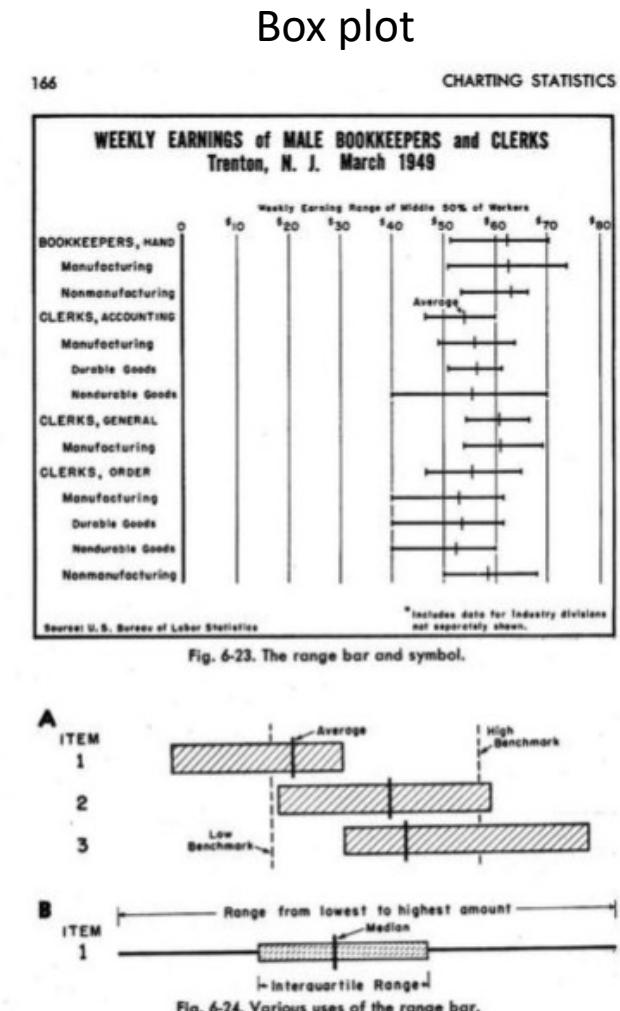
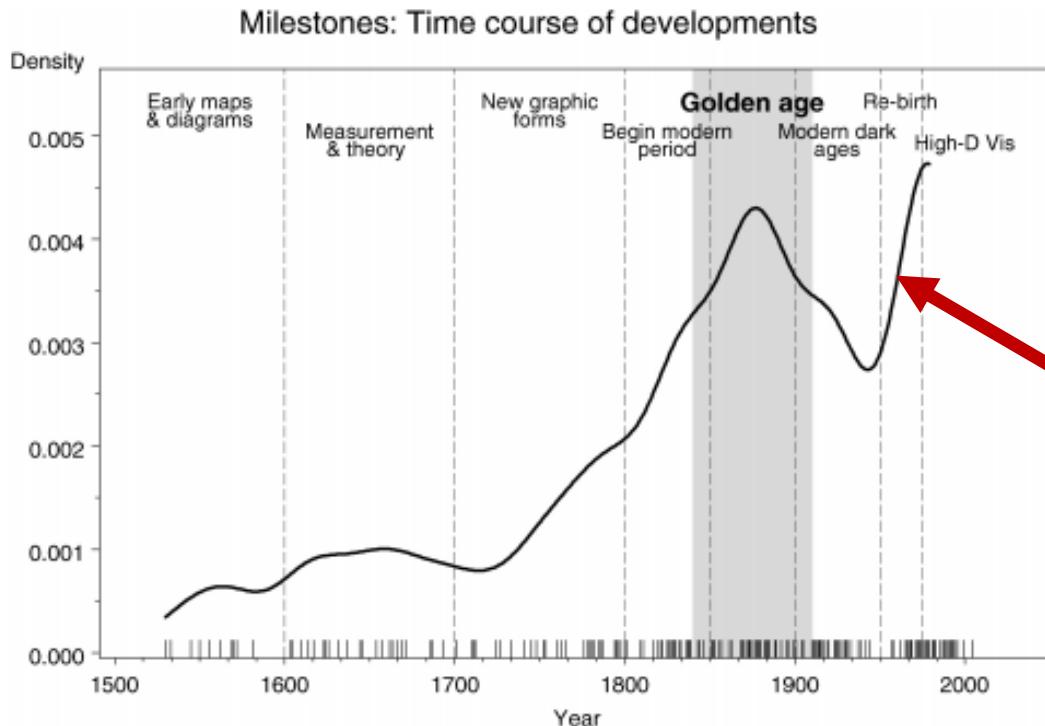
A very brief history of data visualization

“Graphical dark ages” around 1950



A very brief history of data visualization

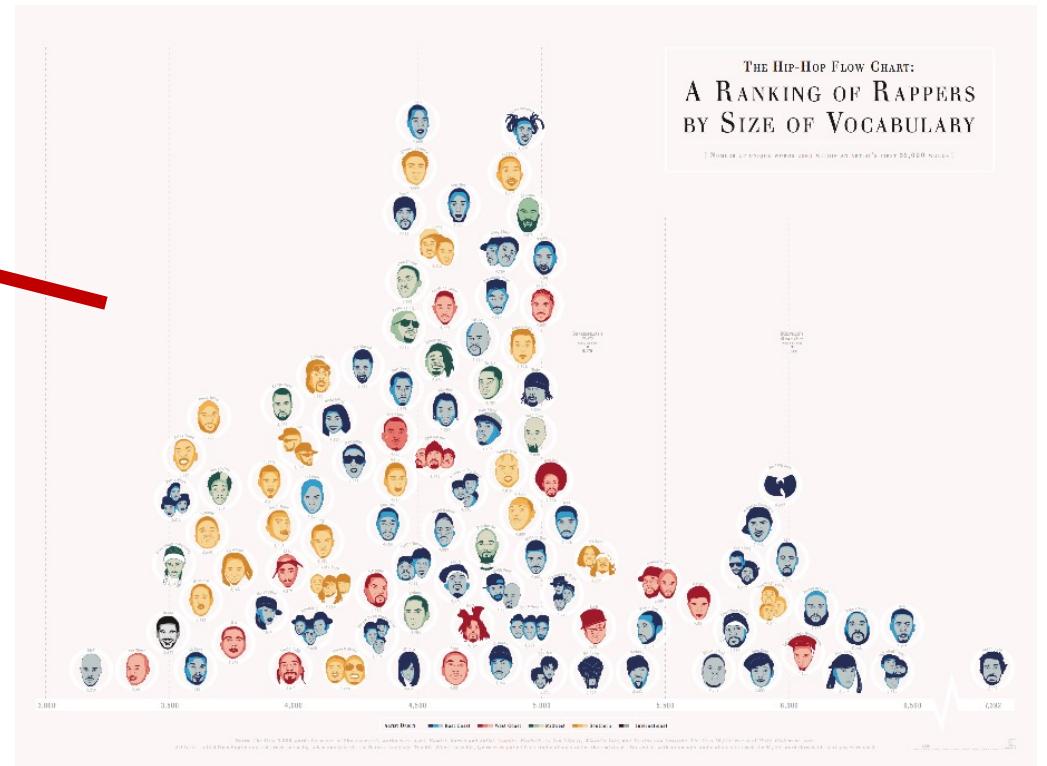
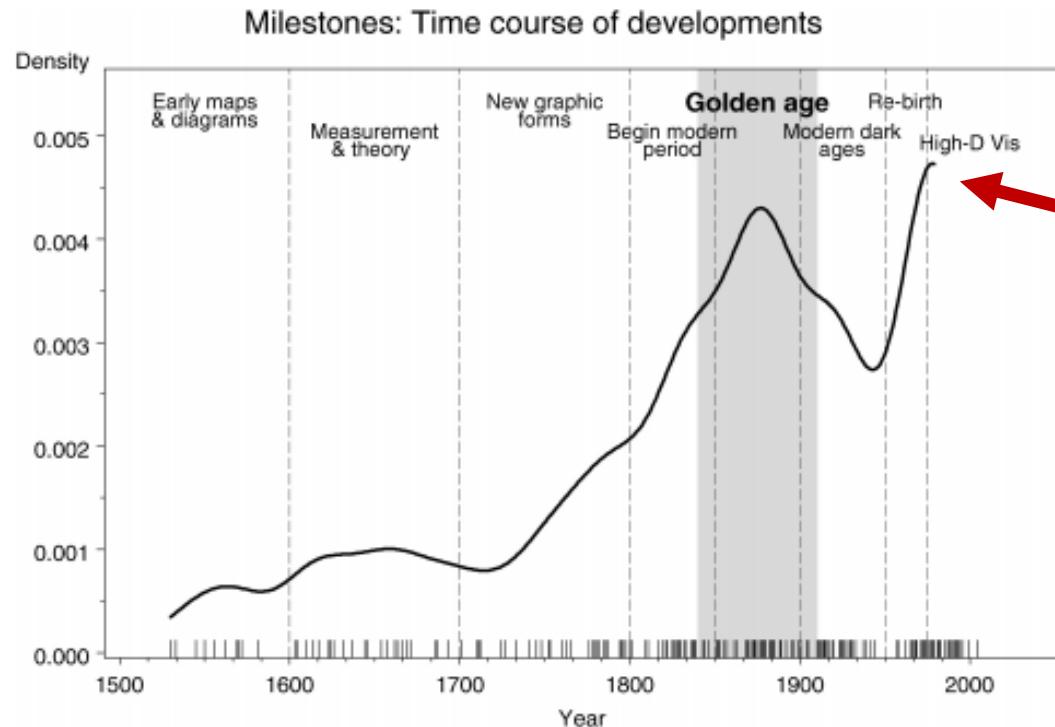
Currently undergoing a “Graphical re-birth”



