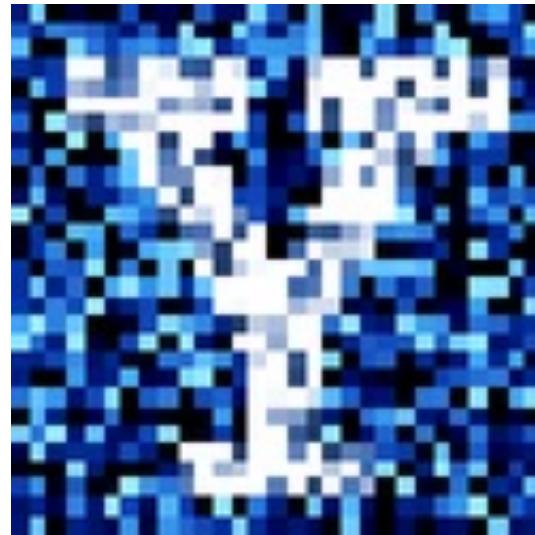


YData: Introduction to Data Science



Class 01: Introduction

Overview

Course overview

- Introductions
- Syllabus and logistics



What is data science?

- Brief history of data science

If there is time: Intro to Python

Office hours and contact information

Ethan Meyers (he/him)

Email: ethan.meyers@yale.edu

Note: I'm going to be a little distracted by a new neural network...

Office hours:

- Mondays and Wednesdays, 2-3pm
 - (subject to change)

Office: Kline Tower, room 1253

- <https://yale.zoom.us/j/96419395464>

Teaching Assistants

Preceptor

- Shivam Sharma: shivam.sharma@yale.edu



Teaching Fellows

- Alex Amari: alex.amari@yale.edu

Undergraduate Learning Assistants

- James Poe: james.poe@yale.edu
- Mark Ayiah: mark/ayiah@yale.edu
- Sloane Huey: sloane.huey@yale.edu

TA office hours are on the calendar on Canvas

Introductions

Let's do some quick introductions

Create groups of ~4 people:

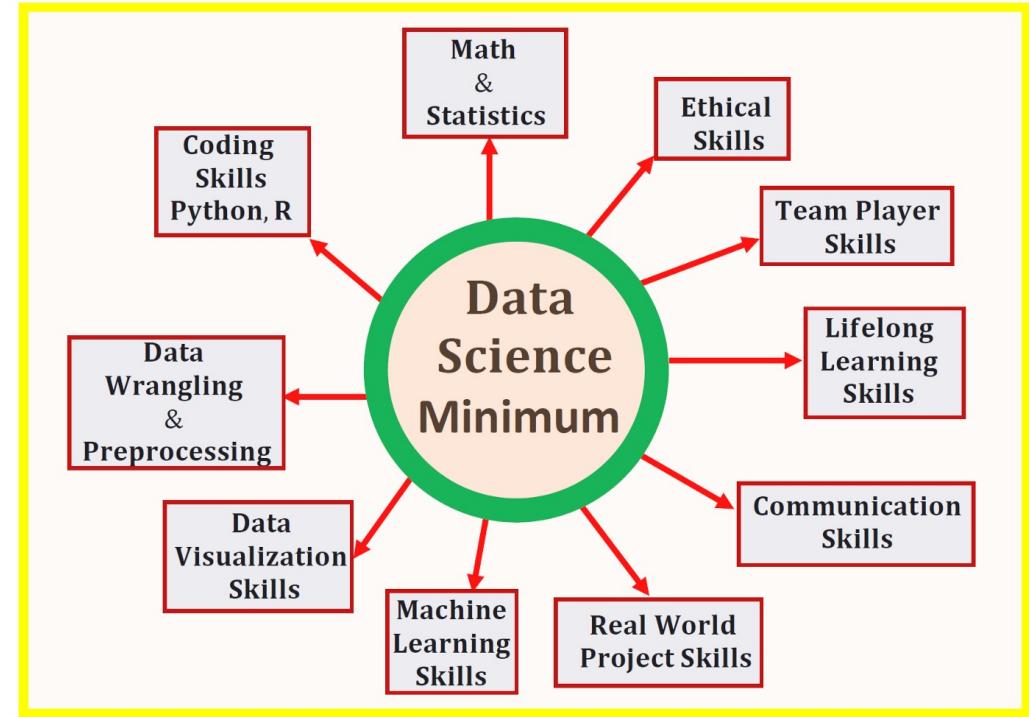
- Your name and preferred gender pronouns
- Your major/grad dept (research area)
- Why you are interested in this class
- Anything else you would like to share with your group



About this class

Topics covered

- What is Data Science?
- Python basics
- Descriptive statistics
- Array computations
- Manipulating data tables
- Data visualization
- Mapping
- Text manipulation and data cleaning
- Statistical perspective: hypothesis tests and confidence intervals
- Machine learning perspective: supervised and unsupervised learning



Tentative plan for the semester: subject to change!

Week	Date	Topic	HW Assigned	HW Due
1	Aug 29	Class overview	0	
2	Sep 3-5	Introduction to Python	1	8-Sep
3	Sep 10-12	Descriptive statistics and plots	2	15-Sep
4	Sep 17-19	Array computations	3	22-Sep
5	Sep 24-26	Tables and data manipulation	4	29-Sep
6	Oct 1-3	Data visualization	5	6-Oct
7	Oct 8-10	Review and midterm exam		
8	Oct 15	Interactive graphics		
	Oct 17	October break		
9	Oct 22-24	Mapping and text manipulation	6	27-Oct
10	Oct 29-31	For loops and writing functions	7	3-Nov
11	Nov 5-7	Statistics perspective: hypothesis tests	Draft of final project	10-Nov
12	Nov 12-14	Statistics perspective: confidence intervals	8	24-Nov
13	Nov 19-21	Machine Learning	9	1-Dec
	Nov 26-28	November recess		
	Dec 3-5	Ethics and review	Final project	8-Dec
Final exam	Dec 16 (Mon) 7pm	In person final exam		

Learning goals

1. Understand concepts in data science

- Learn basic computational skills for analyzing data
- Understand concepts in statistics and machine learning

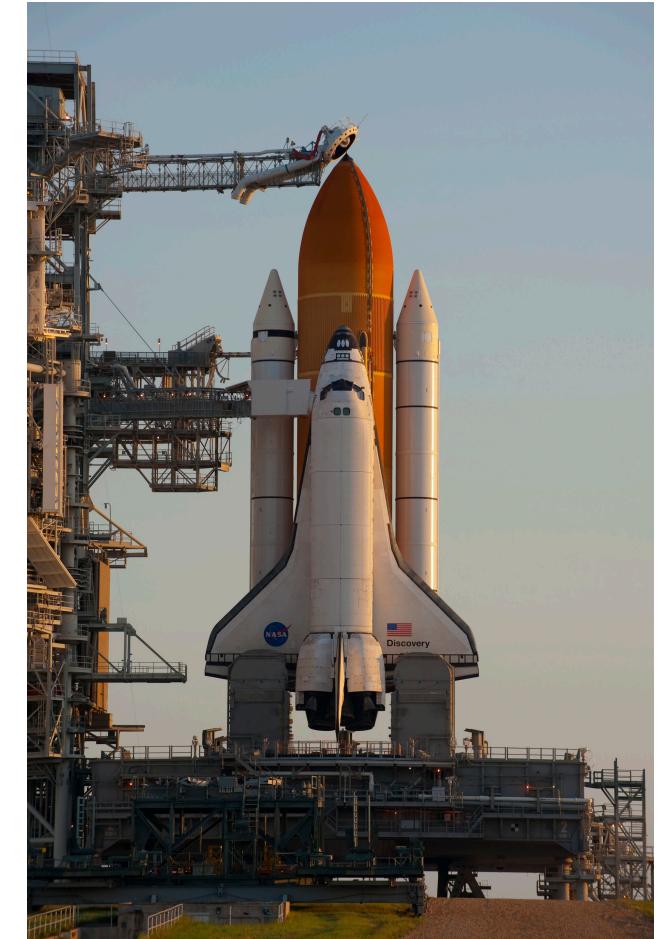
2. Gain practical data science skills applicable to any domain

3. See how data science analyses can be applied to real-world data from a variety of domains

- There will be ~weekly readings on data science related topics

There are no prerequisites for this class

- E.g., no prior knowledge of statistics or programming is required



Course structure

Two lectures per week

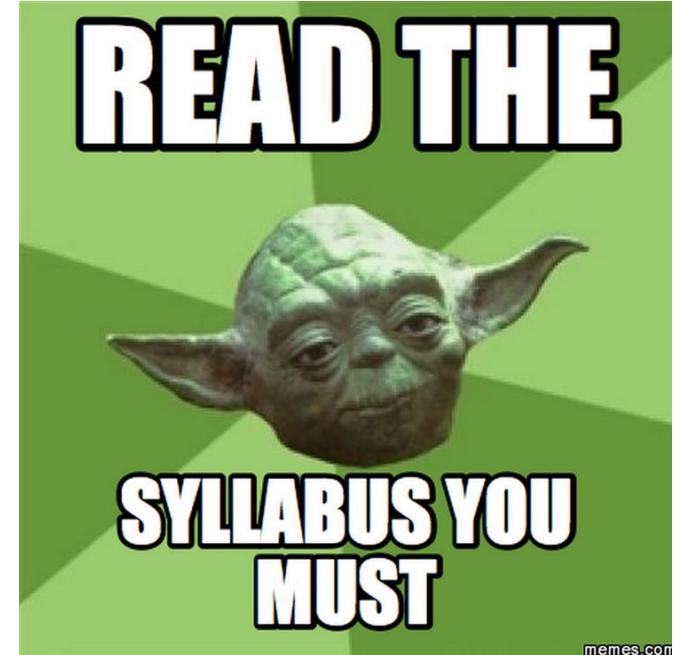
Weekly Python practice sessions (optional)

Weekly homework assignments

A class project

Weekly drop-in office hours to get help on homework (see Canvas)

Midterm and final exam



Python practice sessions



Shivam is hosting one-hour practice sessions each week

- Each session will be offered at three different times each week
- Please fill in class survey to let us know what times work for you

Sessions will be a great opportunity to practice Python and get your questions answered!

Highly recommended to attend these sessions

- Attendance is optional
- If you attend all sessions and score below a median cutoff score on the exams, up to 3 points will be added to your score

Class readings

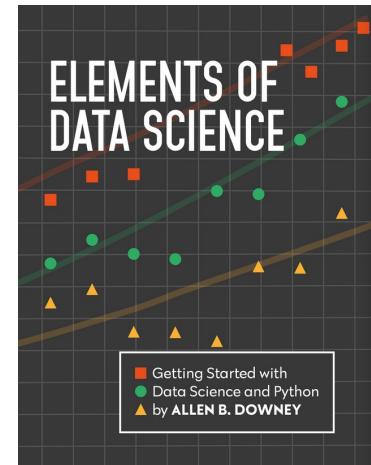
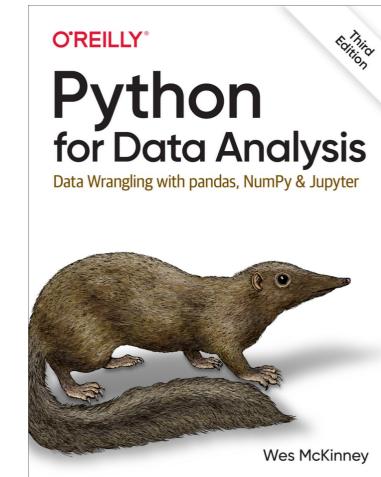
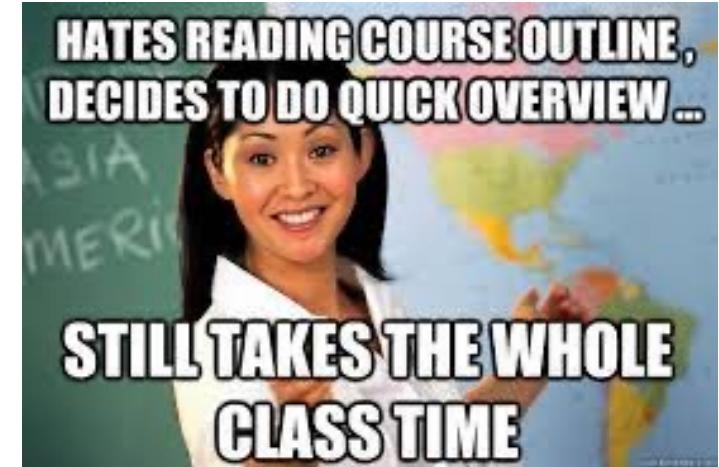
There will be short readings on data science topics approximately every other week

- These will be on Canvas

Readings will also be taken from:

- Brett M (2020). [Data Science for Everyone: course text](#)
- McKinney (2022). [Python for Data Analysis, 3E](#)
- Downey (2024). [Elements of Data Science](#)
- Other sources posted to Canvas/on the Internet

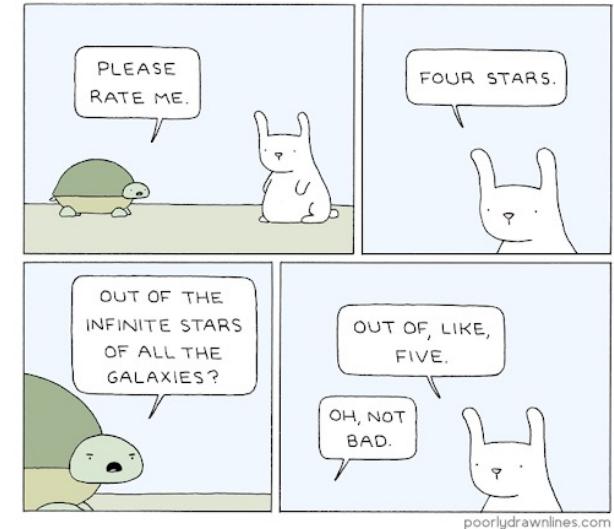
Resources related to programming will also be posted on Canvas under the appropriate class



Assignments and grades

1. Homework problem sets (48%)

- Exploring concepts and analyzing data using Python
- Weekly: 9 in total



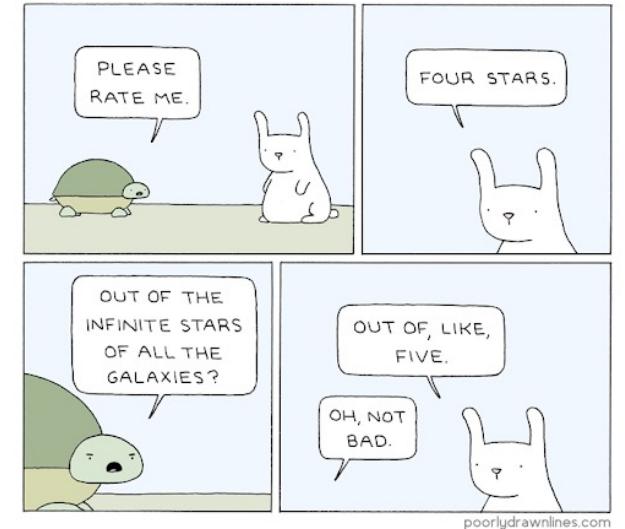
Homework policies

- You may discuss questions with other but the work you turn in must be your own!
- Worksheets assigned on Mondays and are due at 11pm on Sundays
 - (with a 59 minute grace period)
- Late worksheets (90%) credit if turned in by 11pm on Monday
 - For any other extensions a Deans Excuse is needed
- Lowest scoring worksheet will be dropped

Assignments and grades

1. Homework problem sets (48%)

- Exploring concepts and analyzing data using Python
- Weekly: 9 in total



Homework policies

- You can use chatGPT to answer general questions about Python and/or statistical concepts
 - E.g., It is ok to ask chatGPT “What does the np.sum() function in Python do?”
- You are **not** allowed to use it to directly answer homework questions
 - E.g., It is **not** ok to cut and paste a homework question into chatGPT

Assignments and grades

2. Project (10%)

- A draft of your class project is due 2/3^{rds} of the way through the semester
- You will give and receive feedback from your peers
- Final version of the project will be turned in at the end of the semester

3. Exams (40% total)

- Midterm: October 10th during the regular class time (15%)
- Final Exam: Monday December 16th at 7pm (25%)
- **To take the class you must be able to attend these exams at their scheduled times!**

4. Participation (2%)

- Active asking and answering questions on Ed Discussions

Grade distribution

Grade cut-off are

- A [94-100], A- [90-94), B+ [87-90), B [80-84), etc.
 - I might slightly modify these downward if the class too hard

No strict grade distribution but roughly:

- 25% A, 25% A-, 25% B+, 25% everything else

Students generally score high on the homework (> 90) and exam scores tend to be lower (~80)

If an exam is too hard, I sometimes curve them by adding "free points"

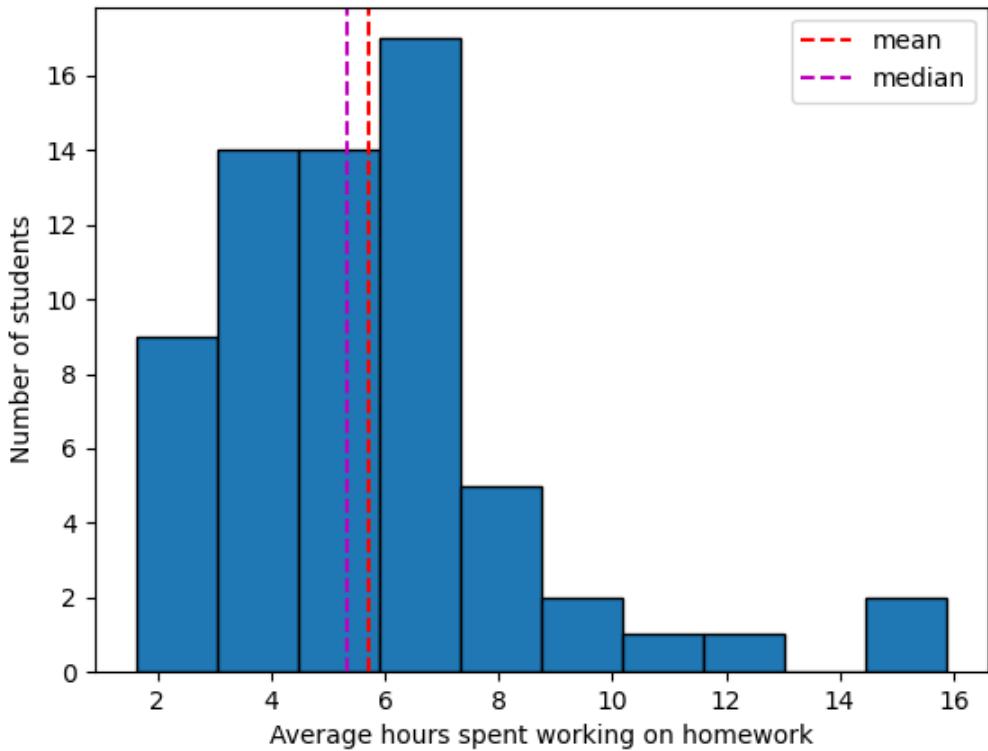
- E.g., if an exam is out of 85 points, I might add a free 15 bonus points so the exam is out of 100



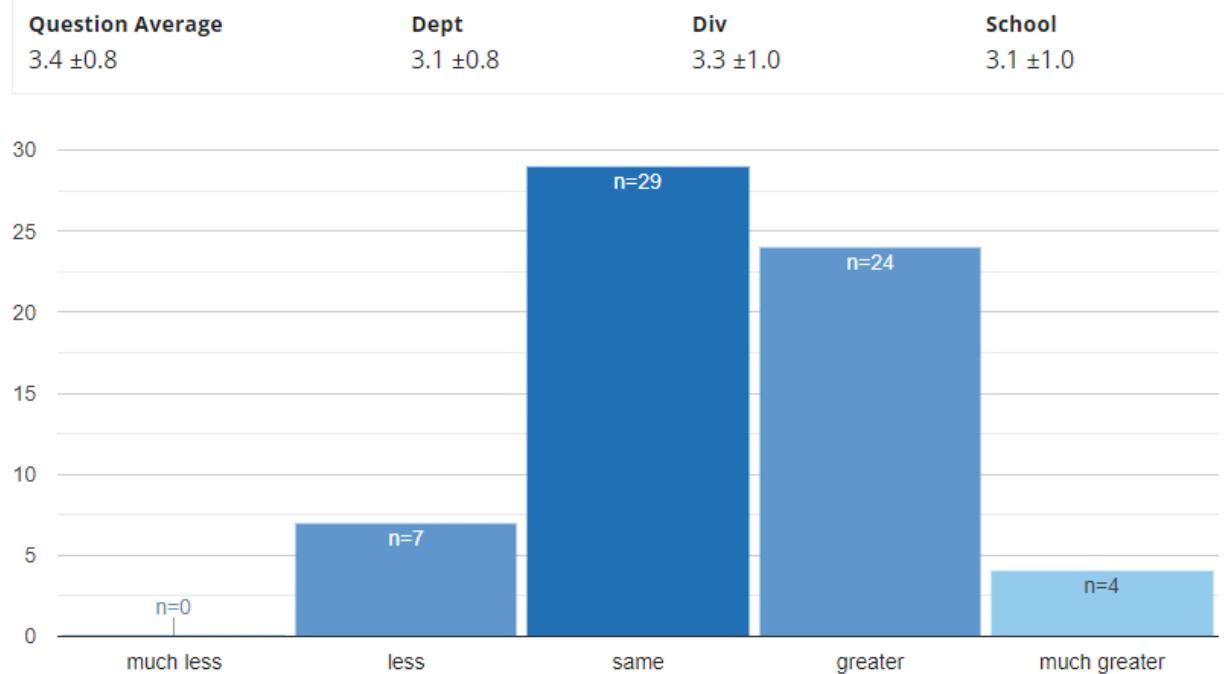
Please focus on learning rather than on grades!

Class workload

Average weekly hours spent working on homework



Relative to other courses you have taken at Yale, the workload of this course was:

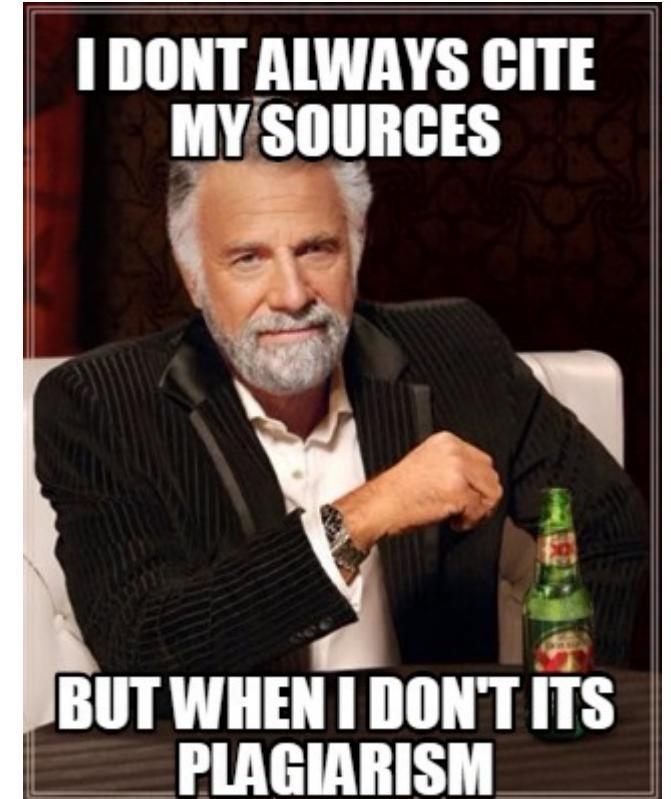


Policies

Accommodation: please let me know if you have accommodations for homework and/or exams

Academic dishonesty: Don't do it!

- You can work with others on the homework but the work you turn in needs to be your own
- Any student who turns in work for credit that is identical, or similar beyond coincidence, to that of another student may face appropriate disciplinary action at the department, college, or university level.
- If you get ideas or words from a website, journal article, book, another person, etc., cite the source in your work.
- You can't talk with others on exam, etc.



A typical homework assignment

The screenshot shows a Jupyter Notebook interface with the following details:

- Title:** hw01.ipynb
- Toolbar:** File, Edit, View, Insert, Runtime, Tools, Help, All changes saved
- Code/Text Buttons:** + Code, + Text
- Search/Filter:** 🔎
- Section:** ▾ 5. Differences between Universities
- Question:** Question 1. (2 points) Suppose you'd like to quantify how dissimilar two universities are, using three quantitative characteristics. The US Department of Education data on [UW](#) and [Cal](#) describes the following three traits (among many others):

Trait	UW	Cal
Average annual cost to attend (\$)	13,566	13,707
Graduation rate (percentage)	83	91
Socioeconomic Diversity (percentage)	25	31

- Description:** You decide to define the dissimilarity between two universities as the maximum of the absolute values of the 3 differences in their respective trait values.
- Instructions:** Using this method, compute the dissimilarity between UW and CAL. Name the result `dissimilarity`. Use a single expression (a single line of code) to compute the answer. Let Python perform all the arithmetic (like subtracting 91 from 83) rather than simplifying the expression yourself. The built-in `abs` function takes absolute values.
- Code Cell:** [] dissimilarity = ...
dissimilarity

Running Jupyter Notebooks

In order to do the homework, you will need to be able to run Jupyter Notebooks

There are a few ways to do this:

- Use the [YCRC Jupyter Server](#)
- Use [Google Colabs](#) with Google drive
- [Install Anaconda on your own computer](#)

Homework 0 allows you to test that you have a working Jupyter Notebook environment

- Homework 0 is not turned in, but please try it soon
- Ask questions on [Ed Discussions](#) or go to office hours to get help

Grace OnDemand



Let's test the YCRC Jupyter notebook server...

Before we get started, let's test the [YCRC Jupyter notebook server](#)

A link to the server is at the top of the class Canvas page

If you can't log in, let us know on the background survey...

Class survey

In order to get to know you and to adjust the class to everyone's interests, please fill out the class survey on canvas

- Under the Quizzes link on the left

Any questions about the class logistics???

- Ask on Ed Discussions!



What is Data Science?

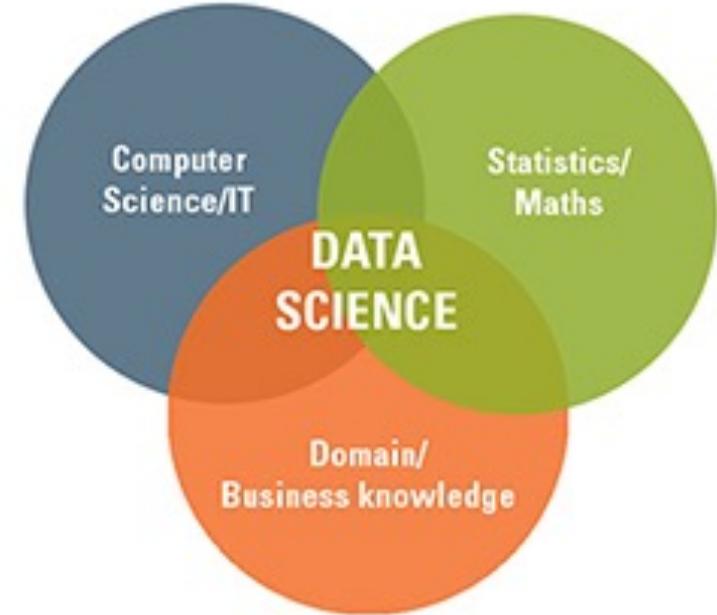
What is Data Science?

Thoughts?



Josh Wills
@josh_wills

Follow



Data Scientist (n.): Person who is better at statistics than any software engineer and better at software engineering than any statistician.

"A Data Scientist is a Statistician who lives in San Francisco"

Let's see if we can gain an understanding of what Data Science is by looking at some history...

Brief history of Data Science: data

The first data we know of:

- The **Ishango bone** is a bone tool and possible mathematical device discovered at in the Democratic Republic of Congo
- Believed to about 20,000 years old



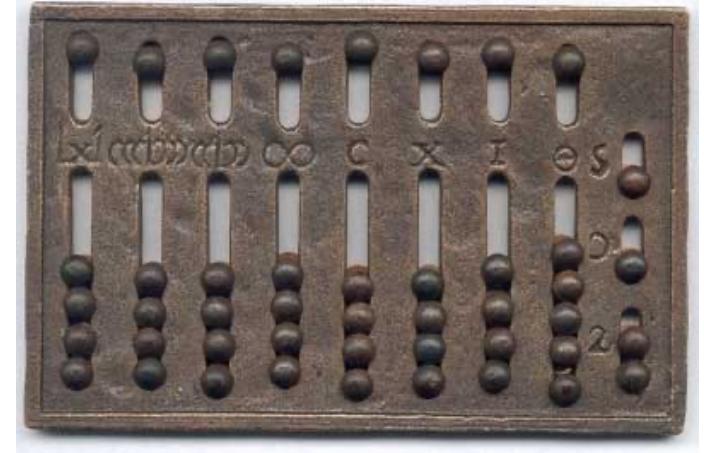
Cuneiform tablets from Uruk, a Mesopotamian settlement 5,000 years old contained transaction data on commodities



Brief history of Data Science: early computational devices

Some early computational devices include:

- The abacus comes from Babylon in 2400 BCE
- Antikythera mechanism (~100 BCE) is an ancient Greek hand-powered device described as the oldest example used to predict astronomical positions and eclipses decades in advance.



Brief history of Data Science: demography and probability

John Graunt (1620-1674) develops statistical census methods that provided a framework for modern demography. He is credited with producing the first life table, giving probabilities of survival to each age.



CAPTAIN JOHN GRAUNT

The mathematics of probability began to be developed in Europe starting in the 17th century

- Fermat and Pascal (1654), Bernoulli (1713), De Moivre (1718), Gauss and Laplace (1812)



Brief history of Data Science: visualization and math

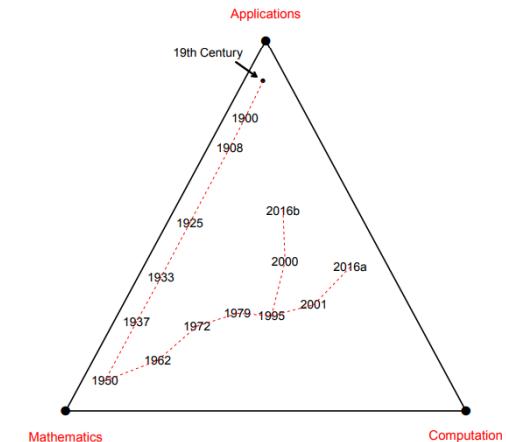
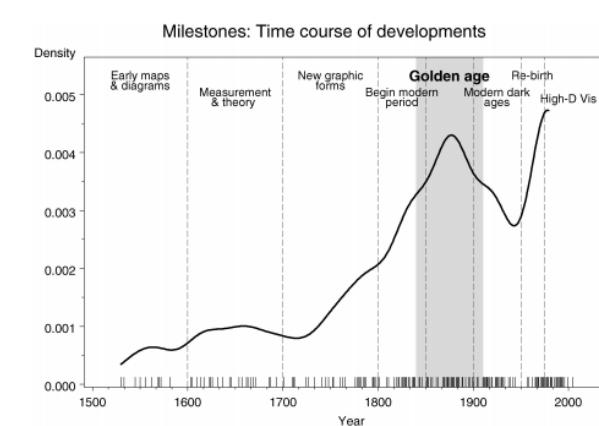
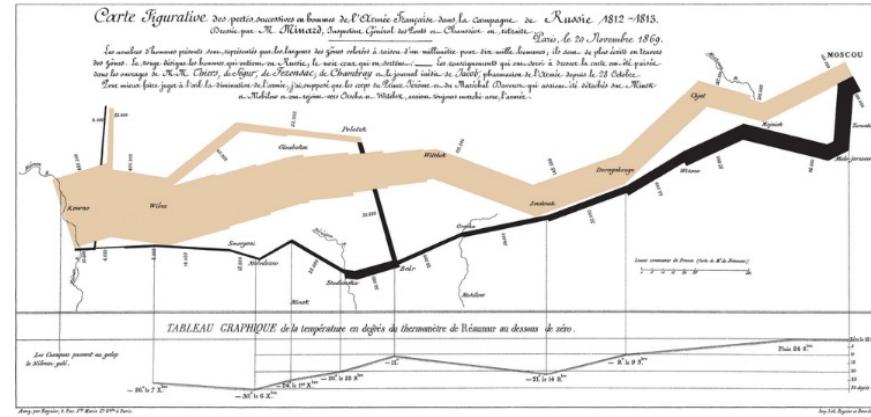
In the second half of the 19th century:

1. The field of Statistics uses probability models to analyze data
 - Galton, Pearson, Fisher, Neyman
 2. Elaborate visualizations of data were published

Probability models dominate Statistics in the first half of the 20th century

Experimental data becomes dominant in the science and medicine in the 2nd half of the 20th century

- E.g., Randomized Controlled Trials in Medicine

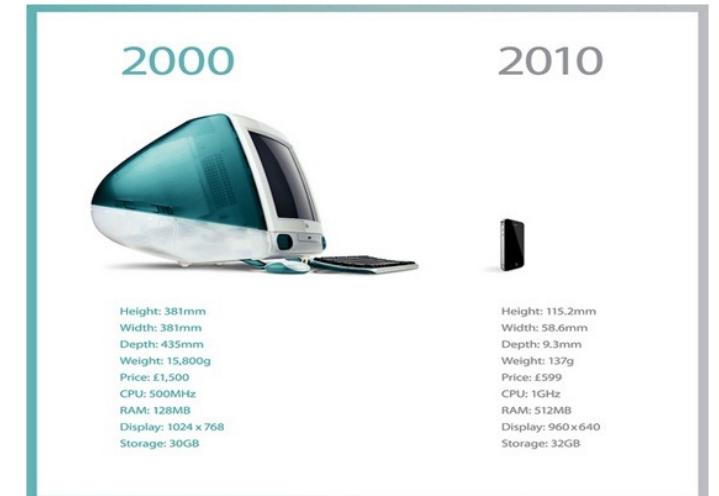


Brief history of Data Science: the rise of computers

Herman Hollerith develops the Hollerith Tabulating Machine for the 1890 census (reduces 10 years of work to 3 months). Creates IBM.

Computer technology develops rapidly over the second half of the 20th century

- Mainframe computers developed in the 1940's
- Relationship database developed in 1970
- Personal computers developed in the 1970's and 1980's
- World Wide Web developed in 1989
- iPhone developed in 2007
- Etc.



Brief history of Data Science: the rise of Data Science

The rise of powerful computers and plentiful data has given rise to new approaches to analyzing data.

- John Tukey (1962) looks for a broadening of data analysis beyond mathematics
- Breiman (2001) describes a mathematical modeling culture and algorithmic culture
- The term "Data Science" starts being used in the 2000's to describe computational approaches to analyzing data
 - E.g., Cleveland 2001

THE FUTURE OF DATA ANALYSIS¹

BY JOHN W. TUKEY

Princeton University and Bell Telephone Laboratories

Statistical Science
2001, Vol. 16, No. 3, 199–231

Statistical Modeling: The Two Cultures

Leo Breiman

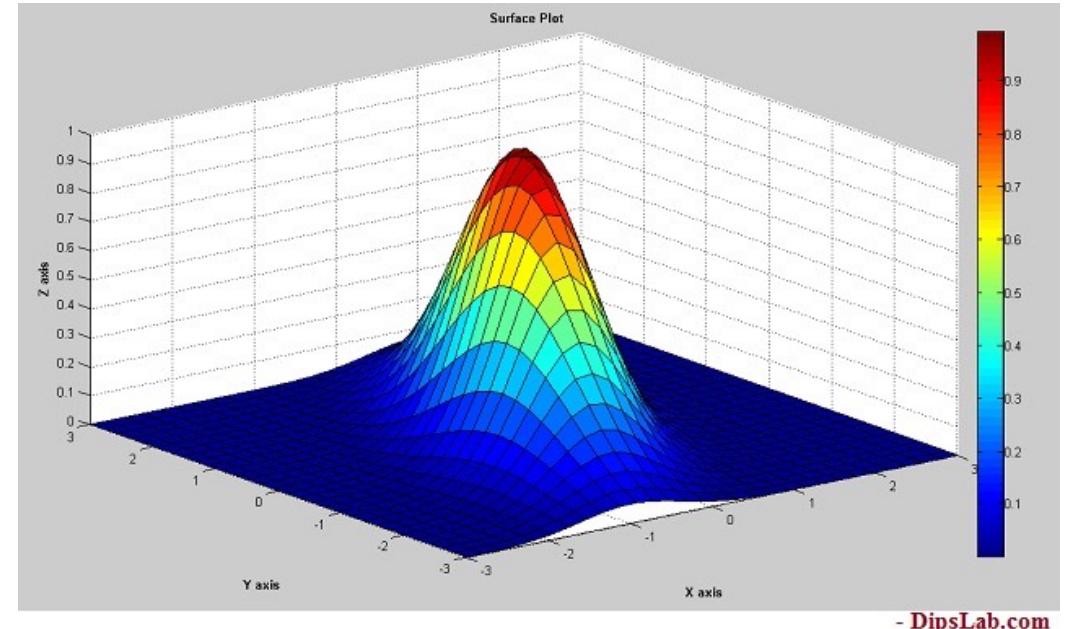
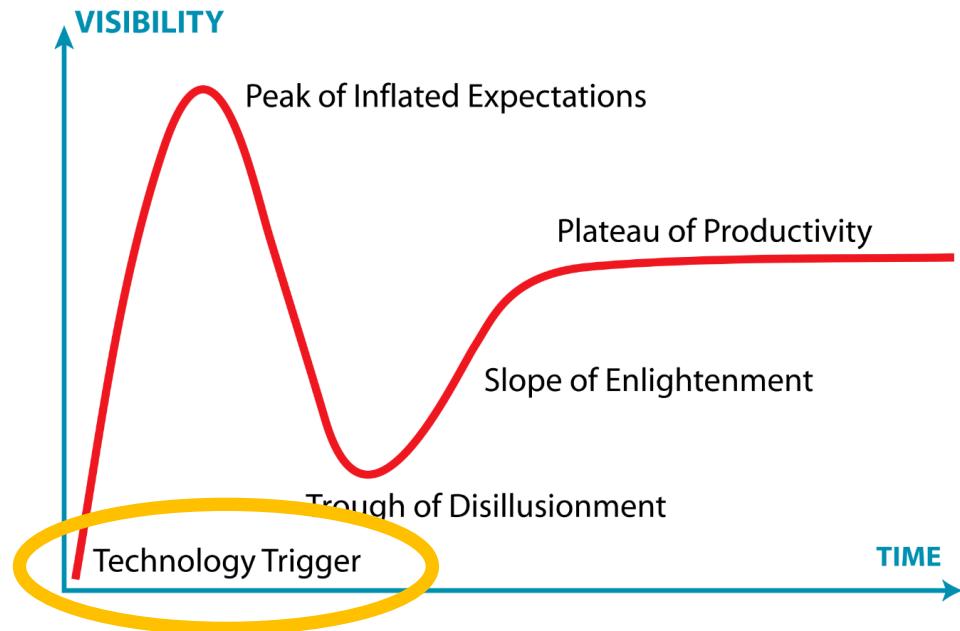
International Statistical Review (2001), 69, 1, 21–26, Printed in Mexico
© International Statistical Institute

Data Science: an Action Plan for Expanding the Technical Areas of the Field of Statistics

William S. Cleveland

Statistics Research, Bell Laboratories, 600 Mountain Avenue, Murray Hill NJ07974, USA
E-mail: wsc@research.bell-labs.com

Brief history of Data Science: Technology Trigger



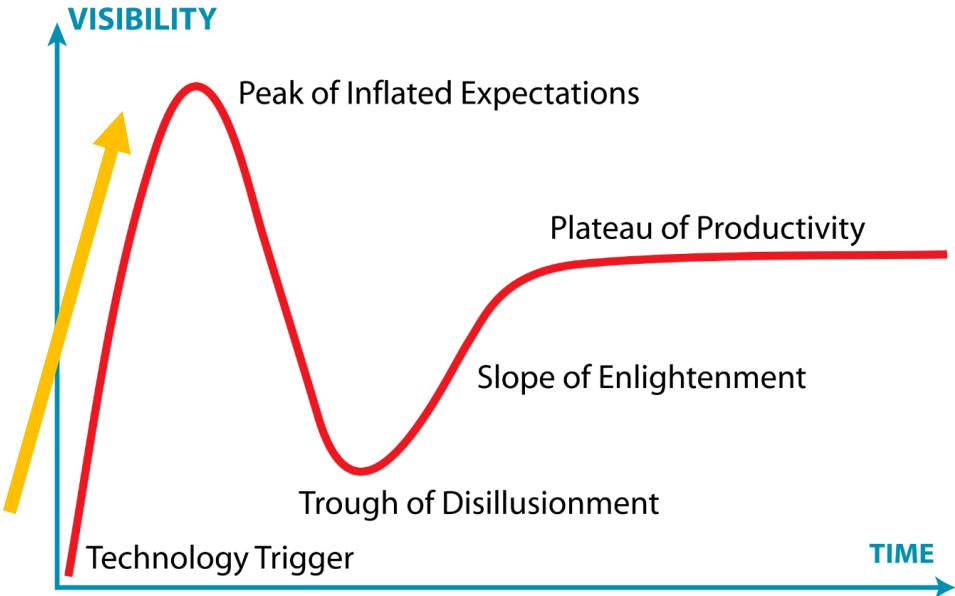
- DipsLab.com

Early 2000's rise of the internet, personal computers and data analysis programming languages changes how data can be analyzed

- MATLAB
- Python: matplotlib 2003, numpy 2005, scikit-learn 2007, pandas 2009

Brief history of Data Science

2009



Data Science rises with data science competitions, blog posts, and industry jobs

- Data Scientists viewed as “unicorns” because they had to know both statistics and how to program

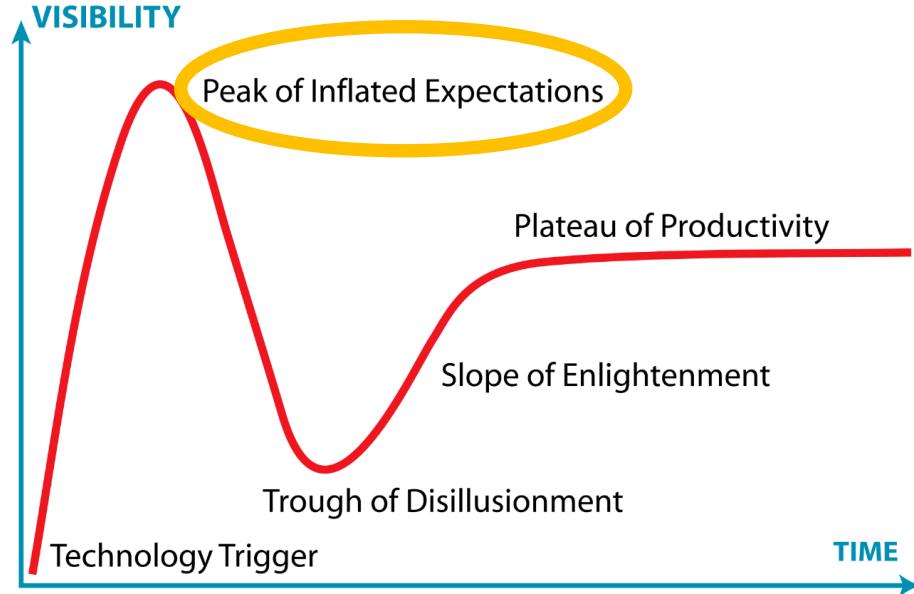
2010

kaggle

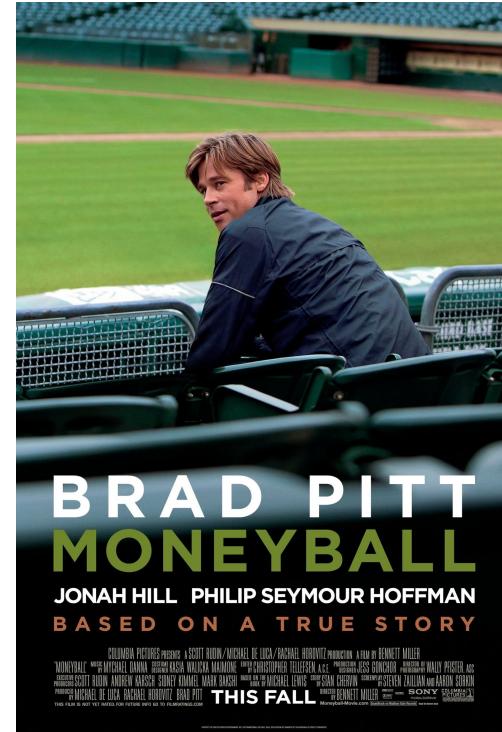
Blog: 2009-2010

okcupid

Brief history of Data Science



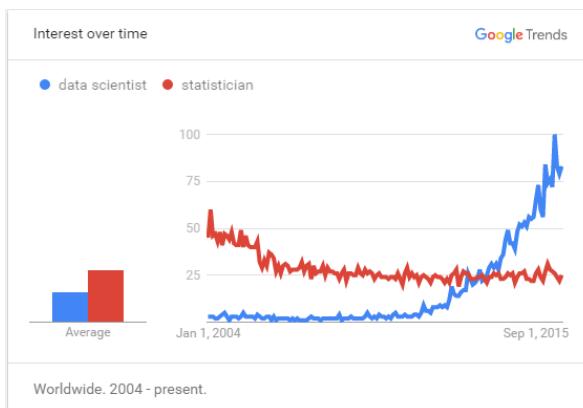
2011



2012



Data Science hits
“peak of inflated
expectations”
around 2012-14



Harvard
Business
Review

Latest Magazine Ascend Topics Podcasts Store The Big Idea Data & Visuals Case Selections