[Spear 1952](#), [Tukey 1970](#)

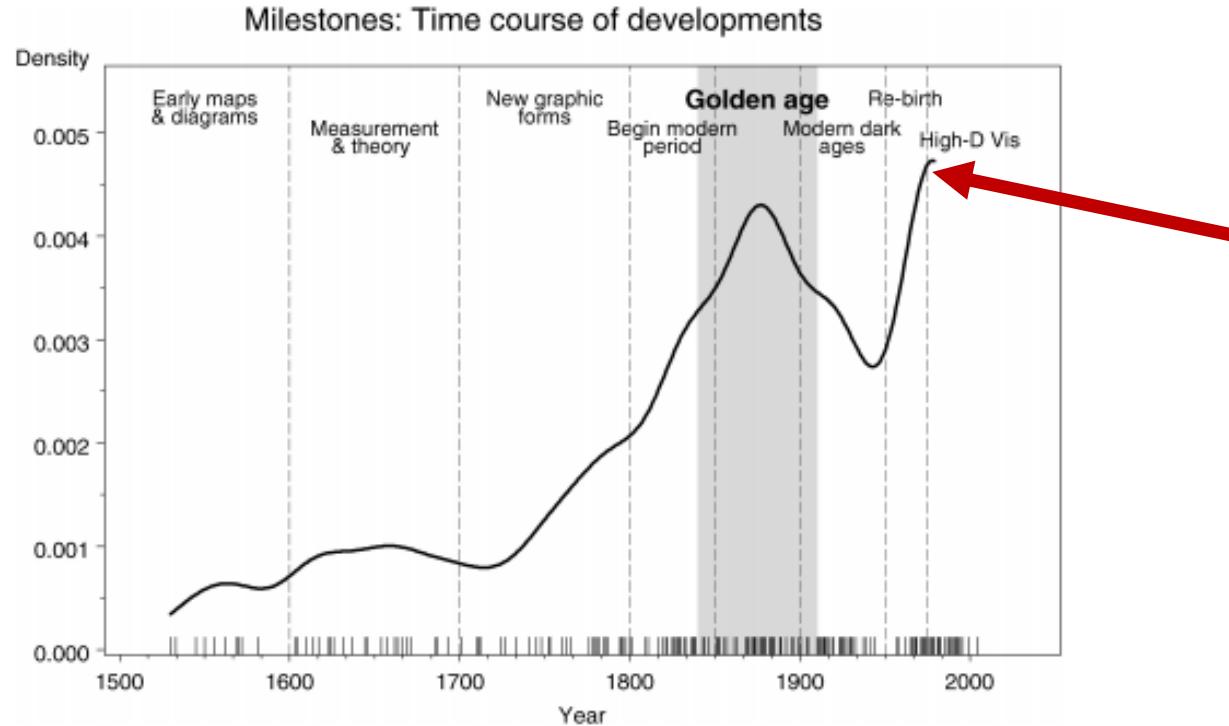
A very brief history of data visualization

Currently undergoing a “Graphical re-birth”



A very brief history of data visualization

Currently undergoing a “Graphical re-birth”

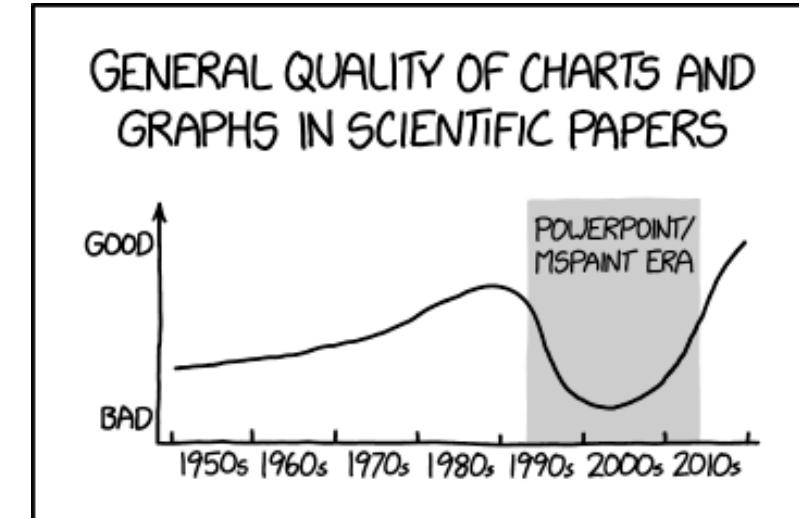
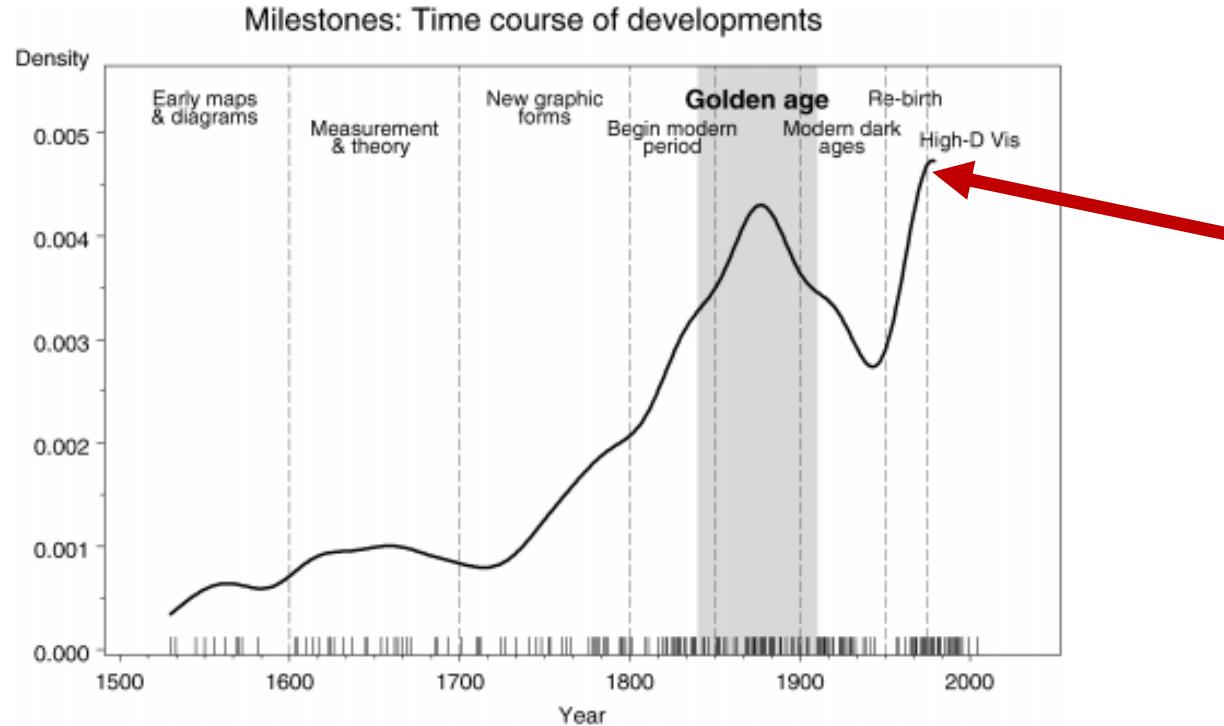


Hans Rosling's gapminder

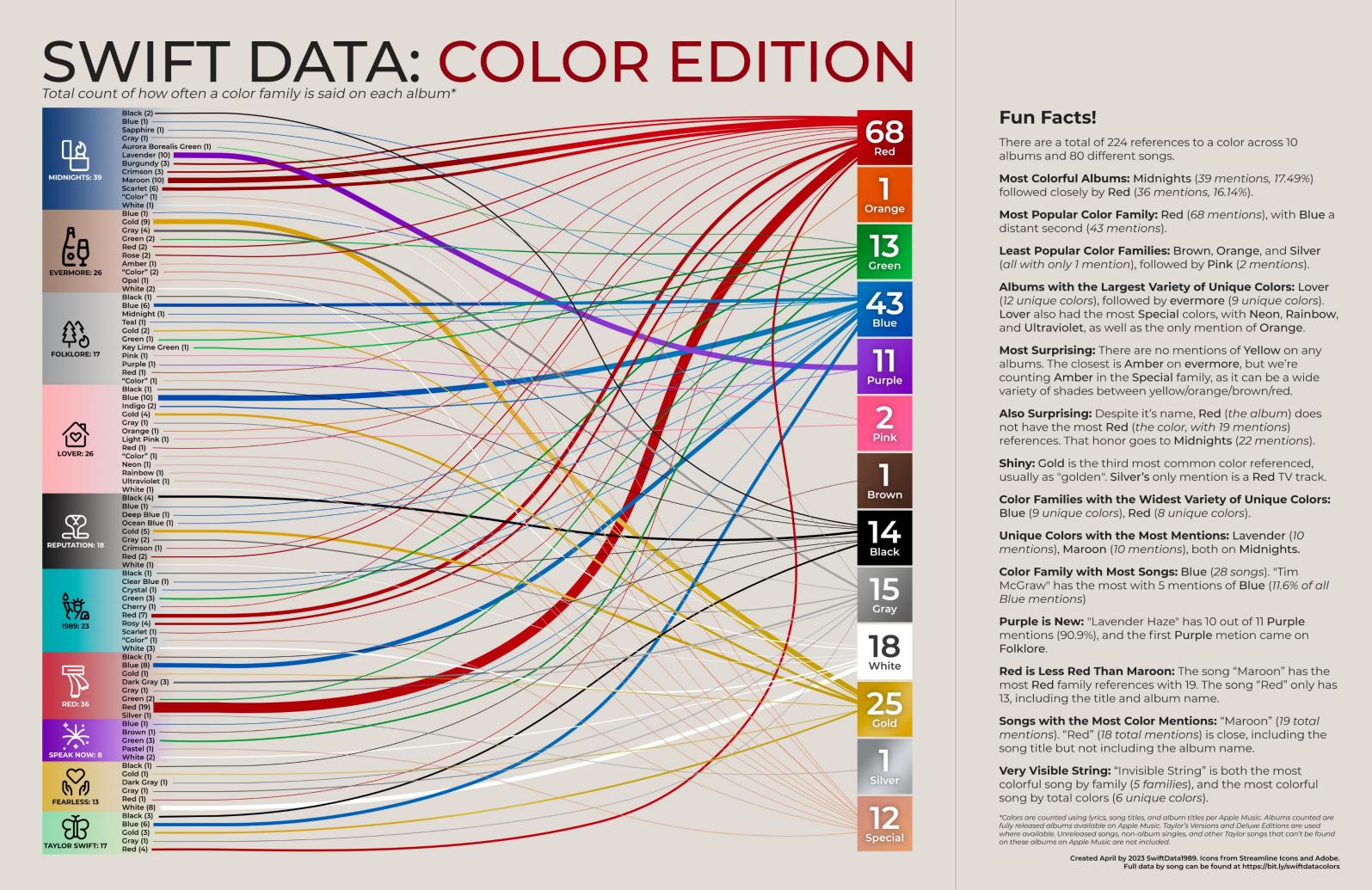
- [Simple version](#)
- [TV special effects](#)
- [Ted Talk](#)

A very brief history of data visualization

Currently undergoing a “Graphical re-birth”



A very brief history of data visualization

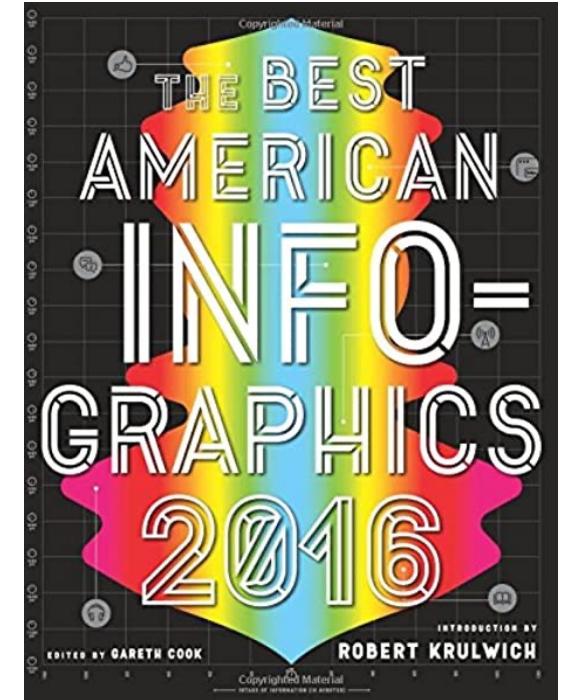


Coming up on homework 4: find an interesting data visualization...