2012

Data Scientist: The Sexiest Job of the 21st Century

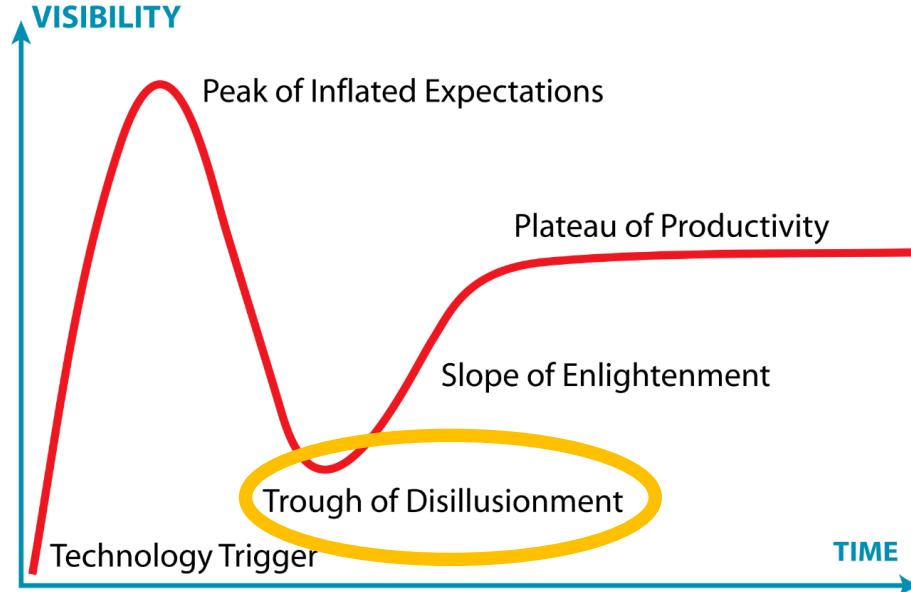
Meet the people who can coax treasure out of messy, unstructured data. by Thomas H. Davenport and DJ Patil

From the Magazine (October 2012)

Subscribe

Sign In

Brief history of Data Science: Technology Trigger



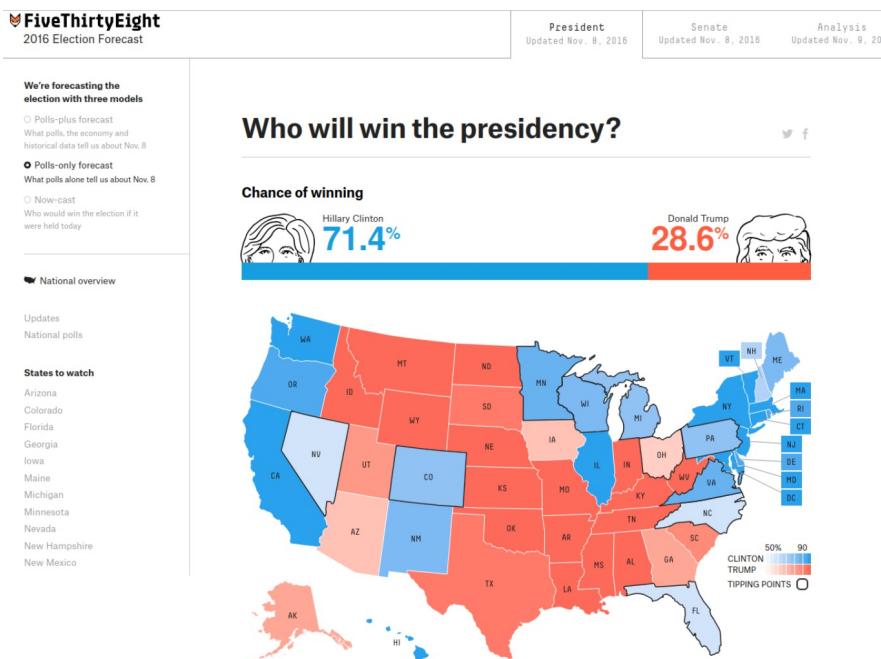
PROPUBLICA

Machine Bias

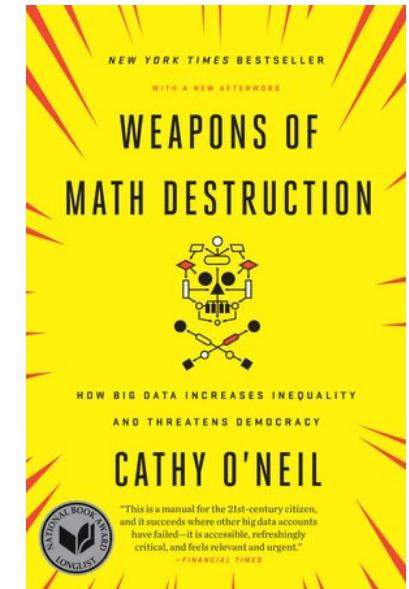
There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica

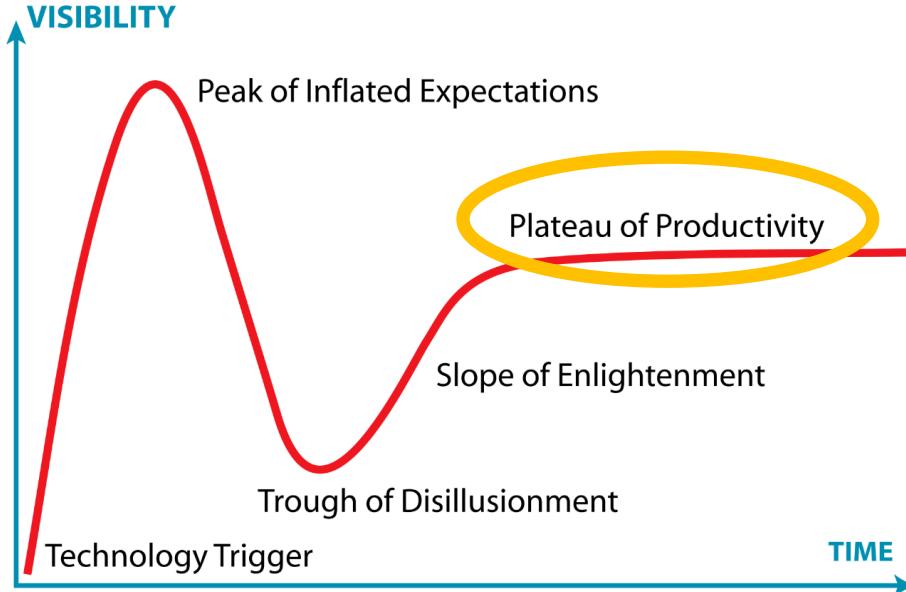
May 23, 2016



Negative consequences of predictive models were highlighted circa 2016



Brief history of Data Science: now



Data Scientists roles in many industries

- E.g., Data journalism ([NYTimes TheUpshot](#))

Many universities have Data Science programs

- In March 2017 Yale renames the Department of Statistics to be the Department of Statistics and Data Science

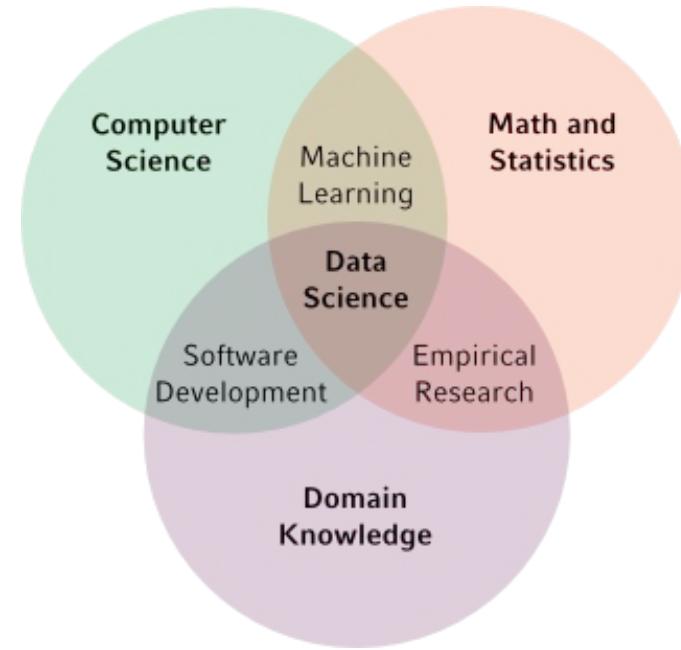
The cover features a dark blue map of the United States with various data visualization icons (charts, graphs, circles) overlaid. The title 'DATA SCIENCE' is at the top, followed by 'IS FOR EVERYONE'. Below the map, the date 'FEBRUARY 2024' and authors 'By Carlo Salerno and Frank Steemers' are listed. Logos for 'THE burningglass INSTITUTE' and 'ExcelinEd' are at the bottom.

Yale
S&DS

What is Data Science?

Data Science is a broadening of data analyses:

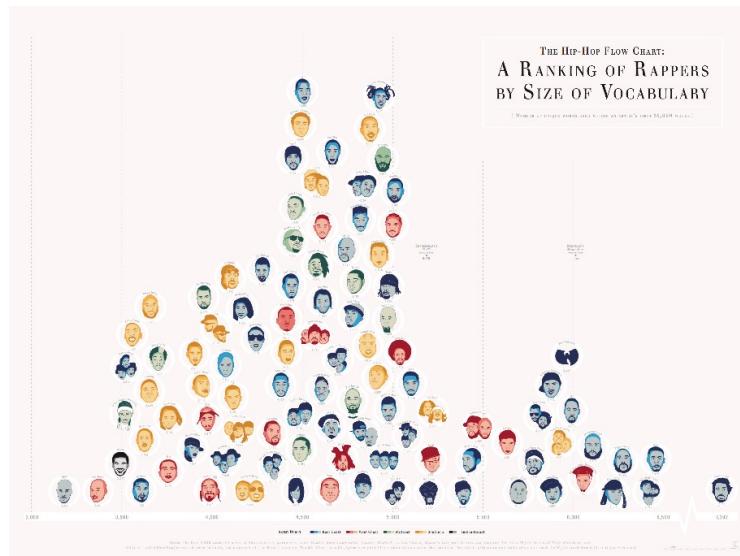
- Beyond what traditional Statistical mathematical/inferential analyses
- To using more computation



Classical statistical analysis

Descriptives								
	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
Sulfur dioxide	5	18.840	9.6919	4.3344	6.806	30.874	5.1	30.1
Nitrous dioxide	6	6.617	3.9448	1.6105	2.477	10.757	2.2	11.9
Oxygen	4	4.975	3.4092	1.7046	-.450	10.400	2.1	9.3
Total	15	10.253	8.6514	2.2338	5.462	15.044	2.1	30.1

ANOVA					
Bronchial reactivity	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	559.450	2	279.725	6.873	.010
Within Groups	488.408	12	40.701		
Total	1047.857	14			



Examples:

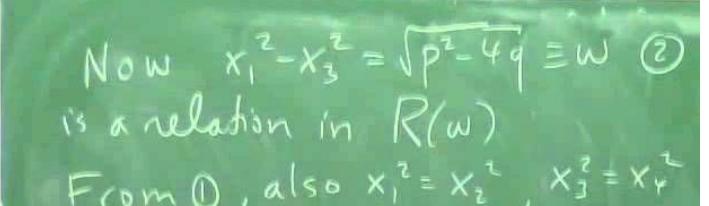
- [NYC city bikes](#)
- [Wind map visualization](#)

New ways to choose the best methods

Statistics focuses on mathematical models (probability distributions) to analyze data

- Best methods are the ones that have mathematical guarantees (proofs)

The proof is in the math



Now $x_1^2 - x_3^2 = \sqrt{p^2 - 4q} \equiv w \quad ②$
is a relation in $R(w)$
From ①, also $x_1^2 = x_2^2, x_3^2 = x_4^2$

Data Science empirically evaluates data analysis methods

- Best methods are the one that gives the most insight in practice

The proof is in the pudding



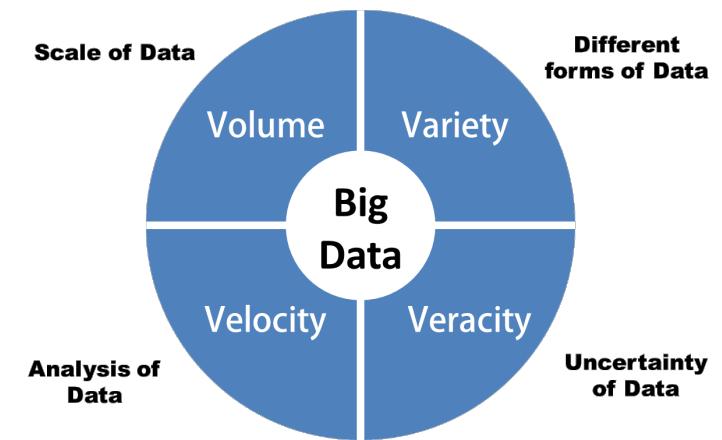
Big Data

New insights:

- Lots of new data from Internet, sensors etc., can be mined to transform our understanding in a range of fields
 - E.g., health, cosmology, social sciences, etc.

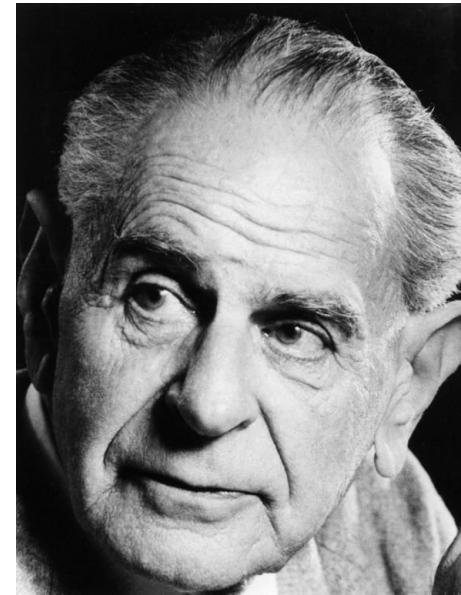
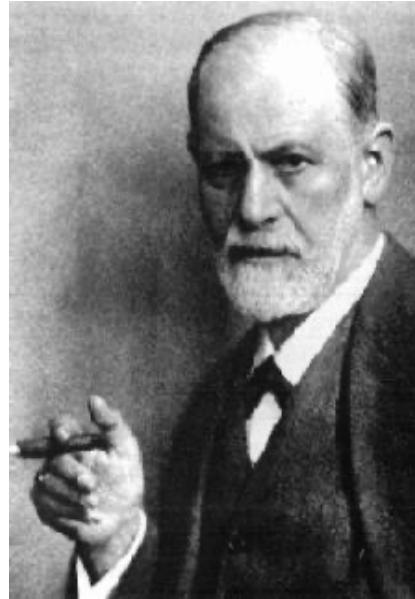
New analysis and approaches:

- Hypothesis test pick up on very small (meaningless) effects with very large samples
- Data manipulation and programming are needed to extract insights
- Also, new standards for choosing the best data analysis methods



[Data Science vs. Statistician video](#)

Short paper to read from the book Everybody lies



Much of Freud's theory dealt with the subconscious

- E.g., Freudian slips

Karl Popper claimed that Freud's theories were unscientific because they couldn't be falsified

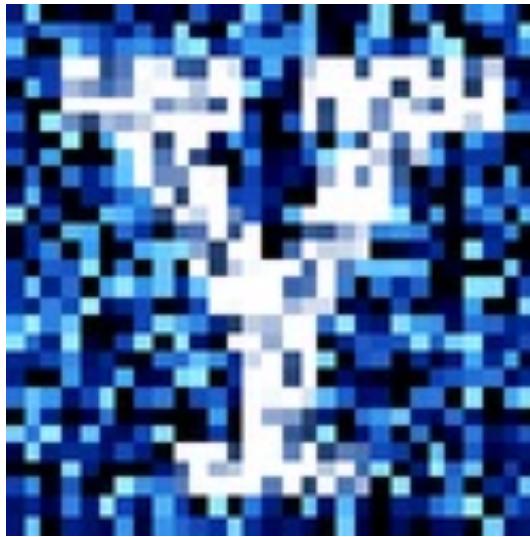
- i.e., can come up with any 'just so' story to explain a behavior

New data science analyses might make it possible to actually test Freud's theories

Things to do for next class...

1. Complete class survey
2. Do short reading on Canvas from the book "Everybody lies"
3. Make sure you can log into the YCRC Jupyter Server

YData: Introduction to Data Science



Class 02: Introduction to Python

Overview

Very quick continuation of the history of data science

Quick Discussion of the reading from “Everybody Lies”

Intro to Python

- Expressions
- Names
- Call expressions (functions)
- Data types
 - Numbers and strings
- If there is time
 - Lists



Let's test the YCRC Jupyter notebook server...

Before we get started, please log into the [YCRC Jupyter notebook server](#)

A link to the server is at the top of the class Canvas page

Can everyone log in?

Announcements

Please fill out two surveys

- 1. Background survey
- 2. Practice session and study group survey

If you have not done so already, fill these out by 11pm tonight!

Practice sessions for this week are:

- Thursday 5-6, and 6-7
- Friday 3-4, and 4-5
- Subject to change in future week (fill out survey #2)



Announcement: Homework 1

Homework 1 has been posted!

```
import YData
```

```
YData.download.download_homework(1)
```

It is due on Gradescope on **Sunday September 8th** at 11pm

- **Be sure to mark each question on Gradescope!**

The history of Data Science

(a very incomplete list)

Data



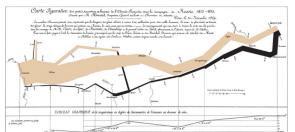
Ishango bone (20,000 BCE)



Cuneiform tablets (4,000 BCE)



Quipus in South America (1100-1500)



Probability

Key Take Away

Probability models
dominated data analysis
prior to using
computational methods

Computers



Abacus (2400 BCE)



Antikythera mechanism (100 BCE)



Analytical Engine (1800's)



Hollerith Tabulating Machine (1890)

Demographics (1600's)

Golden age of data visualization

(1850-1900)

Big data (now)

Initial development (1600's)

Probability in Statistics

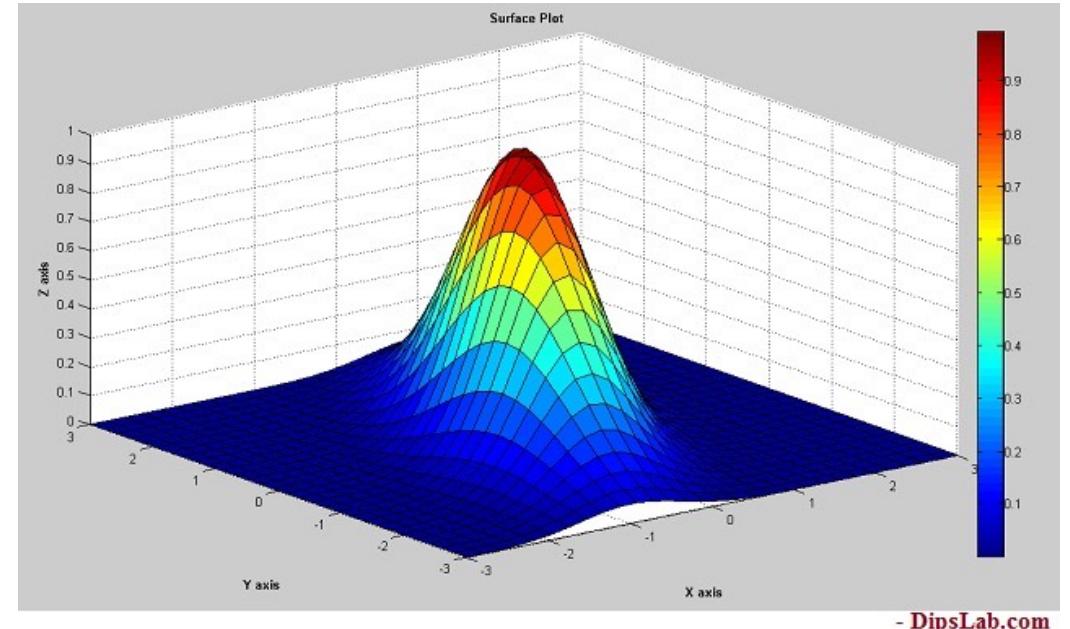
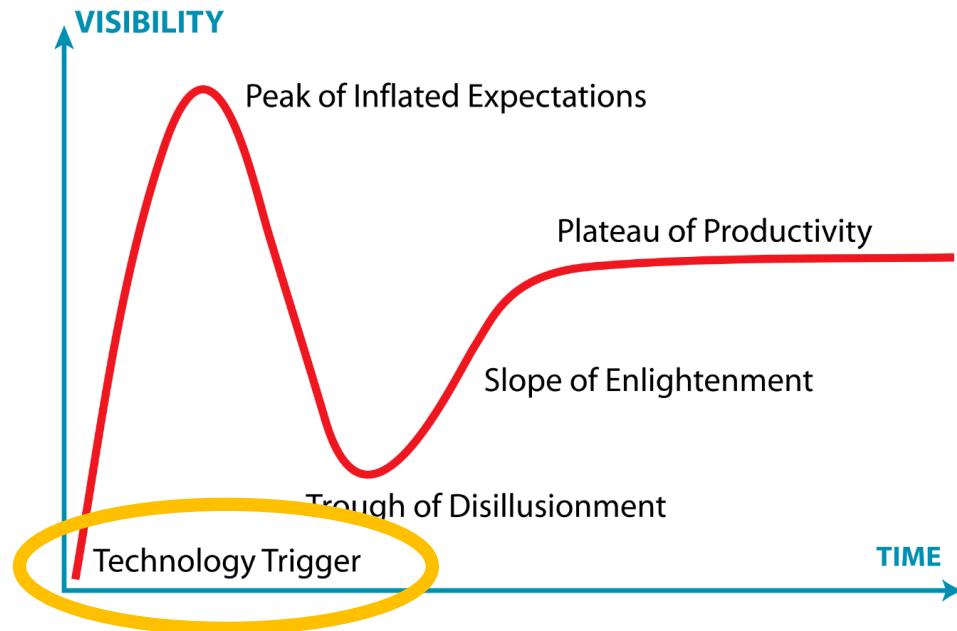
(1820's – 1950's)

Math Stats dominates (1900-1960's)

Mainframes, PCs, Internet,
etc.
(1950-present)

"Big data"

Brief history of Data Science: Technology Trigger

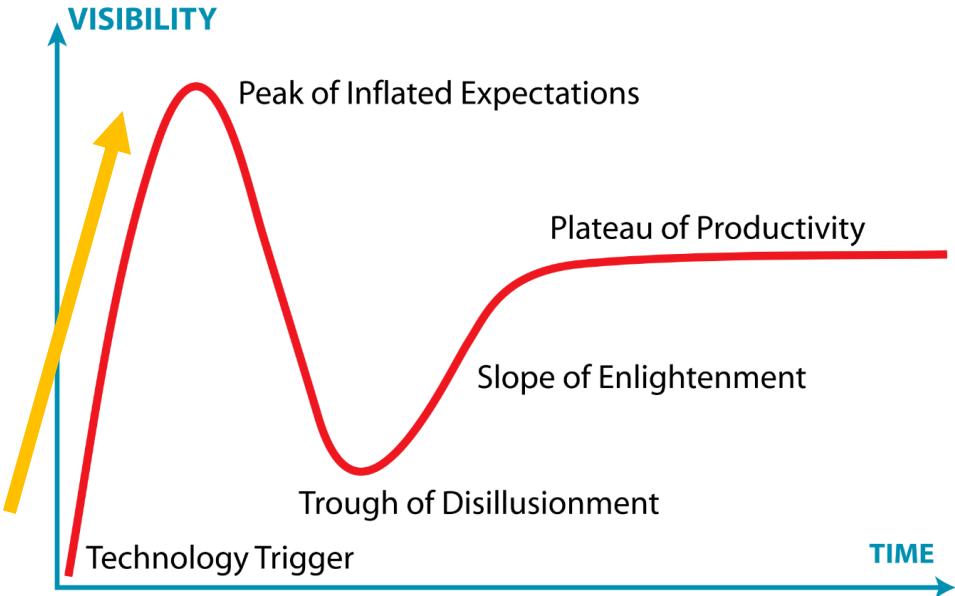


Early 2000's rise of the internet, personal computers and data analysis programming languages changes how data can be analyzed

- MATLAB
- Python: matplotlib 2003, numpy 2005, scikit-learn 2007, pandas 2009

Brief history of Data Science

2009



Data Science rises with data science competitions, blog posts, and industry jobs

- Data Scientists viewed as “unicorns” because they had to know both statistics and how to program

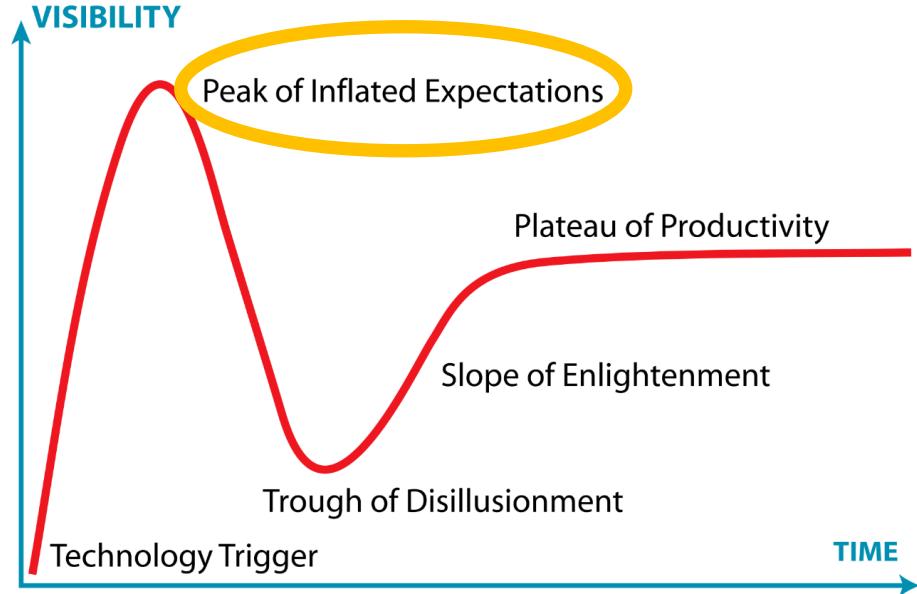
2010

kaggle

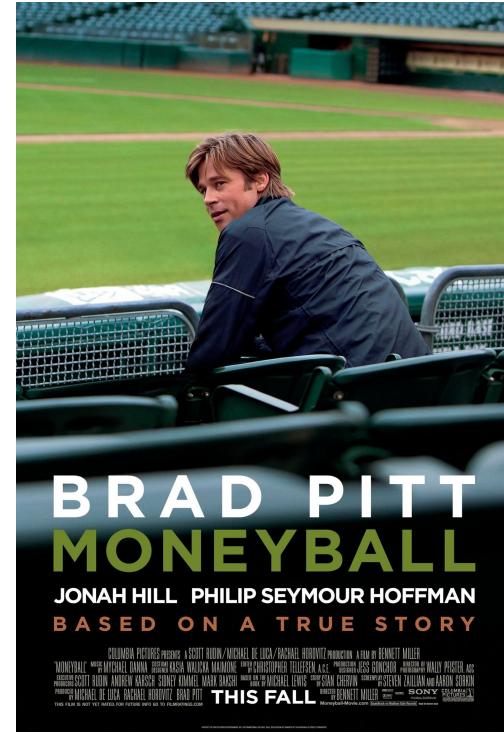
Blog: 2009-2010

okcupid

Brief history of Data Science



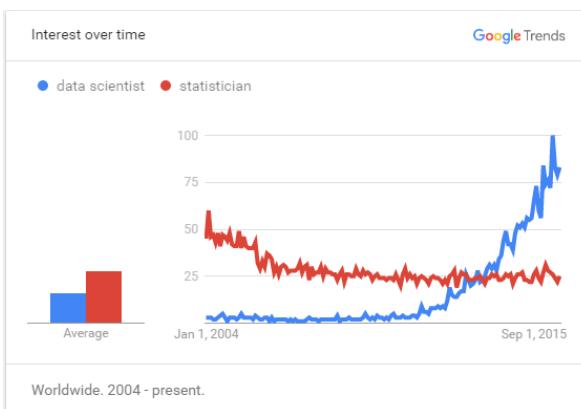
2011



2012



Data Science hits
“peak of inflated
expectations”
around 2012-14



Harvard
Business
Review

Latest Magazine Ascend Topics Podcasts Store The Big Idea Data & Visuals Case Selections

2012

Data Scientist: The Sexiest Job of the 21st Century

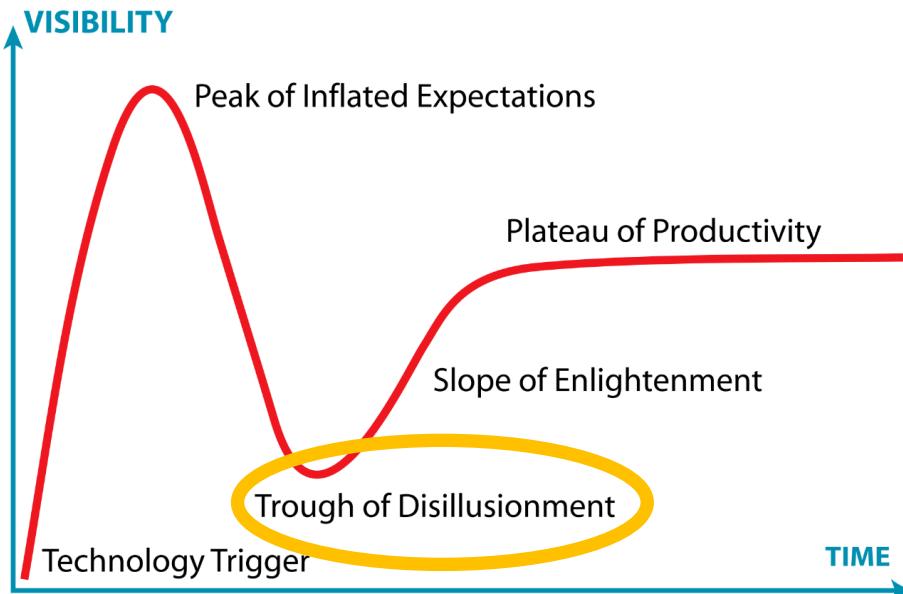
Meet the people who can coax treasure out of messy, unstructured data. by Thomas H. Davenport and DJ Patil

From the Magazine (October 2012)

Subscribe

Sign In

Brief history of Data Science: Technology Trigger



PROPUBLICA
Machine Bias
There's software used across the country to predict future criminals. And it's biased against blacks.
by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica
May 23, 2016

FiveThirtyEight
2016 Election Forecast

President Updated Nov. 8, 2016 Senate Updated Nov. 8, 2016 Analysis Updated Nov. 9, 2016

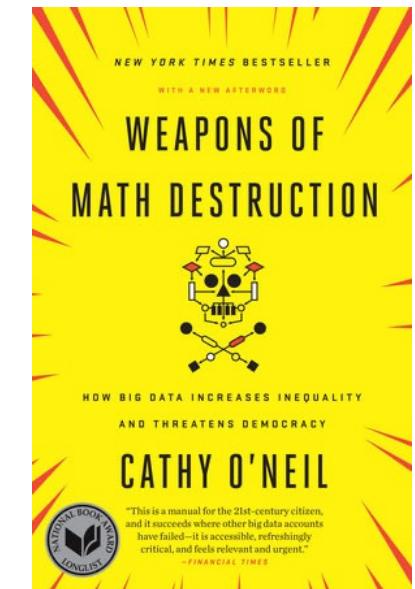
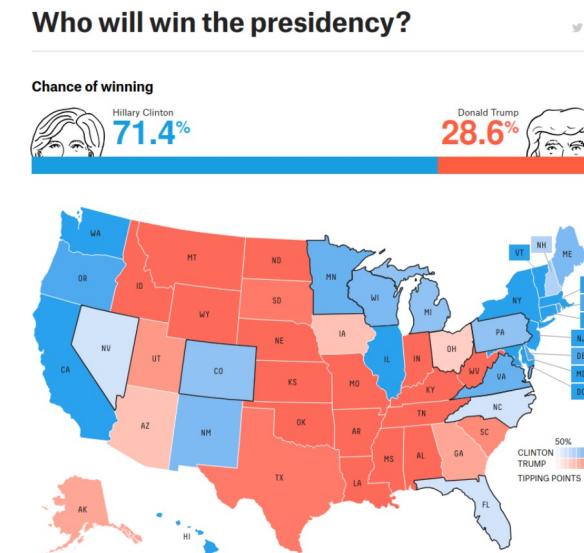
We're forecasting the election with three models

- Polls-plus forecast: What polls, the economy and historical data tell us about Nov. 8
- Polls-only forecast: What polls alone tell us about Nov. 8
- Now-cast: Who would win the election if were held today

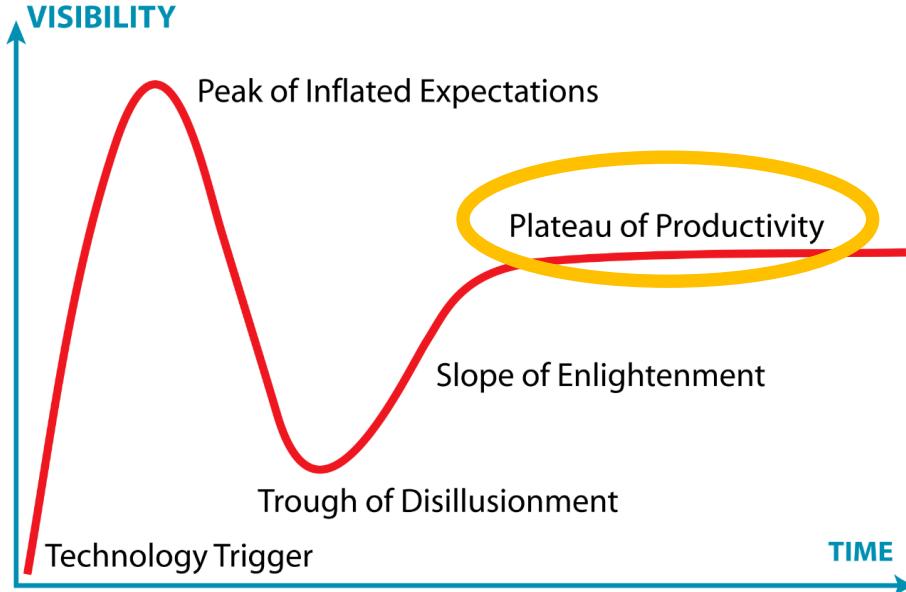
Who will win the presidency?

Candidate	Chance of winning
Hillary Clinton	71.4%
Donald Trump	28.6%

Negative consequences of predictive models were highlighted circa 2016



Brief history of Data Science: now



Data Scientists roles in many industries

- E.g., Data journalism ([NYTimes TheUpshot](#))

Many universities have Data Science programs

- In March 2017 Yale renames the Department of Statistics to be the Department of Statistics and Data Science

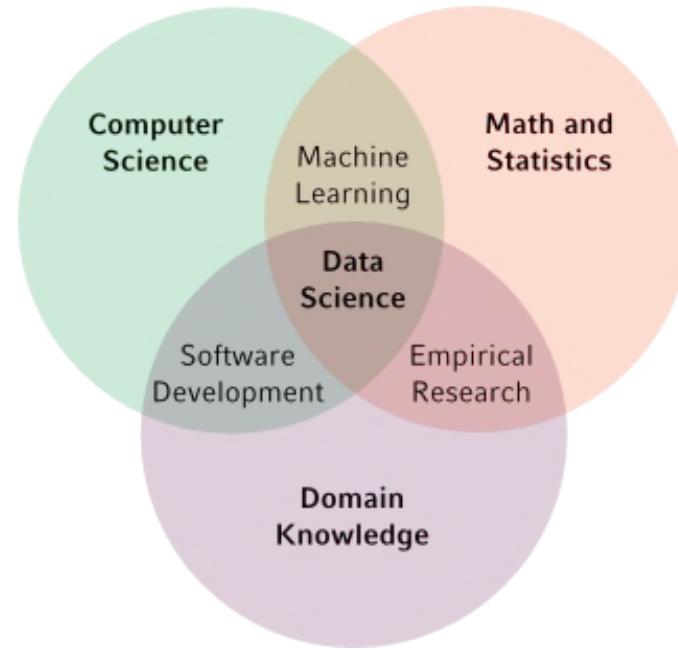
The cover features a dark blue map of the United States with various data visualization icons (charts, graphs, circles) overlaid. The title 'DATA SCIENCE' is at the top, followed by 'IS FOR EVERYONE'. Below the map, the date 'FEBRUARY 2024' and authors 'By Carlo Salerno and Frank Steemers' are listed. Logos for 'THE burningglass INSTITUTE' and 'ExcelinEd' are at the bottom.

Yale
S&DS

What is Data Science?

Data Science is a broadening of data analyses:

- Beyond what traditional statistical mathematical/inferential analyses
- To using more computation



Classical statistical analysis

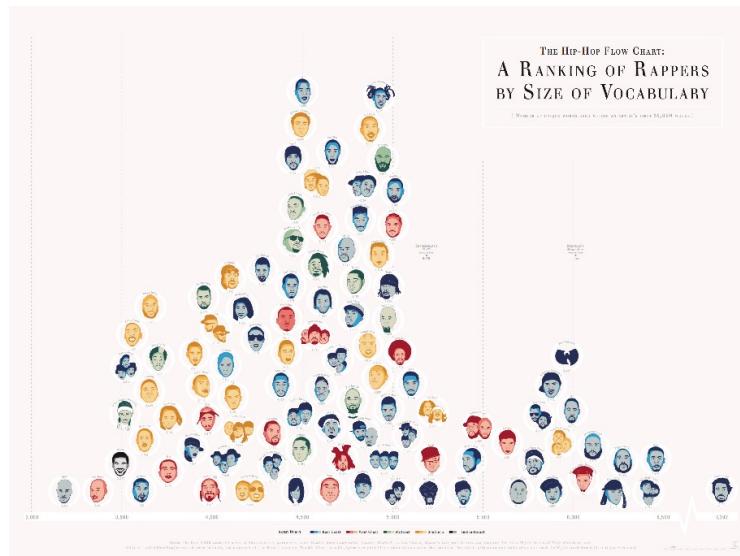
Descriptives

Bronchial reactivity	N	Descriptives						
		Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
Sulfur dioxide	5	18.840	9.6919	4.3344	6.806	30.874	5.1	30.1
Nitrous dioxide	6	6.617	3.9448	1.6105	2.477	10.757	2.2	11.9
Oxygen	4	4.975	3.4092	1.7046	-.450	10.400	2.1	9.3
Total	15	10.253	8.6514	2.2338	5.462	15.044	2.1	30.1

ANOVA

Bronchial reactivity

Bronchial reactivity	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	559.450	2	279.725	6.873	.010
Within Groups	488.408	12	40.701		
Total	1047.857	14			



Examples:

- [NYC city bikes](#)
- [Wind map visualization](#)

New ways to choose the best methods

Statistics focuses on mathematical models (probability distributions) to analyze data

- Best methods are the ones that have mathematical guarantees (proofs)

The proof is in the math

Now $x_1^2 - x_3^2 = \sqrt{p^2 - 4q} \equiv w \quad ②$
is a relation in $R(w)$
From ①, also $x_1^2 = x_2^2, x_3^2 = x_4^2$.

Data Science empirically evaluates data analysis methods

- Best methods are the one that gives the most insight in practice

The proof is in the pudding



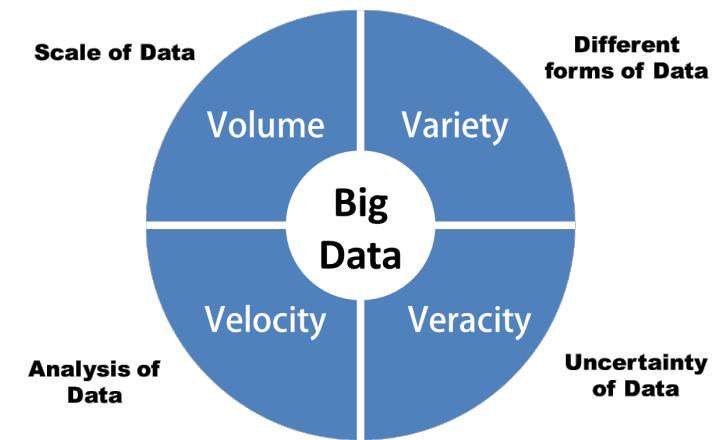
Big Data

New insights:

- Lots of new data from Internet, sensors etc., can be mined to transform our understanding in a range of fields
 - E.g., health, cosmology, social sciences, etc.

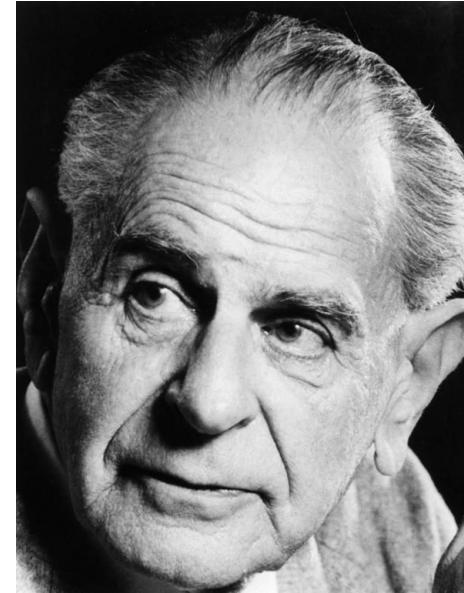
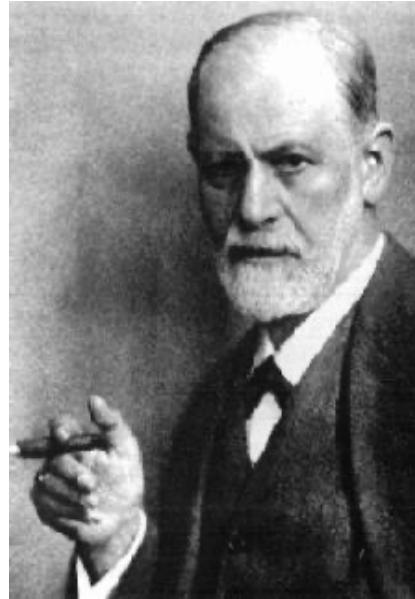
New analysis and approaches:

- Hypothesis test pick up on very small (meaningless) effects with very large samples
- Data manipulation and programming are needed to extract insights
- Also, new standards for choosing the best data analysis methods



[Data Science vs. Statistician video](#)

Thoughts on the reading from Everybody Lies?



Much of Freud's theory dealt with the subconscious

- E.g., Freudian slips

Karl Popper claimed that Freud's theories were unscientific because they couldn't be falsified

- i.e., can come up with any 'just so' story to explain a behavior

New data science analyses might make it possible to actually test Freud's theories

Introduction to Python



Please log into YCRC Jupyter notebook server...

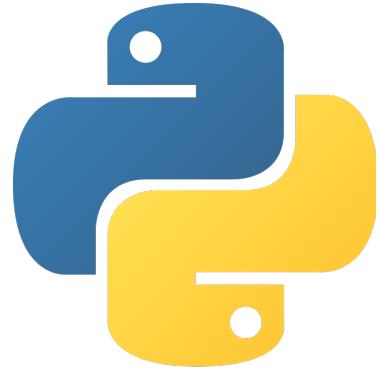
[A link to the server is at the top of
the class Canvas page](#)

Grace OnDemand

If you can't log in, please try to
look at the computer of someone
sitting next to you

Programming languages for Data Science

The two most popular languages for Data Science are:



General purpose programming language

- Can do a lot more than data analysis
- Code is easy to read
- Easy to write larger software packages
- Good machine learning package (scikitlearn)



Focused on data analysis

- Better for creating pdf reports
- Easy to create interactive apps
- RStudio created a great IDE and support

AS SEEN BY USERS OF ...

STATA

R

sas

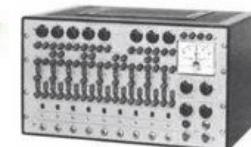
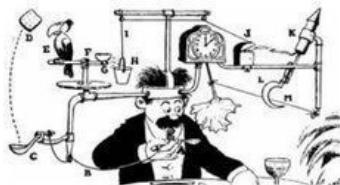
python

SPSS

STATA



R



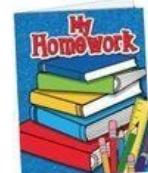
sas



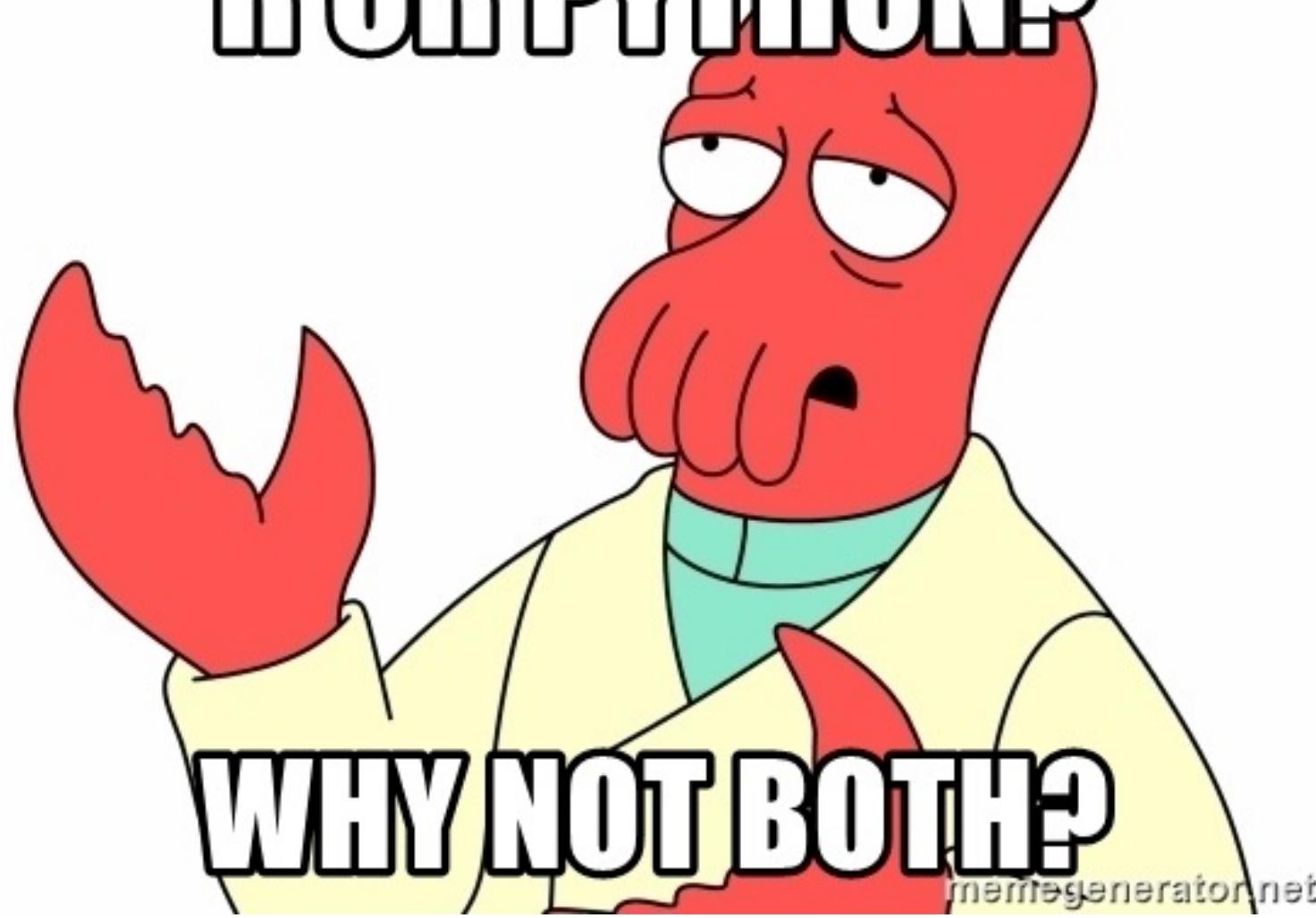
python



SPSS



R OR PYTHON?