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 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")
```

**World's Most Accurate
Pie Chart**



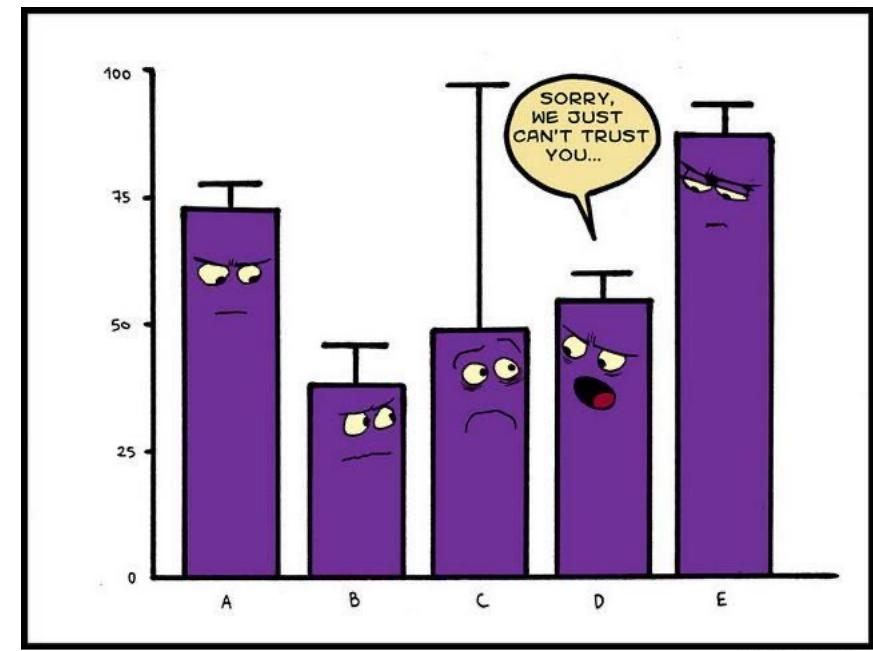
REAL Bar Chart

Review of visualizing one quantitative variable

Q: How can we visualize one quantitative variable?

A: Histograms!

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



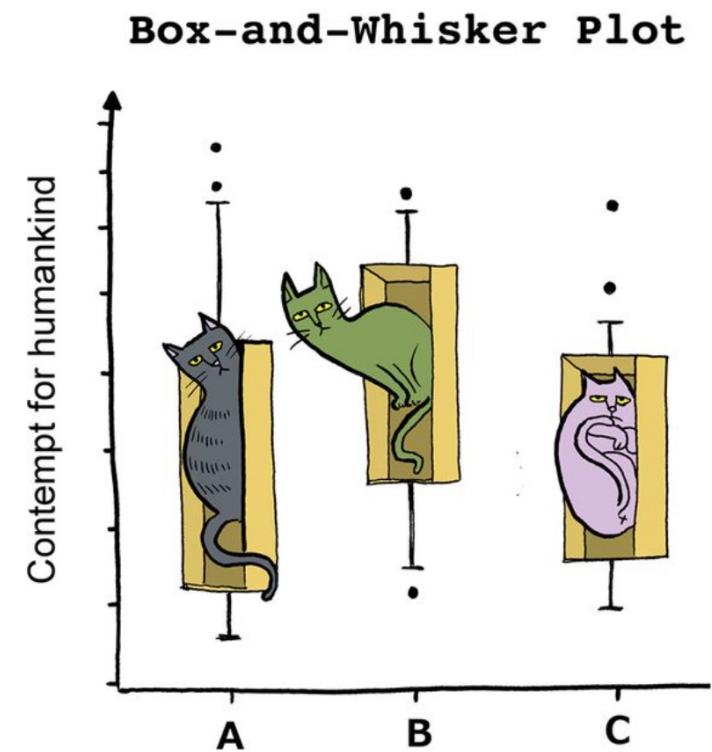
Let's review this in Jupyter!

Review of visualizing one quantitative variable

Q: Do you know of any other ways to visualize quantitative data?

A: Boxplots!

Let's first talk about percentiles...



Percentiles

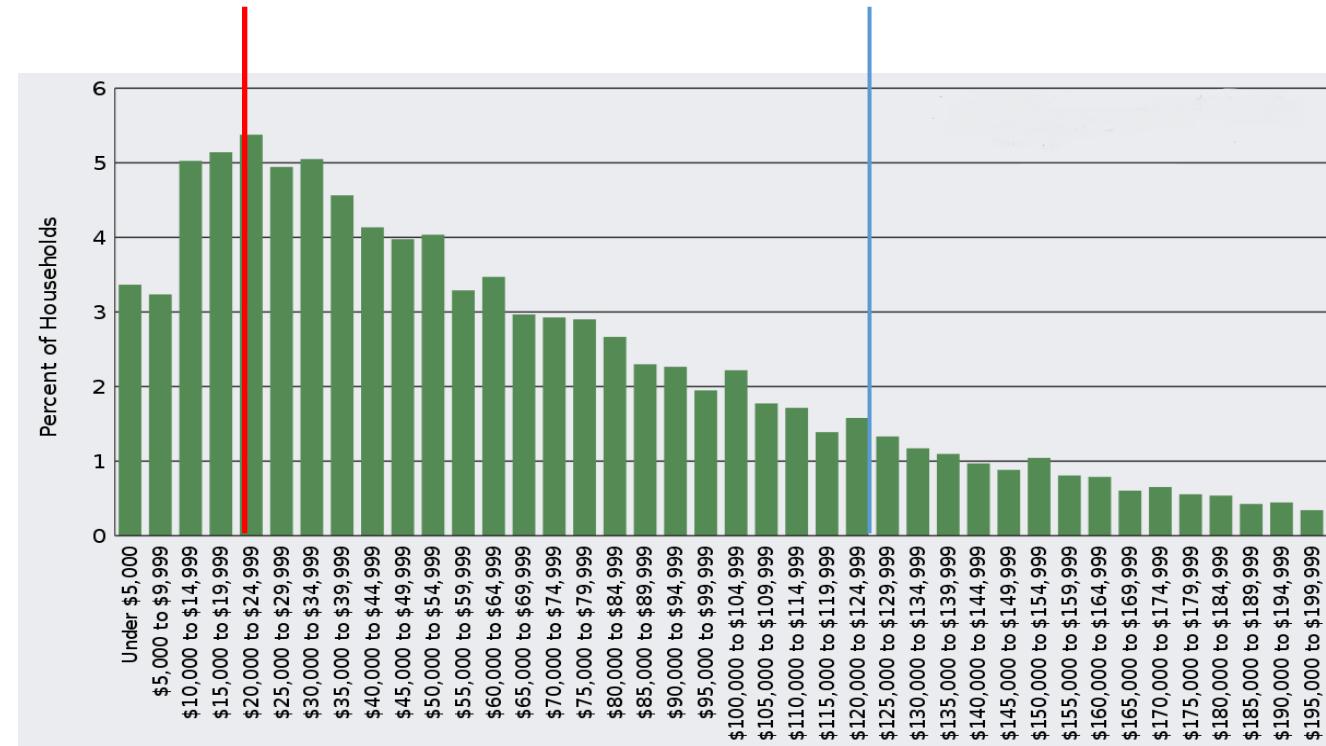
The **Pth percentile** is the value of a quantitative variable which is greater than P percent of the data

20th percentile = \$21,430