WHY NOT BOTH?

memegenerator.net

Terminology: scripts, modules, and packages

A **script** is a piece of code that is run to accomplish a specific task

- E.g., one could write and run a script to download a set of files



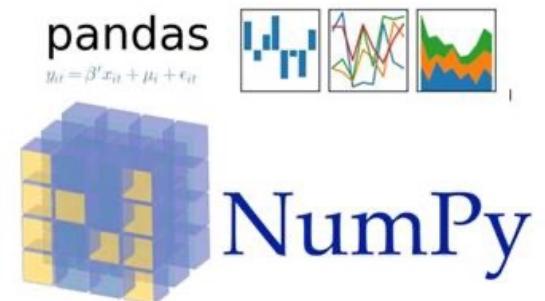
A **module** is a piece of code that reused by different scripts and programs

- Modules are **imported** by other programs/scripts to add commonly used functionality.
- E.g., `import matplotlib.pyplot as plt`



A **package** (library) contains several related modules

Packages are an essential building block in programming. Without packages, you would waste a lot of time writing code that's already been written



Jupyter notebooks

Jupyter notebooks allow one to create an analysis document that contains text, analysis code, and plotted results

Because one can see all the code used to generate results, it allows one to create reproduce analyses

```
[5]: import matplotlib.pyplot as plt  
plt.style.use('classic')  
%matplotlib inline  
import numpy as np  
import pandas as pd  
import seaborn as sns  
sns.set()
```

```
[6]: rng = np.random.RandomState(0)  
x = np.linspace(0, 10, 500)  
y = np.cumsum(rng.randn(500, 6), 0)
```

Next step

Now, create a graph.

```
[7]: plt.plot(x, y)  
plt.legend('ABCDEF', ncol=2, loc='upper left');
```



Programming in Python



Understanding the language fundamentals is important

Learn through practice, not only by reading or watching but by doing

- Like learning to ride a bike

If you need more practice, be sure to attend the Shivam's practice sessions!

Today you can follow along with the class 2 Jupyter notebook which we will download now!

Expressions

Expressions

Expressions describe how a computer should combine pieces of data

- They are evaluated to by the computer and return a value
- E.g., mathematical expressions
 - Multiplication: `3 * 4`
 - Exponentiation: `3**4`

Operation	Operation	Example	Value
Addition	<code>+</code>	<code>2 + 3</code>	5
Subtraction	<code>-</code>	<code>2 - 3</code>	-1
Multiplication	<code>*</code>	<code>2*3</code>	6
Division	<code>/</code>	<code>7/3</code>	2.667
Remainder	<code>%</code>	<code>7 % 3</code>	1
Exponentiation	<code>**</code>	<code>2**.05</code>	1.414

Syntax

The *Syntax* of a language is its set of grammar rules for how expressions can be written

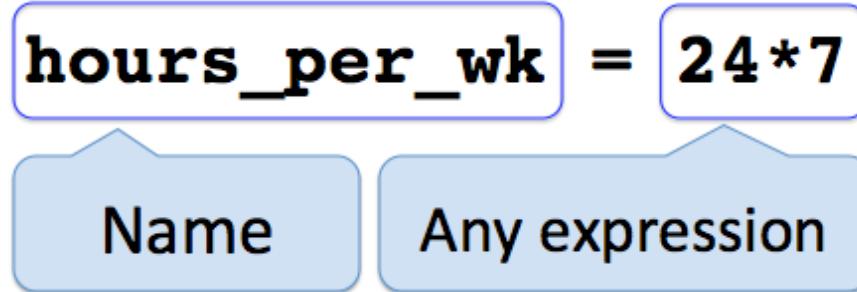
- *SyntaxError* indicates that an expression structure doesn't match any of the rules of the language.
- E.g., failed attempt at exponentiation: 3 * * 4

```
File "<ipython-input-2-012ea60b41dd>", line 1
  3 * * 4
    ^
SyntaxError: invalid syntax
```

Let's explore this in Jupyter!

Names

Assignment statements



Names store the values (from an expression)

- i.e., they are like variables in algebra

Names are assigned values using the `=` symbol

- E.g., `my_number = 7`

Let's explore this in Jupyter!

Call Expressions

Anatomy of a Call Expression

Call expressions are expressions that call functions

- Functions take in one or more values (arguments) and (usually) return another value

What function
to call

Argument to
the function

f(27)

Example: taking the maximum value

What function
to call

First
argument

Second
argument

max(15, 27)

Let's explore this in Jupyter!

Numerical data

Arithmetic operations

Operation	Operation	Example	Value
Addition	+	$2 + 3$	5
Subtraction	-	$2 - 3$	-1
Multiplication	*	$2 * 3$	6
Division	/	$7 / 3$	2.667
Remainder	%	$7 \% 3$	1
Exponentiation	**	$2 ** .05$	1.414

We can store the output of evaluating expression in names

- `my_result = 10 * 2`

Numbers in Python: Ints and Floats

Python has two basic number types

- **int**: an integer of any size
- **float**: a number with an optional decimal part

An int never has a decimal point - a float always does

- 3 **# int of float?**
- 2.7 **# int of float?**

A float might be printed using scientific notation

Notes on Floats

Three limitations of float values:

- They have limited size (but the limit is huge)
- They have limited precision of 15 - 16 decimal places
- After arithmetic, the final few decimal places can be wrong



Let's explore this in Jupyter!

Strings

Text and Strings

A string value is a snippet of text of any length

- 'a'
- 'word'
- "there can be 2 sentences. Here's the second!"

Strings consisting of numbers can be converted to numbers

- `int('12')`
- `float('1.2')`

Any value can be converted to a string

- `str(5)`

Let's explore this in Jupyter!

Types

Every value has a type

We've seen several types so far:

- int: `2`
- Built-in function: `abs()`
- float: `2.2`
- str: `'Red fish, blue fish'`

The `type` function can tell you the type of a value

- `type(2)`
- `type('Red fish')`

An expression's type is based on its value, not how it looks

- `my_text = 2`
- `type(my_text)`

Let's explore this in Jupyter!

Conversions

Strings that contain numbers can be converted to numbers

- `int('12')`
- `float('1.2')`
- `float('one point two')` # Not a good idea!

Any numeric value can be converted to a string

- `str(5)`

Numbers can be converted to other numeric types

- `float(1)`
- `int(1.2)` # DANGER: loses information!

Lists

Lists

Lists are ways to store multiple items

We can create lists using square brackets []

- my_list = [2, 3, 4]

We can also access list items using square brackets []

- my_list[2]

Lists can contain elements of different types

- my_list2 = [5, 6, 'seven']

TO DO LIST

1. make lists
2. look at lists
3. PANIC!

Let's explore this in Jupyter!

Next class: Text manipulation!

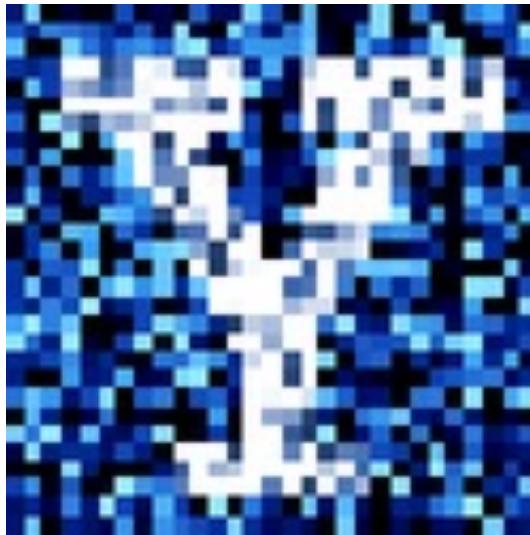
Homework 1 has been posted

```
import YData  
YData.download.download_homework(1)
```

It is due on Gradescope on Sunday September 8th at 11pm

- Be sure to mark each question on Gradescope!

YData: Introduction to Data Science



Class 03: Python basics continued

Overview

Review of an intro to Python

- Expressions, Names, Call expressions

More intro to Python

- Strings
- Types
- Lists
- Text manipulation

If there is time:

- Booleans and comparisons
- Additional string methods



Announcement: Homework 1

Homework 1 has been posted!

```
import YData
```

```
YData.download.download_homework(1)
```

It is due on Gradescope on Sunday September 8th at 11pm

- Be sure to mark each question on Gradescope!

Practice sessions for this week are:

- Thursday (today) 5-6, and 6-7
- Friday (tomorrow) 3-4, and 4-5
- Subject to change in future weeks (fill out survey from Shivam)

Review: Basics of Python

Last class we discussed the basics of Python including...

Expressions:

- $2 + 3$
- 2^{**3}

Names and assignment:

- `class_number = 123`

Numerical representations:

- `my_int = 2`
- `my_float = 3.14159`

Let's quickly practice this in Jupyter!

Strings

Text and Strings

A string value is a snippet of text of any length

- 'a'
- 'word'
- "there can be 2 sentences. Here's the second!"

Strings consisting of numbers can be converted to numbers

- `int('12')`
- `float('1.2')`

Any value can be converted to a string

- `str(5)`

Let's explore this in Jupyter!

Types

Every value has a type

We've seen several types so far:

- int: `2`
- Built-in function: `abs()`
- float: `2.2`
- str: `'Red fish, blue fish'`

The `type` function can tell you the type of a value

- `type(2)`
- `type('Red fish')`

An expression's type is based on its value, not how it looks

- `x = 2`
- `type(x)`

Let's explore this in Jupyter!

Conversions

Strings that contain numbers can be converted to numbers

- `int('12')`
- `float('1.2')`
- `float('one point two')` # Not a good idea!

Any numeric value can be converted to a string

- `str(5)`

Numbers can be converted to other numeric types

- `float(1)`
- `int(1.2)` # DANGER: loses information!

Lists

Lists

Lists are ways to store multiple items

We can create lists using square brackets []

- my_list = [2, 3, 4]

We can also access list items using square brackets []

- my_list[2]

Lists can contain elements of different types

- my_list2 = [5, 6, 'seven']

TO DO LIST

1. make lists
2. look at lists
3. PANIC!

Let's explore this in Jupyter!

text
MaNiPuLaTiOn

Text manipulation

80% of a Data Scientists time is cleaning data

- Text manipulation is a big part of cleaning data

20% of a Data Scientists time is complaining about cleaning data

Python has many string methods that are useful for manipulating text and cleaning data!

Terminology: functions and methods

Recall: Functions take in zero or more values (arguments) and (usually) return another value

- E.g., `abs(-5)` # takes the absolute value of -5

Methods are functions that operate on particular pieces of data

The syntax for methods is: `data_object.method()`

Example:

`"hello".upper()`

`"HELLO"` # returns capitalized string

Methods &

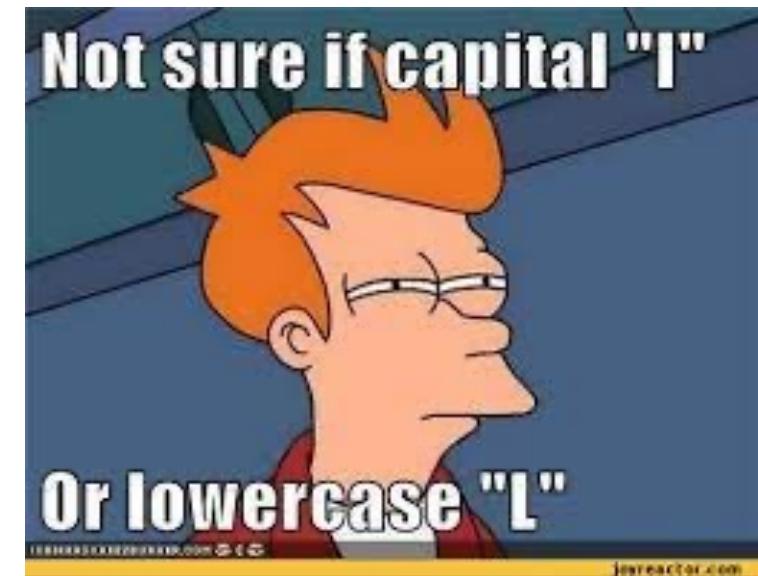


Protocols

Text manipulation: capitalization

Some of the simplest string methods involve changing capitalization

Changing capitalization can be useful when combining (joining) data sets as we will discuss later in the semester.



Text manipulation: capitalization

Python strings have a number of methods to change the capitalization of words including:

- `.capitalize()`: Converts the first character to upper case
- `.lower()`: Converts a string into lower case
- `.upper()`: Converts a string into upper case
- `.title()`: Converts the first character of each word to upper case
- `.swapcase()`: Swaps cases, lower case becomes upper case and vice versa

Let's explore this in Jupyter!

Text manipulation: splitting and joining strings

There are several methods that can help us join strings that are contained into a list into a single string, or conversely, parse a single string into a list of strings. These include:

- `.split(separator_string)`: Splits the string at the specified separator, and returns a list
- `.splitlines()`: Splits the string at line breaks and returns a list
- `.join(a_list)`: Converts the elements of an iterable into a string

Let's explore this in Jupyter!

Text manipulation: finding and replacing substrings

Some methods for locating a substring within a larger string include:

`.count(substring)`: Returns the number of times a specified value occurs in a string

`.replace(original_str, replacement_str)`: Replace a substring with a different string

Also:

`.startswith(substring)`: Returns true if the string starts with the specified value

`.endswith(substring)` : Returns true if the string ends with the specified value

Let's explore this in Jupyter!

Booleans and comparisons

Comparisons

We can use mathematical operators to compare numbers and strings

- Results return Boolean values **True** and **False**

Comparison	Operator	True example	False Example
Less than	<	$2 < 3$	$2 < 2$
Greater than	>	$3 > 2$	$3 > 3$
Less than or equal	\leq	$2 \leq 2$	$3 \leq 2$
Greater or equal	\geq	$3 \geq 3$	$2 \geq 3$
Equal	\equiv	$3 \equiv 3$	$3 \equiv 2$
Not equal	\neq	$3 \neq 2$	$2 \neq 2$

True is equal to 1
False is equal to 0

True + True + False
is equal to...

We can compare strings alphabetically

- $a < b$

2

Let's explore this in Jupyter!

Text manipulation: checking string properties

There are also many functions to check properties of strings including:

- `.isalnum()`: Returns True if all characters in the string are alphanumeric
- `.isalpha()`: Returns True if all characters in the string are in the alphabet
- `.isnumeric()`: Returns True if all characters in the string are numeric
- `.isspace()`: Returns True if all characters in the string are whitespaces
- `.islower()`: Returns True if all characters in the string are lower case
- `.isupper()`: Returns True if all characters in the string are upper case
- `.istitle()`: Returns True if the string follows the rules of a title

Let's explore this in Jupyter!

Additional string methods

Text manipulation: string padding

Often we want to remove extra spaces (called "white space") from the front or end of a string

Conversely, sometimes we want to add extra spaces to make a set of strings the same length

- This is known as "string padding"

Python strings have a number of methods that can pad/trim strings including:

- `.strip()`: Returns a trimmed version of the string (i.e., with no leading or trailing white space)
 - Also, `.rstrip()` and `.lstrip()`: Returns a right/left trim version of the string
- `.center(num)`: Returns a centered string (with equal padding on both sides)
 - Also `.ljust(num)` and `.rjust(num)`: Returns a right justified version of the string
- `.zfill(num)`: Fills the string with a specified number of 0 values at the beginning

Let's explore this in Jupyter!

Text manipulation: filling in strings with values

There are a number of ways to fill in strings parts of a string with particular values.

Perhaps the most useful is to use "f strings", which have the following syntax such as:

- `value_to_fill = "my_value"`
- `f"my string {value_to_fill} will be filled in"`

Let's explore this in Jupyter!

Brief mention: regular expressions

More complex text manipulation can be done using “regular expressions”

```
import re  
bool(re.match("m.ss", "mess"))
```

We might discuss regular expression later in the semester...



YData: Introduction to Data Science



Class 04: Descriptive statistics and plots

Overview

Continuation of python basics:

- Comparisons
- Quick discussion of additional string methods

Statistics and data visualizations:

- Categorical data: Proportions, bar plots and pie charts
- Quantitative data: mean median and histograms
- If there is time:
 - Measures of spread: standard deviation and z-scores



Announcement: Homework 2

Homework 2 has been posted!

```
import YData
```

```
YData.download.download_homework(2)
```

It is due on Gradescope on Sunday September 15th at 11pm

- Be sure to mark each question on Gradescope!

Notes:

- There is an ~18 page reading from the book "Data and the American Dream" that you need to do, so I recommend you get started on this soon.

Review: Lists

Last class we discussed lists

- Suppose we have the list: `z = [1, 2, 3, 2, 1, [8, 9, 10]]`

Retrieving items from a list and slicing:

- `z[2]`
- `z[1:4:2]`

List functions and methods:

- `len(z)`
- `z.count(2)`
- `new_list = z + [1, 2, 3]` *# returns a new list and saves it to the name new_list*
- `z.append(7)` *# modifies the list z (and returns None)*

Functions on lists of numbers:

- `sum()`, `max()`, `min()`

Review: String methods

Last class we also discussed string methods:

- Suppose we have a string `my_string = "Hello Yale"`



String methods

- `my_string.upper()`
- `my_string.replace("Yale", "Whale")`
- `string_list = my_string.split(" ")`
- `", ".join(string_list)`

Let's do a very quick warmup exercise in Jupyter!

Booleans and comparisons

Comparisons

We can use mathematical operators to compare numbers and strings

- Results return Boolean values **True** and **False**

Comparison	Operator	True example	False Example
Less than	<	$2 < 3$	$2 < 2$
Greater than	>	$3 > 2$	$3 > 3$
Less than or equal	\leq	$2 \leq 2$	$3 \leq 2$
Greater or equal	\geq	$3 \geq 3$	$2 \geq 3$
Equal	\equiv	$3 \equiv 3$	$3 \equiv 2$
Not equal	\neq	$3 \neq 2$	$2 \neq 2$

True is equal to 1

False is equal to 0

True + True + False
is equal to...

2

We can compare strings alphabetically

- '**a**' < '**b**'

Let's explore this in Jupyter!

String methods: checking string properties

There are also many functions to check properties of strings including:

- `.isalnum()`: Returns True if all characters in the string are alphanumeric
- `.isalpha()`: Returns True if all characters in the string are in the alphabet
- `.isnumeric()`: Returns True if all characters in the string are numeric
- `.isspace()`: Returns True if all characters in the string are whitespaces
- `.islower()`: Returns True if all characters in the string are lower case
- `.isupper()`: Returns True if all characters in the string are upper case
- `.istitle()`: Returns True if the string follows the rules of a title

Let's explore this in Jupyter!

Additional string methods

String methods: string padding

Often we want to remove extra spaces (called "white space") from the front or end of a string

Conversely, sometimes we want to add extra spaces to make a set of strings the same length

- This is known as "string padding"

Python strings have a number of methods that can pad/trim strings including:

- `.strip()`: Returns a trimmed version of the string (i.e., with no leading or trailing white space)
 - Also, `.rstrip()` and `.lstrip()`: Returns a right/left trim version of the string
- `.center(num)`: Returns a centered string (with equal padding on both sides)
 - Also `.ljust(num)` and `.rjust(num)`: Returns a right justified version of the string
- `.zfill(num)`: Fills the string with a specified number of 0 values at the beginning

Let's explore this in Jupyter!

String methods: filling in strings with values

There are a number of ways to fill in strings parts of a string with particular values

Perhaps the most useful is to use "f strings", which have the following syntax such as:

- `value_to_fill = "my_value"`
- `f"my string {value_to_fill} will be filled in"`

Let's explore this in Jupyter!

Brief mention: regular expressions

More complex text manipulation can be done using “regular expressions”

```
import re  
bool(re.match("m.ss", "mess"))
```

We might discuss regular expression later in the semester...



A brief introduction to statistics and
data visualization...

The Bechdel test

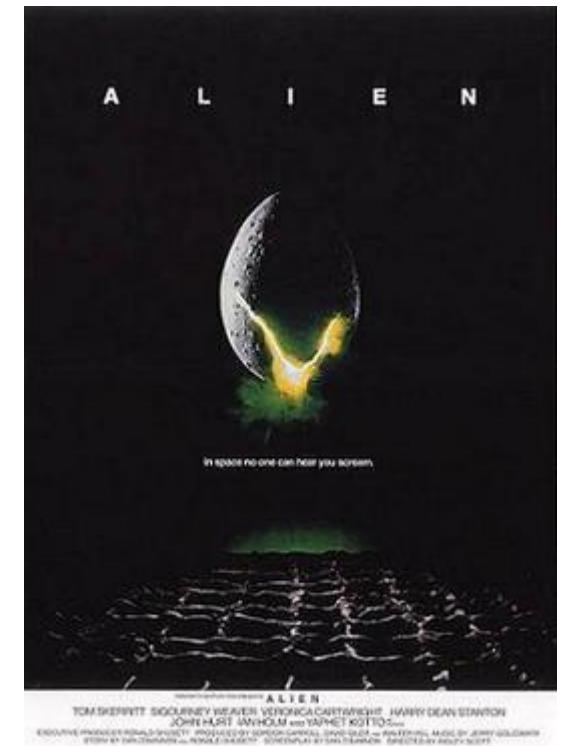


1. Has to have ≥ 2 women
2. Who talk to each other
3. About something other than a man

Where do samples/data come from?

Suppose we had a random sample of 1794 movies

- The *sample size* is 1794 (n = 1794)



The Bechdel data

Variables

Cases

	title	clean_test	binary	budget	domgross	budget_2013	domgross_2013
	21 & Over	notalk	FAIL	13000000	25682380.0	13000000	25682380.0
	Dredd 3D	ok	PASS	45000000	13414714.0	45658735	13611086.0
	12 Years a Slave	notalk	FAIL	20000000	53107035.0	20000000	53107035.0
	2 Guns	notalk	FAIL	61000000	75612460.0	61000000	75612460.0
	42	men	FAIL	40000000	95020213.0	40000000	95020213.0

Categorical and Quantitative Variables

Cases

Categorical Variable

Quantitative Variable

	title	clean_test	binary	budget	domgross	budget_2013	domgross_2013
21 & Over		notalk	FAIL	13000000	25682380.0	13000000	25682380.0
Dredd 3D		ok	PASS	45000000	13414714.0	45658735	13611086.0
12 Years a Slave		notalk	FAIL	20000000	53107035.0	20000000	53107035.0
2 Guns		notalk	FAIL	61000000	75612460.0	61000000	75612460.0
42		men	FAIL	40000000	95020213.0	40000000	95020213.0

Categorical data

statistics

A ***statistic*** is a number computed from a sample of data

Quantitative Variable

	title	clean_test	binary	budget	domgross	budget_2013	domgross_2013
21 & Over		notalk	FAIL	13000000	25682380.0	13000000	25682380.0
Dredd 3D		ok	PASS	45000000	13414714.0	45658735	13611086.0
12 Years a Slave		notalk	FAIL	20000000	53107035.0	20000000	53107035.0
2 Guns		notalk	FAIL	61000000	75612460.0	61000000	75612460.0
42		men	FAIL	40000000	95020213.0	40000000	95020213.0

A blue oval highlights the 'domgross_2013' column. A red arrow points from this column to the value 95174784.

Proportions

For a *single **categorical variable***, the main **statistic** of interest is the *proportion* in each category

$$\text{Proportion in a category} = \frac{\text{number in that category}}{\text{total number}}$$

Proportions

For a *single categorical variable*, the main *statistic* of interest is the *proportion* in each category

- E.g., the proportion of movies that passed the Bechdel test

Proportion passed =

$$\frac{\text{number of movies that passed}}{\text{total number}}$$

Let's explore this in Jupyter!

Categorical Variable

title	clean_test	binary
21 & Over	notalk	FAIL
Dredd 3D	ok	PASS
12 Years a Slave	notalk	FAIL
2 Guns	notalk	FAIL
42	men	FAIL

Visualizing Categorical Data

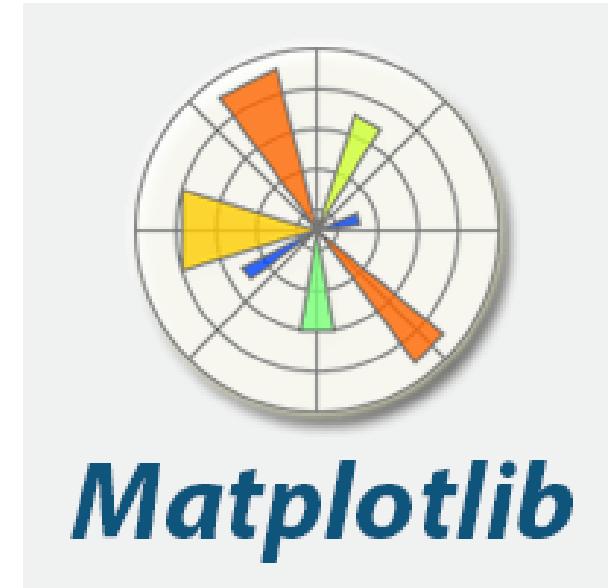
Plotting data

To create basic visualizations in Python we can use the matplotlib library

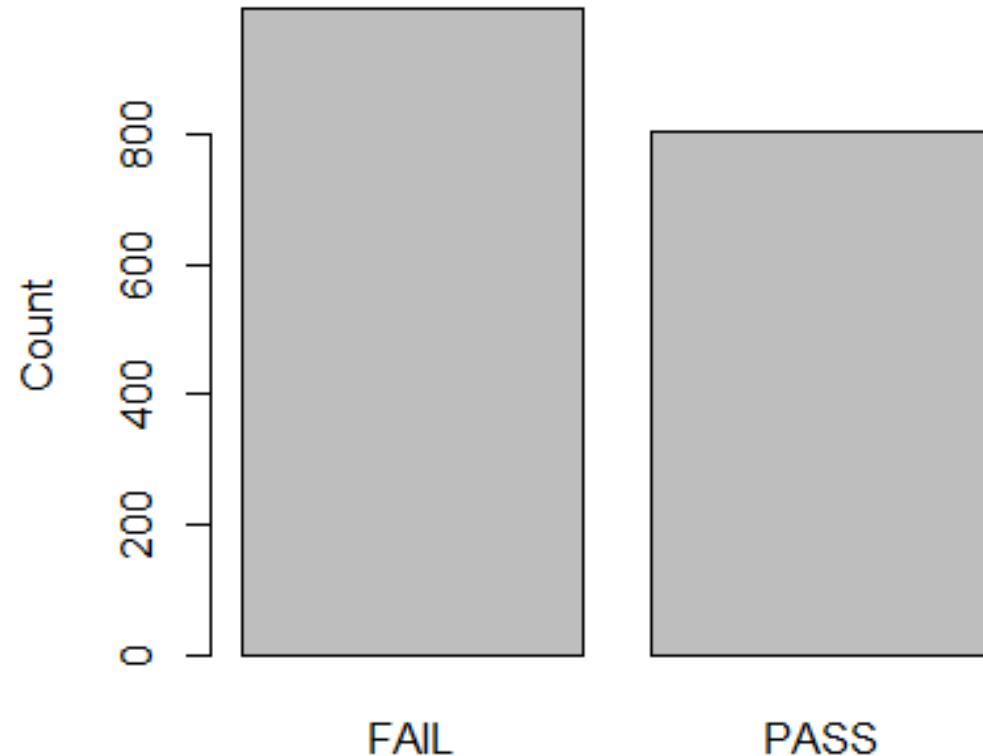
```
import matplotlib.pyplot as plt
```

We can then create plots using functions such as:

- `plt.plot()`
- `plt.bar()`
- `plt.hist()`
- etc.

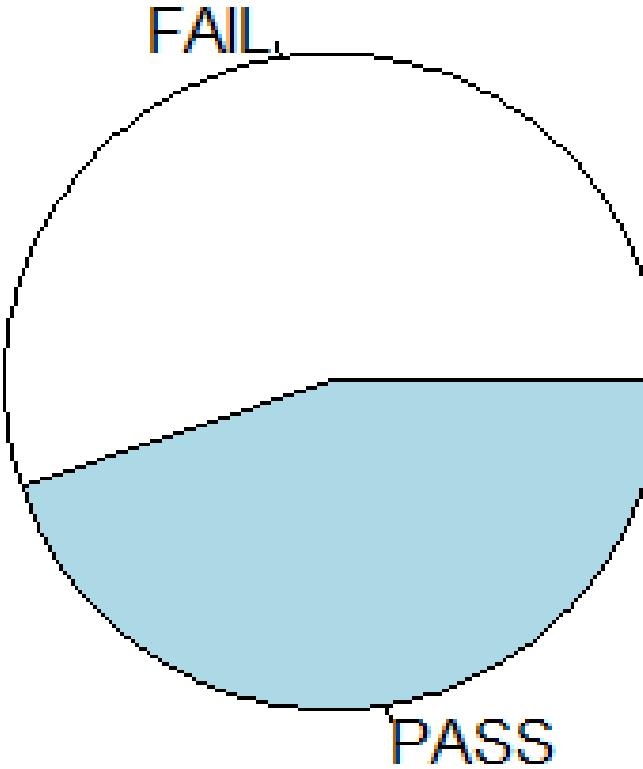


Visualizing categorical data: The Bar Chart



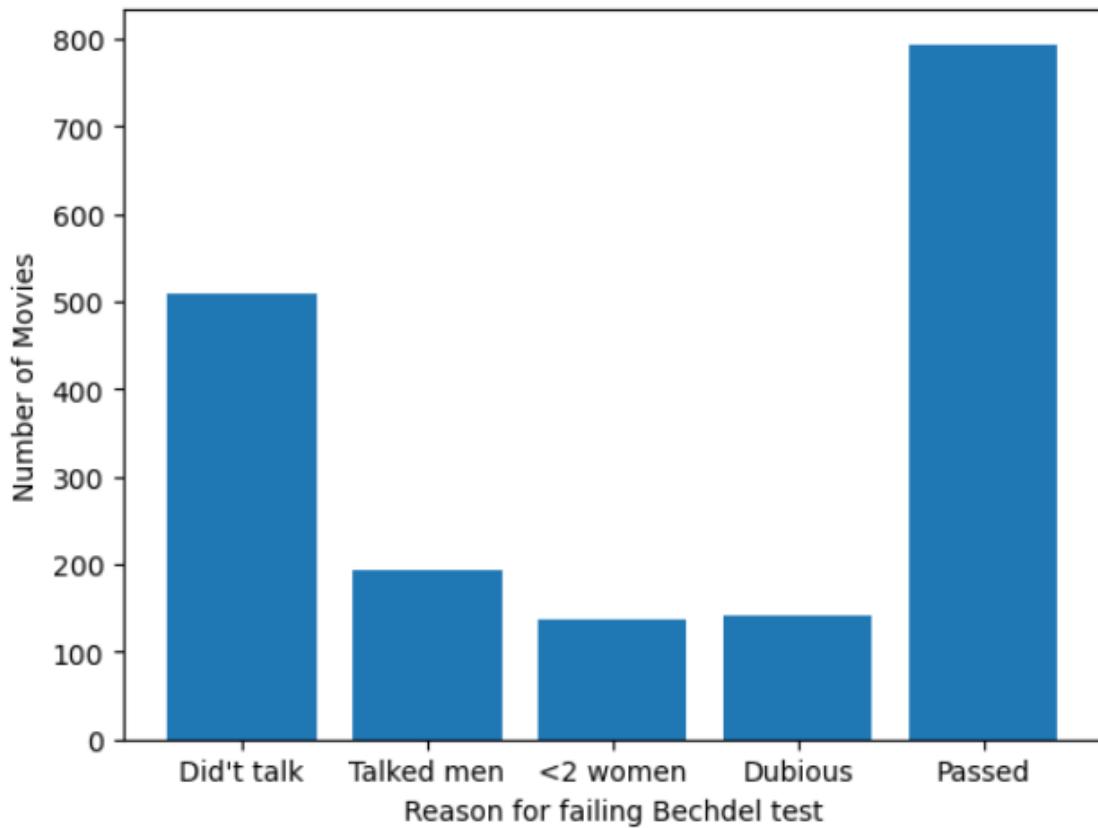
Matplotlib: `plt.bar(labels, data)`

Visualizing categorical data: The Pie Chart



Matplotlib: `plt.pie(data)`

Labeling axes!



```
plt.bar(reason_names, reason_counts);
```

```
plt.xlabel("Reason failed");
```

```
plt.ylabel("Number of movies")
```

Quantitative data

Quantitative data

To explore quantitative data, let's look at how much revenue each movie made in the United States in (2013) inflation adjusted dollars

- `domgross_2013`

Quantitative Variable

	<code>title</code>	<code>clean_test</code>	<code>binary</code>	<code>budget</code>	<code>domgross</code>	<code>budget_2013</code>	<code>domgross_2013</code>
	21 & Over	notalk	FAIL	13000000	25682380.0	13000000	25682380.0
	Dredd 3D	ok	PASS	45000000	13414714.0	45658735	13611086.0
	12 Years a Slave	notalk	FAIL	20000000	53107035.0	20000000	53107035.0
	2 Guns	notalk	FAIL	61000000	75612460.0	61000000	75612460.0
	42	men	FAIL	40000000	95020213.0	40000000	95020213.0

Visualizing quantitative data: histograms

Movie US revenue (in millions of dollars):

- 25.68, 13.61, 53.11, 236.84, ...

To create a histogram we create a set of intervals

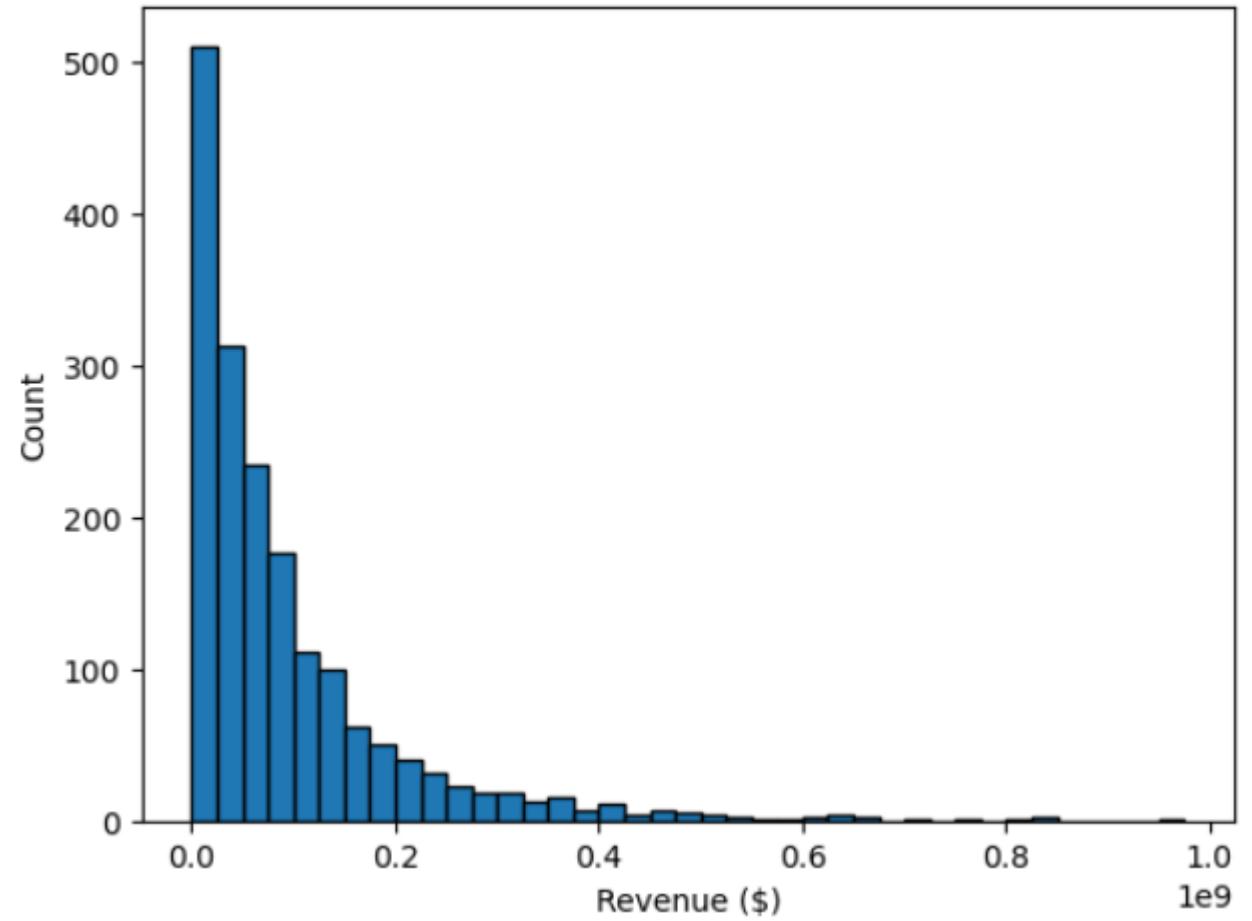
- 0-25, 25-50, 50-75, ... 200-250, 250-300

We count the number of points that fall in each interval

We create a bar chart where the height of the bars is the counts in each bin

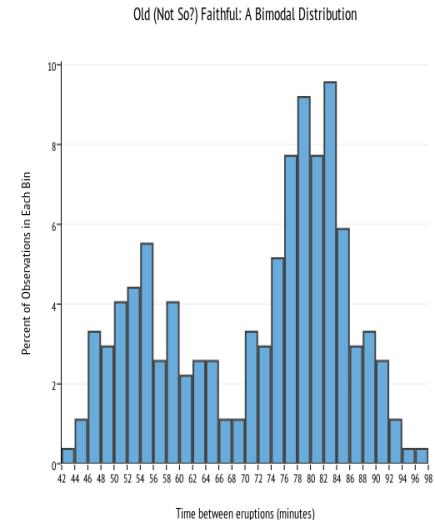
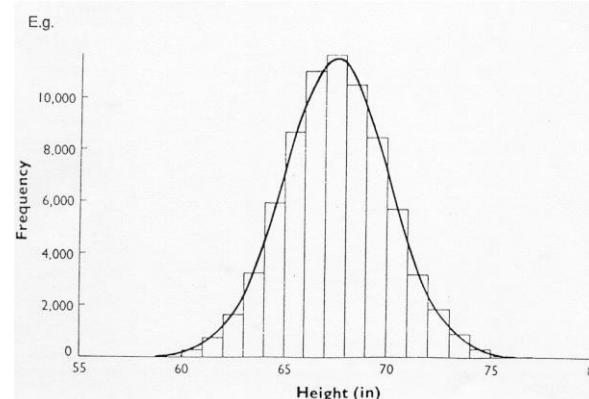
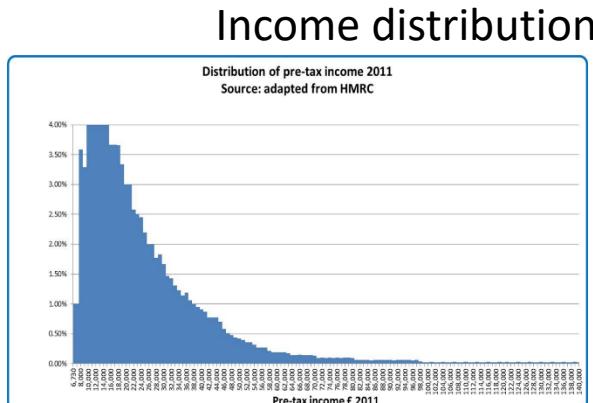
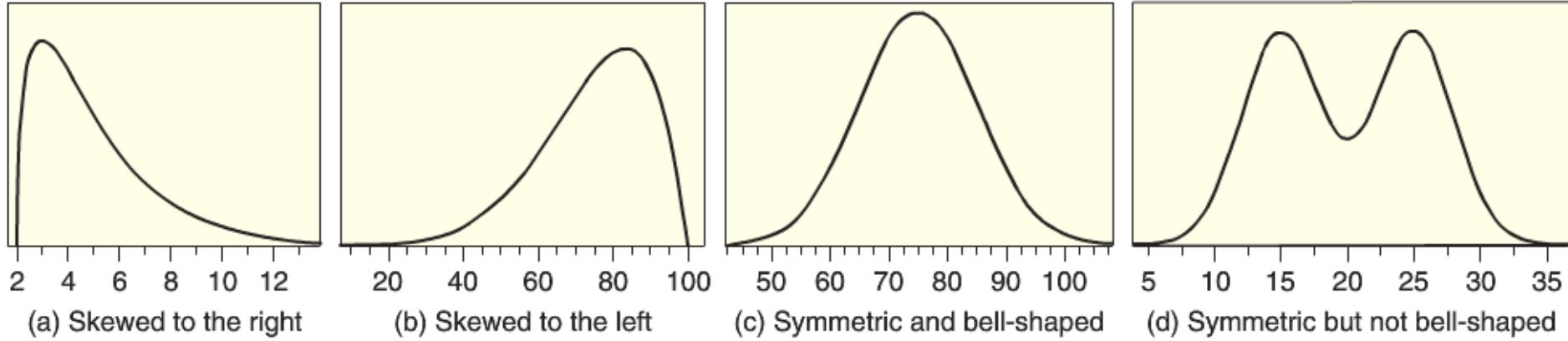
Histograms – movie US revenue

Domgross range	Frequency Count
(0 – 25]	510
(25 – 50]	312
(50 – 75]	234
(75 – 100]	176
(100 – 125]	111
(125 – 150]	99
(150 – 175]	62
(175 – 200]	51
(200 – 225]	40
(225 – 250]	32



Matplotlib: `plt.hist(data)`

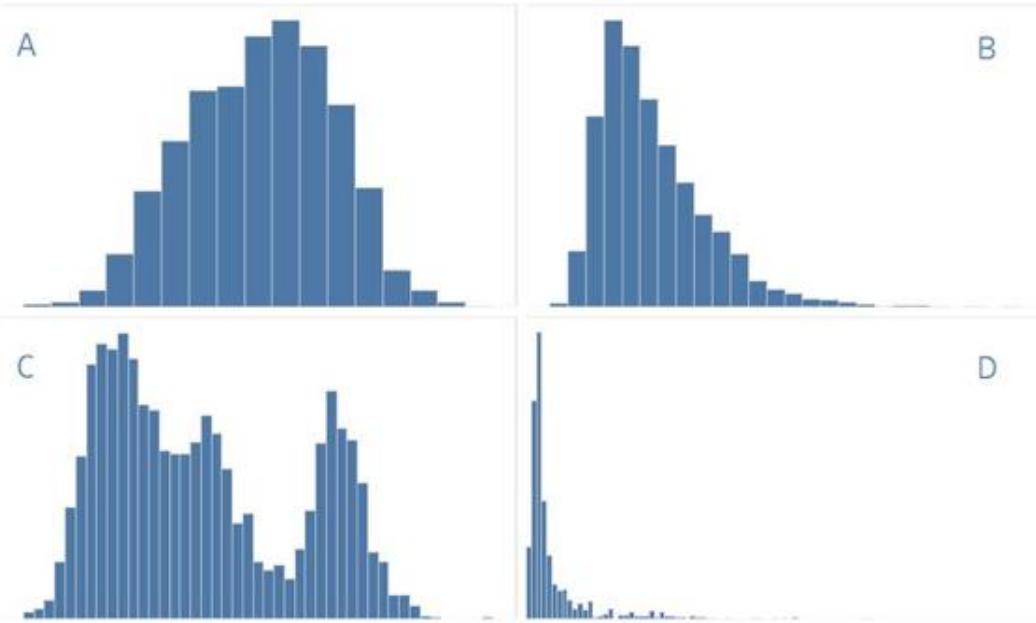
Common shapes of data distributions





Neat facts – the average NFL player is:

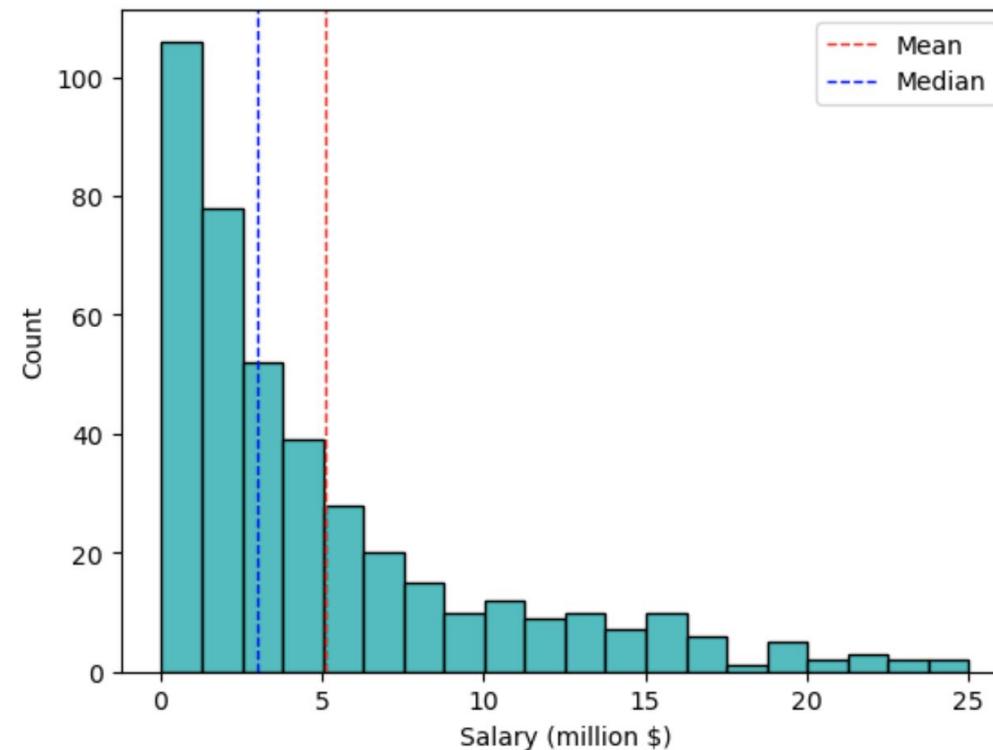
- 1. **Age:** Is about 25 years old
- 2. **Height:** Is just over 6'2" in height
- 3. **Weight:** Weighs a little more than 244lbs
- 4. **Salary:** Makes slightly less than \$1.5M in salary per year



Question: Can you tell which histogram goes with which trait?