80th percentile = \$112, 254

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

np.quantile(ndarray, q)



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)

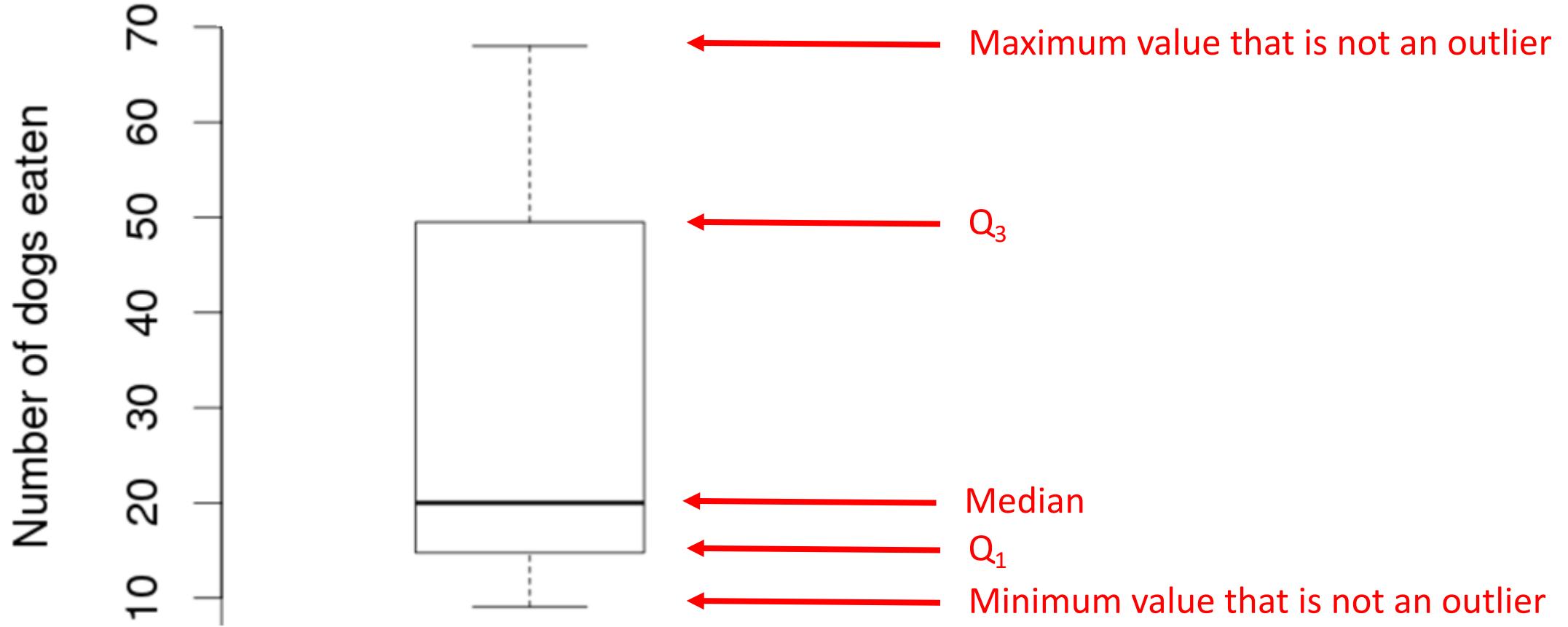
Roughly divides the data into fourths

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

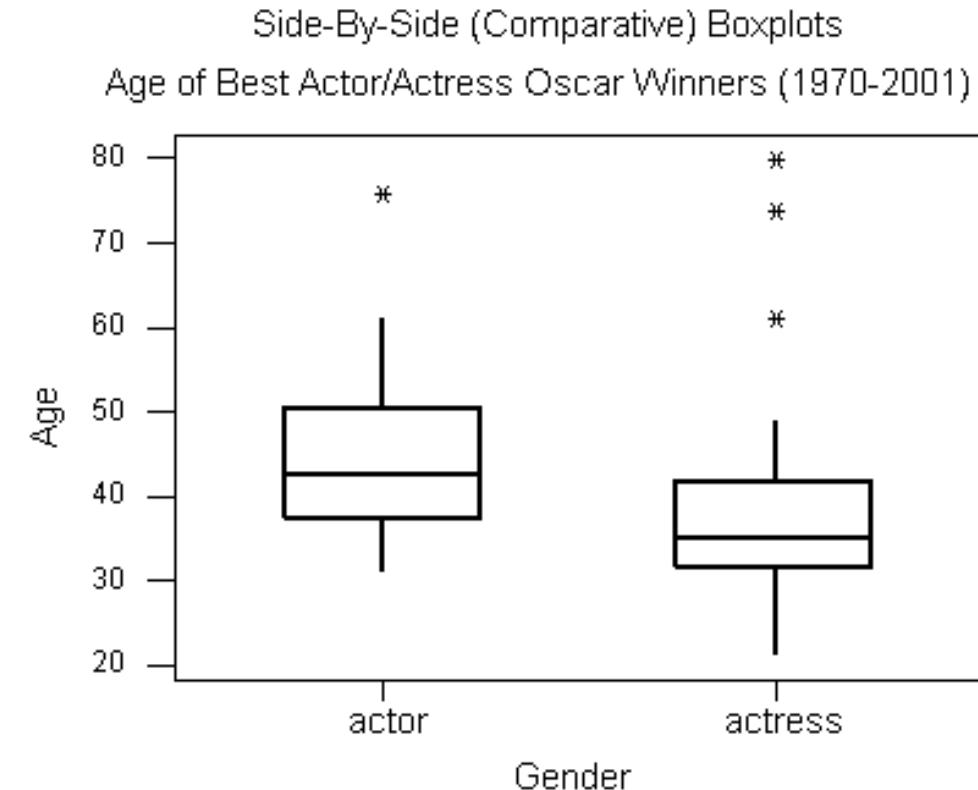
Box plot of the number of hot dogs eaten by the men's contest winners 1980 to 2010



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.



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

Let's explore this in Jupyter!

Visualizing time series

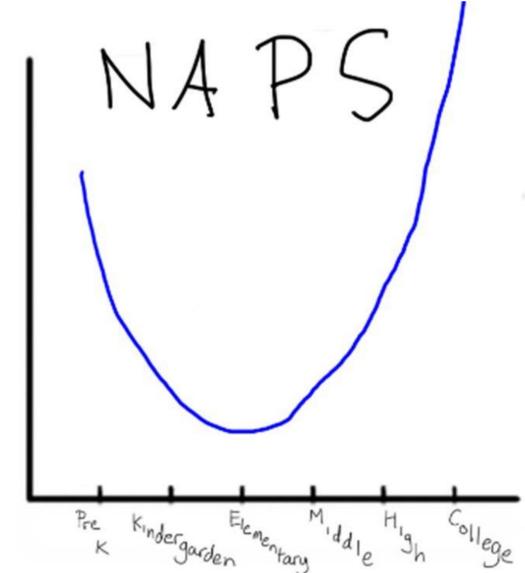
Q: How can we visualize a time series?

A: Line plot

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

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