Quantitative data: statistics for central tendency

Two statistics for measuring the “central value” of a sample of quantitative data are the ***mean*** and the ***median***



The mean

$$\text{Mean} = \frac{\text{Sum of all data values}}{\text{Number of data values}}$$

$$\text{Mean} = \frac{x_1 + x_2 + x_3 + \dots + x_n}{n} = \sum_{i=1}^n \frac{x_i}{n} = \frac{1}{n} \sum_{i=1}^n x_i$$

```
import statistics  
statistics.mean(data_list)
```

The median

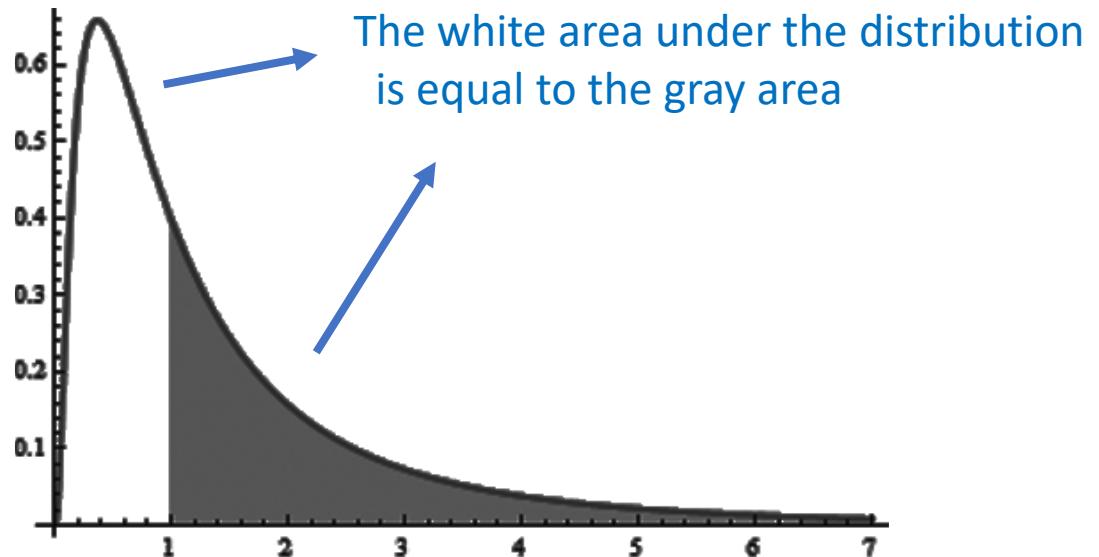
The **median** is a value that splits the data in half

- i.e., half the values in the data are smaller than the median and half are larger

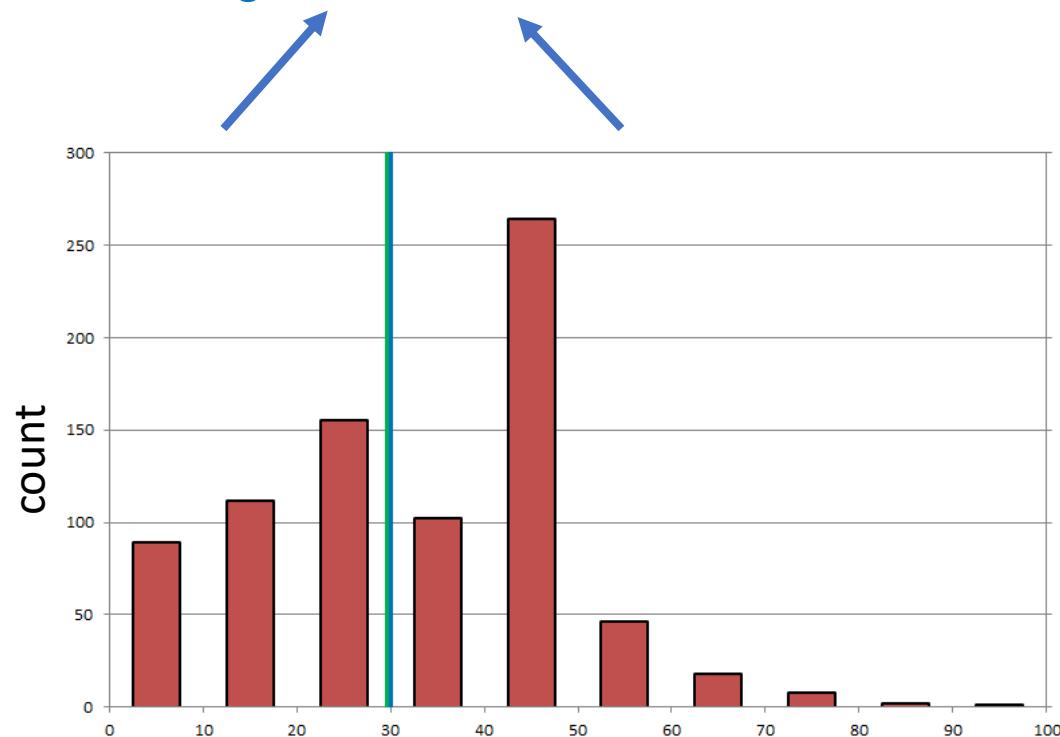
To calculate the median for a data sample of size n , sort the data and then:

- If n is odd: The middle value of the sorted data
- If n is even: The average of the middle two values of the sorted data

The median



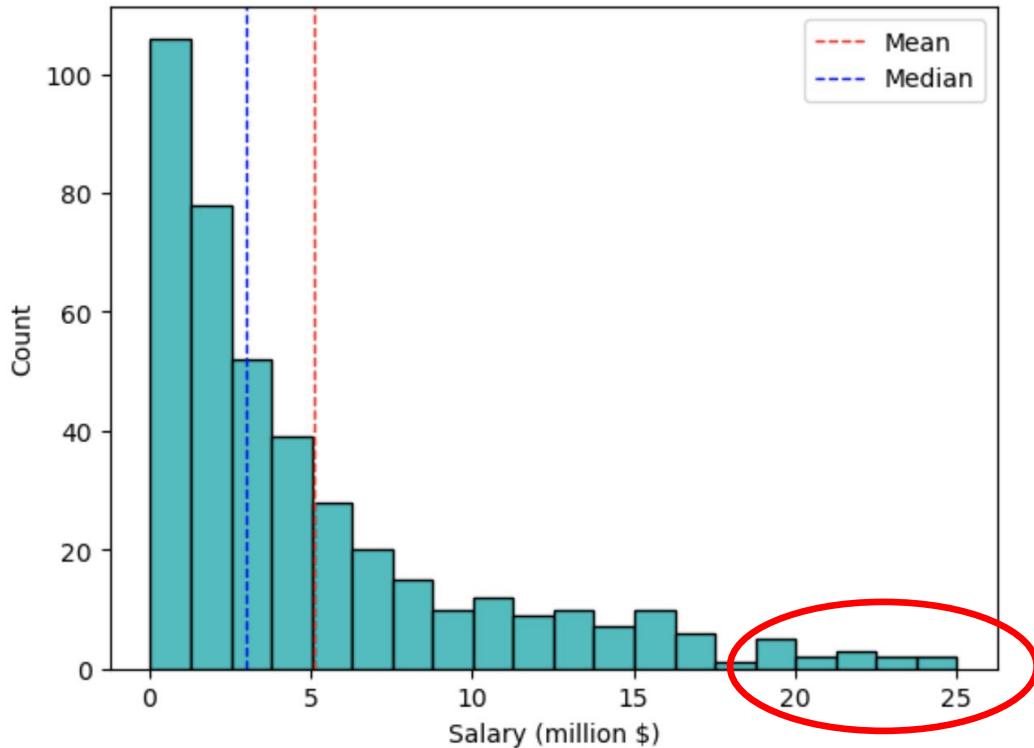
The sum of the heights of the bars on the left is equal to the sum of the heights of the bars on the right



```
import statistics  
statistics.median(data_list)
```

Outliers

An **outlier** is an observed value that is notably distinct from the other values in a dataset by being much smaller or larger than the rest of the data.

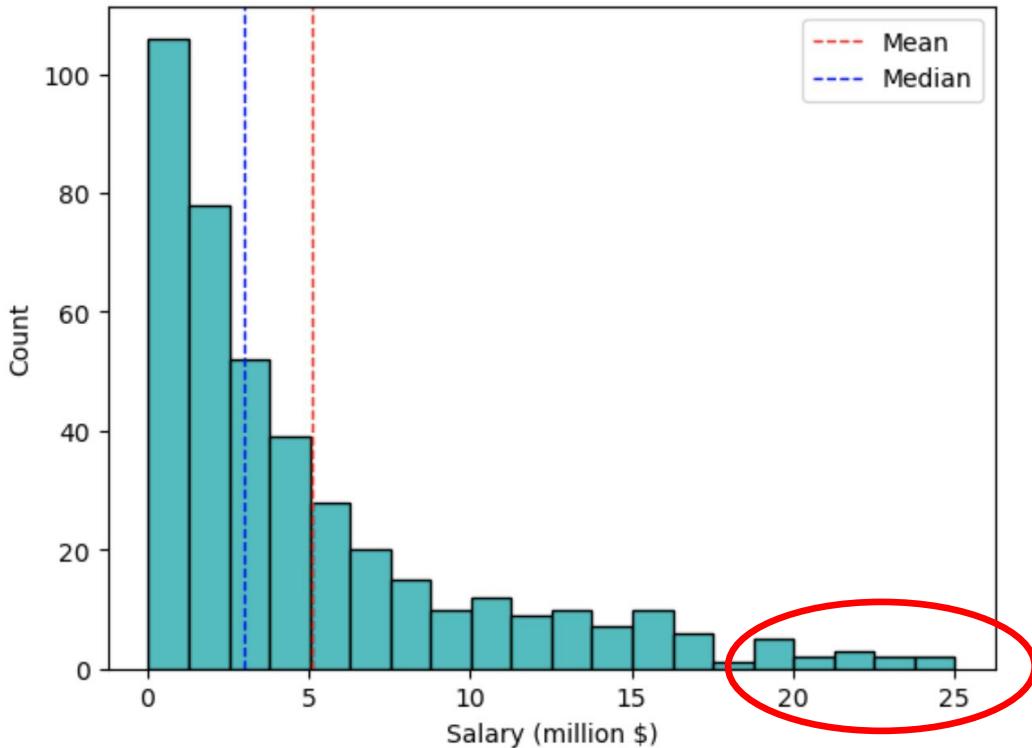


Outliers can potentially have a large influence on the statistics you calculate

One should examine outliers to understand what is causing them

- If there are due to an error, remove them
- Otherwise, need to think about how to treat them
 - Could be interesting phenomenon
 - Could restrict data to a particular range of values
 - Etc.

Outliers' impact on mean and median



The median is *resistant* to outliers

- i.e., not affected much by outliers

The mean is not resistant to outliers

What is the mean and median of the data: 1, 2, 3, 4, 990?

- Mean = 200
- Median = 3

Bechdel outliers



**ANY
QUESTIONS?**

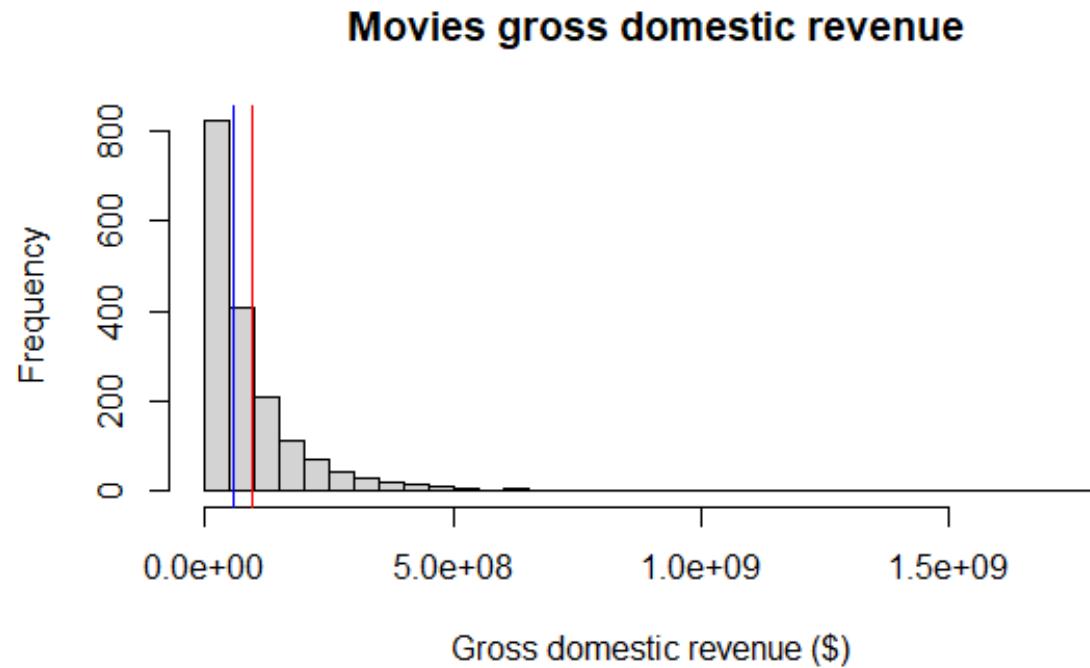


Measures of spread



Characterizing the spread

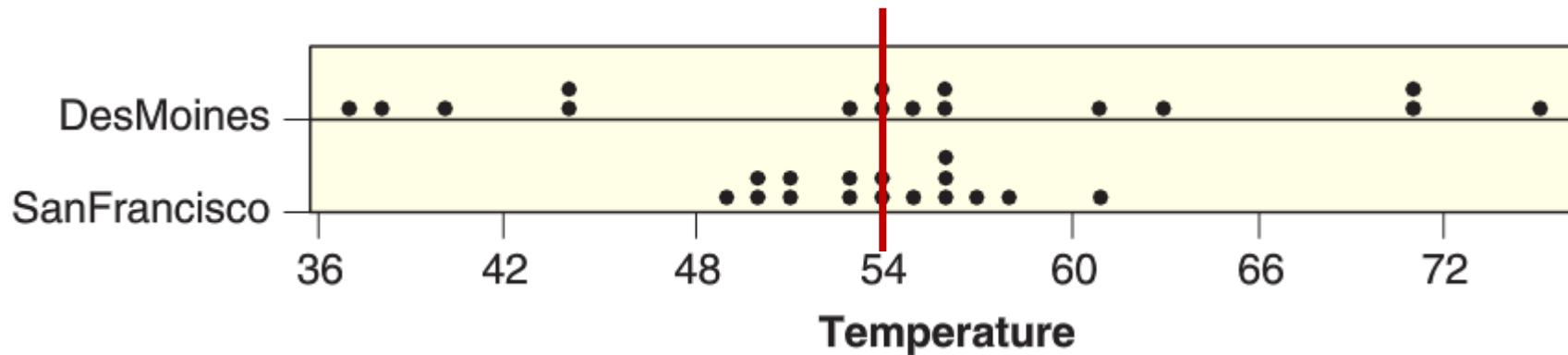
The mean and median are numbers that tell us about the center of a distribution



We can also use numbers to characterize how data is spread

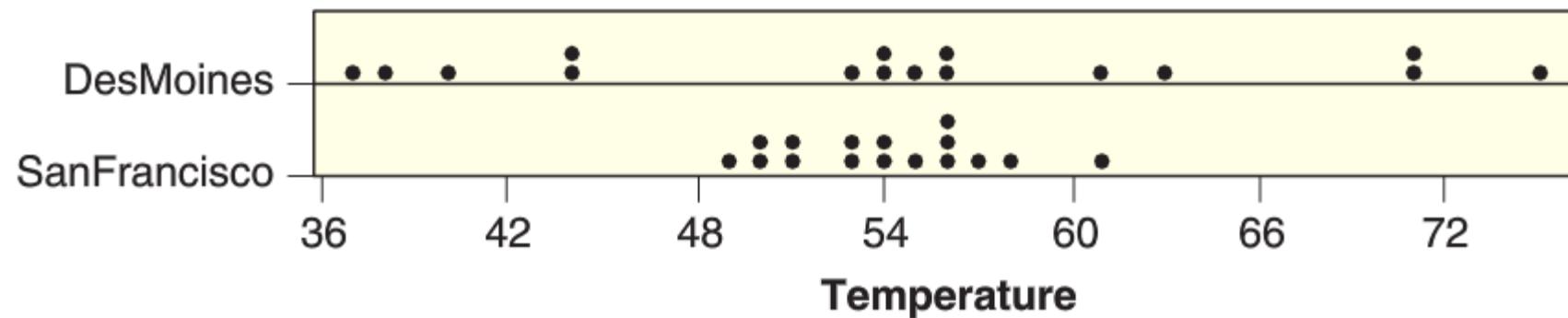
Average monthly temperature: Des Moines vs. San Francisco

Data measured on April 14th from 1997 to 2010:



Mean temperature (°F): Des Moines = 54.49 San Fran = 54.01

Which has the larger standard deviation?



$$s_{DM} = 11.73 \text{ } ^\circ\text{F}$$

$$s_{SF} = 3.38 \text{ } ^\circ\text{F}$$

The standard deviation

The standard deviation can be computed using the following formula:

$$s = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (x_i - \bar{x})^2}$$

Standard deviation measures roughly how far the data are from their average



Example: computing the standard deviation

Suppose we had a sample with $n = 4$ points:

$$x_1 = 8, \quad x_2 = 2, \quad x_3 = 6, \quad x_4 = 4,$$

We can compute the mean using the formula:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{1}{4} \cdot (x_1 + x_2 + x_3 + x_4) = \frac{1}{4} \cdot (8 + 2 + 6 + 4)$$

The standard deviation can be computed using the formula:

$$s = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (x_i - \bar{x})^2} \quad (\text{remember order of operations!})$$

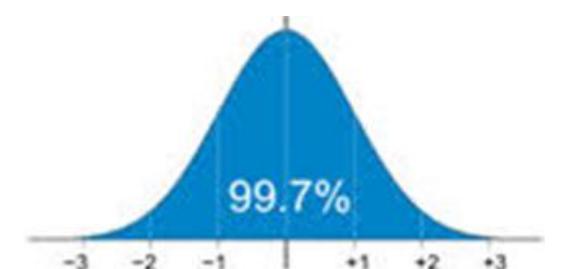
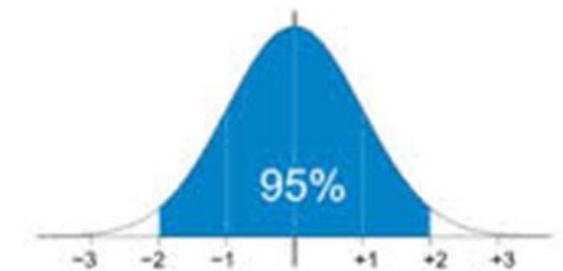
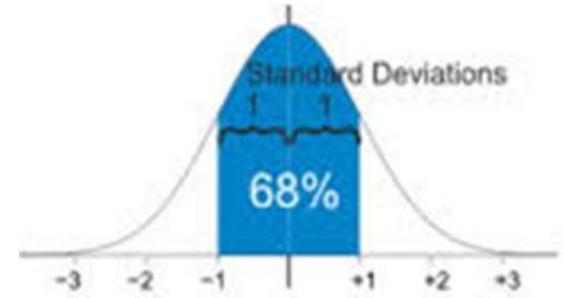
`statistics.stdev(data_list)`

Normally distributed data

No matter what the shape of the distribution, the bulk of the data are in the range "average \pm a few SDs"

If the data is “normally distributed” (bell shaped distribution) than the following holds:

Range	Proportion
Average \pm 1 SDs	68% of the data
Average \pm 2 SDs	95% of the data
Average \pm 3 SDs	99.7% of the data



Chebyshev's Inequality

No matter what the shape of the distribution, the bulk of the data are in the range "average \pm a few SDs"

Chebyshev's Inequality: No matter what the shape of the distribution, the proportion of values in the range "average $\pm z \cdot \text{SDs}$ " is at least $1 - 1/z^2$

Range	Proportion		
Average \pm 2 SDs	at least	$1 - 1/4$	(75%)
Average \pm 3 SDs	at least	$1 - 1/9$	(88.88...%)
Average \pm 4 SDs	at least	$1 - 1/16$	(93.75%)
Average \pm 5 SDs	at least	$1 - 1/25$	(96%)

Let's briefly explore standard deviations in Jupyter!

Z-scores

Standardized units

Items in the world are often measured on very different scales

How can we create a standard scale to quantify unusual/large/impressive values?

Z-scores measure how many SDs a value is from average:

$$\text{z-score}(x_i) = \frac{x_i - \bar{x}}{s}$$

- Negative z: value below average
- Positive z: value above average
- z = 0: value equal to average



Which Accomplishment is most impressive?

LeBron James is a basketball player who had the following statistics in 2011:

- Field goal percentage (FGPct) = 0.510
- Points scored = 2111
- Assists = 554
- Steals = 124



The summary statistics of the NBA in 2011 are given below

$$\text{z-score}(x_i) = \frac{x_i - \bar{x}}{s}$$

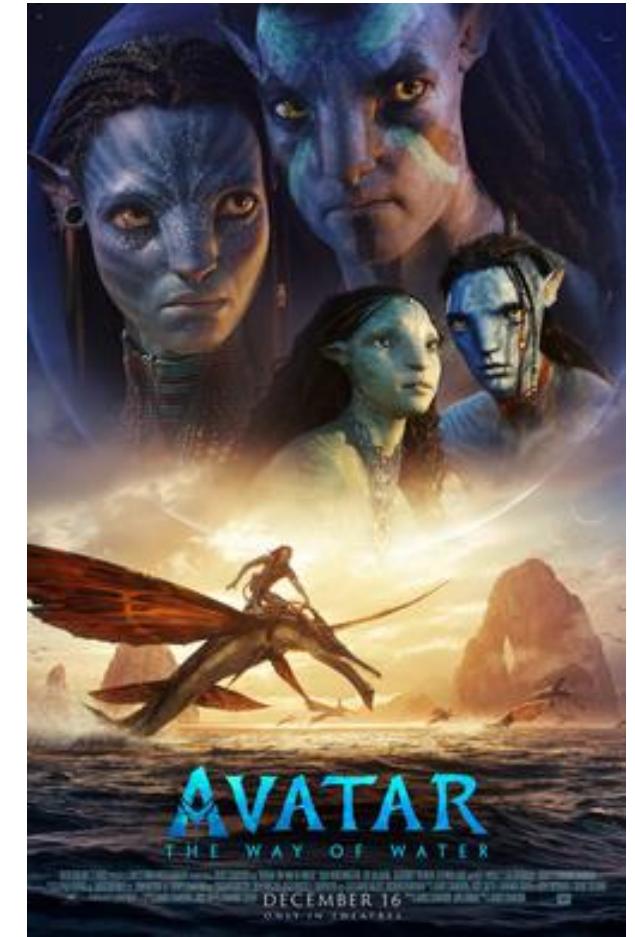
	Mean	Standard Deviation
FGPct	0.464	0.053
Points	994	414
Assists	220	170
Steals	68.2	31.5

Question: Relative to his peers, which statistic is most and least impressive?

Relationships between two quantitative variables

Do movies with larger budgets make more money?

Q: How could we visualize the data to see if this is true?



Scatterplot

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

Each axis represents the value of one variable

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.

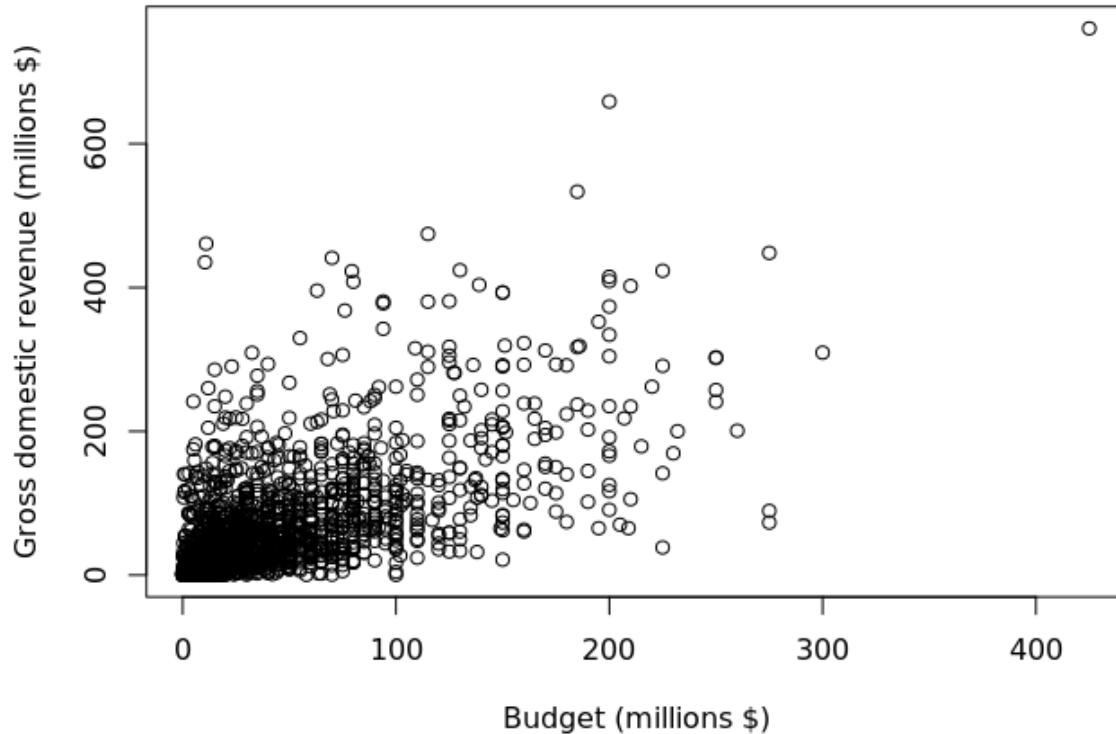
Do movies with larger budgets make more money?

Q: If we want to create a scatter plot to address our question, what variables should we use in our plot?



Relationship movie money spent and made

Bechel movies relationship between buget and revenue



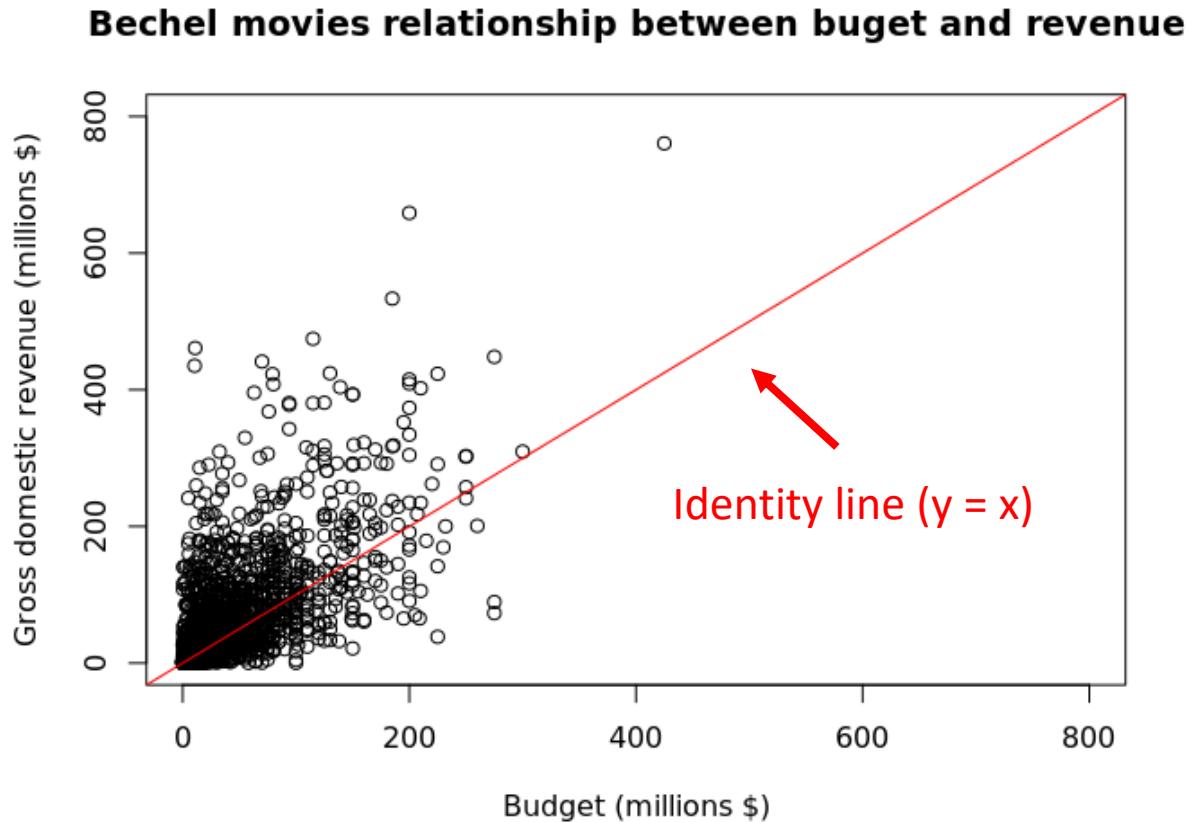
Do movies with larger budgets make more money?

Do most movies make money?

- How could we create a more informative scatter plot of this data?

Matplotlib: `plt.plot(x, y)`

Relationship movie money spent and made



Do movies with larger budgets make more money?

Do most movies make money?

- How could we create a more informative scatter plot of this data?

Matplotlib: `plt.plot(x, y)`

Let's try it in Python!

Questions when looking at scatterplots

Do the points show a clear trend?

Does it go upward or downward?

How much scatter around the trend?

Does the trend seem be linear (follow a line) or is it curved?

Are there any outlier points?

Questions when looking at scatterplots

Do the points show a clear trend?

Does it go upward or downward?

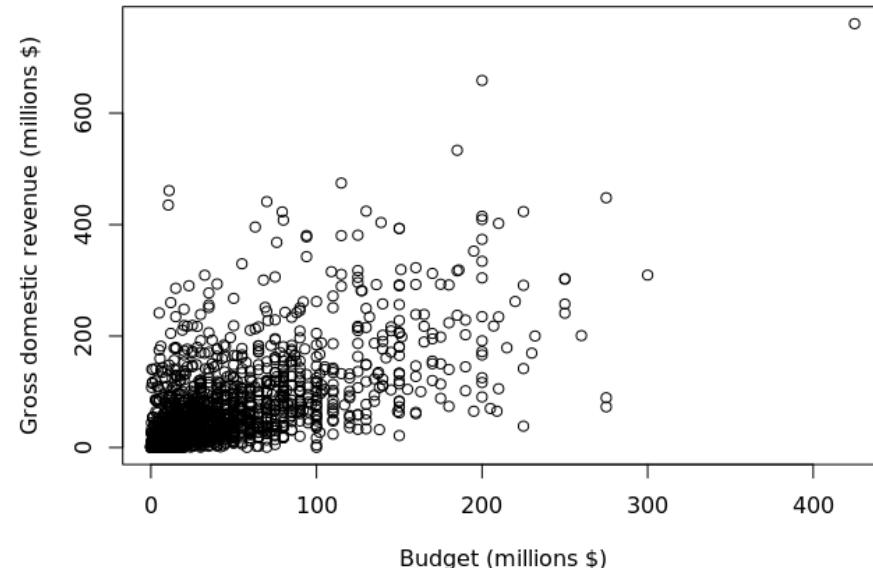
How much scatter around the trend?

Does the trend seem be linear (follow a line) or is it curved?

Are there any outlier points?

Budget and revenue

Bechel movies relationship between buget and revenue



Positive, negative, no correlation

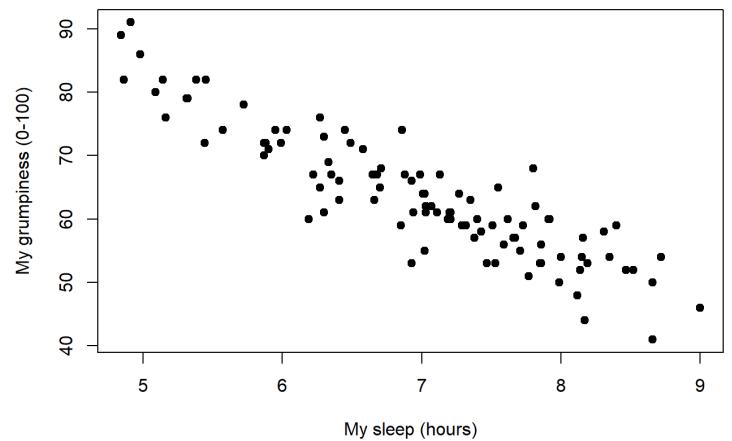
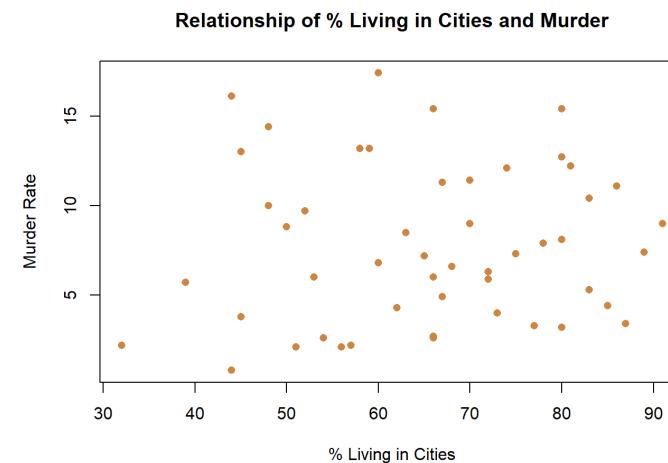
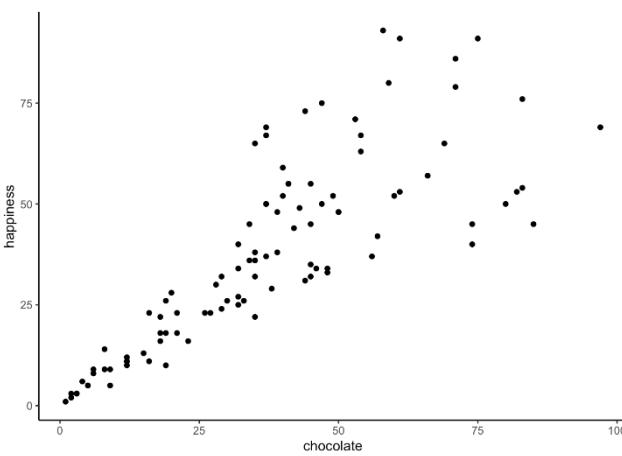
Do the points show a clear trend?

Does it go upward or downward?

How much scatter around the trend?

Does the trend seem be linear (follow a line) or is it curved?

Are there any outlier points?



The correlation coefficient

The **correlation** is measure of the strength and direction of a linear association between two variables

$$r = \frac{1}{(n - 1)} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right)$$

```
statistics.correlation(x, y)
```

Properties of the correlation

Correlation is always between -1 and 1: $-1 \leq r \leq 1$

The sign of r indicates the direction of the association

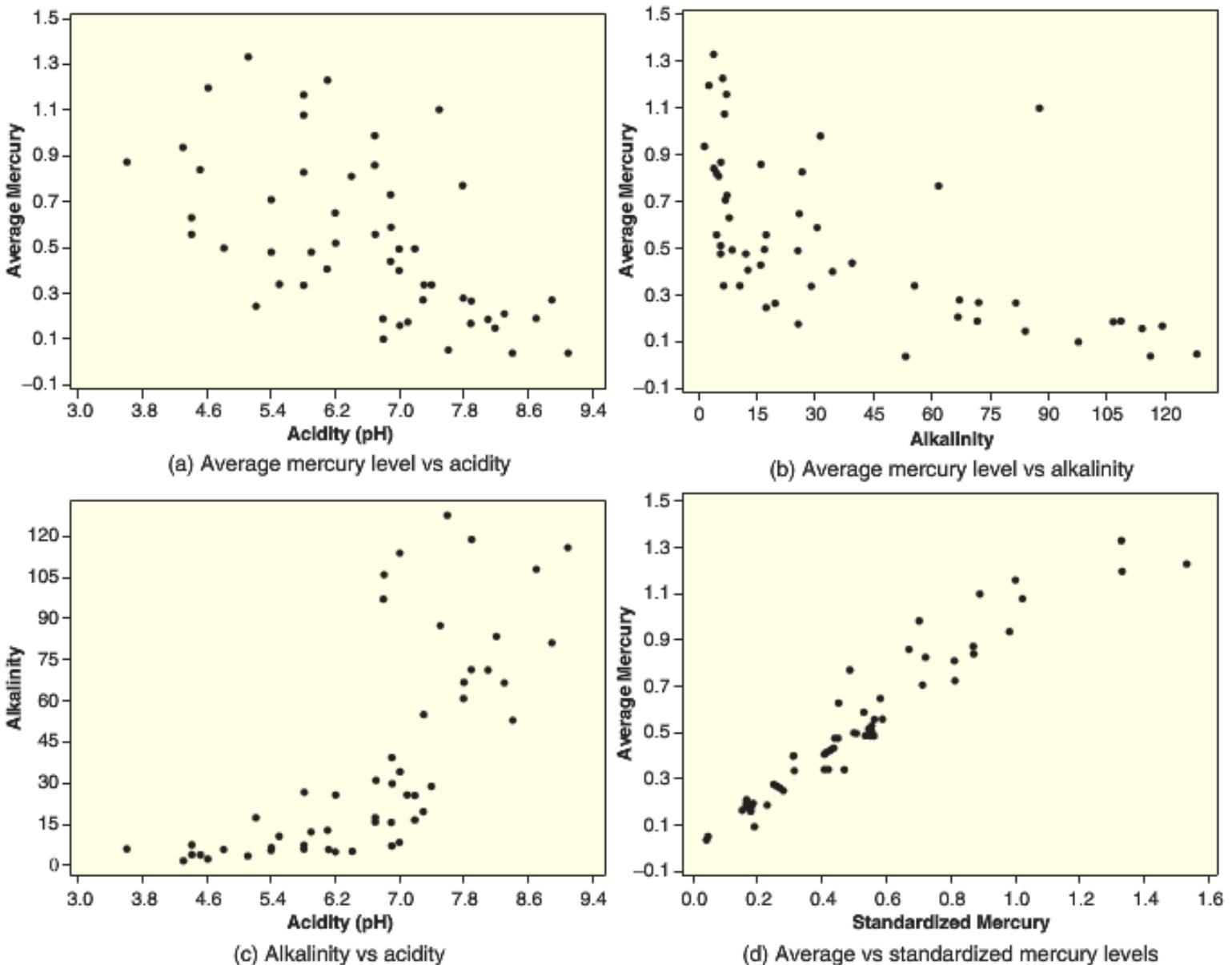
Values close to ± 1 show strong linear relationships, values close to 0 show no linear relationship

Correlation is symmetric: $r = \text{cor}(x, y) = \text{cor}(y, x)$

$$r = \frac{1}{(n-1)} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right)$$

Florida lakes

Correlation game



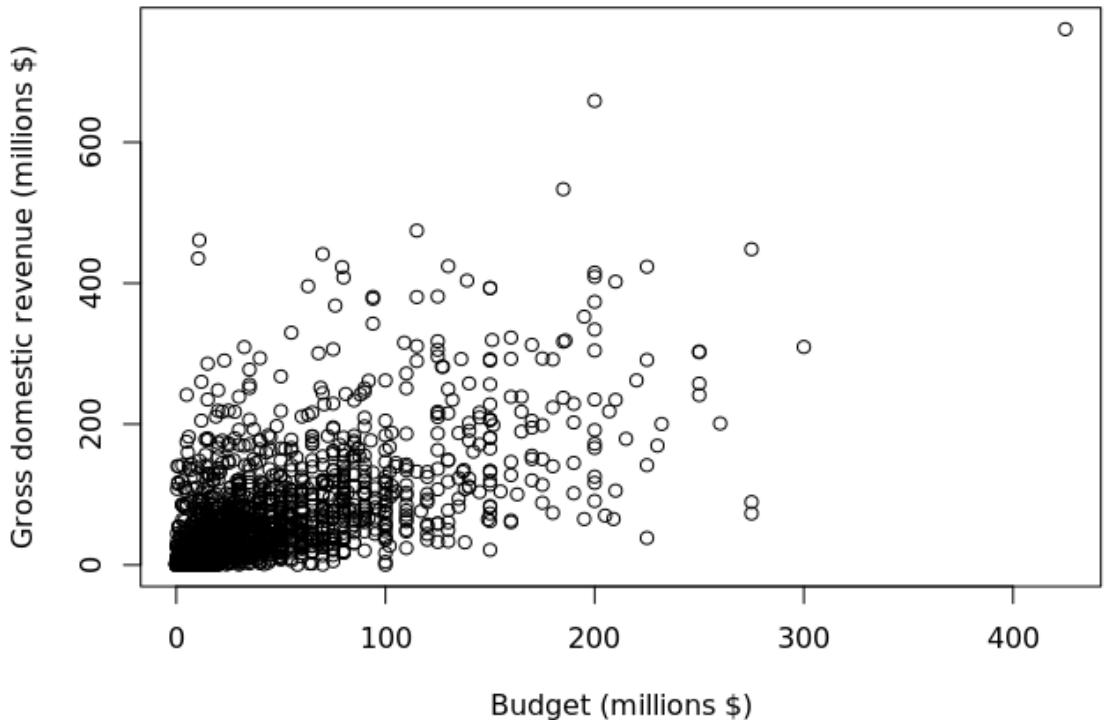
Movie budget and revenue correlation?

The **correlation** is measure of the strength and direction of a linear association between two variables

r = ?

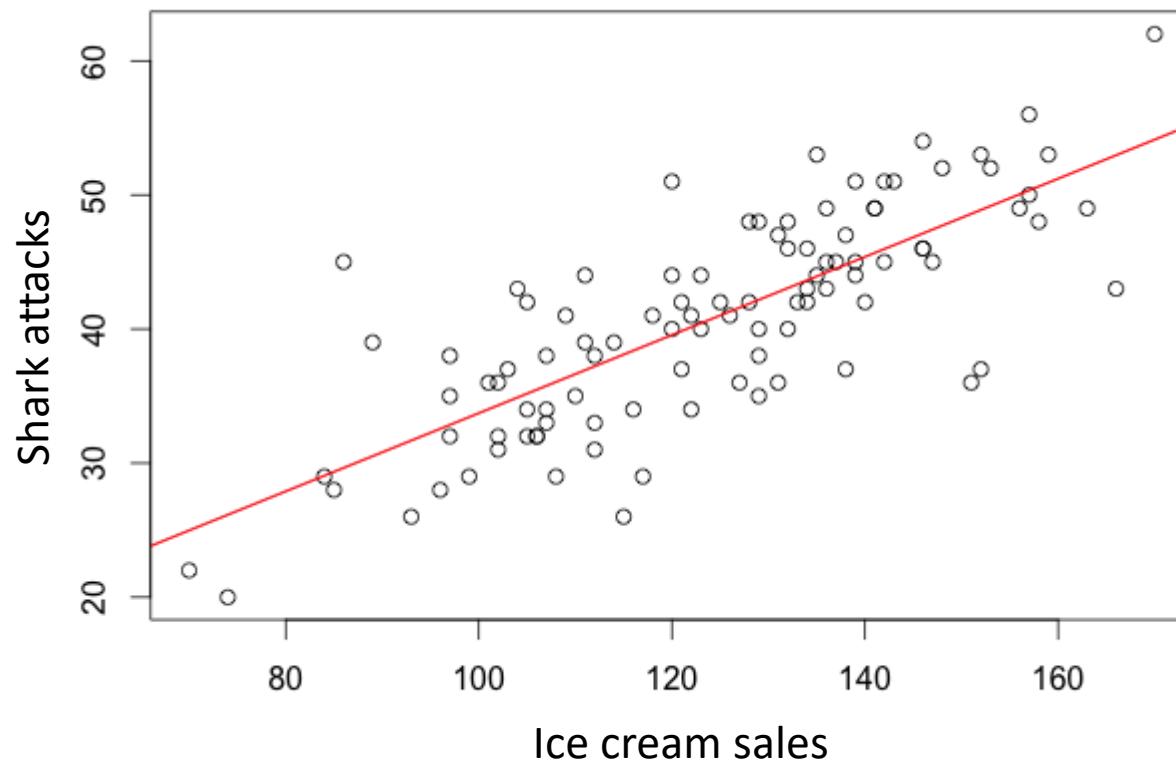
Let's calculate the correlation in Python!