```
plt.legend();
```



ChatGPT



You

Can you create a figure using matplotlib that shows two line plots on the same figure with a legend that plots something funny?

ChatGPT

python

 Copy code

```
import matplotlib.pyplot as plt
import numpy as np

# Hours of the day from 0 to 24
x = np.arange(0, 25, 1)

# Simulate "Hours of Sleep" vs. "Happiness Level"
# (assuming more sleep equals more happiness, with a drop-off after 8 hours)
y1 = np.where(x < 8, x, 8)

# Simulate "Number of Coffees Consumed" vs. "Energy Level"
# (let's make it funny by showing increasing energy levels with each coffee)
y2 = np.where(x < 5, x**2, -3*x + 15)

plt.figure(figsize=(10, 5))

plt.plot(x, y1, label='Happiness Level (More Sleep)', linestyle='-', marker='o')
plt.plot(x, y2, label='Energy Level (Coffee Intake)', linestyle='--', marker='x')

plt.title('Sleep and Coffee: The Keys to Happiness and Energy?')
plt.xlabel('Hours of the Day')
plt.ylabel('Arbitrary Units')
plt.xticks(np.arange(0, 25, 1))
plt.yticks(np.arange(0, max(np.max(y1), np.max(y2))+1, 5))
plt.legend()
```

ChatGPT



You

Can you actually generate the image for me to download?

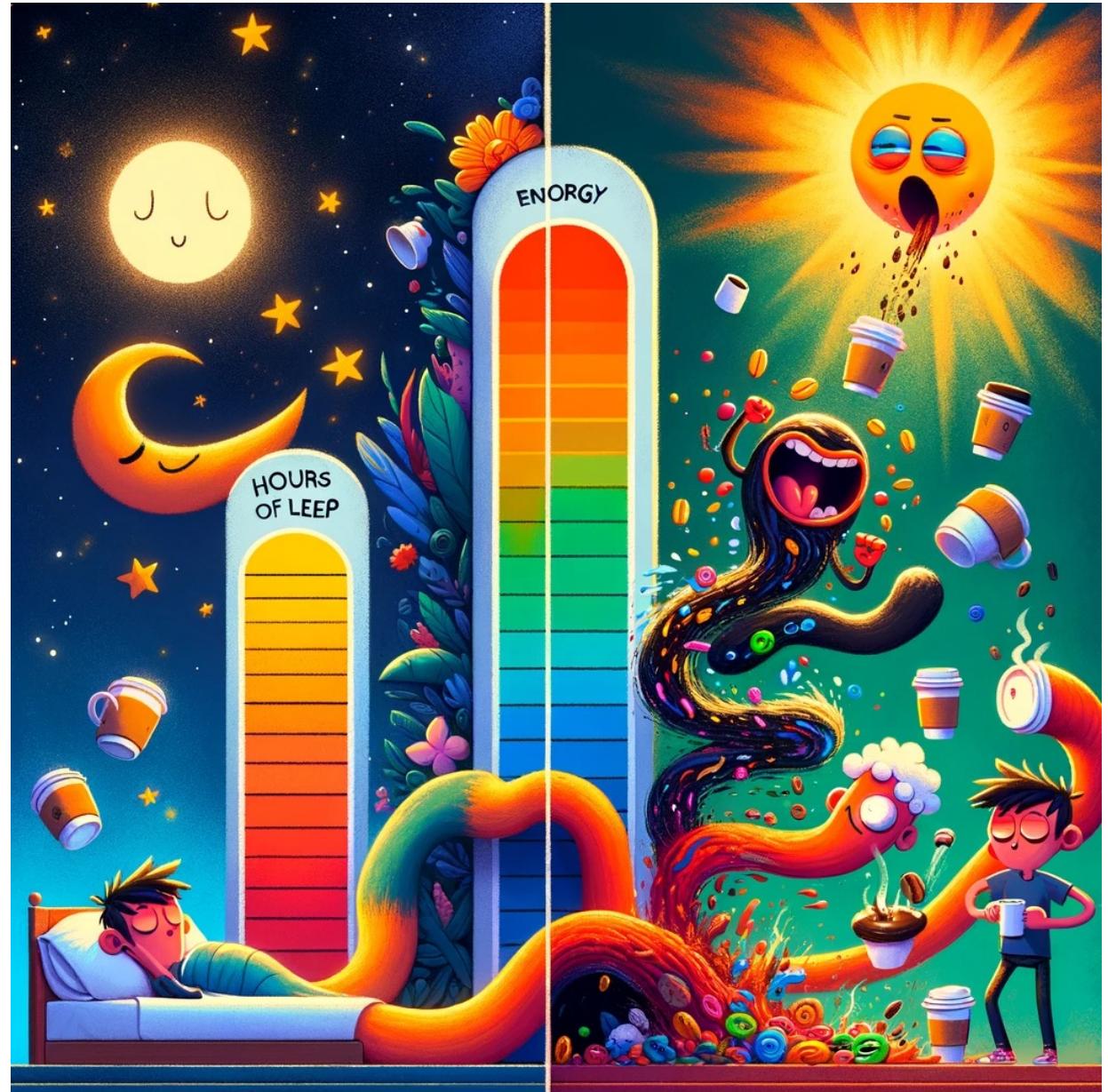
ChatGPT

Prompt for DALL-E:

Create a whimsical and abstract illustration that contrasts two scenarios side by side. On the left, depict a character sleeping soundly under a starry night, with a happiness meter above showing a high level of happiness. This meter is filled with warm colors (yellows and oranges) and labeled "Hours of Sleep." On the right, illustrate the same character throughout the day, progressively drinking cups of coffee. Start with the character looking energetic with the first few cups, shown by a rising "Energy Level" meter filled with vibrant greens. As the character continues to drink more coffee, depict a humorous twist where the character becomes increasingly jittery and exhausted, with the energy meter now plummeting and turning into chaotic reds and blues. The background transitions from morning to evening, emphasizing the time progression. Include a playful comparison in the form of a graph floating between the two scenarios, humorously correlating "Hours of Sleep" with "Happiness Level" and "Number of Coffees Consumed" with a rollercoaster-like "Energy Level." Ensure the overall tone is light-hearted, emphasizing the playful narrative of balancing sleep and coffee intake.

ChatGPT

Let's explore this in Jupyter!

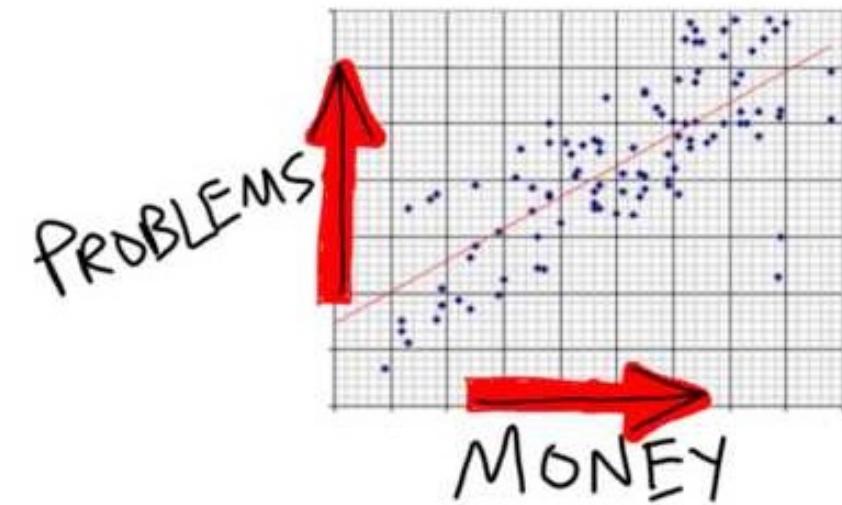


Review of visualizing two quantitative variables

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

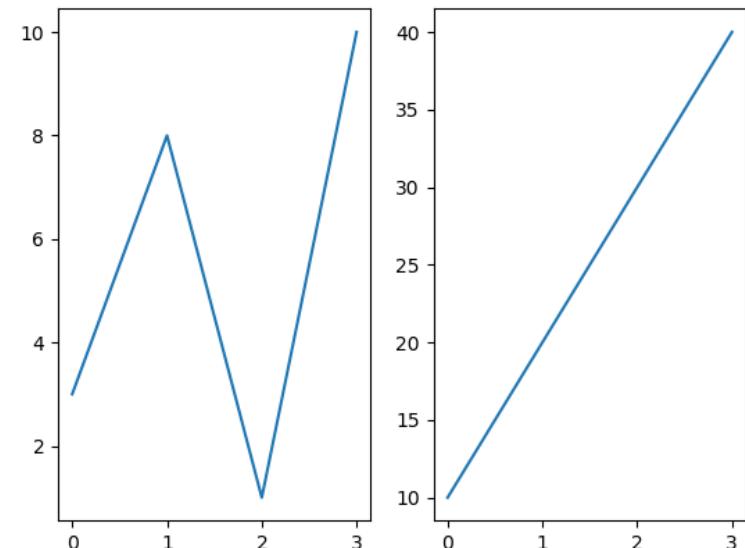
Matplotlib makes it easy to create multiple subplots within a larger figure

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

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

plot on the first subplot

plot on the second subplot

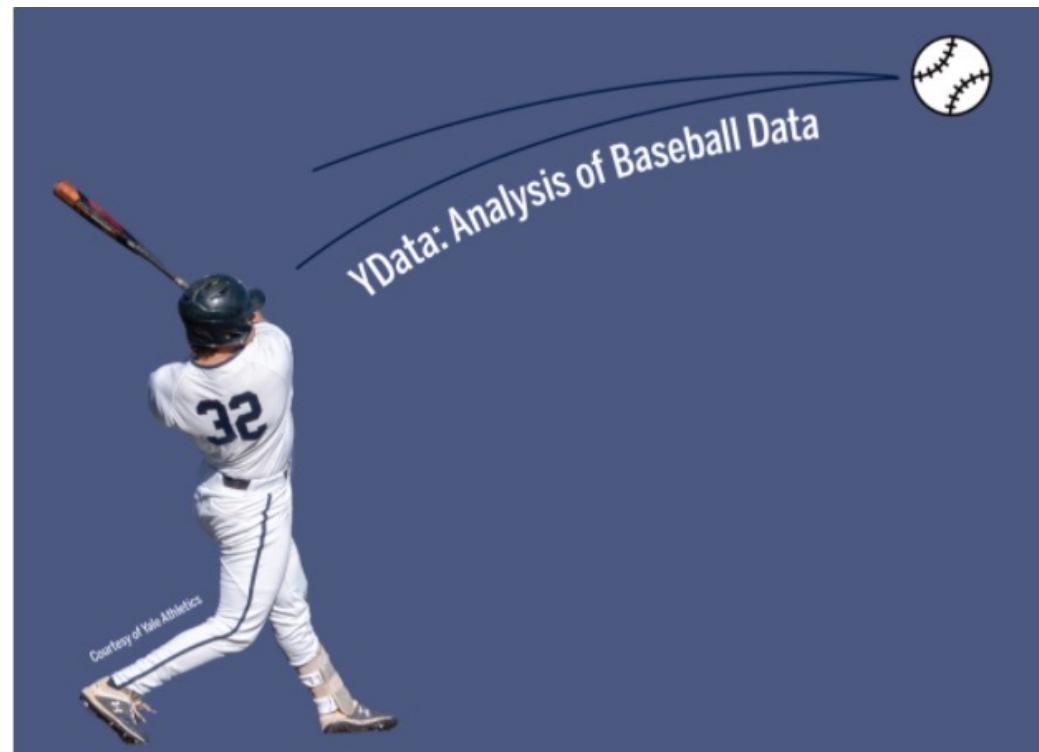


Let's explore this in Jupyter!

Using matplotlib as a canvas

We can also use matplotlib 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



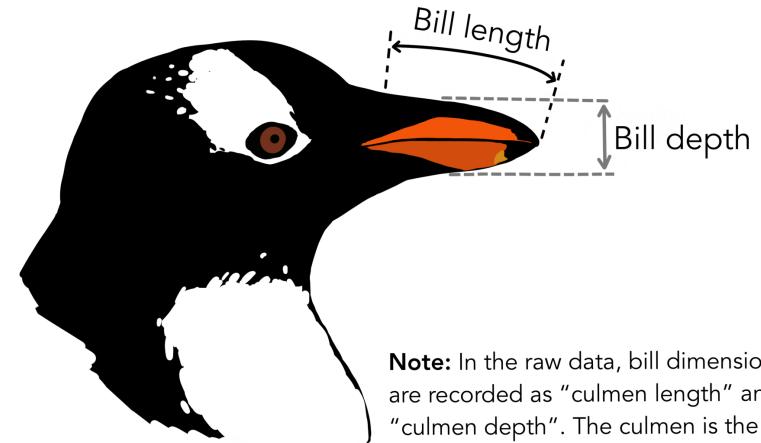
To make plots better looking we can set a theme