Bechel movies relationship between buget and revenue



Correlation caution #1

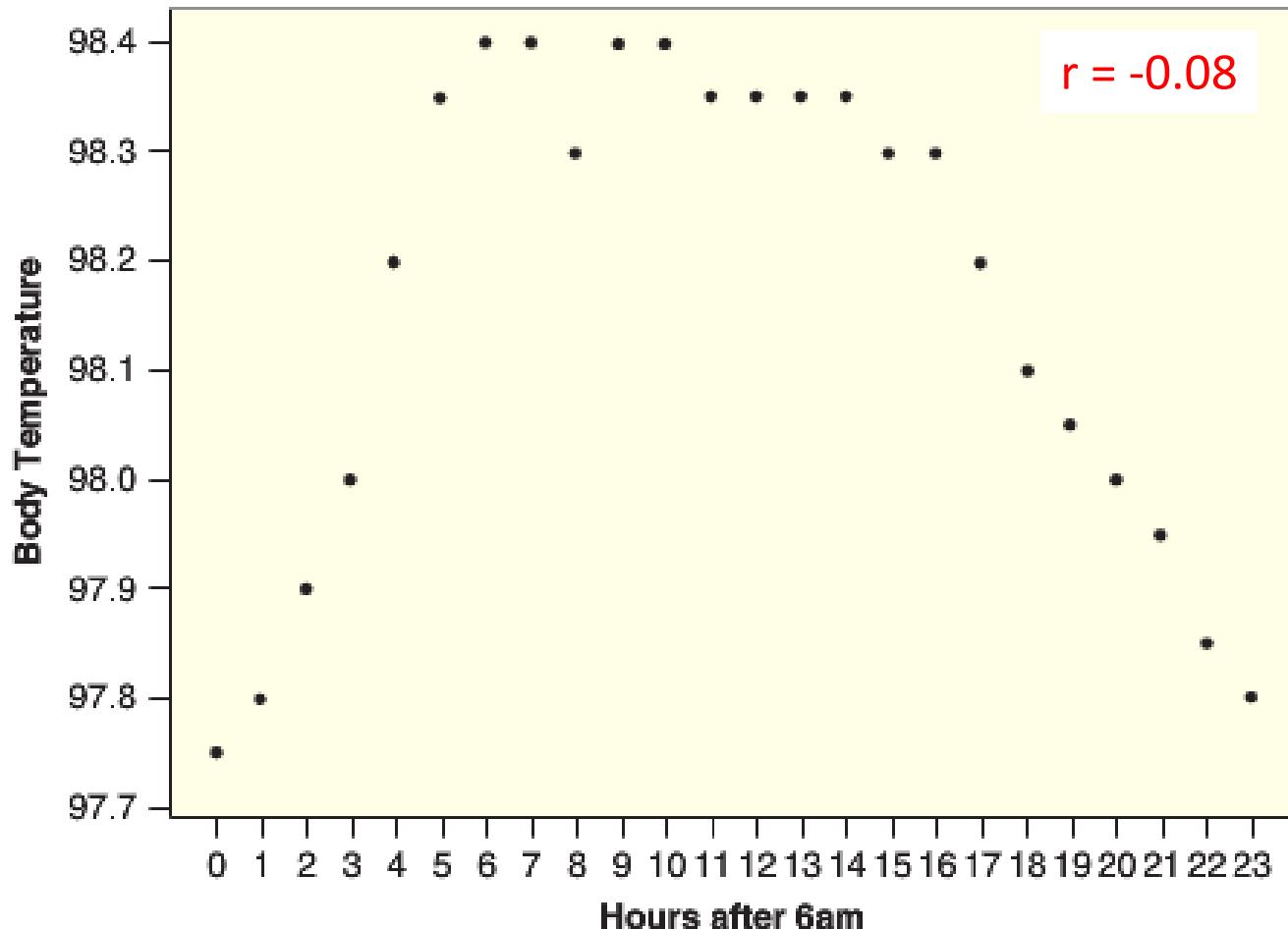
A strong positive or negative correlation does not (necessarily) imply a cause and effect relationship between two variables



Correlation caution #2

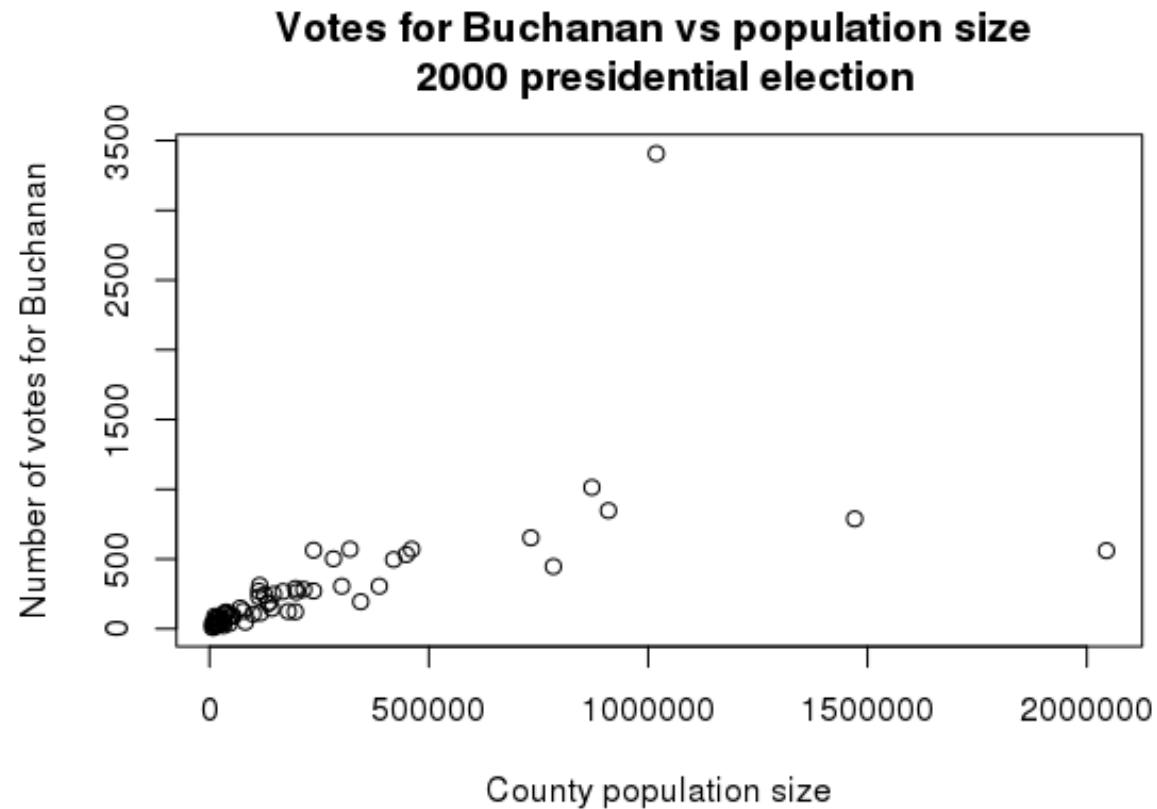
A correlation near zero does not (necessarily) mean that two variables are not associated. Correlation only measures the strength of a linear relationship.

Body temperature as a function of time of the day



Correlation caution #3

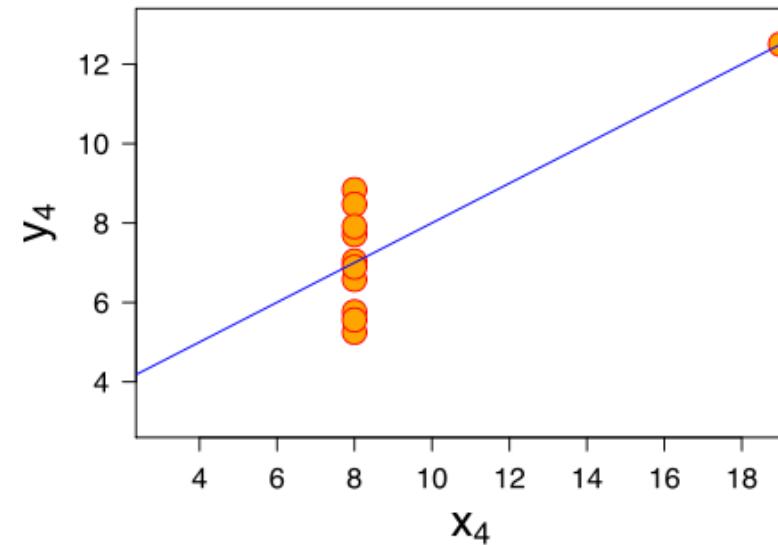
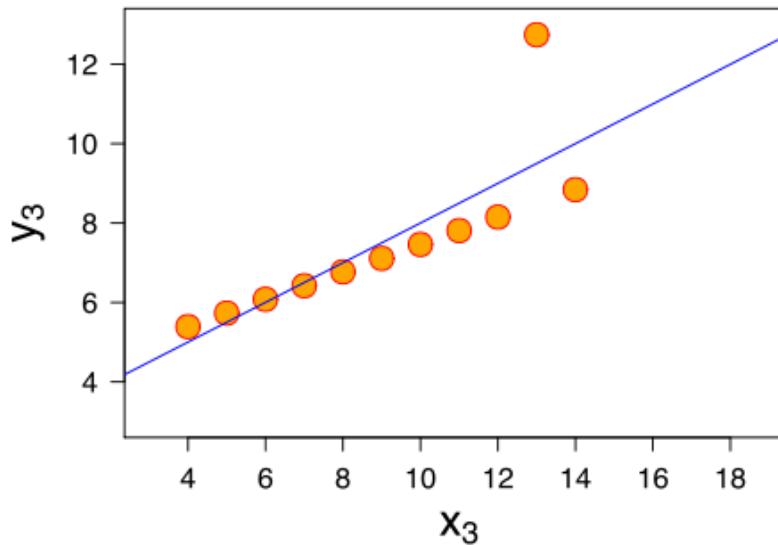
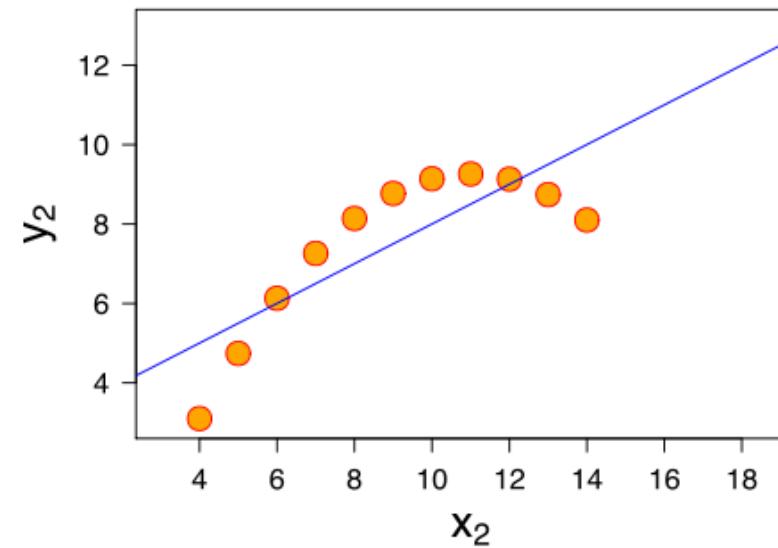
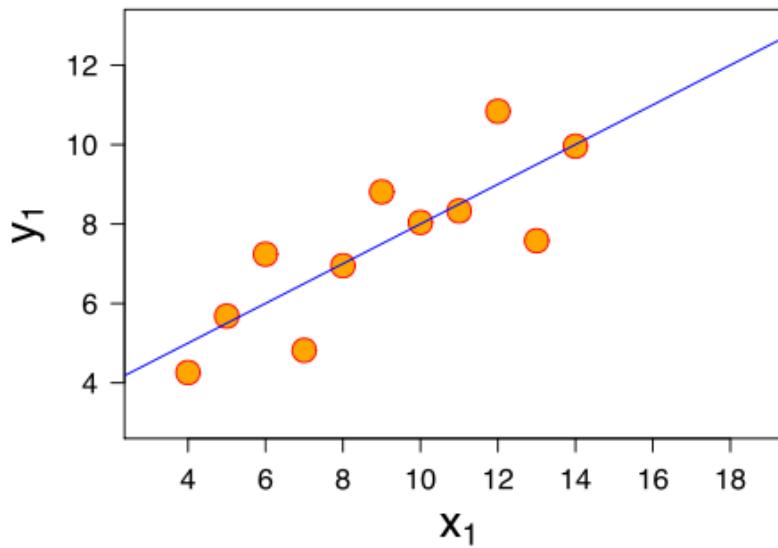
Correlation can be heavily influenced by outliers. Always plot your data!



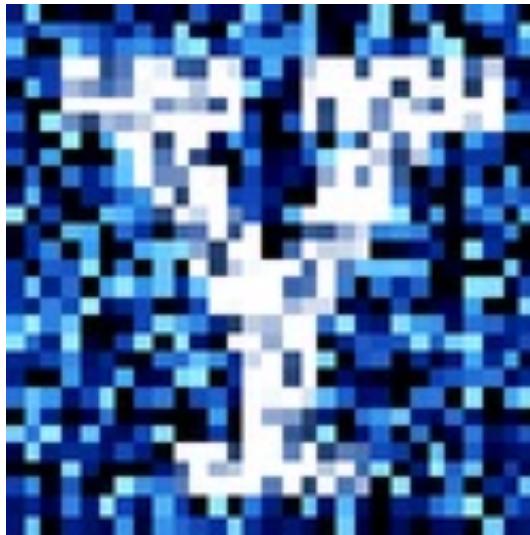
With Palm Beach
 $r = 0.61$

Without Palm Beach
 $r = .78$

Anscombe's quartet ($r = 0.81$)



YData: Introduction to Data Science



Class 05: Descriptive statistics and plots continued

Overview

Quick review of categorical data

Statistics and data visualizations continued:

- Quantitative data: mean median and histograms
- Measures of spread (standard deviation)
- z-scores
- Scatter plots and correlation



Announcement: Homework 2

Homework 2 has been posted!

```
import YData
```

```
YData.download.download_homework(2)
```

It is due on Gradescope on Sunday September 15th at 11pm

- Be sure to mark each question on Gradescope!

Notes:

- There is an ~18 page reading from the book "Data and the American Dream" that you need to do, so I recommend you get started on this soon.

Very quick review

Last class we spoke about comparisons and Booleans, and more string methods

- `3 < 5` # TRUE
- `"123".isnumeric()`
- `f"The number {my_num} in a string"`



We discussed structure data

Categorical data main statistic:

- Proportion = $\frac{\text{number in category}}{\text{total number}}$

`bechdel.count("PASS")/len(bechdel)`

Categorical Variable

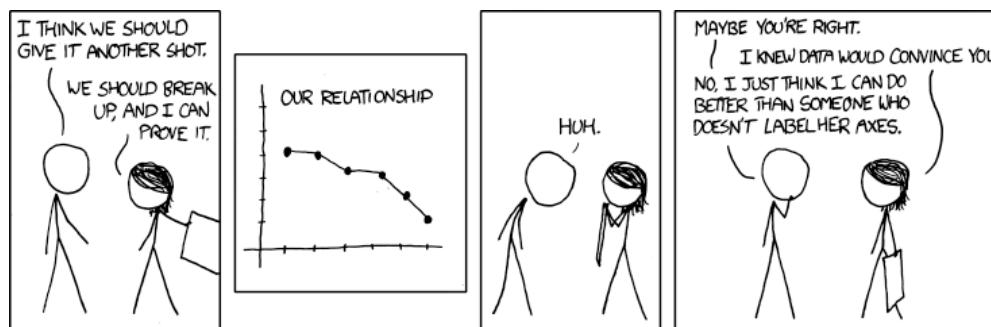
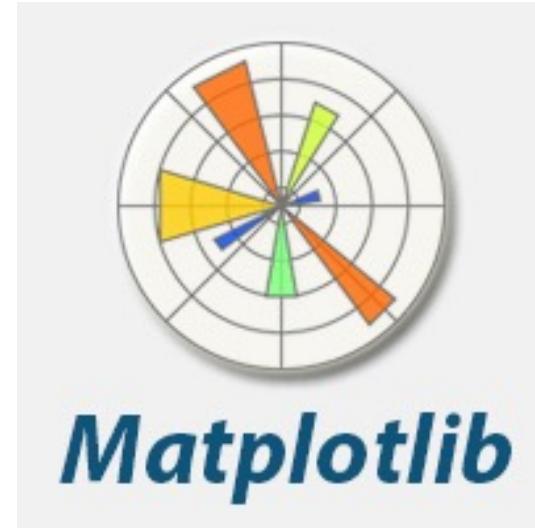
title	clean_test	finaly	budget	domgross	budget_2013	domgross_2013
21 & Over	notalk	FAIL	13000000	25682380.0	13000000	25682380.0
Dredd 3D	ok	PASS	45000000	13414714.0	45658735	13611086.0
12 Years a Slave	notalk	FAIL	20000000	53107035.0	20000000	53107035.0
2 Guns	notalk	FAIL	61000000	75612460.0	61000000	75612460.0
42	men	FAIL	40000000	95020213.0	40000000	95020213.0

Quantitative Variable

Very quick review

Visualize categorical data

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



If you don't want exes, label your axes!

Quantitative data

Quantitative data

To explore quantitative data, let's look at how much revenue each movie made in the United States in (2013) inflation adjusted dollars

- [domgross_2013](#)

Quantitative Variable

	title	clean_test	binary	budget	domgross	budget_2013	domgross_2013
	21 & Over	notalk	FAIL	13000000	25682380.0	13000000	25682380.0
	Dredd 3D	ok	PASS	45000000	13414714.0	45658735	13611086.0
	12 Years a Slave	notalk	FAIL	20000000	53107035.0	20000000	53107035.0
	2 Guns	notalk	FAIL	61000000	75612460.0	61000000	75612460.0
	42	men	FAIL	40000000	95020213.0	40000000	95020213.0

Visualizing quantitative data: histograms

Movie US revenue (in millions of dollars):

- 25.68, 13.61, 53.11, 236.84, ...

To create a histogram we create a set of intervals

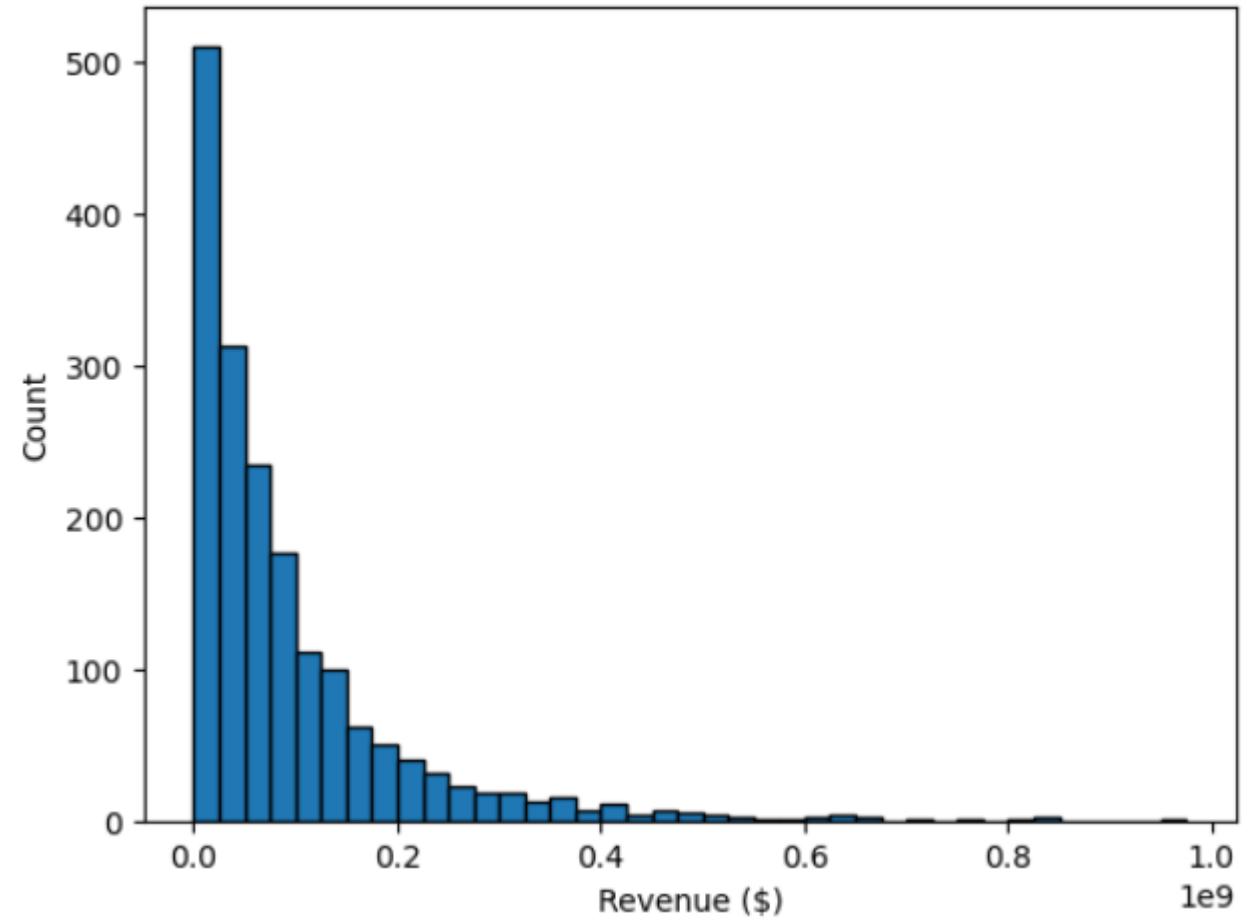
- 0-25, 25-50, 50-75, ... 200-250, 250-300

We count the number of points that fall in each interval

We create a bar chart where the height of the bars is the counts in each bin

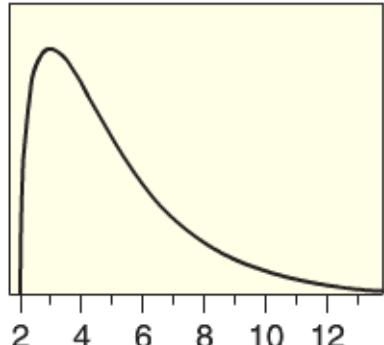
Histograms – movie US revenue

Domgross range	Frequency Count
(0 – 25]	510
(25 – 50]	312
(50 – 75]	234
(75 – 100]	176
(100 – 125]	111
(125 – 150]	99
(150 – 175]	62
(175 – 200]	51
(200 – 225]	40
(225 – 250]	32

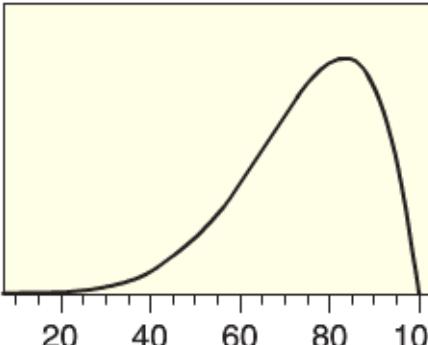


Matplotlib: `plt.hist(data)`

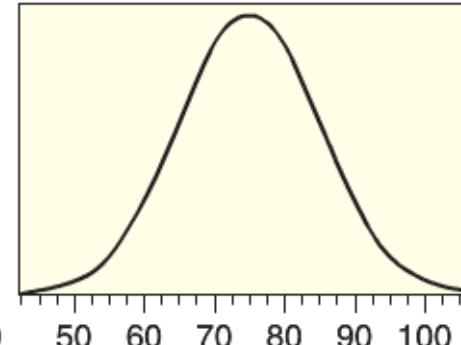
Common shapes of data distributions



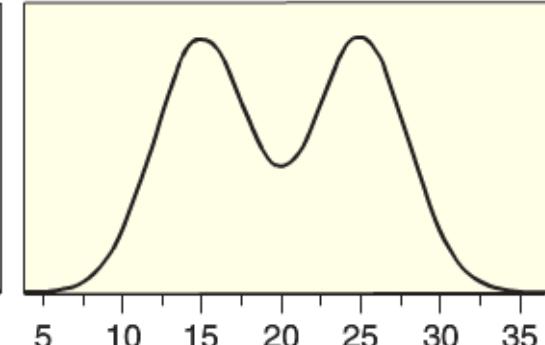
(a) Skewed to the right



(b) Skewed to the left



(c) Symmetric and bell-shaped

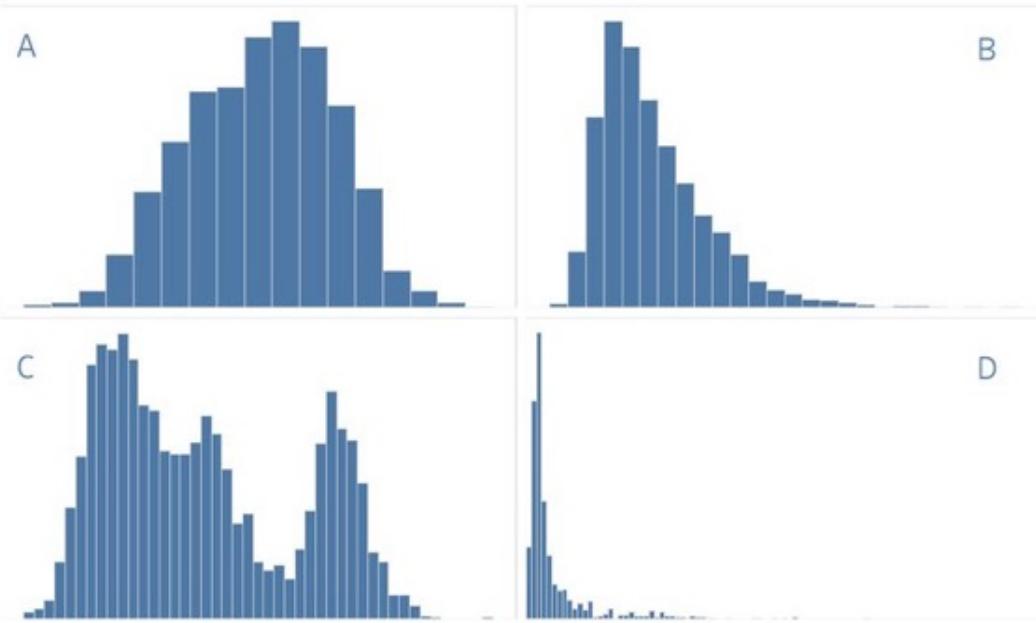


(d) Symmetric but not bell-shaped



Neat facts – the average NFL player is:

- 1. **Age:** Is about 25 years old
- 2. **Height:** Is just over 6'2" in height
- 3. **Weight:** Weighs a little more than 244lbs
- 4. **Salary:** Makes slightly less than \$1.5M in salary per year

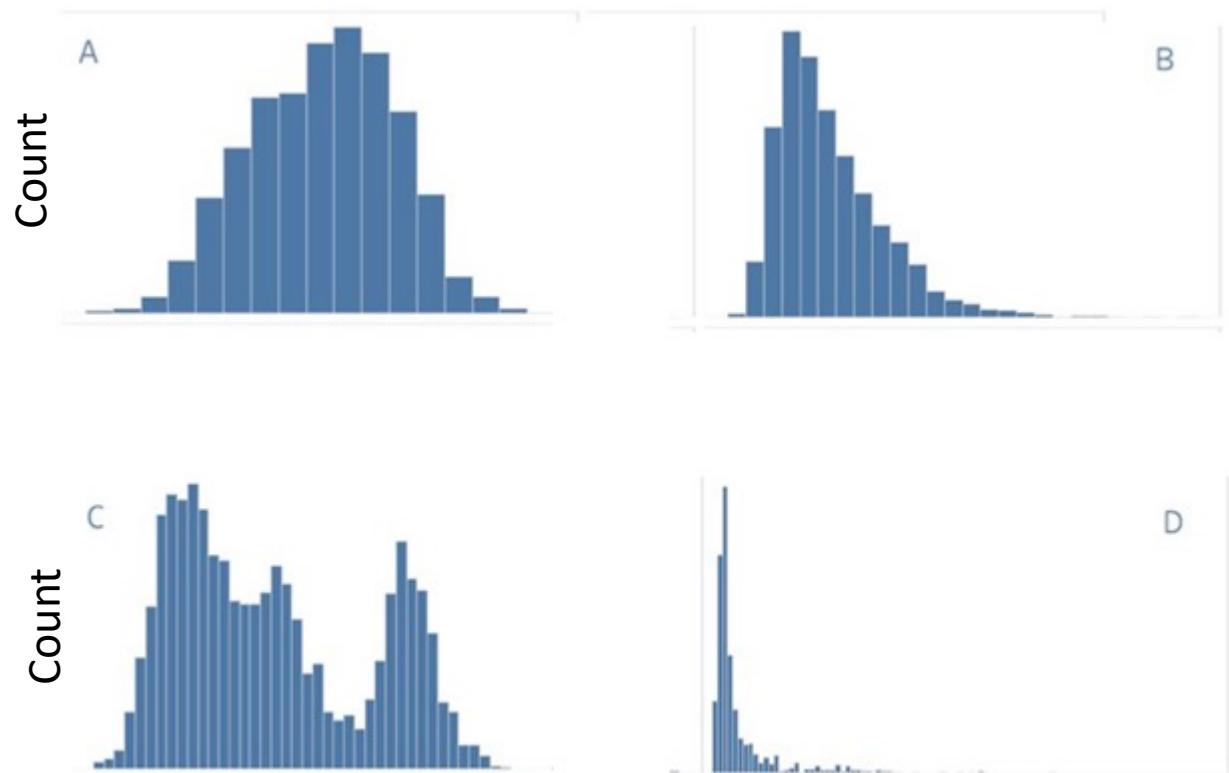
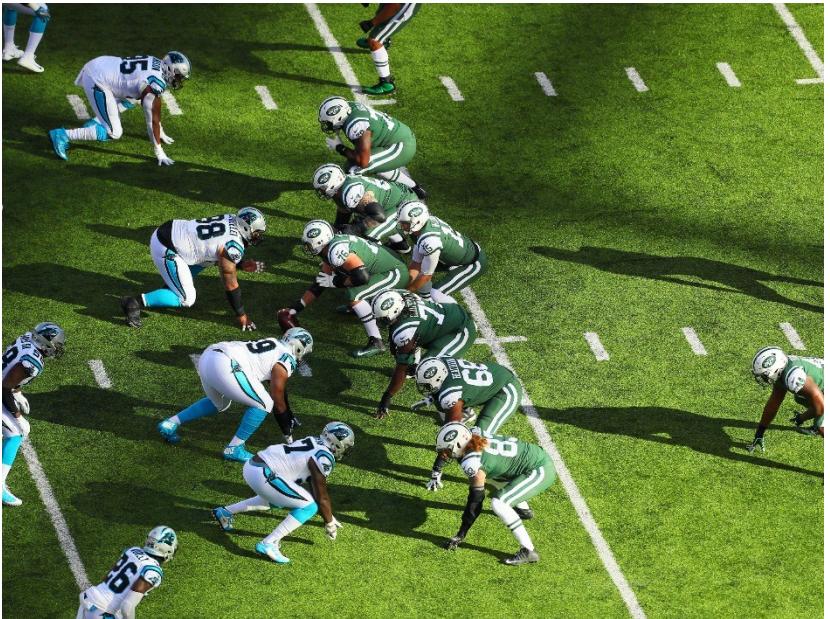


Question: Can you tell which histogram goes with which trait?

Task is to add the labels: **Age, Height, Weight, and Salary**

- Hint: There are a wide range of positions in football that have very different roles
 - E.g., placekickers only play for small fractions of the game, while quarterbacks are essentially to a team's success

First: what is the label for the y-axis?



Always label your axes!

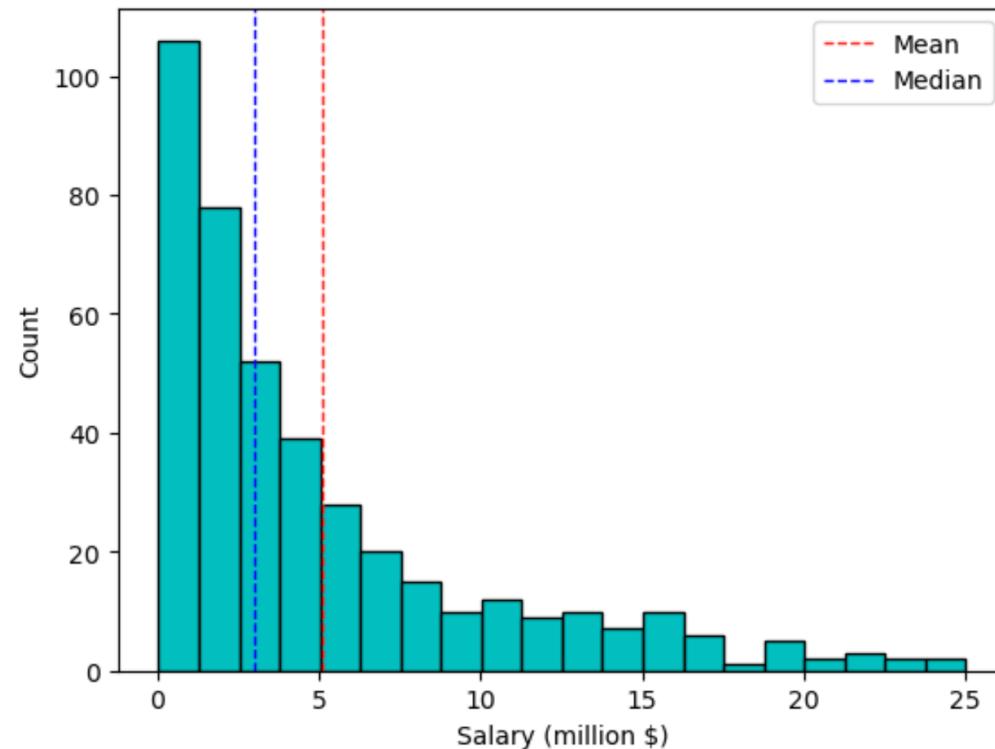
```
plt.ylabel("y label")
```

```
plt.xlabel("x label")
```

```
plt.title("my title")
```

Quantitative data: statistics for central tendency

Two statistics for measuring the “central value” of a sample of quantitative data are the ***mean*** and the ***median***



The mean

$$\text{Mean} = \frac{\text{Sum of all data values}}{\text{Number of data values}}$$

$$\text{Mean} = \frac{x_1 + x_2 + x_3 + \dots + x_n}{n} = \frac{1}{n} \sum_{i=1}^n x_i$$

```
import statistics  
statistics.mean(data_list)
```

The median

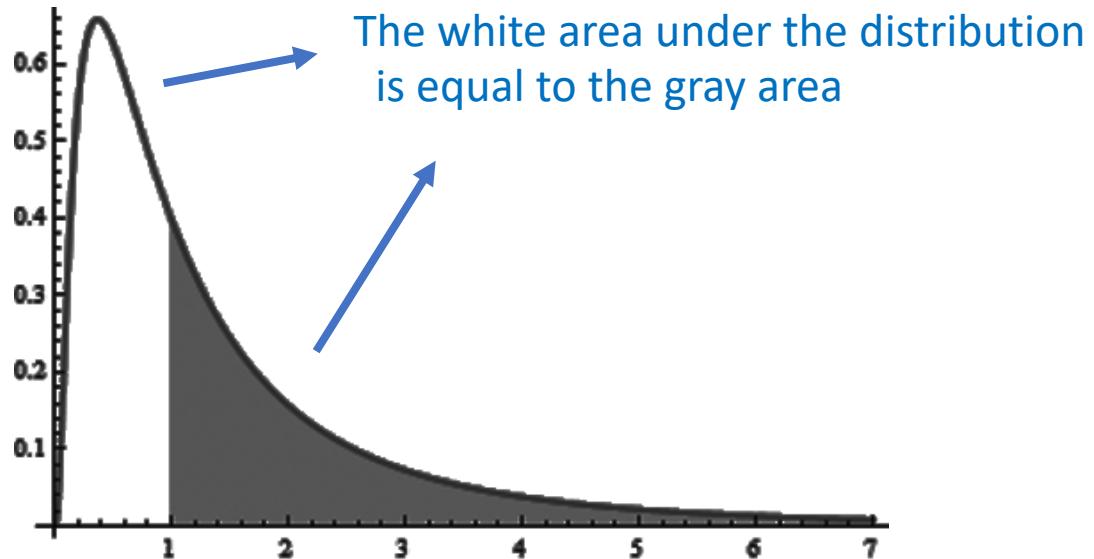
The **median** is a value that splits the data in half

- i.e., half the values in the data are smaller than the median and half are larger

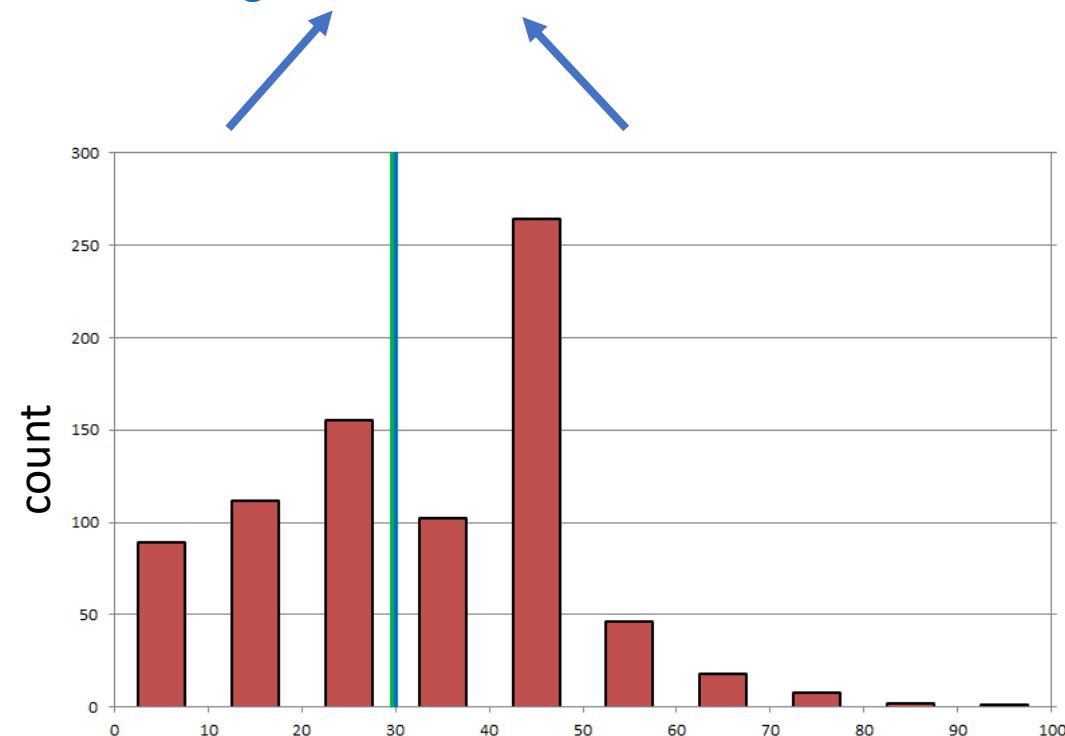
To calculate the median for a data sample of size n , sort the data and then:

- If n is **odd**: The middle value of the sorted data
- If n is **even**: The average of the middle two values of the sorted data

The median



The sum of the heights of the bars on the left is equal to the sum of the heights of the bars on the right

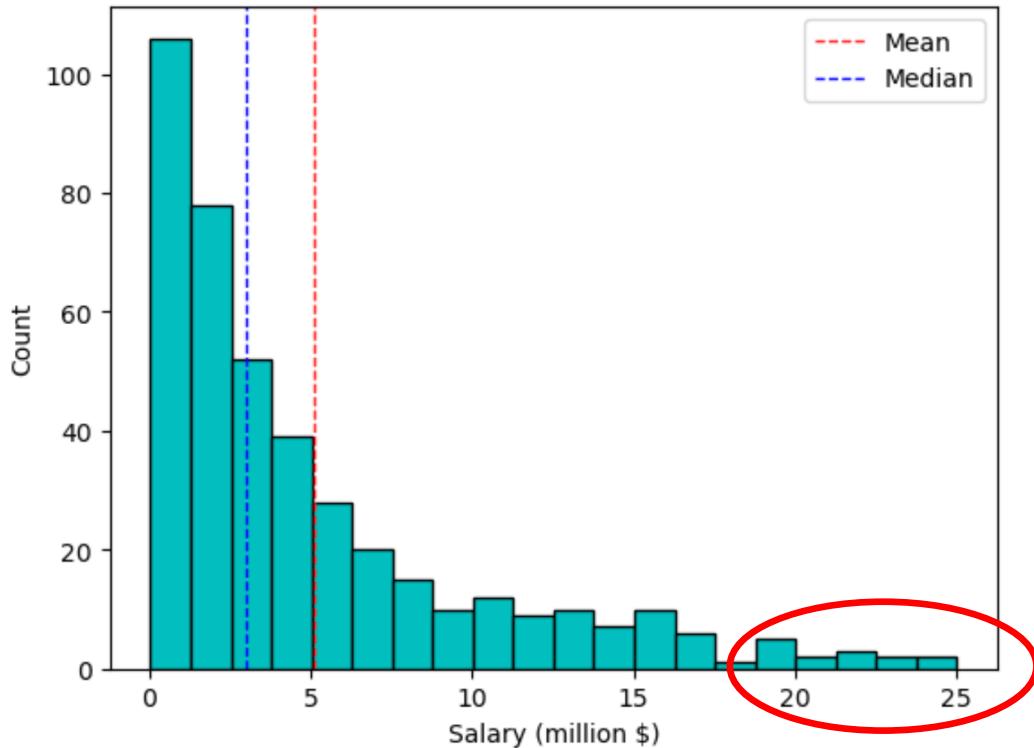


```
import statistics  
statistics.median(data_list)
```

Let's explore this in Jupyter!

Outliers

An **outlier** is an observed value that is notably distinct from the other values in a dataset by being much smaller or larger than the rest of the data.



Outliers can potentially have a large influence on the statistics you calculate

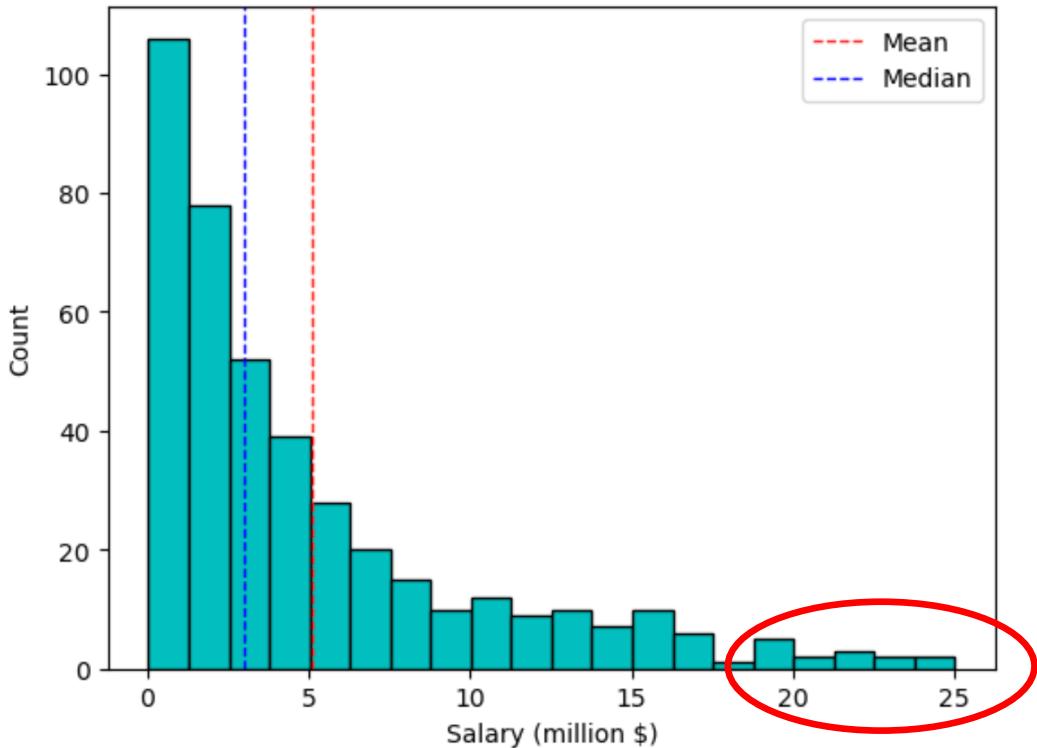
One should examine outliers to understand what is causing them

- If there are due to an error, remove them
- Otherwise, need to think about how to treat them
 - Could be interesting phenomenon
 - Could restrict data to a particular range of values
 - Etc.

Bechdel outliers



Outliers' impact on mean and median



The median is *resistant* to outliers

- i.e., not affected much by outliers

The mean is not resistant to outliers

What is the mean and median of the data: 1, 2, 3, 4, 990?

- Mean = 200
- Median = 3

**ANY
QUESTIONS?**

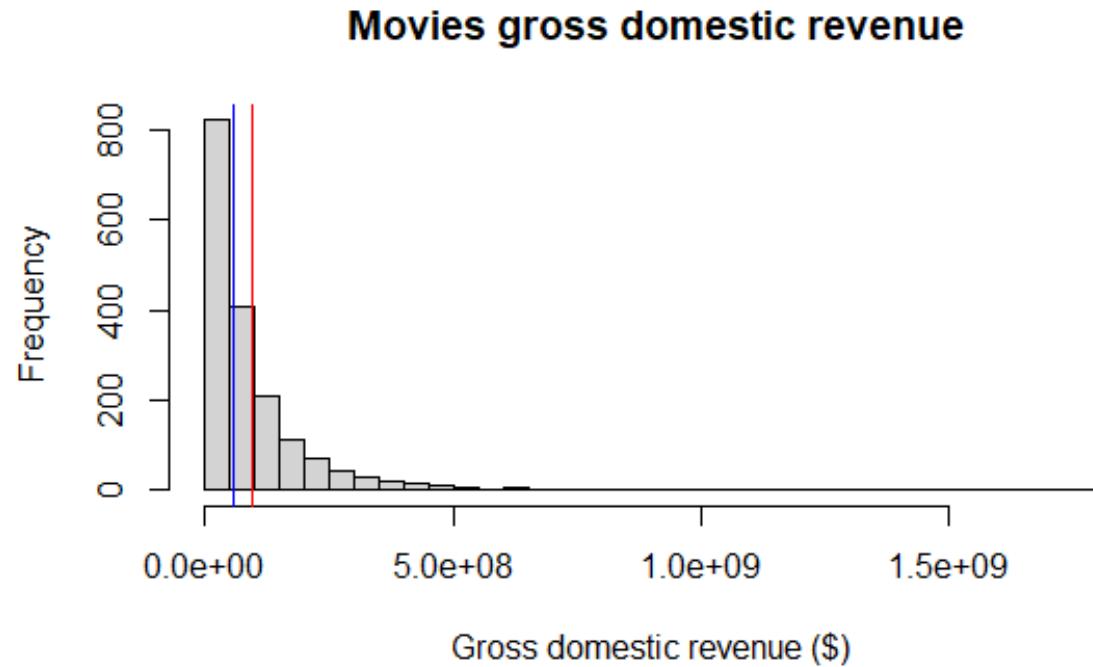


Measures of spread



Characterizing the spread

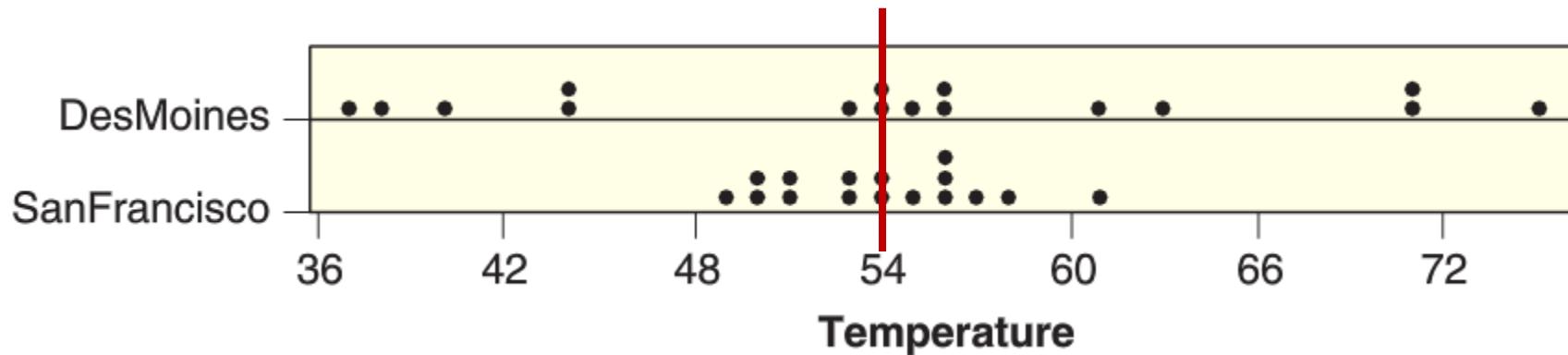
The mean and median are numbers that tell us about the center of a distribution



We can also use numbers to characterize how data is spread

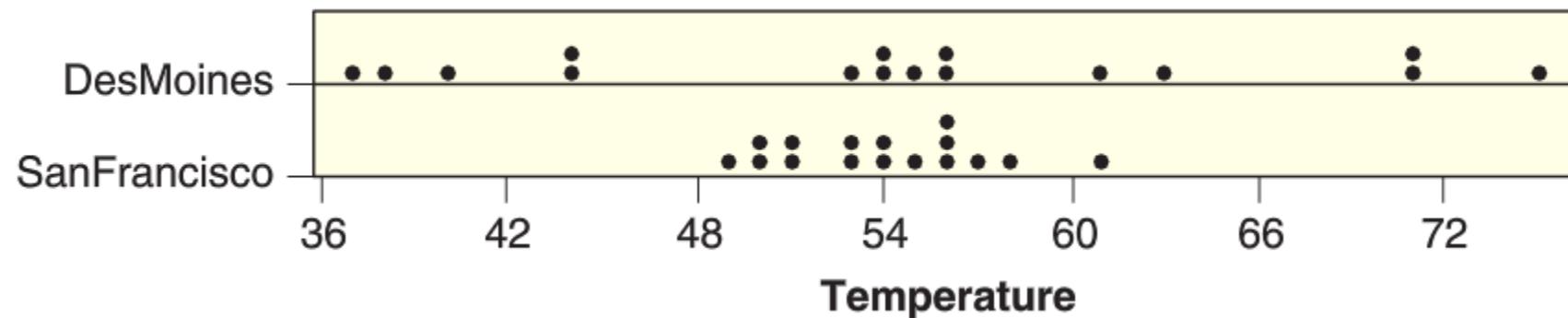
Average monthly temperature: Des Moines vs. San Francisco

Data measured on April 14th from 1997 to 2010:



Mean temperature (°F): Des Moines = 54.49 San Fran = 54.01

Which has the larger standard deviation?



$$s_{DM} = 11.73 \text{ } ^\circ\text{F}$$

$$s_{SF} = 3.38 \text{ } ^\circ\text{F}$$

The standard deviation

The standard deviation can be computed using the following formula:

$$s = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (x_i - \bar{x})^2}$$

Standard deviation measures roughly how far the data are from their average



Example: computing the standard deviation

Suppose we had a sample with $n = 4$ points:

$$x_1 = 8, \quad x_2 = 2, \quad x_3 = 6, \quad x_4 = 4,$$

We can compute the mean using the formula:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{1}{4} \cdot (x_1 + x_2 + x_3 + x_4) = \frac{1}{4} \cdot (8 + 2 + 6 + 4)$$

The standard deviation can be computed using the formula:

$$s = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (x_i - \bar{x})^2} \quad (\text{remember order of operations!})$$

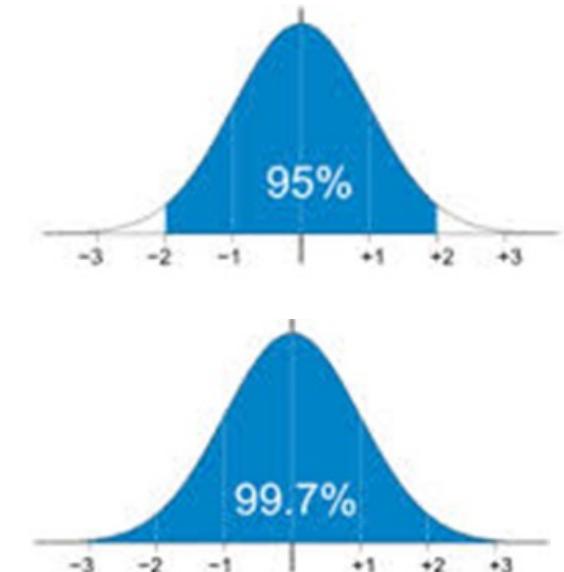
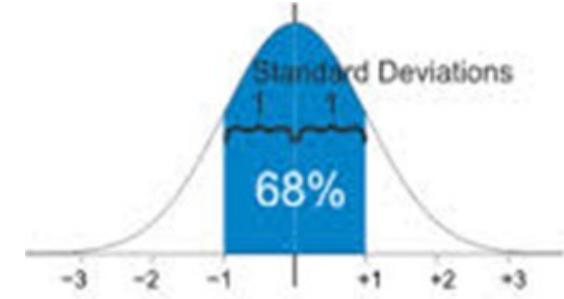
`statistics.stdev(data_list)`

Normally distributed data

The bulk of the data are in the range "average \pm a few SDs"

If the data is “normally distributed” (bell shaped distribution) than the following holds:

Range	Proportion
Average \pm 1 SDs	68% of the data
Average \pm 2 SDs	95% of the data
Average \pm 3 SDs	99.7% of the data



Chebyshev's Inequality

The bulk of the data are in the range "average \pm a few SDs"

Chebyshev's Inequality: No matter what the shape of the distribution, the proportion of values in the range "average $\pm z \cdot \text{SDs}$ " is at least $1 - 1/z^2$

Range	Proportion		
Average \pm 2 SDs	at least	$1 - 1/4$	(75%)
Average \pm 3 SDs	at least	$1 - 1/9$	(88.88...%)
Average \pm 4 SDs	at least	$1 - 1/16$	(93.75%)
Average \pm 5 SDs	at least	$1 - 1/25$	(96%)

Let's briefly explore standard deviations in Jupyter!

Z-scores

Standardized units

Items in the world are often measured on very different scales

How can we create a standard scale to quantify unusual/large/impressive values?

Z-scores measure how many SDs a value is from average:

$$\text{z-score}(x_i) = \frac{x_i - \bar{x}}{s}$$

- Negative z: value below average
- Positive z: value above average
- z = 0: value equal to average



Which Accomplishment is most impressive?

LeBron James is a basketball player who had the following statistics in 2011:

- Field goal percentage (FGPct) = 0.510
- Points scored = 2111
- Assists = 554
- Steals = 124



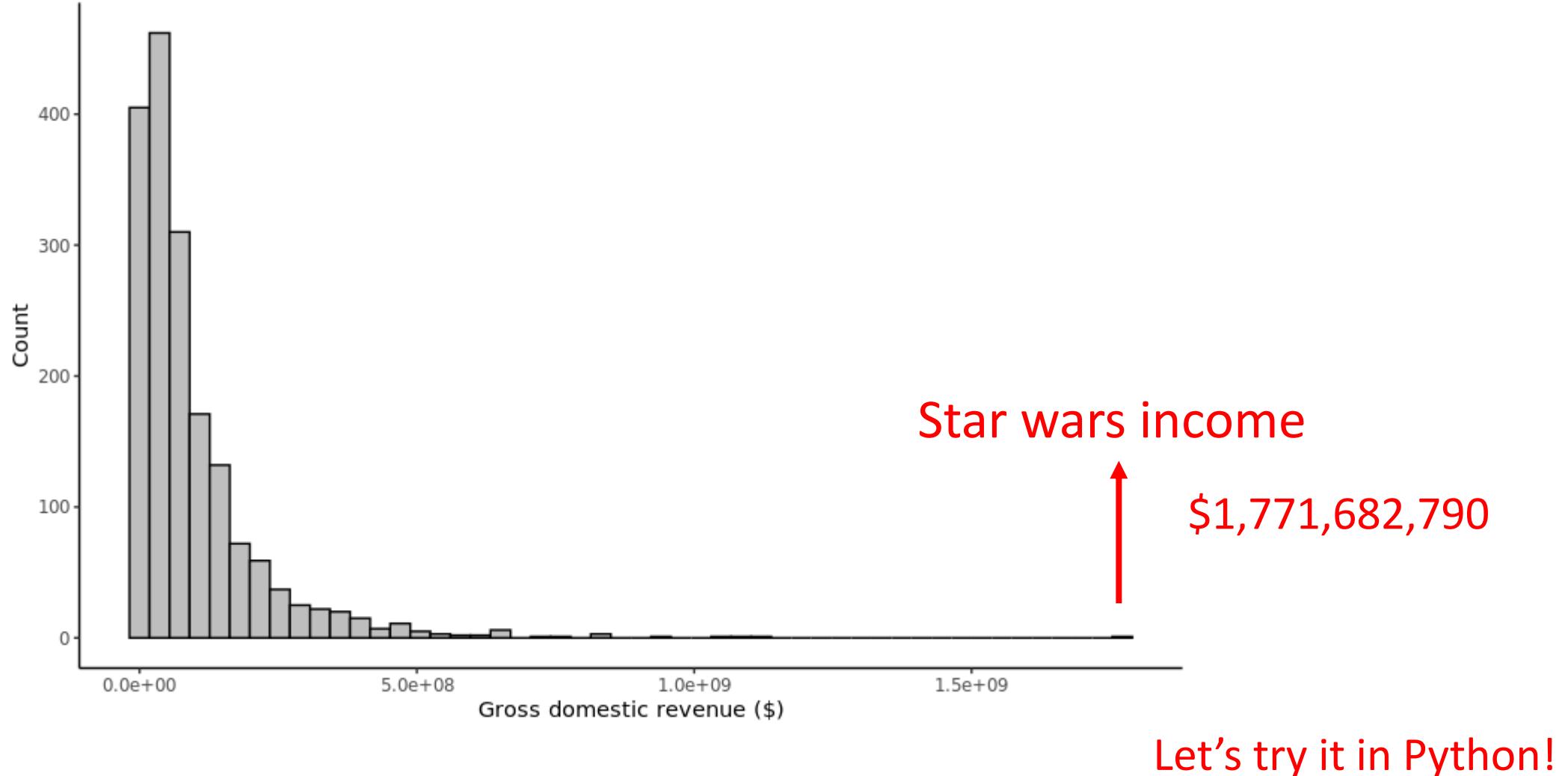
The summary statistics of the NBA in 2011 are given below

$$\text{z-score}(x_i) = \frac{x_i - \bar{x}}{s}$$

	Mean	Standard Deviation
FGPct	0.464	0.053
Points	994	414
Assists	220	170
Steals	68.2	31.5

Question: Relative to his peers, which statistic is most and least impressive?

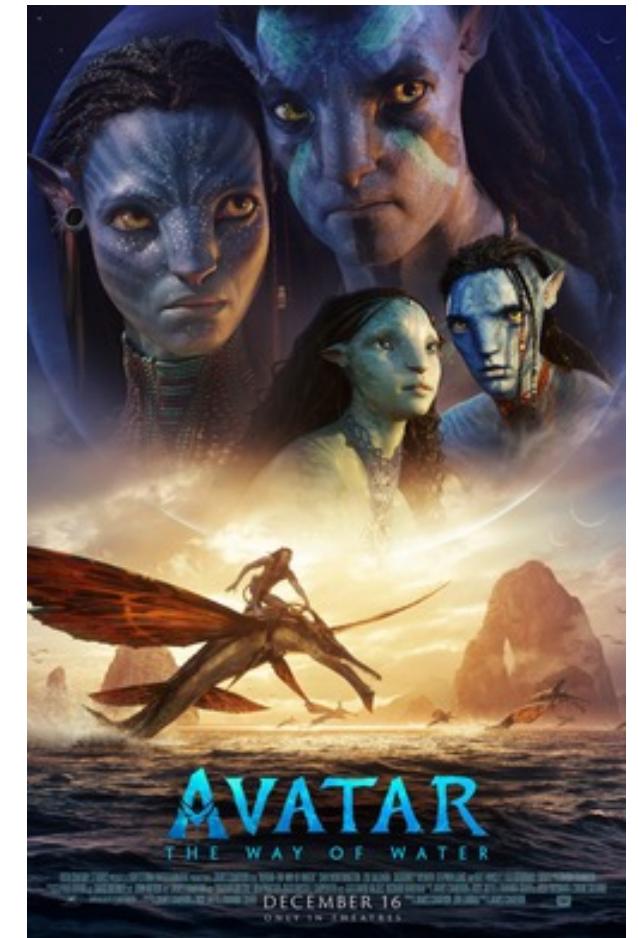
What is star wars' z-score?



Relationships between two quantitative variables

Do movies with larger budgets make more money?

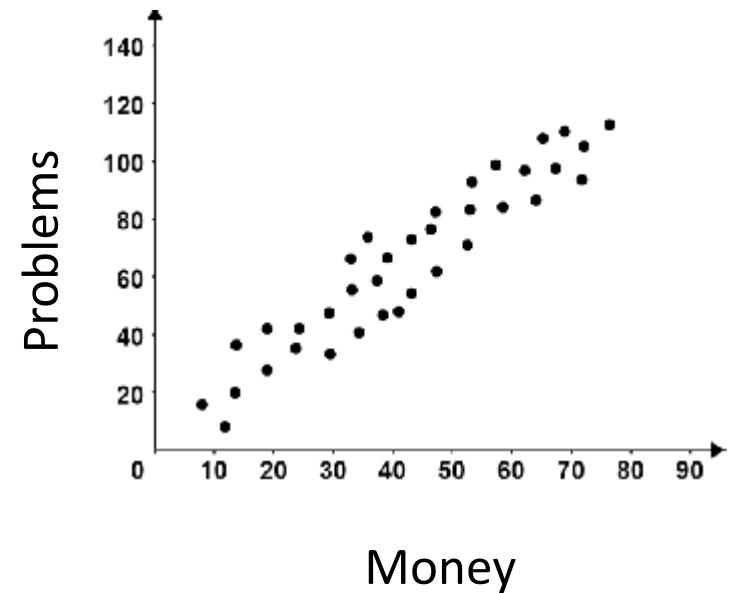
Q: How could we visualize the data to see if this is true?



Visualizing two quantitative variables: 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.

```
plt.plot(x, y, '.')
```

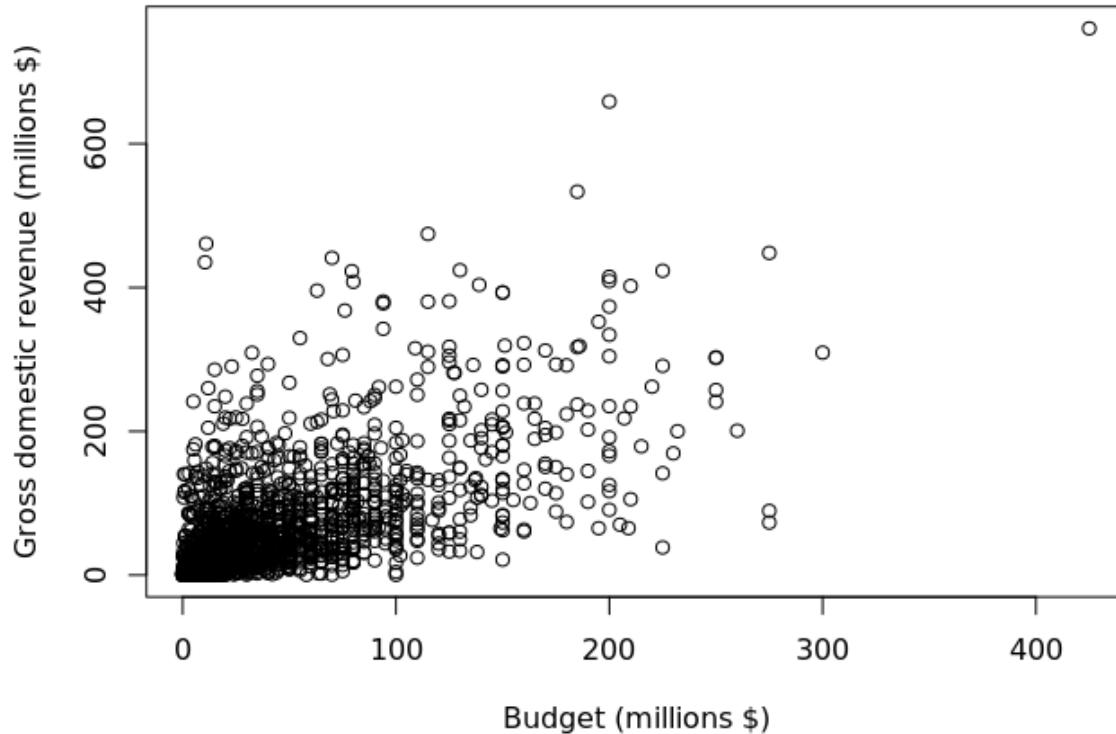
Do movies with larger budgets make more money?

Q: If we want to create a scatter plot to address whether movies with larger budgets make more money, what variables should we use in our plot?



Relationship movie money spent and made

Bechel movies relationship between buget and revenue



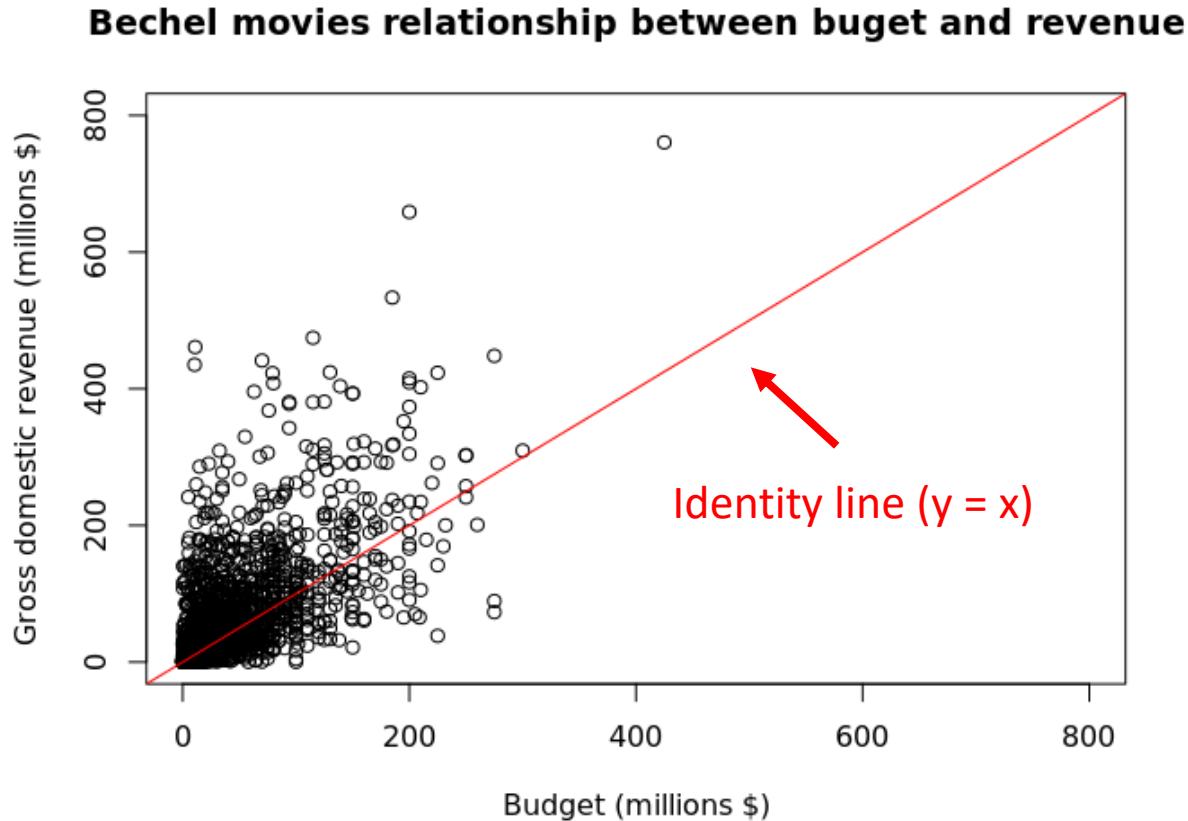
Do movies with larger budgets make more money?

Do most movies make money?

- How could we create a more informative scatter plot of this data?

Matplotlib: `plt.plot(x, y)`

Relationship movie money spent and made



Do movies with larger budgets make more money?

Do most movies make money?

- How could we create a more informative scatter plot of this data?

Matplotlib: `plt.plot(x, y)`

Questions when looking at scatterplots

Do the points show a clear trend?

Does it go upward or downward?

How much scatter around the trend?

Does the trend seem be linear (follow a line) or is it curved?

Are there any outlier points?

Questions when looking at scatterplots

Do the points show a clear trend?

Does it go upward or downward?

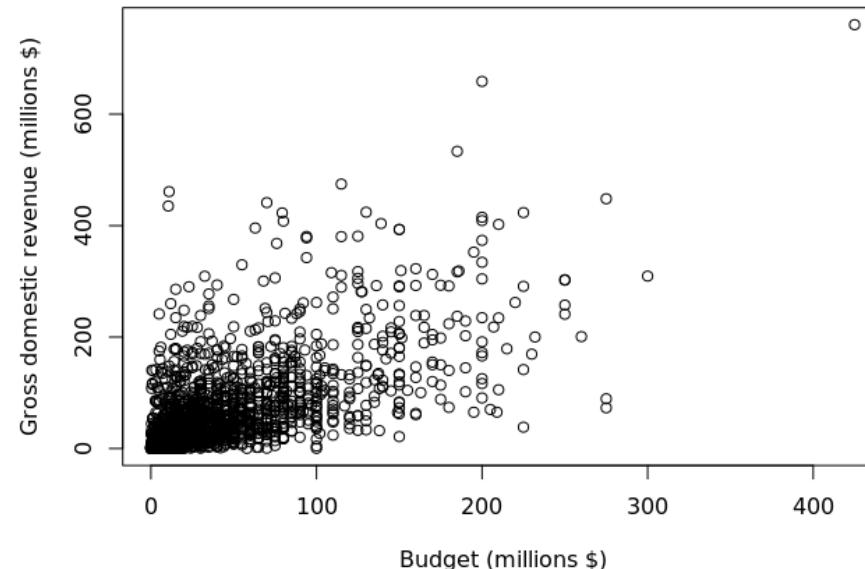
How much scatter around the trend?

Does the trend seem be linear (follow a line) or is it curved?

Are there any outlier points?

Budget and revenue

Bechel movies relationship between buget and revenue



Positive, negative, no correlation

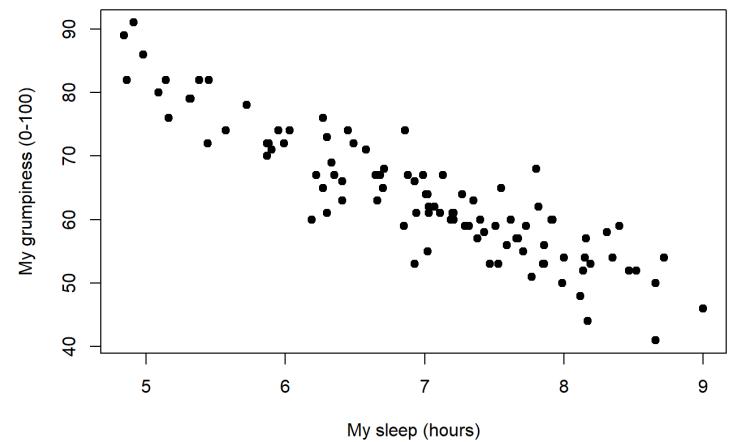
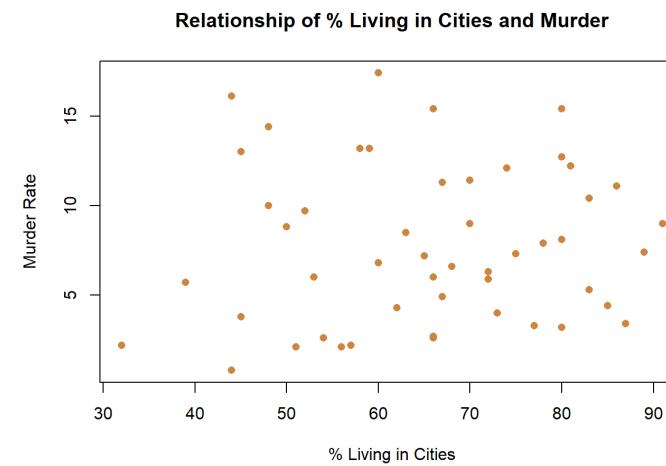
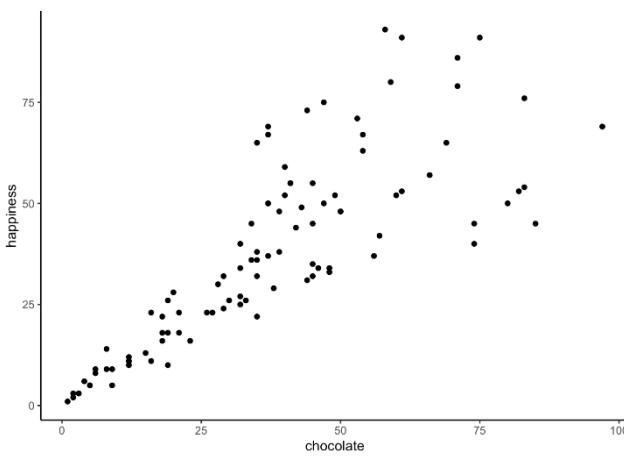
Do the points show a clear trend?

Does it go upward or downward?

How much scatter around the trend?

Does the trend seem be linear (follow a line) or is it curved?

Are there any outlier points?



The correlation coefficient

The **correlation** is measure of the strength and direction of a linear association between two variables

$$r = \frac{1}{(n - 1)} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right)$$

```
statistics.correlation(x, y)
```

Properties of the correlation

Correlation is always between -1 and 1: $-1 \leq r \leq 1$

The sign of r indicates the direction of the association

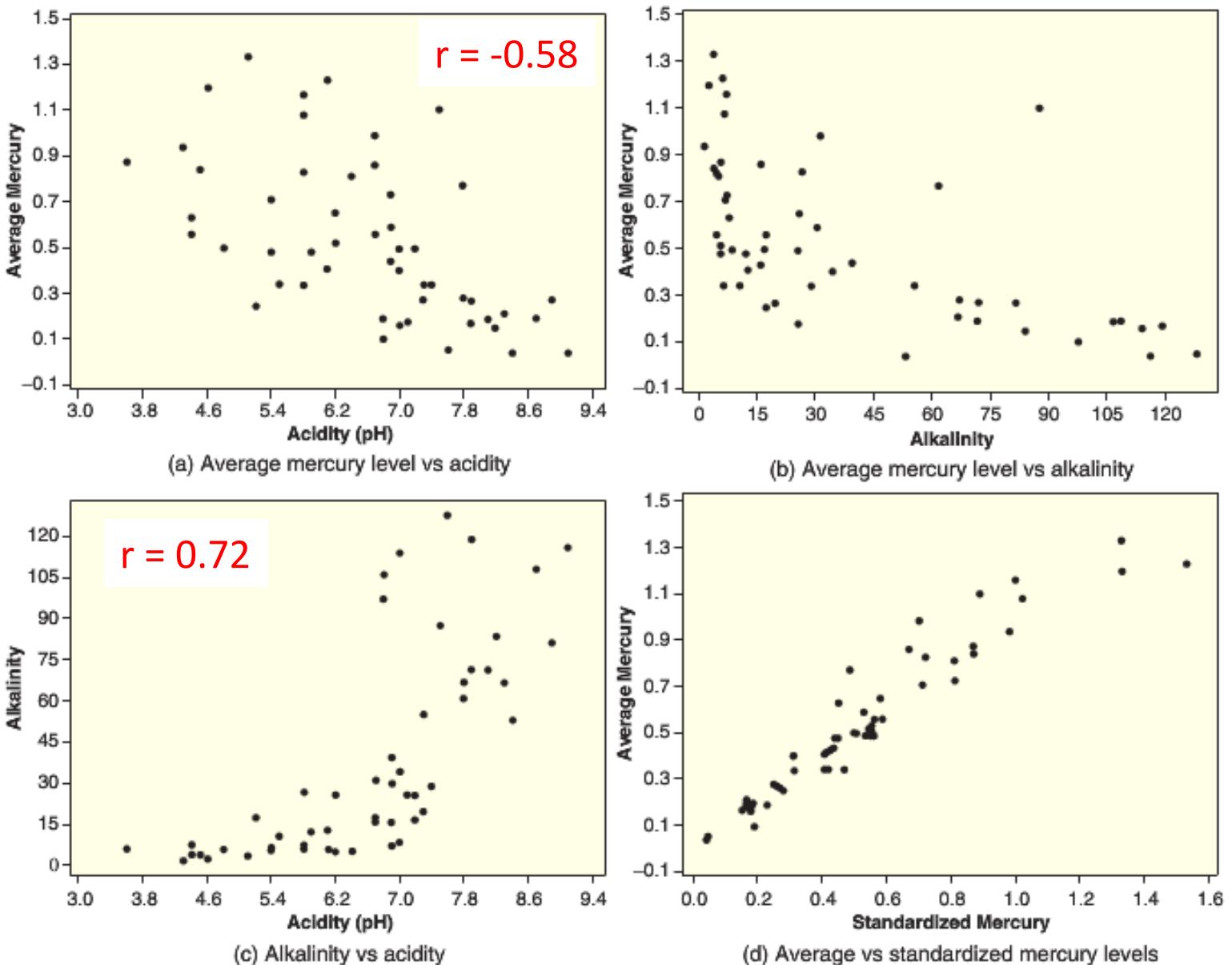
Values close to ± 1 show strong linear relationships, values close to 0 show no linear relationship

Correlation is symmetric: $r = \text{cor}(x, y) = \text{cor}(y, x)$

$$r = \frac{1}{(n-1)} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right)$$

Florida lakes

Correlation game



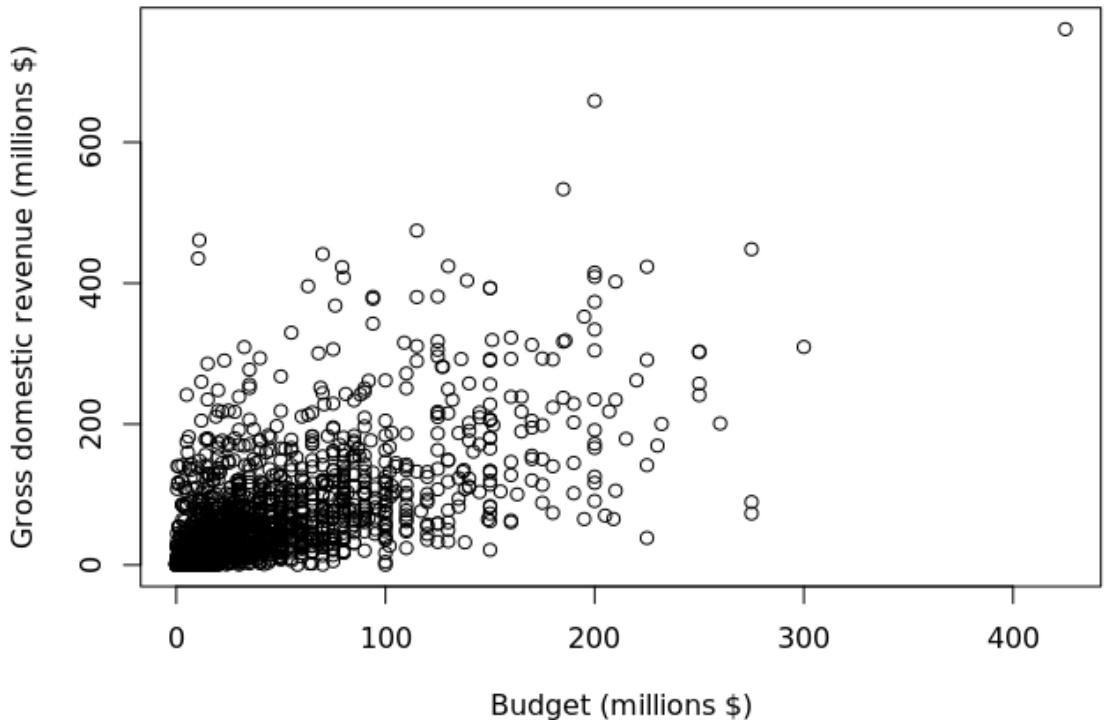
Movie budget and revenue correlation?

The **correlation** is measure of the strength and direction of a linear association between two variables

r = ?

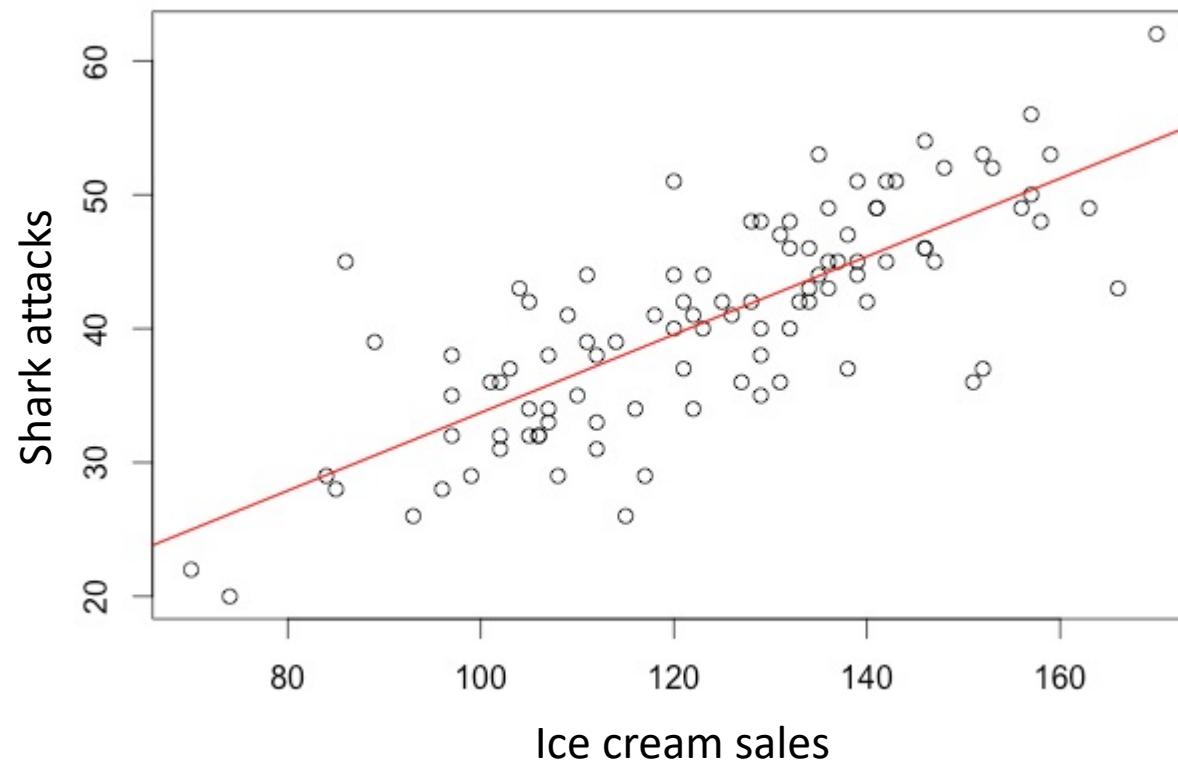
Let's calculate the correlation in Python!

Bechel movies relationship between buget and revenue



Correlation caution #1

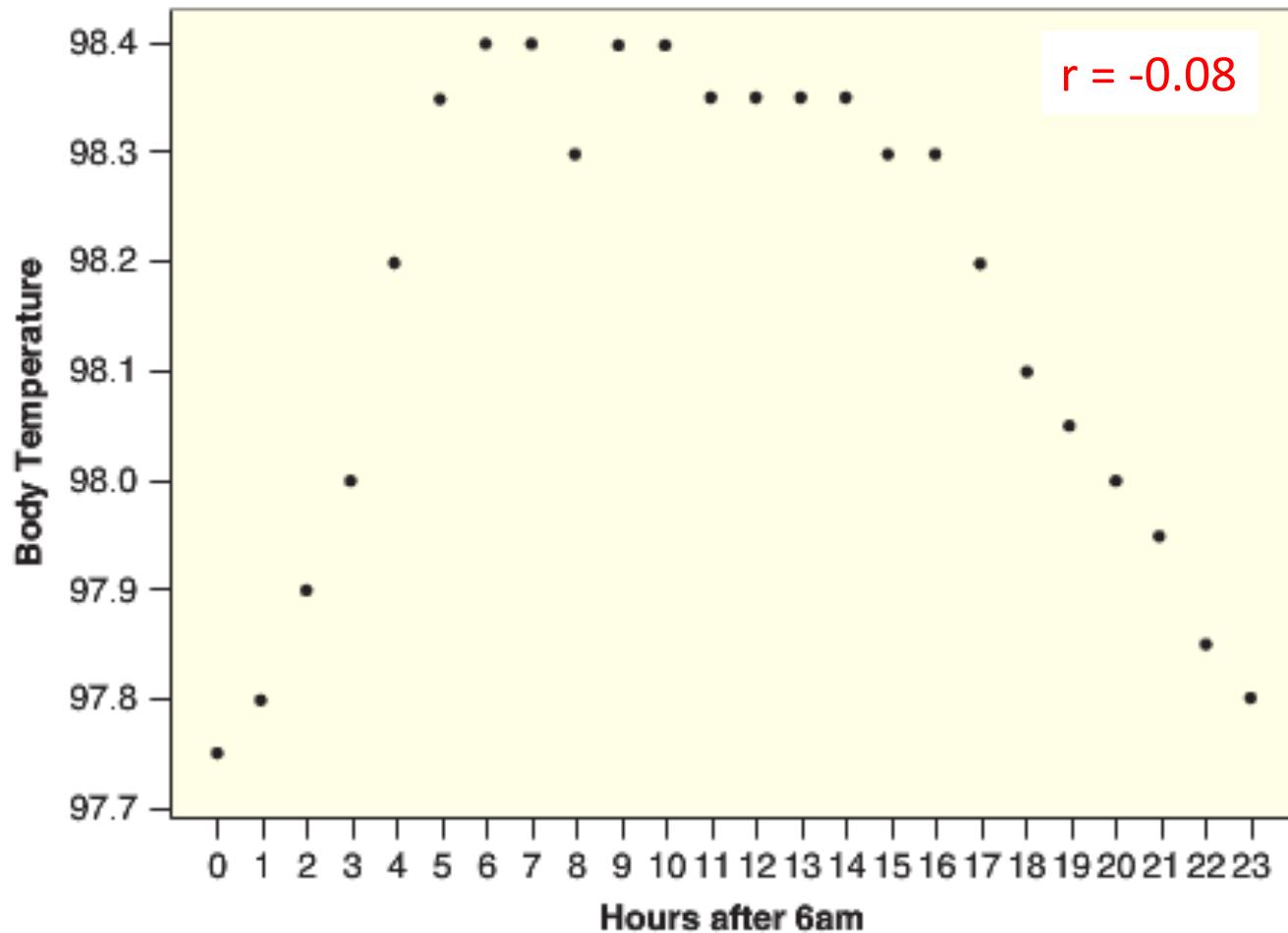
A strong positive or negative correlation does not (necessarily) imply a cause and effect relationship between two variables



Correlation caution #2

A correlation near zero does not (necessarily) mean that two variables are not associated. Correlation only measures the strength of a linear relationship.

Body temperature as a function of time of the day

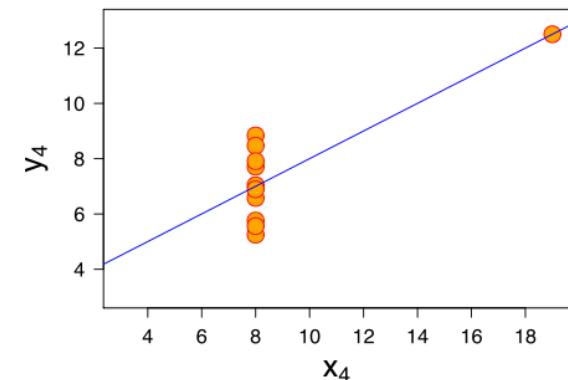
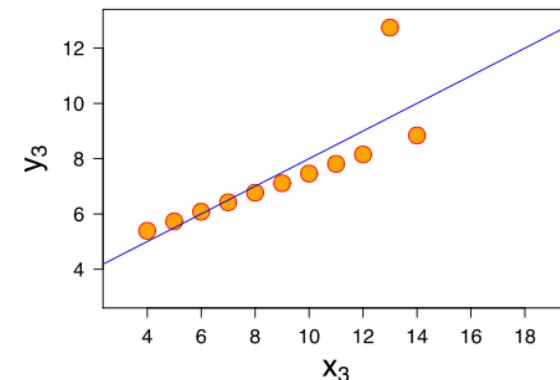
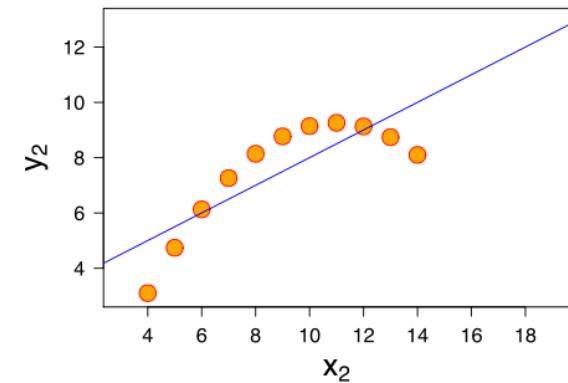
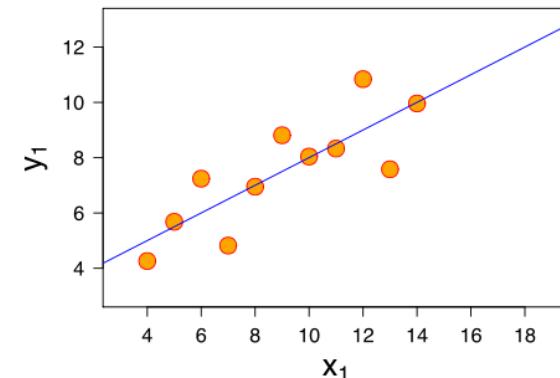


Correlation caution #3

Correlation can be heavily influenced by outliers. Always plot your data!

Example:
“Anscombe’s quartet”

$r = 0.81$ for all plots



Next week: array computations...

Homework 2 has been posted!

```
import YData
```

```
YData.download.download_homework(2)
```

It is due on Gradescope on Sunday September 15th at 11pm

- Be sure to mark each question on Gradescope!

Notes:

- There is an ~18 page reading from the book "Data and the American Dream" that you need to do, so I recommend you get started on this soon.

YData: Introduction to Data Science



Class 06: Array computations

Overview

Brief review of statistics and visualizations

NumPy arrays

- Creating arrays
- Array computations
- Boolean masking
- If there is time: Percentiles and boxplots



Announcement

Homework 3 has been posted!

It is due on Gradescope on **Sunday September 22nd at 11pm**

- **Be sure to mark each question on Gradescope!**

Alison Bechdel is giving a talk on Wednesday September 18th at 4pm in this classroom



THE CHUBB FELLOWSHIP • TIMOTHY DWIGHT COLLEGE • YALE UNIVERSITY

ALISON BECHDEL

PROFESSOR IN THE PRACTICE
ENGLISH DEPARTMENT AND FILM & MEDIA STUDIES

...and I became a lesbian cartoonist.
Reflections on a Curious Career

LECTURE AND Q&A

Wednesday, September 18, 2013
4:00–5:30PM

Sheffield-Sterling-Strathcona Hall,
1 Prospect Street, Room 114

Doors open at 3:30 for seating

Admission is free to the Yale Community and the General Public.
Please register here if you plan to attend. No tickets required.