```
import seaborn as sns
```

```
sns.set_theme()
```

Inspiration: Palmer penguins

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



Bill length
Bill depth

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

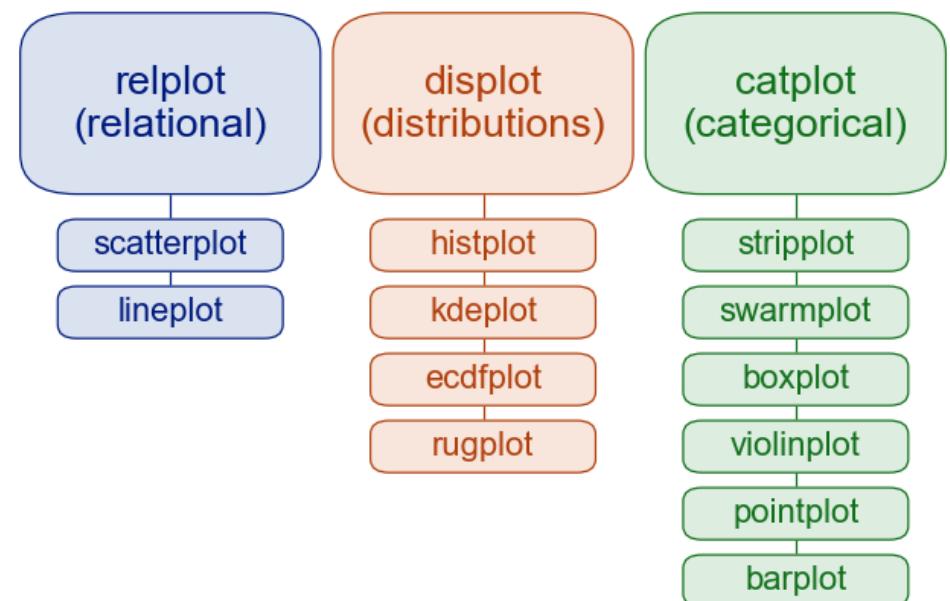
Seaborn figure level plots

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()`

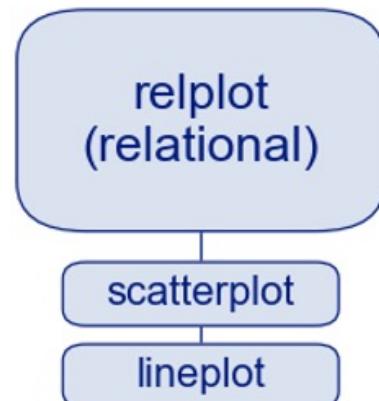
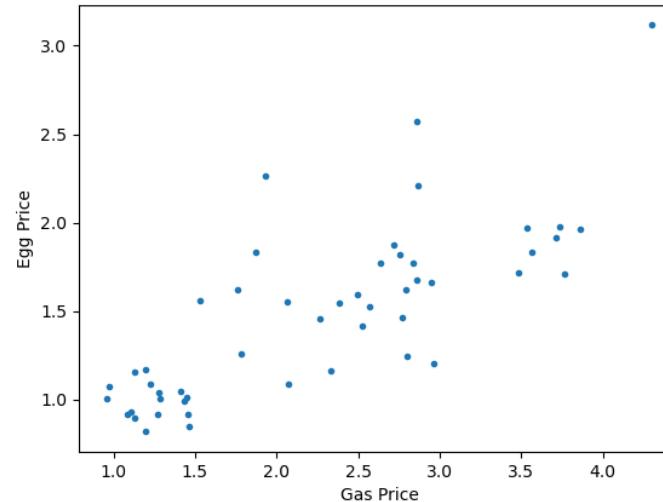
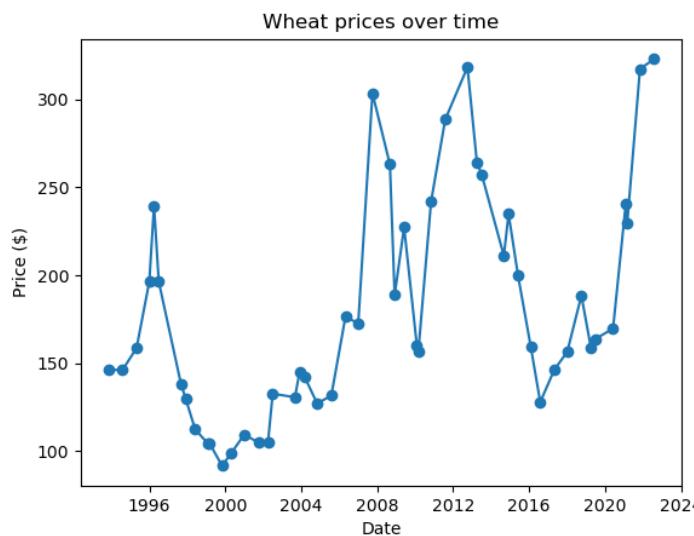
Figure level plots



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!

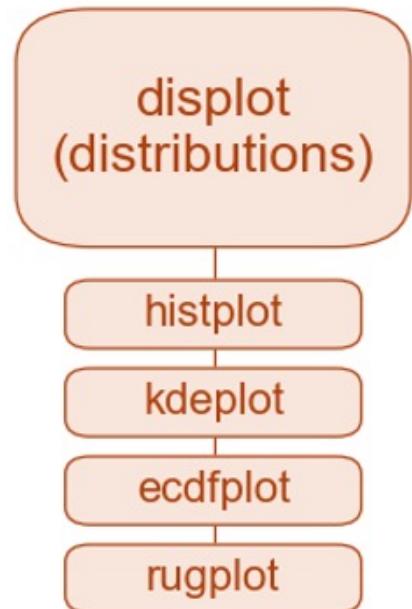