Supported by the Department of English and the Program of Film and Media Studies

Please direct questions to:
chubb.fellowship@yale.edu

Assignments and grades

Homework policies

- You can use chatGPT to answer general questions about Python and/or statistical concepts
 - E.g., It is ok to ask chatGPT “What does the np.sum() function in Python do?”
- You are **not** allowed to use it to directly answer homework questions
 - E.g., It is **not** ok to cut and paste a homework question into chatGPT



Quick review: categorical data

Categorical data

$$\text{Proportion} = \frac{\text{number in category}}{\text{total number}}$$

bechdel.`count("PASS")/len(bechdel)`

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

```
plt.bar(labels, data)
```

```
plt.pie(data)
```

Quick review: one quantitative variable data

Basics statistics and plots:

`plt.hist(data)`

$$\text{statistics.mean(data)} = \frac{1}{n} \sum_{i=1}^n x_i$$

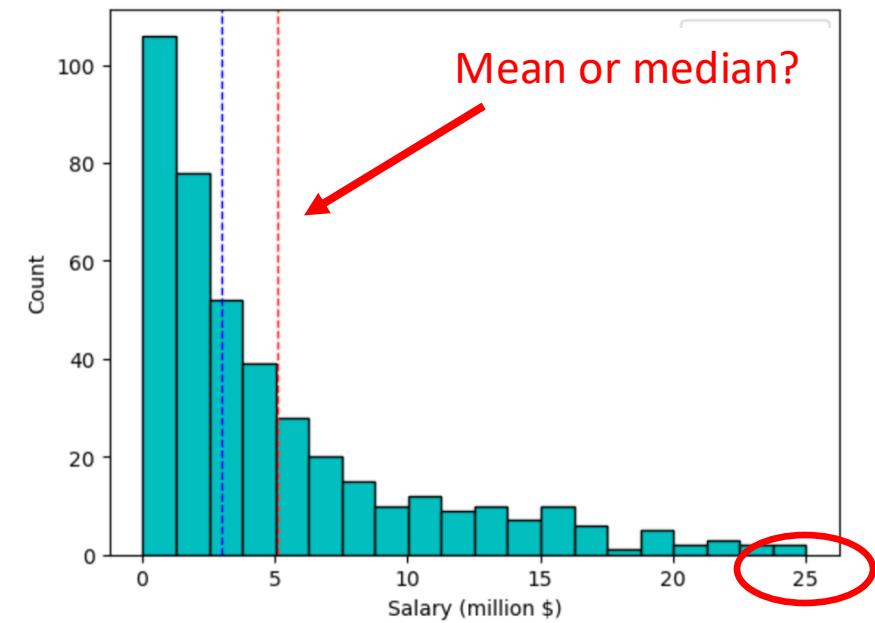
`statistics.median(data)`

Quantitative data spread

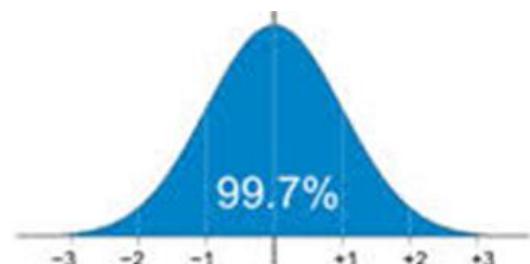
`statistics.stdev(data)`

$$s = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\text{z-score}(x_i) = \frac{x_i - \bar{x}}{s}$$



Outliers



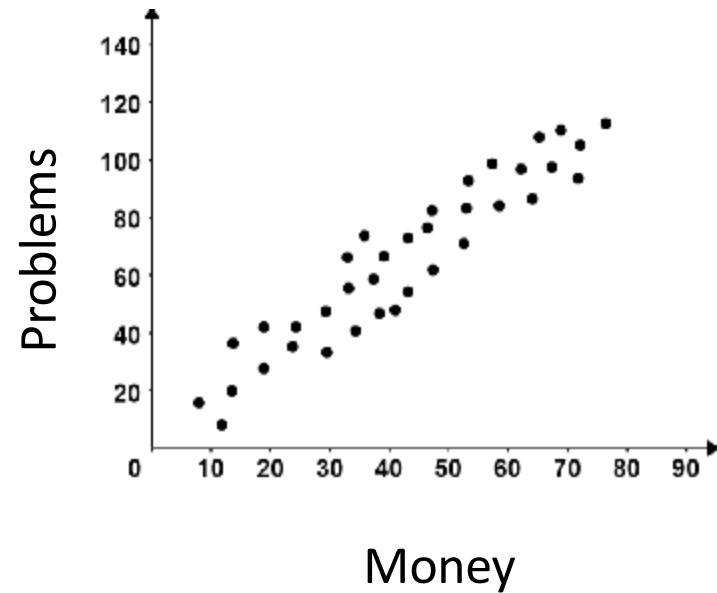
Quick review: two quantitative variables data

Basics statistics and plots:

```
plt.plot(x, y, '.')
```

```
statistics.correlation(x, y)
```

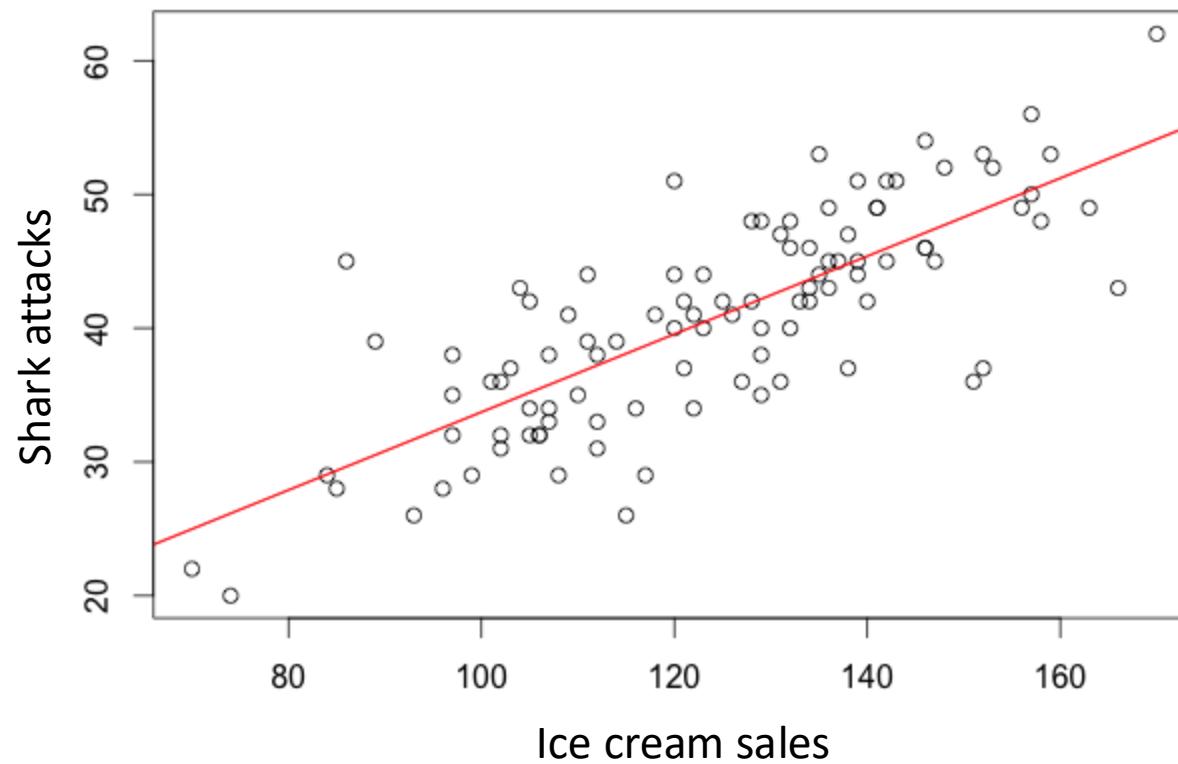
$$r = \frac{1}{(n-1)} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right)$$



- Correlation is always between -1 and 1: $-1 \leq r \leq 1$
- The sign of r indicates the direction of the association
- Values close to ± 1 show strong linear relationships, values close to 0 show no linear relationship
- Correlation is symmetric: $r = \text{cor}(x, y) = \text{cor}(y, x)$

Correlation caution #1

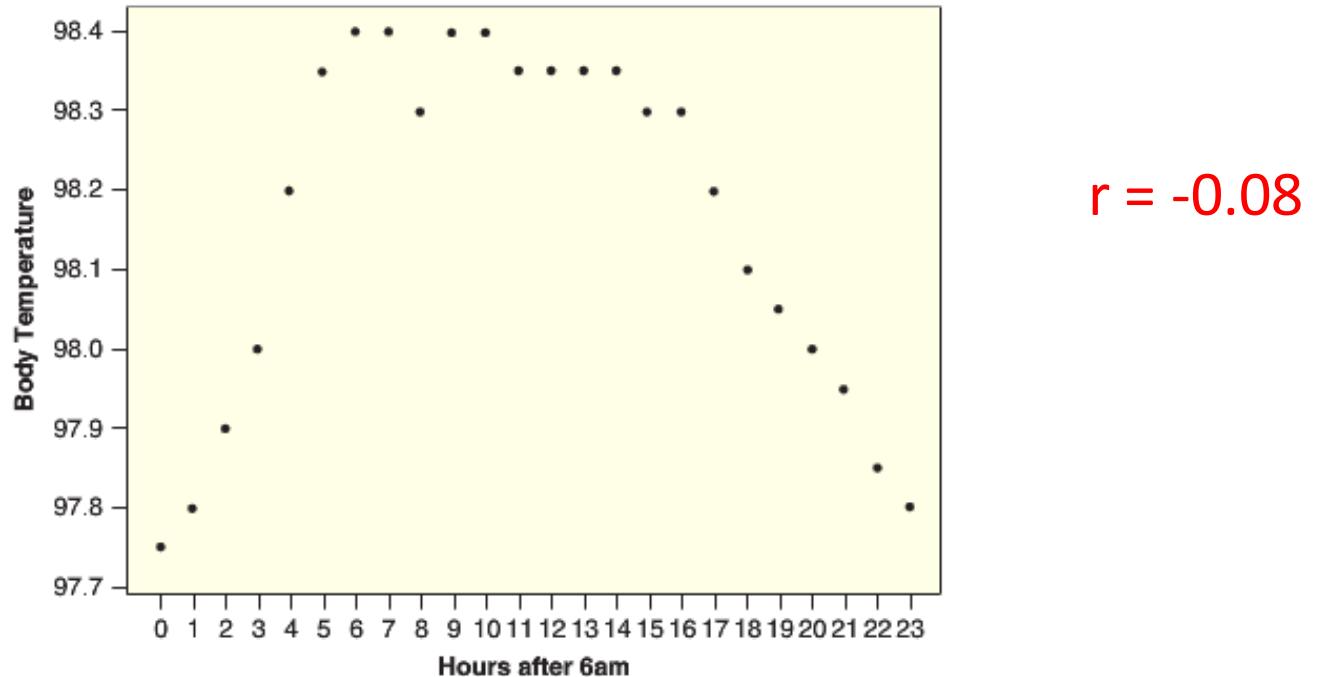
A strong positive or negative correlation does not (necessarily) imply a cause and effect relationship between two variables



Correlation caution #2

A correlation near zero does not (necessarily) mean that two variables are not associated. Correlation only measures the strength of a linear relationship.

Example: Body temperature as a function of time of the day

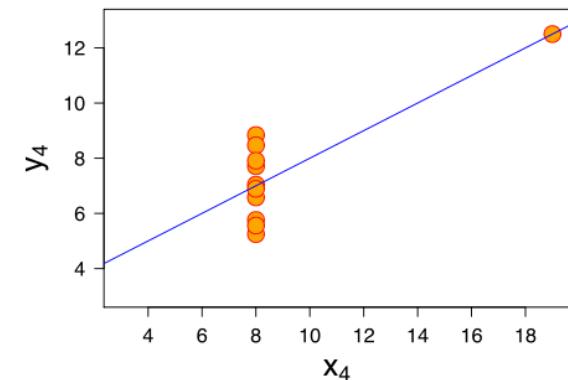
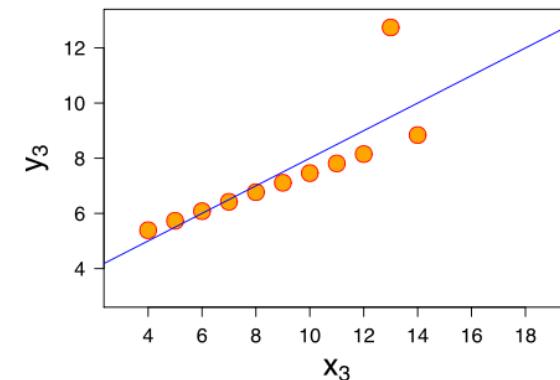
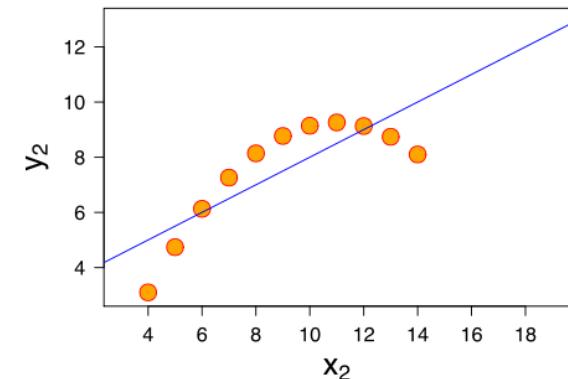
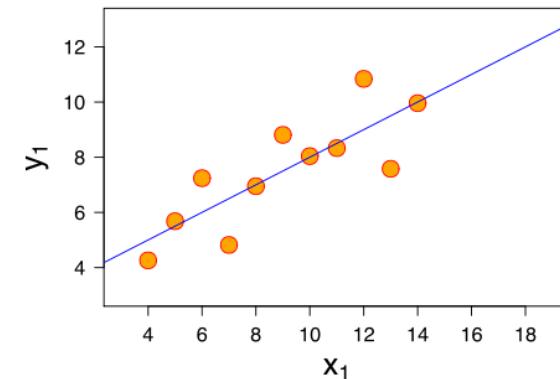


Correlation caution #3

Correlation can be heavily influenced by outliers. Always plot your data!

Example:
“Anscombe’s quartet”

$r = 0.81$ for all plots



Warm-up exercises: NBA salaries

Let's practice analyzing data by explore salaries of NBA players
• (from the 2022-2023 season)



Player Name	Salary	Position	Team	PTS
Stephen Curry	48070014	PG	GSW	29.4
John Wall	47345760	PG	LAC	11.4
Russell Westbrook	47080179	PG	LAL/LAC	15.9
LeBron James	44474988	PF	LAL	28.9
Kevin Durant	44119845	PF	BRK/PHO	29.1
Bradley Beal	43279250	SG	WAS	23.2
Kawhi Leonard	42492492	SF	LAC	23.8
Paul George	42492492	SF	LAC	23.8
Giannis Antetokounmpo	42492492	PF	MIL	31.1
Damian Lillard	42492492	PG	POR	32.2

Array computations

Arrays

Often, we are processing data that is all of the same type

- For example, we might want to do processing on a data set of numbers
 - e.g., if we were analyzing salary data

When we have data that is all of the same type, there are more efficient ways to process data than using a list

- i.e., methods that are faster and take up less memory

In Python, the *NumPy package* offers ways to store and process data that is all of the same type using a data structure called a *ndarray*

There are also functions that operate on ndarrays that can do computations very efficiently.



ndarrays

We can import the NumPy package using: `import numpy as np`

We can then create an array by passing a list to the `np.array()` function

- `my_array = np.array([1, 2, 3])`

We can get elements of an array using similar syntax as using a list

- `my_array[1] # what does this return`

ndarrays have properties that tell us the type and size

- `my_array.dtype` # get the type of elements stored in the array
- `my_array.shape` # get the dimension of the array
- `my_array.astype('str')` # convert the numbers to strings
- `sequential_nums = np.arange(1, 10)` # creates numbers 1 to 9

Let's explore this in Jupyter!

NumPy functions on numerical arrays

The NumPy package has a number of functions that operate very efficiently on numerical ndarrays

- `np.sum()`
- `np.max(), np.min()`
- `np.mean(), np.median()`
- `np.diff()` # takes the difference between elements
- `np.cumsum()` # cumulative sum

There are also "broadcast" functions that operate on all elements in an array

- `my_array = np.array([12, 4, 6, 3, 4, 3, 7, 4])`
- `my_array * 2`
- `my_array2 = np.array([10, 9, 2, 8, 9, 3, 8, 5])`
- `my_array - my_array2`

FRED: Federal Reserve Economic Data

[Federal Reserve Economic Data \(FRED\)](#) is a database maintained by the Research division of the Federal Reserve Bank of St. Louis that has more than 816,000 economic time series from various sources.

They cover:

- E.g., Consumer price indexes
- Employment and population
- Gross domestic product
- Producer price indexes
- Etc.



We can read this data into Python using [pandas_datareader](#)

```
from pandas_datareader.fred import FredReader  
FredReader("GASREGW").read()
```

Let's explore numpy functions using the [price of gas](#) from FRED!

Boolean arrays

It is often to compare all values in an ndarray to a particular value

- `my_array = np.array([12, 4, 6, 3, 4, 3, 7, 4])`
- `my_array < 5` # any guesses what this will return
 - `array([False, True, False, True, True, True, False, True])`

This can be useful for calculating proportions

- `True == 1` and `False == 0`
- Taking the sum of a Boolean array gives the total number of `True` values
- The number of `True`'s divided by the length is the proportion
 - Or we can use the `np.mean()` function

Categorical Variable

PLAYER	POSITION	TEAM	SALARY
str	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

Proportion centers =
$$\frac{\text{number of centers}}{\text{total number}}$$

Let's explore this in Jupyter!

Boolean masking

We can also use Boolean arrays to return values in another array

- This is called "Boolean masking", "Boolean subsetting" or "Boolean indexing"

```
my_array = np.array([12, 4, 6, 3])
boolean_mask = np.array([False, True, False, True, True])

smaller_array = my_array[boolean_mask]
```

This can be useful for calculating statistics on data that meet particular criteria:

- `np.mean(my_array[my_array < 5])` # what does this do?

Boolean masking

Suppose you wanted to get the average movie revenue for movies that passed the Bechdel test

- [domgross_2013](#): Movie revenue
- [bechdel](#): whether a movie passed the Bechdel test

Can you do it?



Let's explore this in Jupyter!

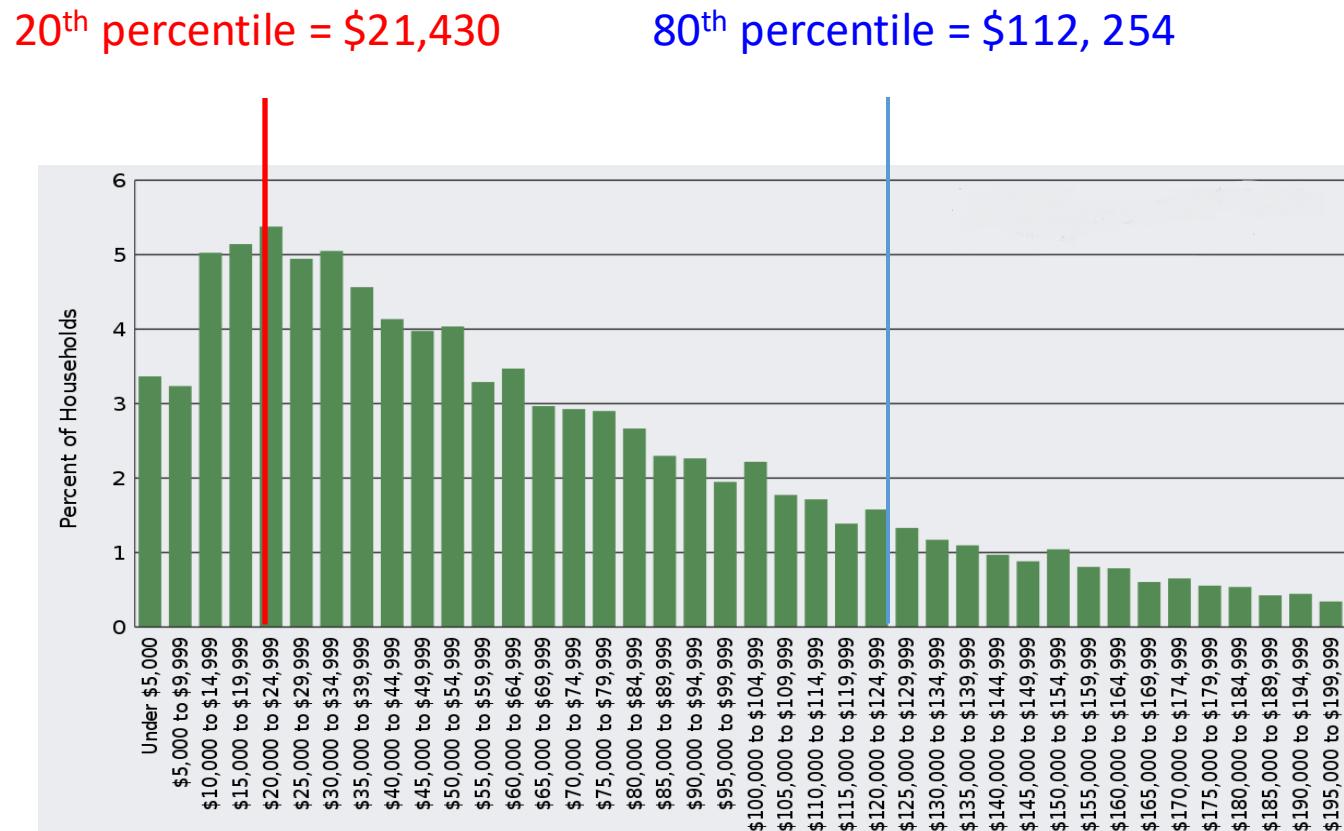
Percentiles

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?

We can calculate percentiles using `np.percentile()`

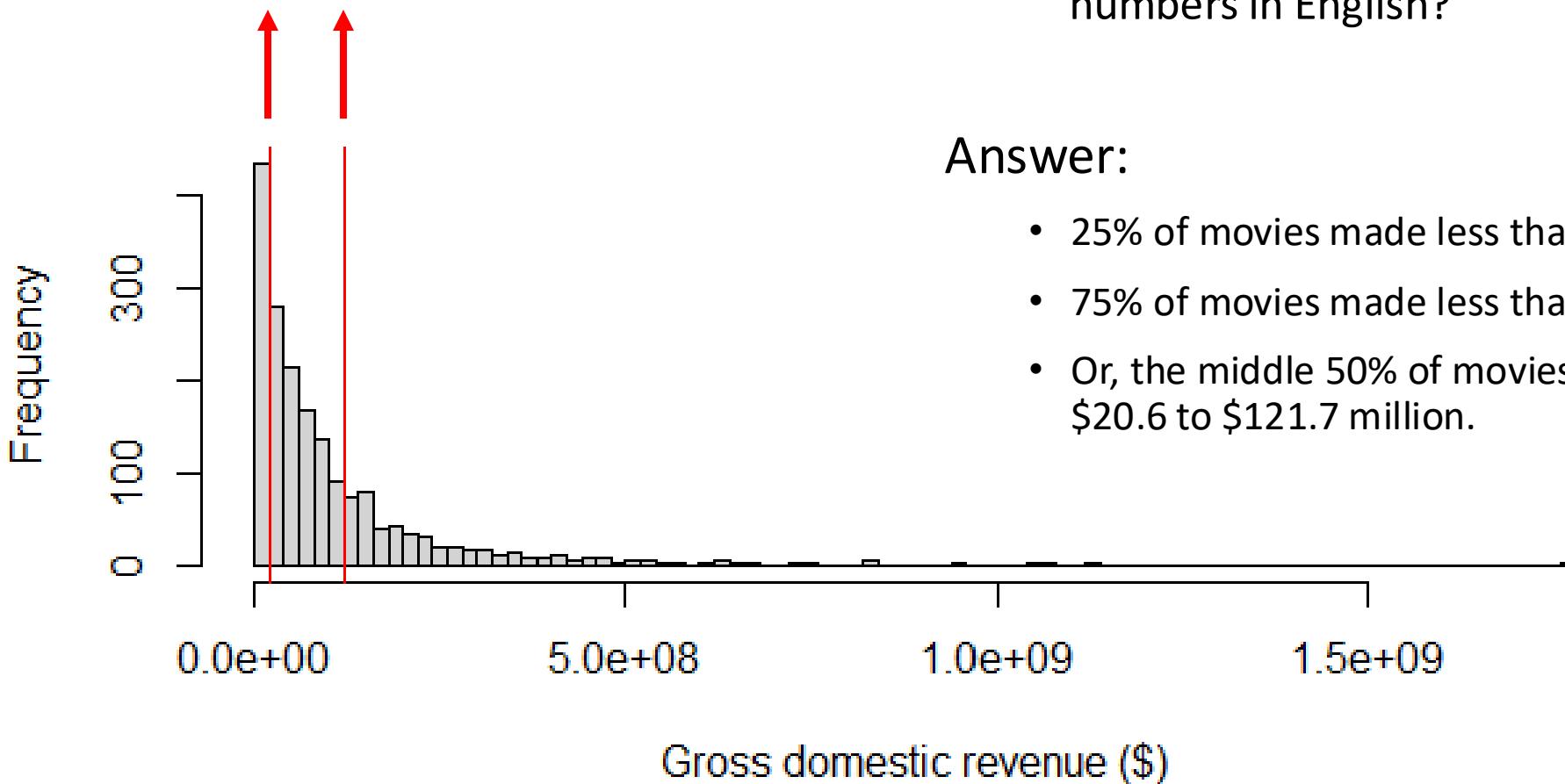
`np.percentile(data, [20, 80])`



Movie revenue percentiles

25th percentile
= \$20.6 million

75th percentile
= \$121.7 million



How do we interpret these numbers?

- i.e., how would we describe these numbers in English?

Answer:

- 25% of movies made less than \$20.6 million
- 75% of movies made less than \$121.7 million
- Or, the middle 50% of movies made between \$20.6 to \$121.7 million.

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

Range and Interquartile Range

Range = maximum – minimum

Interquartile range (IQR) = $Q_3 - Q_1$

Let's calculate these statistics on Bechdel movie revenue data!

Box plots and outliers

Detecting of outliers

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

Smaller than: $Q_1 - 1.5 * \text{IQR}$

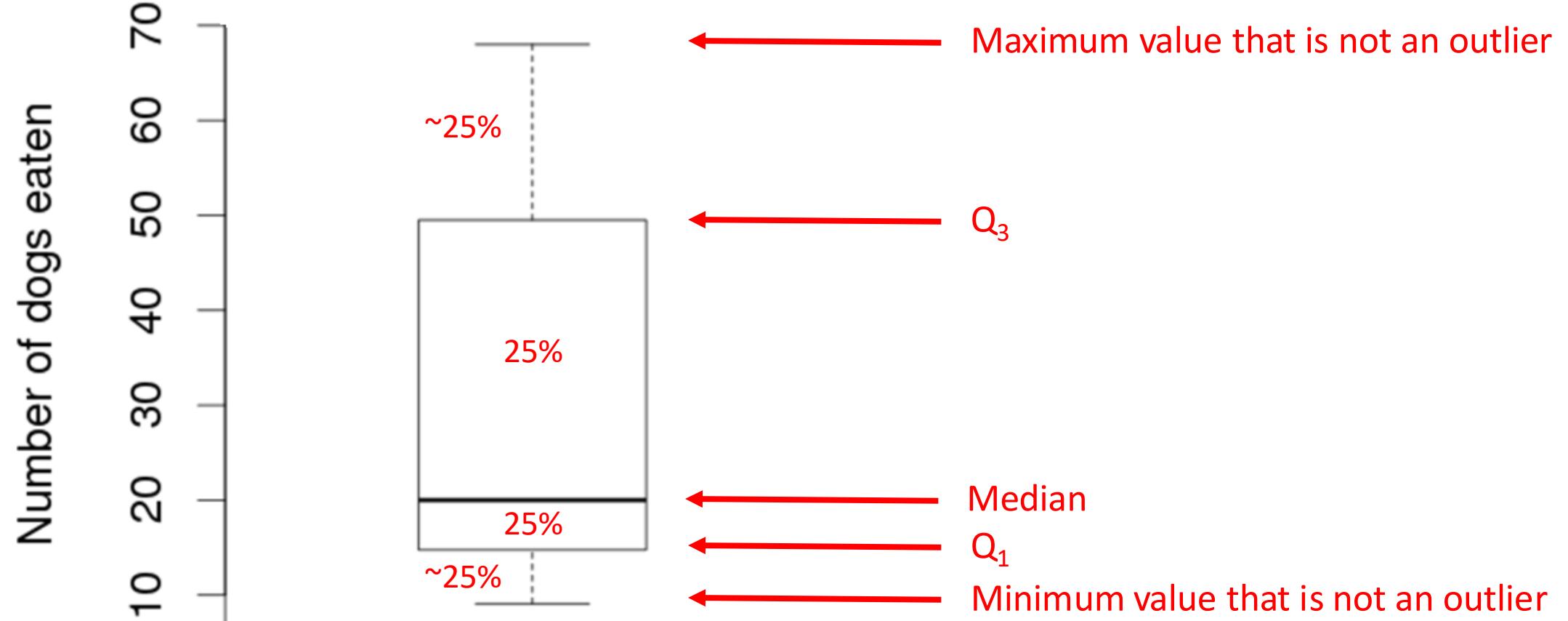
Larger than: $Q_3 + 1.5 * \text{IQR}$

Box plots

A **box plot** is a graphical display of the five-number summary and consists of:

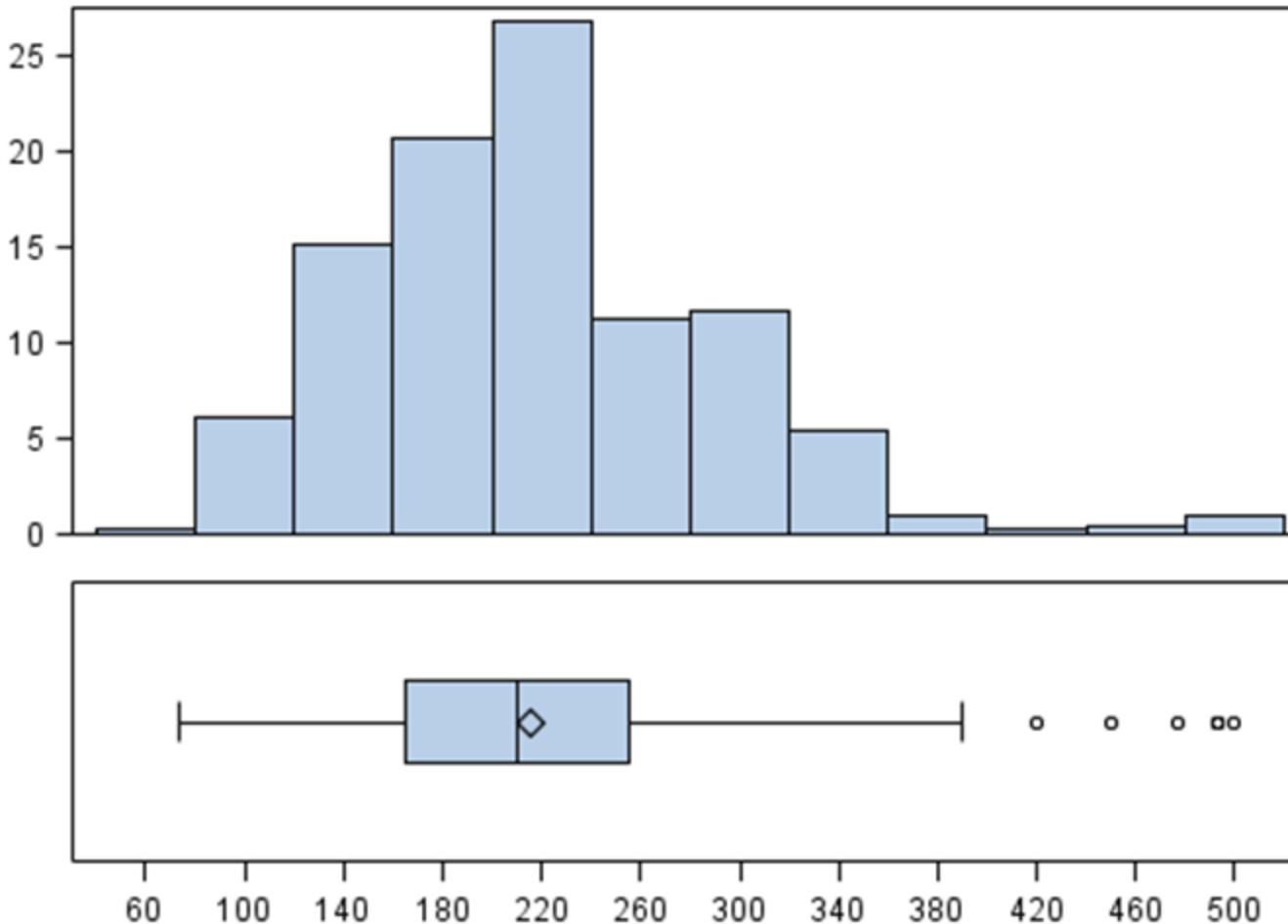
1. Drawing a box from Q_1 to Q_3
2. Dividing the box with a line (or dot) drawn at the median
3. Draw a line from each quartile to the most extreme data value that is not an outlier
4. Draw a dot/asterisk for each outlier data point.

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



Matplotlib: `plt.boxplot(data, labels)`

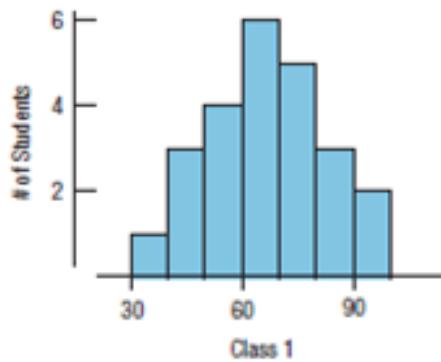
Box plots extract key statistics from histograms



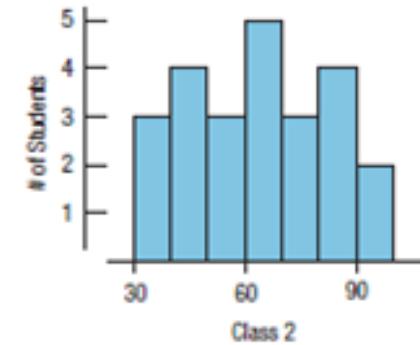
Box plots extract key statistics from histograms

Question: which Box plot goes with which histogram?

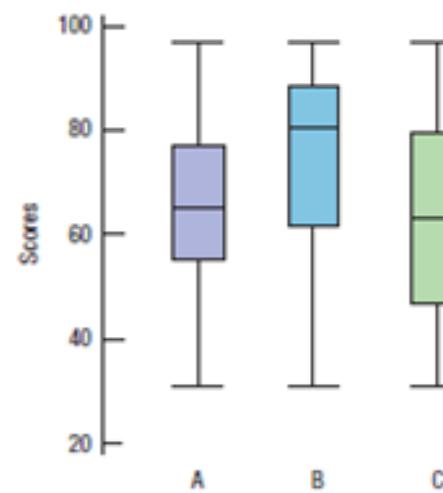
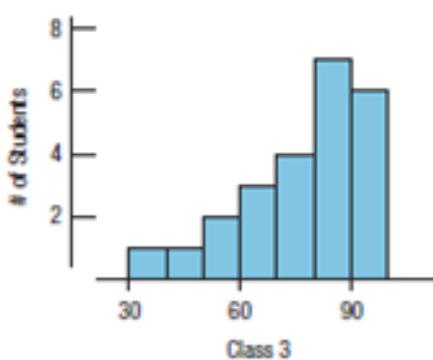
Histogram 1



Histogram 2



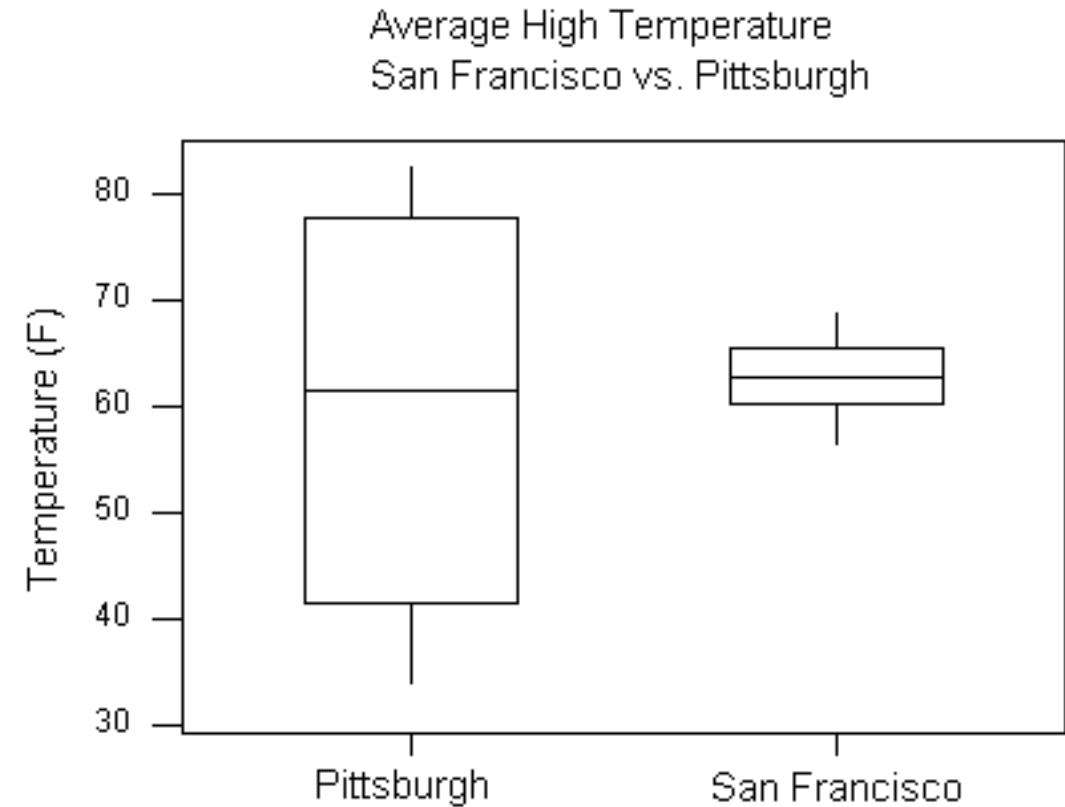
Histogram 3



Comparing quantitative variables across categories

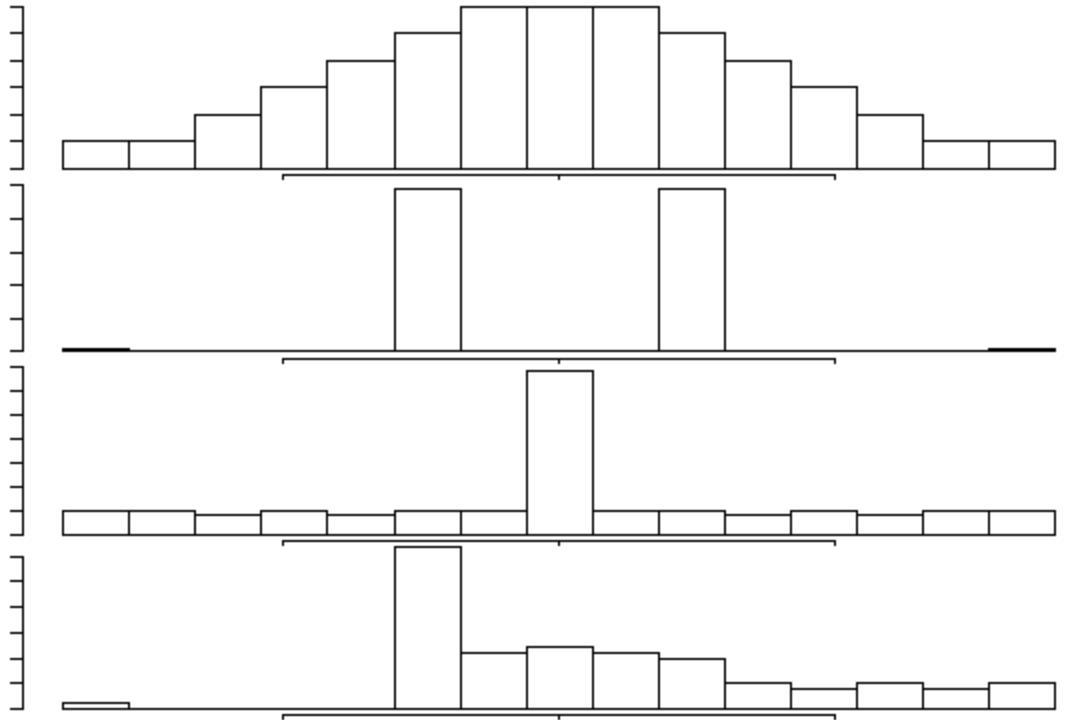
Often one wants to compare quantitative variables across categories

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



```
plt.boxplot([data1, data2], labels = ["name 1", "name 2"])
```

Box plots don't capture everything



Do you think the box plots for these distributions look similar?

Let's explore side-by-side boxplots on the Bechdel data to try to see if movies that pass the Bechdel test make a larger profit!

YData: Introduction to Data Science



Class 07: Array computation continued

Overview

Quick review of:

- NumPy arrays
- Numerical computations

More numpy:

- Boolean masking
- Higher dimensional numerical arrays
- Image manipulation

If there is time:

- Tuples and dictionaries
- Introduction to pandas Series and DataFrames



Announcement: practice session locations

Thursdays

- 5 pm to 6 pm
- 10 Hillhouse Avenue, Dunham Lab, Room 120

Fridays

- 2 pm to 3 pm
- 3 pm to 4 pm
- 4 pm to 5 pm
- 493 College Street, Room 106



Announcement: Homework 3

Homework 3 is due on Gradescope on Sunday September 22nd at 11pm

- **Be sure to mark each question on Gradescope!**

Notes:

- On problem 3, if the images are not showing up make sure to run the cells where the images are embedded
 - If you figure it out, help other people on Ed!

Quick review: ndarrays

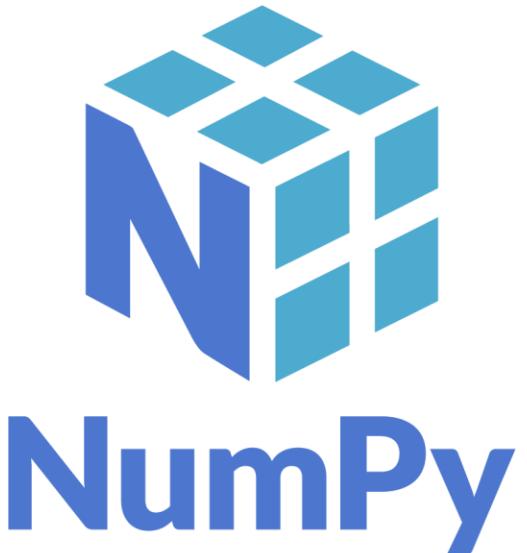
The *NumPy package* efficiently stores and processes data that is all of the same type using *ndarray*

```
import numpy as np

my_array = np.array([1, 2, 3]) # creating an ndarray
my_array[0]                  # accessing the 0th element

my_array.dtype                # get the type of elements
my_array.shape                # get the dimension
my_array.astype('str')        # convert to strings

sequential_nums = np.arange(1, 10) # creates numbers 1 to 9
```



Quick review: functions on numerical arrays

The NumPy functions:

- `np.sum()`
- `np.max(), np.min()`
- `np.mean(), np.median()`
- `np.diff()` # takes the difference between elements
- `np.cumsum()` # cumulative sum

There are also "broadcast" functions that operate on all elements in an array

- `my_array = np.array([12, 4, 6, 3, 4, 3, 7, 4])`
- `my_array * 2`
- `my_array2 = np.array([10, 9, 2, 8, 9, 3, 8, 5])`
- `my_array - my_array2`

Quick review: Boolean arrays

It is often to compare all values in an ndarray to a particular value

- `my_array = np.array([12, 4, 6, 3, 4, 3, 7, 4])`
- `my_array < 5`
 - `array([False, True, False, True, True, True, False, True])`

This can be useful for calculating proportions

- `True == 1` and `False == 0`
- Taking the sum of a Boolean array gives the total number of `True` values
- The number of `True`'s divided by the length is the proportion
 - Or we can use the `np.mean()` function

Categorical Variable

PLAYER	POSITION	TEAM	SALARY
str	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

Proportion centers =
$$\frac{\text{number of centers}}{\text{total number}}$$

Warm up: Number journey!

Please download the class 6 Jupyter notebook

- `import YData`
- `YData.download_class_code(6)`



Please complete the following number journey in the class 6 notebook:

- **Step 1:** Create an ndarray called *my_array* that has the numbers: 12, 4, 6, 3, 4, 3, 7, 4
- **Step 2:** Create an array *my_array2* that consists of the values of *my_array* minus the mean value of *my_array*.
- **Step 3:** Create *my_array3* which is a Boolean array that has True values for the positive values in *my_array2*
- **Step 4:** Calculate and print the total number of True values in *my_array3*

Let's take a number journey now...

Boolean masking

Boolean masking

We can also use Boolean arrays to return values in another array

- This is called "Boolean masking", "Boolean subsetting" or "Boolean indexing"

```
my_array = np.array([12, 4, 6, 3])
boolean_mask = np.array([False, True, False, True, True])

smaller_array = my_array[boolean_mask]
```

This can be useful for calculating statistics on data that meet particular criteria:

- `np.mean(my_array[my_array < 5])` # what does this do?

Boolean masking

Suppose you wanted to get the average movie revenue for movies that passed the Bechdel test

- [domgross_2013](#): Movie revenue
- [bechdel](#): whether a movie passed the Bechdel test

Can you do it?



Let's explore this in Jupyter!