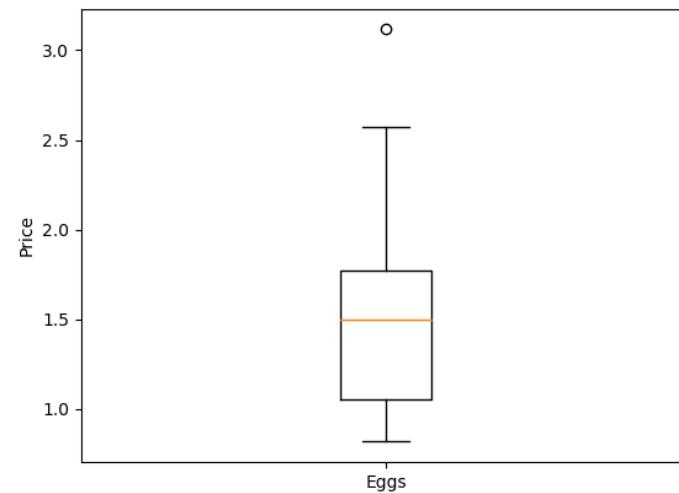
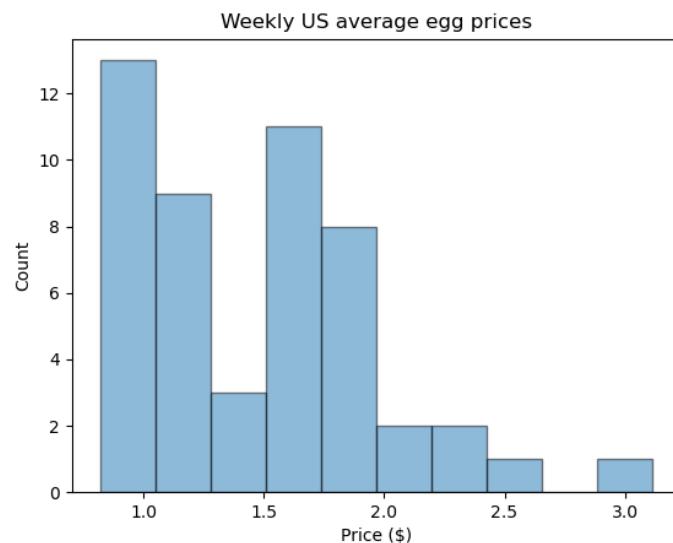


Let's explore this in Jupyter!

Plots for a single quantitative variable

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

- Histograms and boxplots

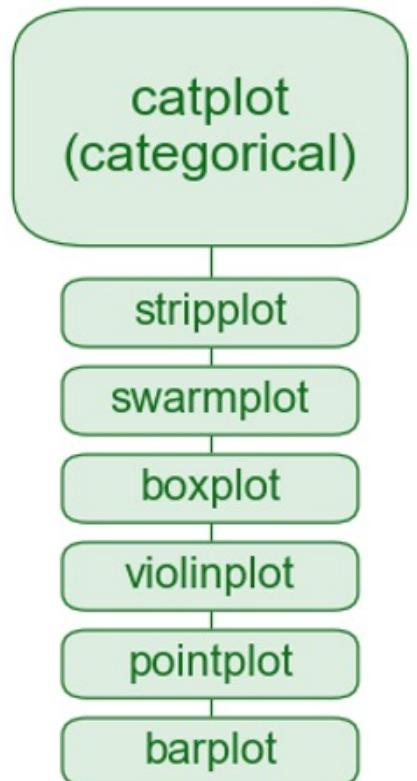
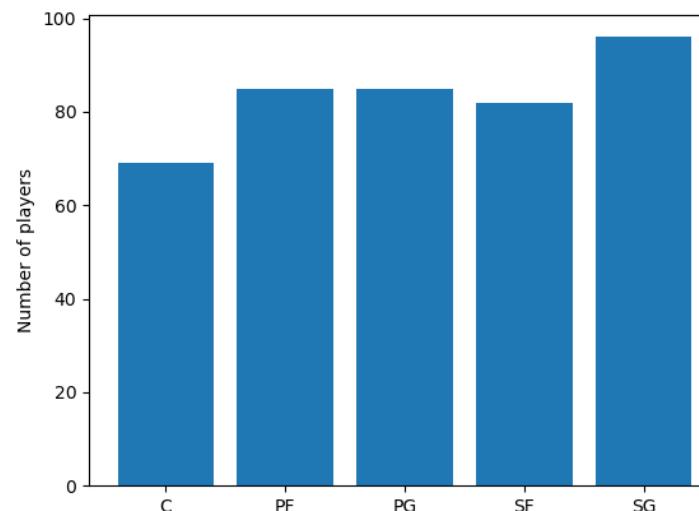
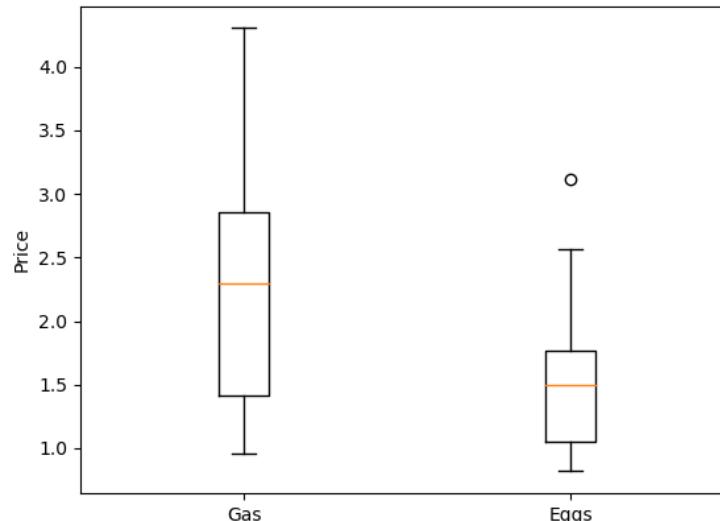


Let's explore this in Jupyter!

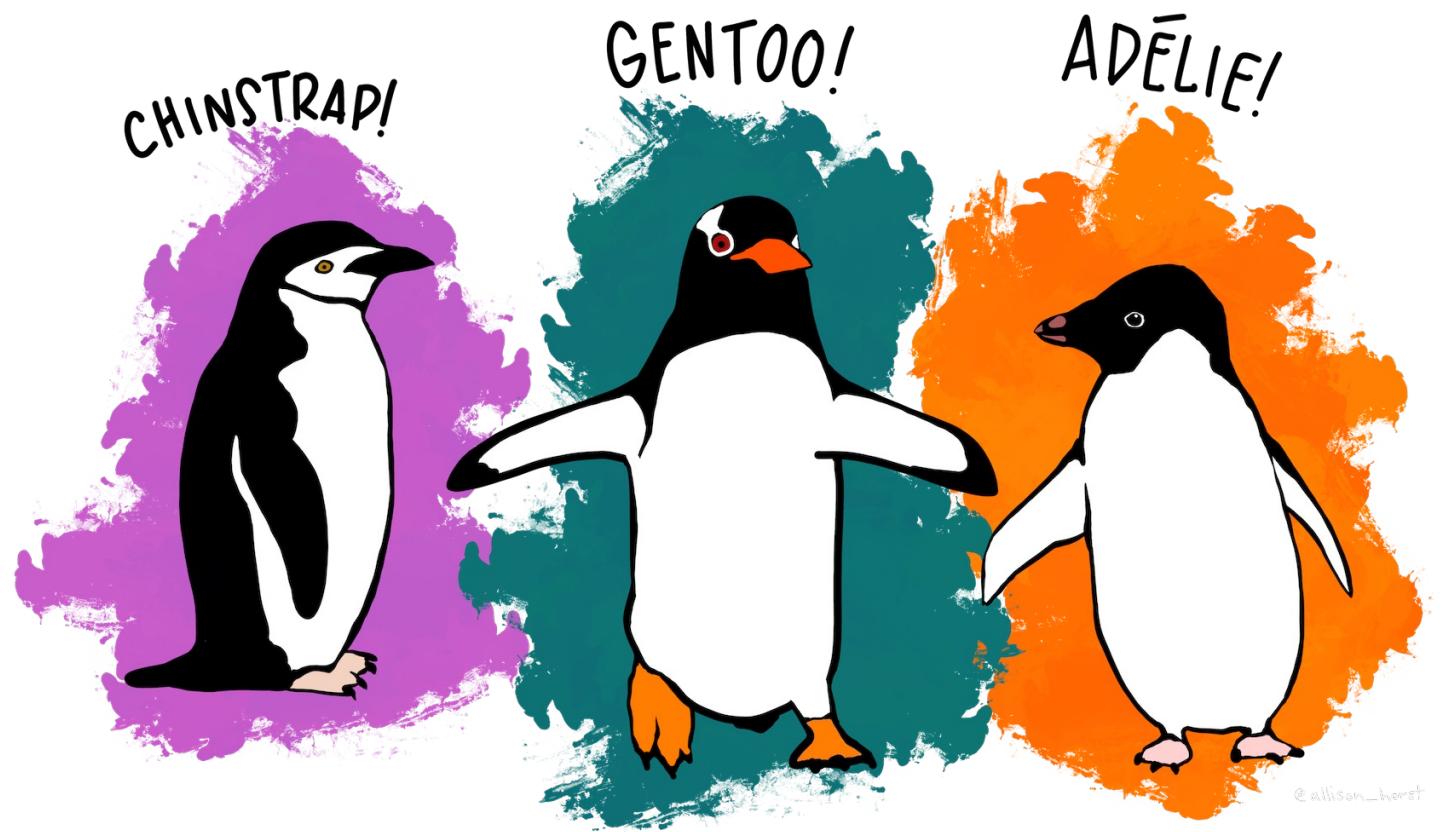
Plots for quantitative data comparing across different categorical levels

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!



@allison_horst

Up next: Interactive graphics