Percentiles

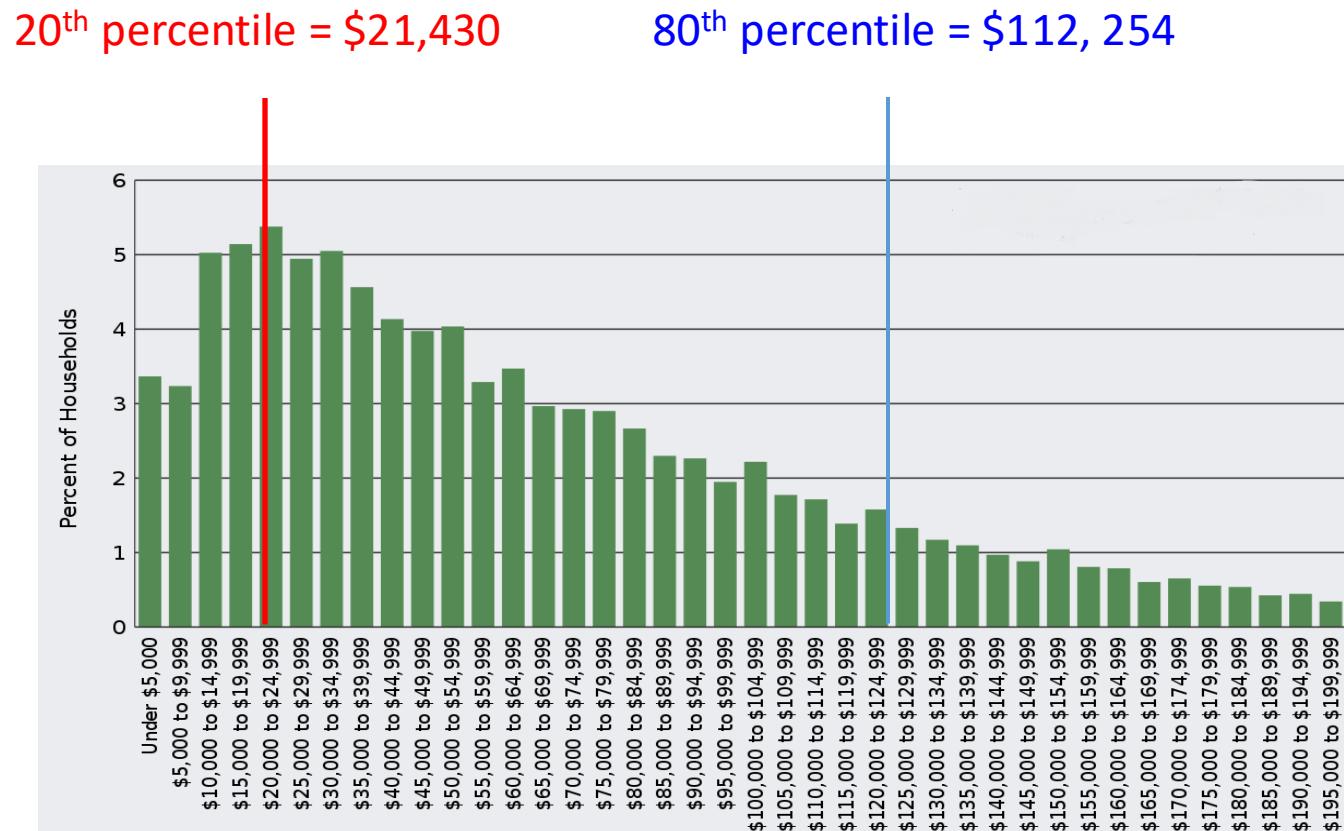
Percentiles

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?

We can calculate percentiles using `np.percentile()`

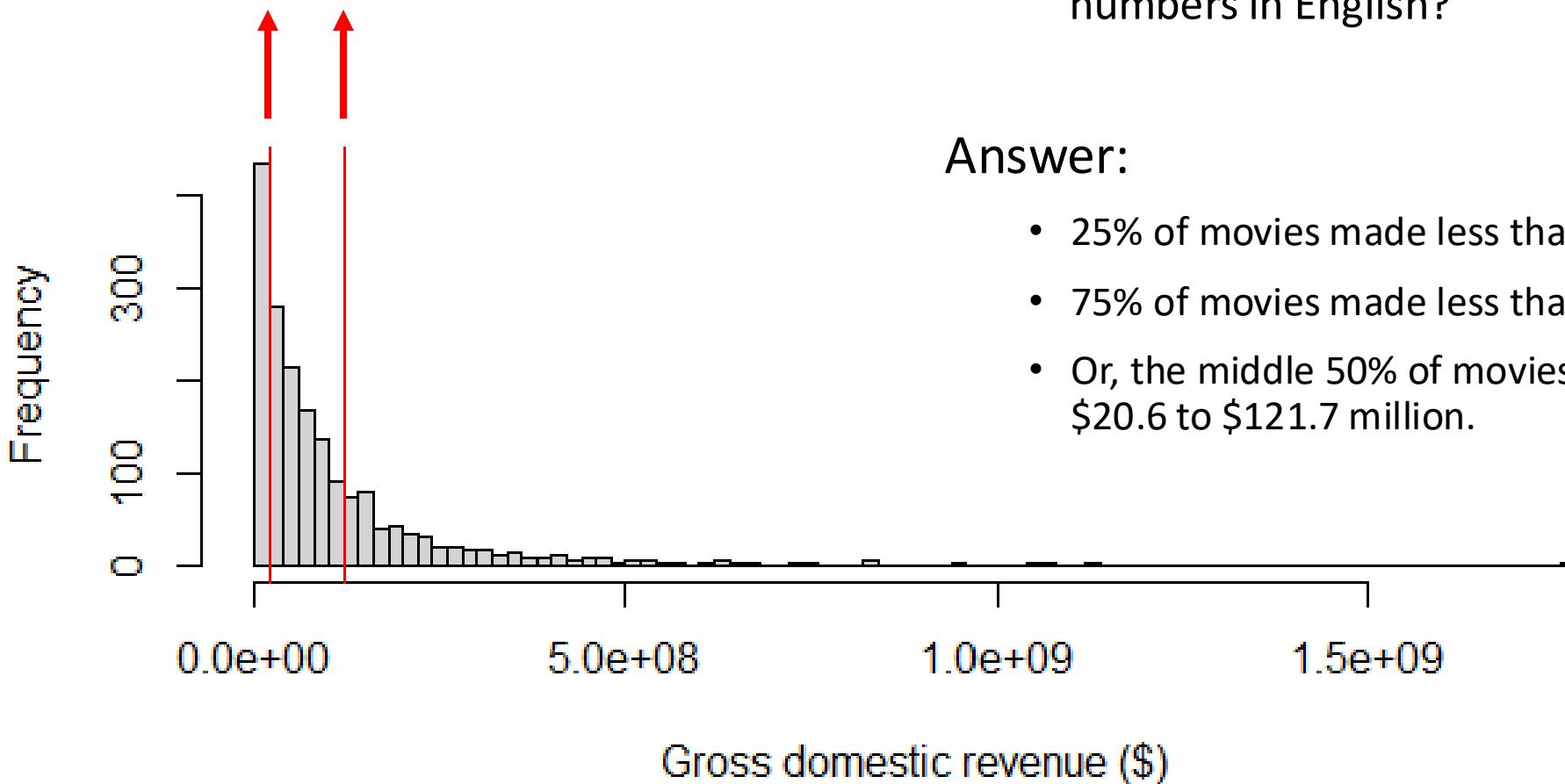
`np.percentile(data, [20, 80])`



Movie revenue percentiles

25th percentile
= \$20.6 million

75th percentile
= \$121.7 million



How do we interpret these numbers?

- i.e., how would we describe these numbers in English?

Answer:

- 25% of movies made less than \$20.6 million
- 75% of movies made less than \$121.7 million
- Or, the middle 50% of movies made between \$20.6 to \$121.7 million.

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

Range and Interquartile Range

Range = maximum – minimum

Interquartile range (IQR) = $Q_3 - Q_1$

Let's calculate these statistics on Bechdel movie revenue data!

Box plots and outliers

Detecting of outliers

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

Smaller than: $Q_1 - 1.5 * \text{IQR}$

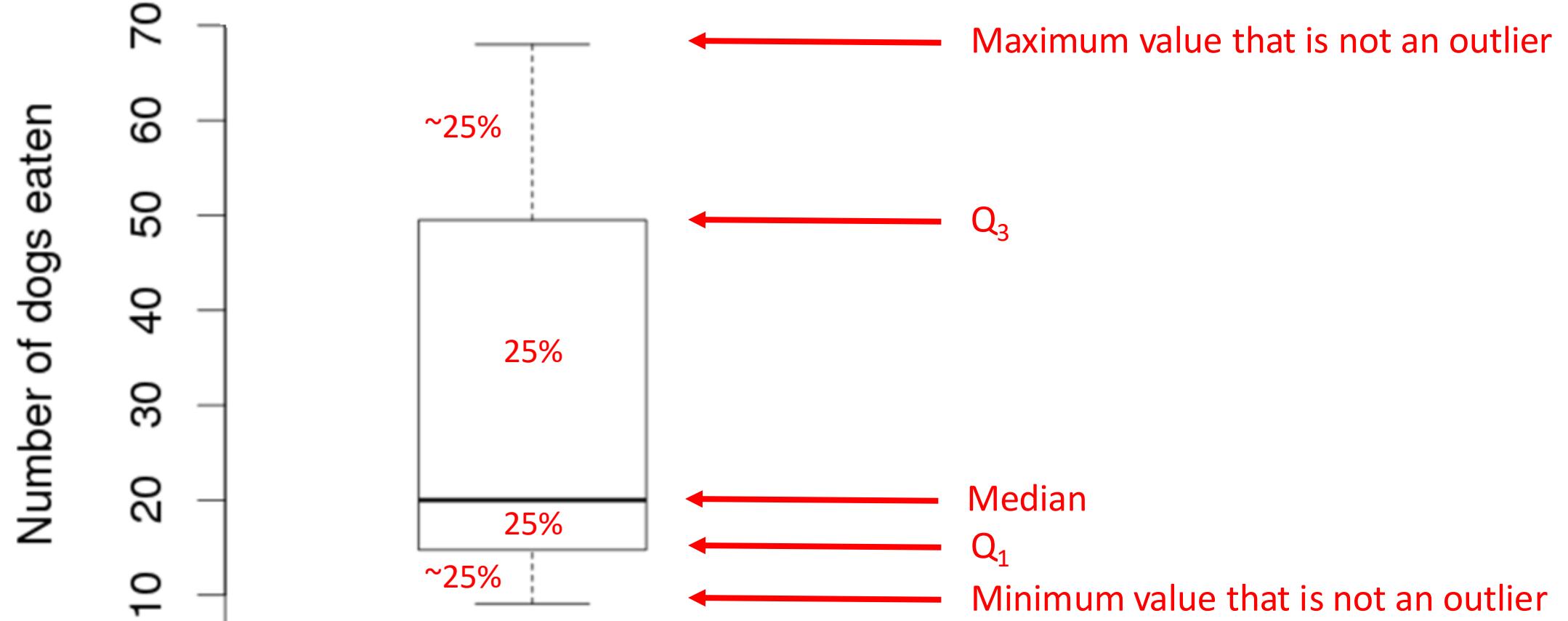
Larger than: $Q_3 + 1.5 * \text{IQR}$

Box plots

A **box plot** is a graphical display of the five-number summary and consists of:

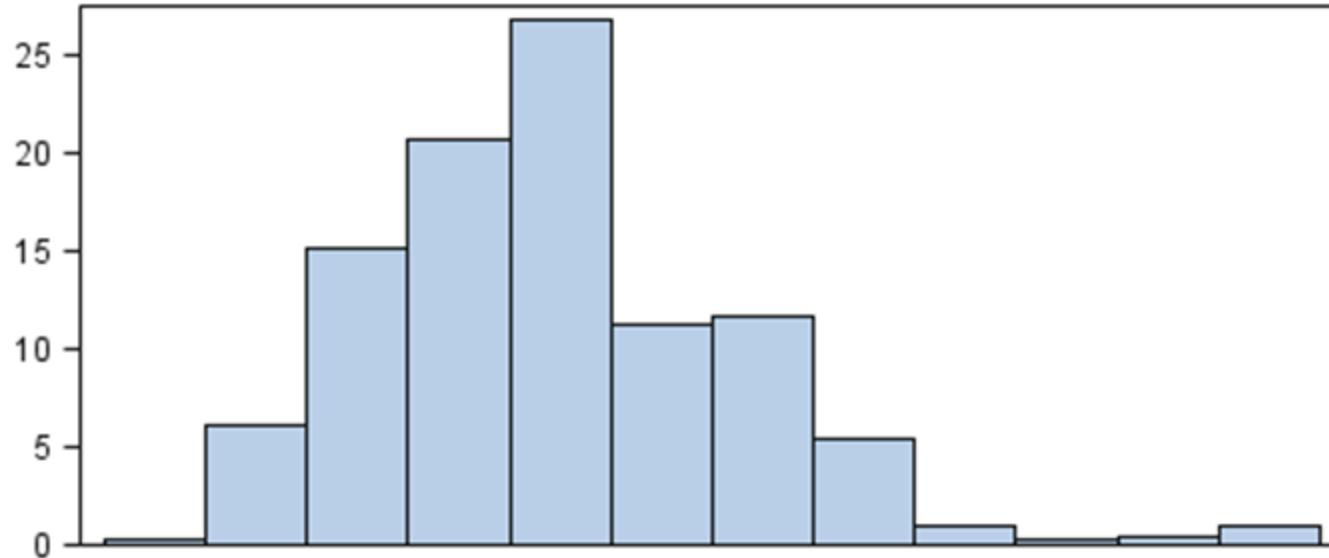
1. Drawing a box from Q_1 to Q_3
2. Dividing the box with a line (or dot) drawn at the median
3. Draw a line from each quartile to the most extreme data value that is not an outlier
4. Draw a dot/asterisk for each outlier data point.

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



Matplotlib: `plt.boxplot(data, labels)`

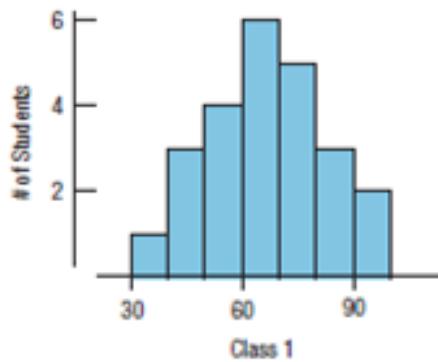
Box plots extract key statistics from histograms



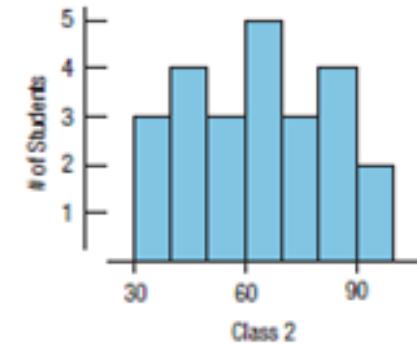
Box plots extract key statistics from histograms

Question: which Box plot goes with which histogram?

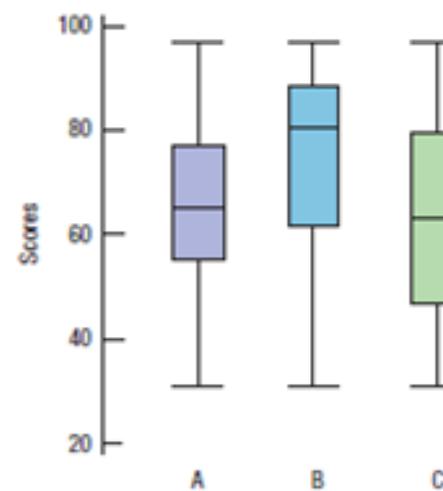
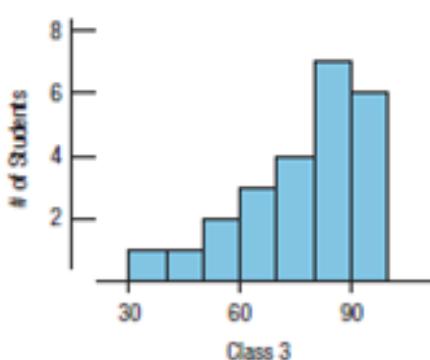
Histogram 1



Histogram 2



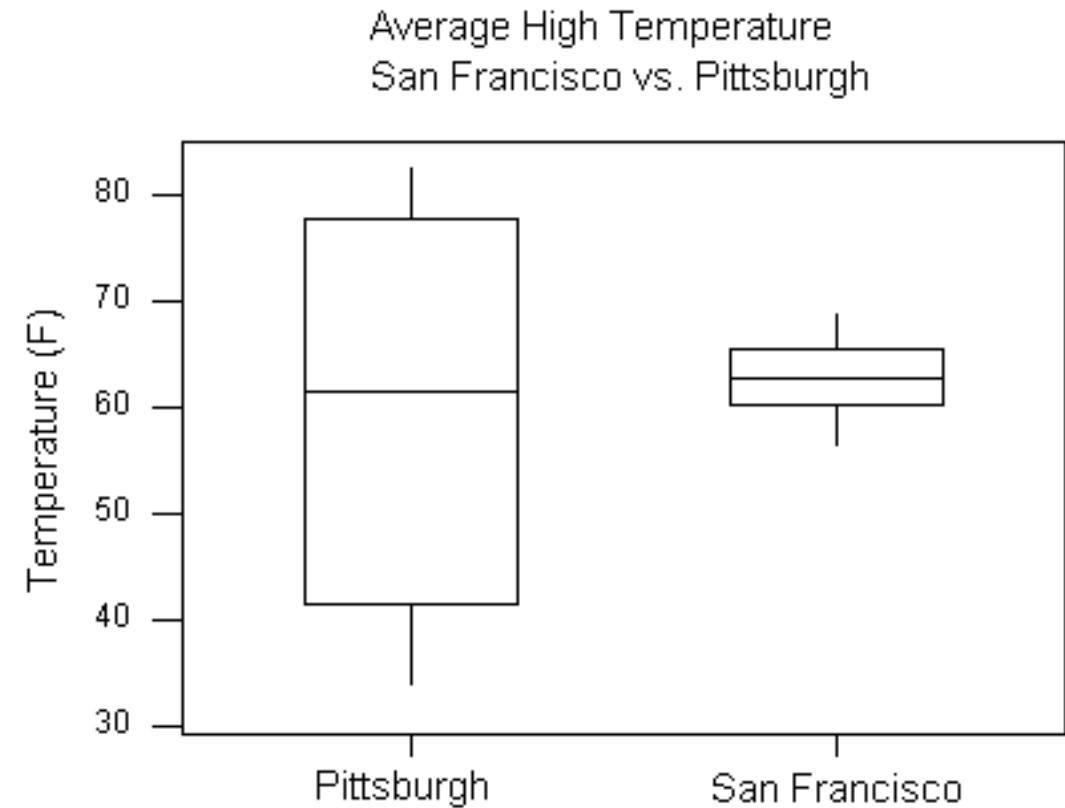
Histogram 3



Comparing quantitative variables across categories

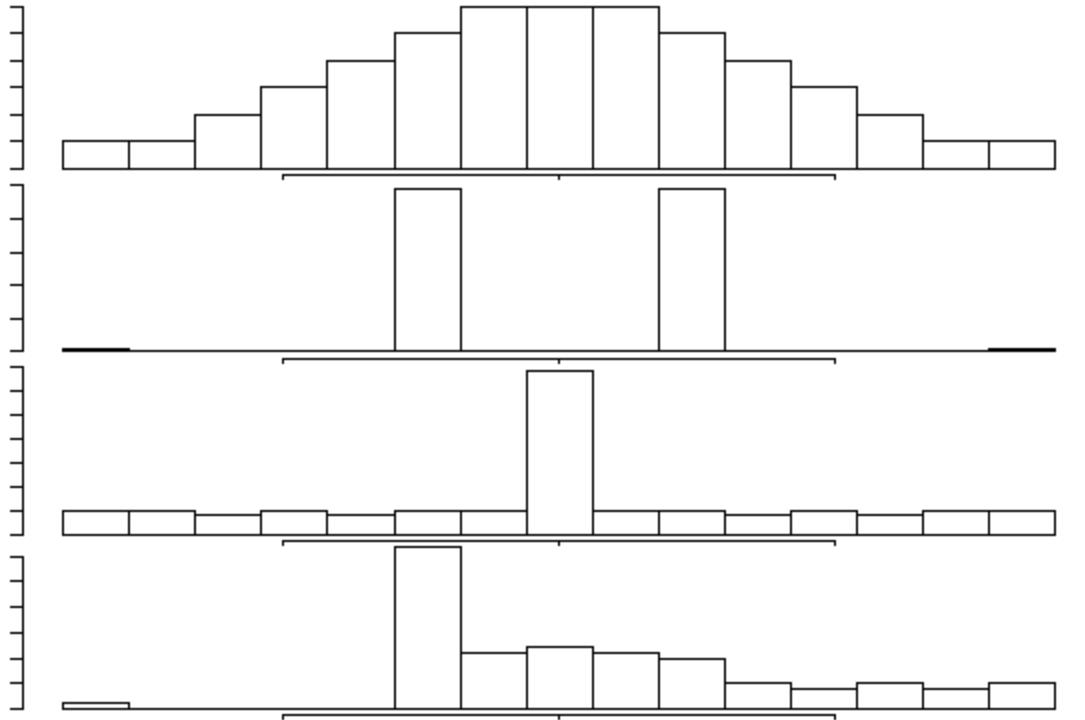
Often one wants to compare quantitative variables across categories

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



```
plt.boxplot([data1, data2], labels = ["name 1", "name 2"])
```

Box plots don't capture everything



Do you think the box plots for these distributions look similar?

Let's explore side-by-side boxplots on the Bechdel data to try to see if movies that pass the Bechdel test make a larger profit!

Boolean arrays

It is often to compare all values in an ndarray to a particular value

- `my_array = np.array([12, 4, 6, 3, 4, 3, 7, 4])`
- `my_array < 5`
 - `array([False, True, False, True, True, True, False, True])`

This can be useful for calculating proportions

- `True == 1` and `False == 0`
- Taking the sum of a Boolean array gives the total number of `True` values
- The number of `True`'s divided by the length is the proportion
 - Or we can use the `np.mean()` function

Categorical Variable

PLAYER	POSITION	TEAM	SALARY
str	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

Proportion centers =
$$\frac{\text{number of centers}}{\text{total number}}$$

2D array

5.2	3.0	4.5
9.1	0.1	0.3

shape: (2, 3)

Higher dimensional arrays

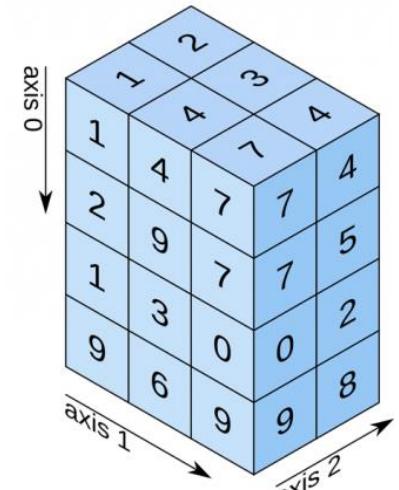
We can make higher dimensional arrays

- (matrices and tensors)

```
my_matrix = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

```
my_matrix
```

3D array



We can slice higher dimensional array

- `my_matrix[0:2, 0:2]`

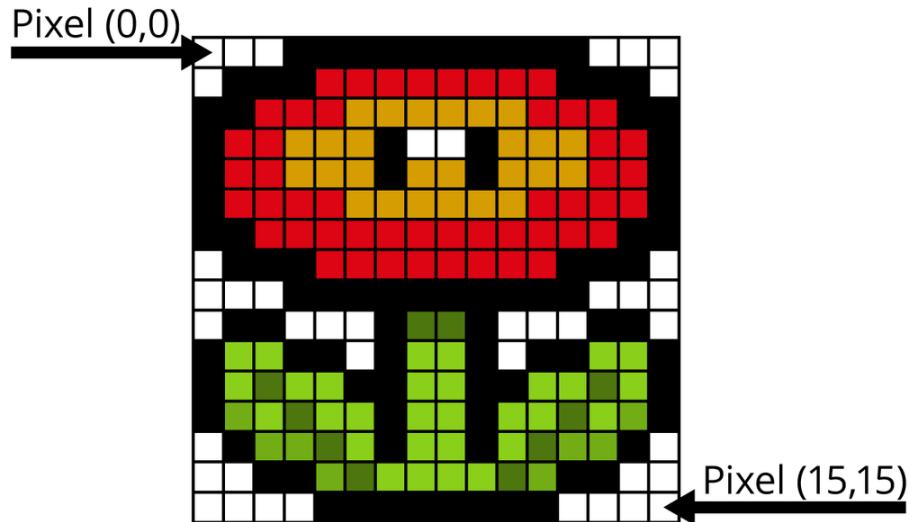
We can apply operations to rows, columns, etc.

- `np.sum(my_matrix, axis = 0)` # sum the values down rows

Let's explore this in Jupyter!

Image processing

We can use higher dimensional numpy arrays to store and manipulate images



Digital images are made up of pixels

Each pixel consists of a red (R), green (G), and Blue (B) color channel

- i.e., we have an “RGB image”

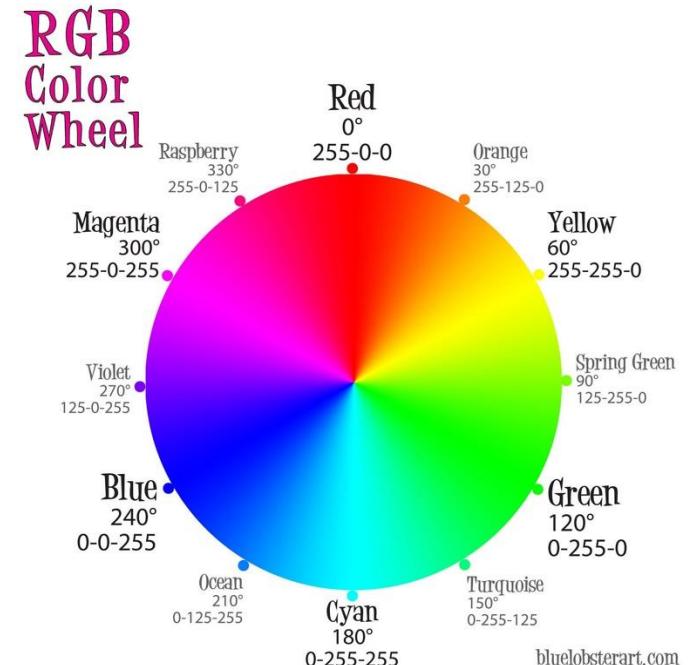
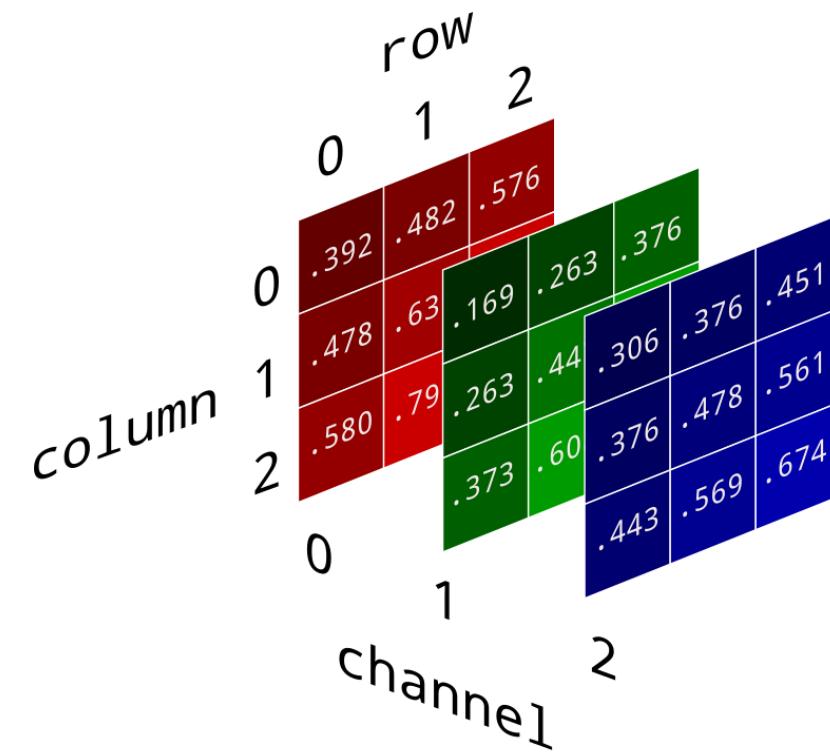


Image processing

We can use 3-dimemsional numerical arrays to store digital RGB images

We can use masking and other array operations to process images



Let's explore this in Jupyter!

Tuples and Dictionaries

Tuples

Tuples are like lists but they are immutable; i.e., once they are created we can't change the values in a tuple.

We can create a tuple using:

- `my_tuple = (10, 20, 30)`

Like lists, we can access elements of tuples using square brackets

- `my_tuple[1]`

We can't change values in tuples:

- `my_tuple[1] = 50 # Error!!!`

Tuples

We can assign values in tuples into regular names using “tuple unpacking”

- `my_tuple = (10, 20, 30)`
- `val1, val2, val3 = my_tuple`
- `val3`

Let's explore this in Jupyter!

Dictionaries



Dictionaries allow you to look up **values** based on a **key**

- i.e., you supply a “key” and the dictionary returns the stored value

We can create dictionaries using the syntax:

- `my_dict = { 'key1': 5, 'key2': 20}`

We can retrieve dictionary values by supplying a key using square brackets []

- `my_dict['key2']`

Let's explore this in Jupyter!



Series and Tables

Pandas: Series and DataFrames

“[pandas](#) is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.”

There are two main data structures in pandas:

- **Series**: represent one-dimensional data
- **DataFrames**: represent data tables
 - i.e., relational data



pandas Series

pandas Series are: One-dimensional ndarray with axis labels

- (including time series)

Example: egg_prices

DATE

1980-01-01	0.879
1980-02-01	0.774
1980-03-01	0.812

Index

values



pandas Series

We can access row elements by Index ***name*** using `.loc`

- `egg_prices.loc["1980-01-01"]`

We can access row elements by Index ***number*** using `.iloc`

- `egg_prices.iloc[0]`

Let's explore this in Jupyter!

pandas DataFrames

Pandas DataFrame hold
Table data

This is one of the most
useful formats to extract
insights from datasets

Often we read data into
a DataFrame using:

- `pd.read_csv("file.csv")`

Cases

Variables

PLAYER	POSITION	TEAM	SALARY
str	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

Let's explore this in Jupyter!

YData: Introduction to Data Science



Class 08: Pandas Series and DataFrames

Overview

Quick review of Boolean masking

Tuples and dictionaries

pandas

- Series
- DataFrames
- Selecting columns and rows from DataFrames
- Sorting values and adding new columns
- Calculating aggregate statistics for separate groups
- If there is time: joining DataFrames



Announcement: Homework 4

Homework 4 is due on Gradescope on **Sunday September 29th at 11pm**

- **Be sure to mark each question on Gradescope!**

How was homework 3?



Quick review of Boolean masking

We can also use Boolean arrays to return values in another array

- This is called "Boolean masking" or "Boolean indexing"

```
my_array = np.array([12, 4, 6, 3, 1])  
boolean_mask = np.array([False, True, False, True, True])
```

```
smaller_array = my_array[boolean_mask]
```

This can be useful for calculating statistics on data that meet particular criteria:

```
np.mean(my_array[my_array < 5]) # what does this do?
```

```
boolean_mask = my_array < 5 # breaking it down into steps...
```

```
values_less_than_5 = my_array[boolean_mask]
```

```
np.mean(values_less_than_5)
```

Let's do a warm-up exercise in Jupyter!

Tuples and Dictionaries

Tuples

Tuples are like lists but they are ***immutable***

- i.e., once they are created we can't change the values in a tuple.

We can create a tuple using:

```
my_tuple = (10, 20, 30)
```

Like lists, we can access elements of tuples using square brackets

```
my_tuple[1]
```

We can't change values in tuples:

```
my_tuple[1] = 50 # Error!!!
```

Tuples

We can assign values in tuples into regular names using “tuple unpacking”

```
my_tuple = (10, 20, 30)
```

```
val1, val2, val3 = my_tuple
```

```
val3
```

Let's explore this in Jupyter!

Dictionaries



Dictionaries allow you to look up **values** based on a **key**

- i.e., you supply a “key” and the dictionary returns the stored value

We can create dictionaries using the syntax:

```
my_dict = { 'key1': 5, 'key2': 20}
```

We can retrieve dictionary values by supplying a key using square brackets []

```
my_dict['key2']
```

Let's explore this in Jupyter!



Series and Tables

Pandas: Series and DataFrames

“[pandas](#) is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.”

There are two main data structures in pandas:

- **Series**: represent one-dimensional data
- **DataFrames**: represent data tables
 - i.e., relational data



pandas Series

pandas Series are: One-dimensional ndarray with axis labels

- (including time series)

Example: egg_prices

DATE

1980-01-01	0.879
1980-02-01	0.774
1980-03-01	0.812

Index

values



pandas Series

We can access elements by Index ***name*** using `.loc`

```
egg_prices.loc["1980-01-01"]
```

We can access elements by Index ***number*** using `.iloc`

```
egg_prices.iloc[0]
```

Since pandas Series are just ndarrays with an Index, all the numpy functions will work on Series

Let's explore this in Jupyter!

pandas DataFrames

Pandas DataFrame hold
Table data

This is one of the most
useful formats to extract
insights from datasets

Often we read data into
a DataFrame using:

```
pd.read_csv("file.csv")
```

Cases

Variables

	title	clean_test	binary
21 & Over	notalk	FAIL	
Dredd 3D	ok	PASS	
12 Years a Slave	notalk	FAIL	
2 Guns	notalk	FAIL	
42	men	FAIL	

Motivation: flight information

Questions:

- Is anyone traveling by airplane for Thanksgiving?
- Is anyone flying out of New York City?
- Has anyone bought plane tickets?

Let's look at flight delay data from NYC airports!

Selecting columns from a DataFrame

We can select a column from a DataFrame using square brackets:

```
my_df["my_col"]      # returns a Series!
```

We can select multiple columns from a DataFrame by passing a list into the square brackets

```
my_df[["col1", "col2"]]
```

Let's explore this in Jupyter!

Extracting rows from a DataFrame

We can extract rows from a DataFrame by:

1. The position they appear in the DataFrame
2. The Index values

We use the `.iloc[]` property to extract values by ***position***

`my_df.iloc[0]`

We use the `.loc[]` property to extract values by ***Index value***

`my_df.loc["index_name"]`

Extracting rows from a DataFrame

We can also extract rows through using Boolean masking

For example:

```
bool_mask = my_df["col_name"] == 7  
my_df.loc[bool_mask]
```

Or in one step: `my_df [my_df["col_name"] == 7]`

Let's explore this in Jupyter!

Sorting rows from a DataFrame

We can sort values in a DataFrame using `.sort_values("col_name")`

- `my_df.sort_values("col_name")`

We can sort from highest to lowest by setting the argument `ascending = False`

- `my_df.sort_values("col_name", ascending = False)`

Let's explore this in Jupyter!

Adding new columns and renaming columns

We can add a column to a data frame using square brackets. For example:

```
my_df["new_col"] = values_array  
my_df["new col"] = my_df["col1"] + my_df["col2"]
```

We can rename columns by passing a dictionary to the `.rename()` method.

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

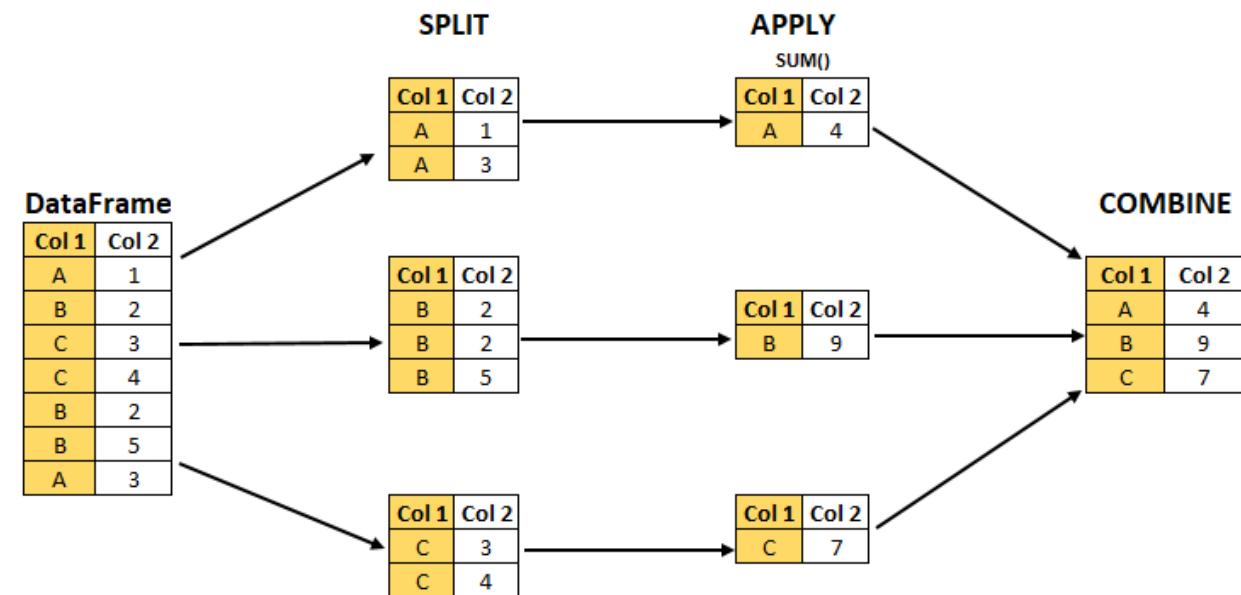
Let's explore this in Jupyter!

Creating aggregate statistics by group

We can get statistics separately by group using the `.groupby()` and `.agg()` methods

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

This implements:
“Split-apply-combine”



Creating aggregate statistics by group

There are several ways to get multiple statistics by group

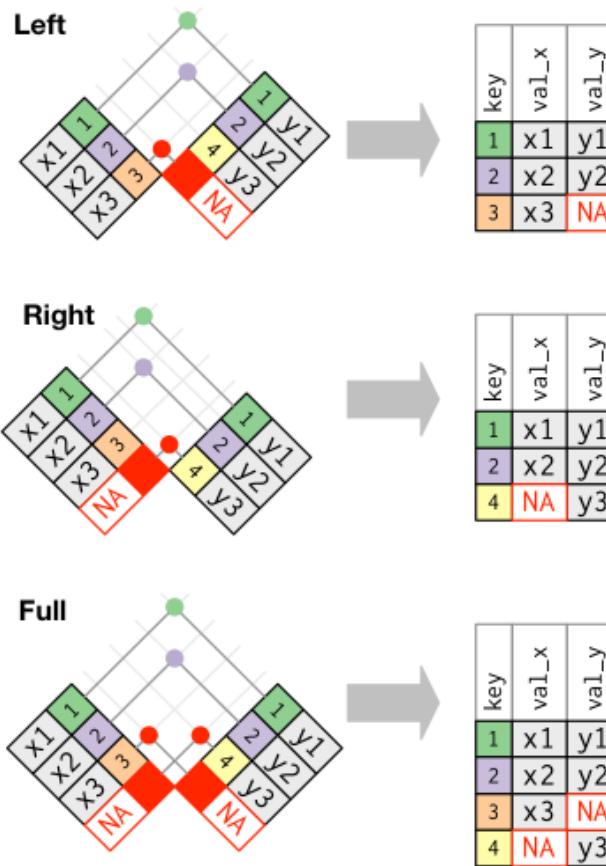
Perhaps the most useful way is to use the syntax:

```
my_df.groupby("group_col_name").agg(  
    new_col1 = ('col_name1', 'statistic_name1'),  
    new_col2 = ('col_name2', 'statistic_name2'),  
    new_col3 = ('col_name3', 'statistic_name3')  
)
```

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

Let's explore this in Jupyter!

Joining data frames



Left and right tables

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

DataFame: x_df

Index `y_vals`

1	y1
2	y2
3	y3

DataFrame: y_df

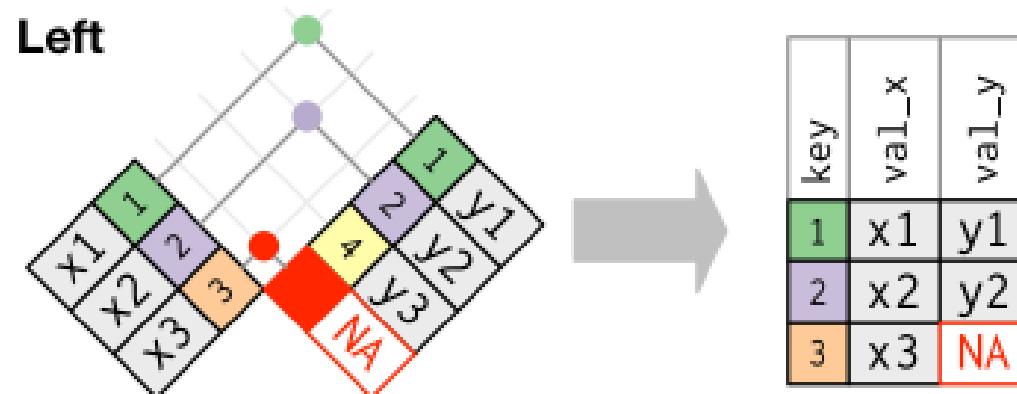
We can join these two DataFrames into a single DataFrame by aligning rows with the same Index value using the general syntax: `x_df.join(y_df)`

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

Left joins

Left joins keep all rows in the left table.

Data from right table is added when there is a matching Index value, otherwise NA as added

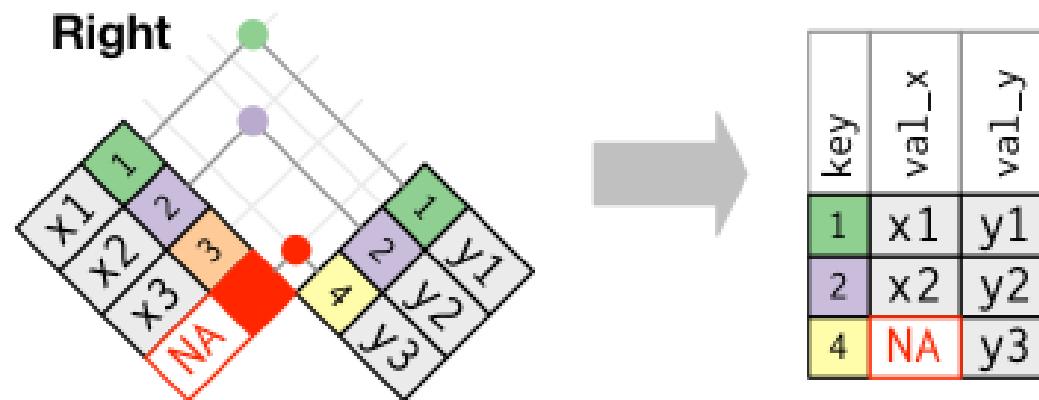


```
x_df.join(y_df, how = "left")
```

Right joins

Right joins keep all rows in the right table.

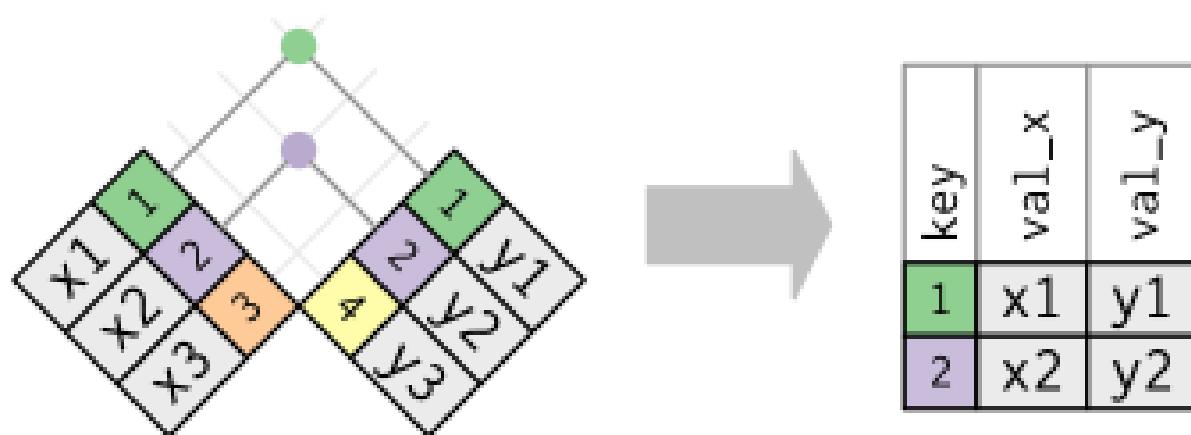
Data from left table added when there is a matching Index value
otherwise NA as added



```
x_df.join(y_df, how = "right")
```

Inner joins

Inner joins only keep rows in which there are matches between the Index values in both tables.

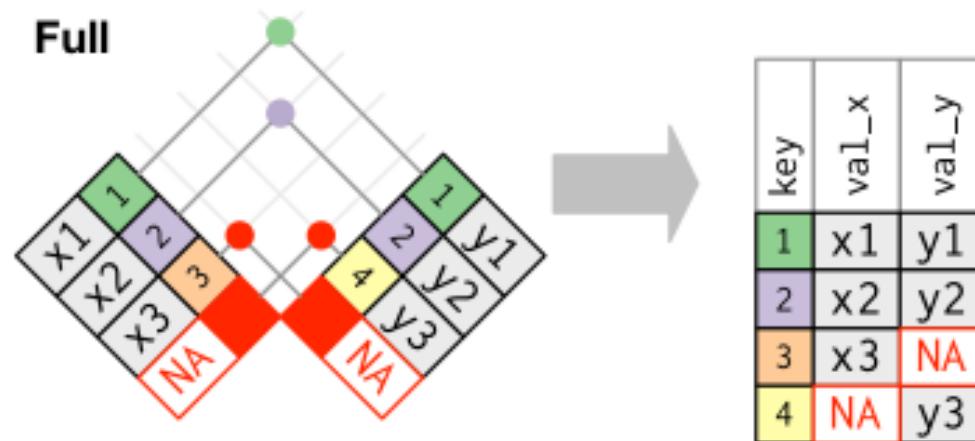


```
x_df.join(y_df, how = "inner")
```

Full (outer) joins

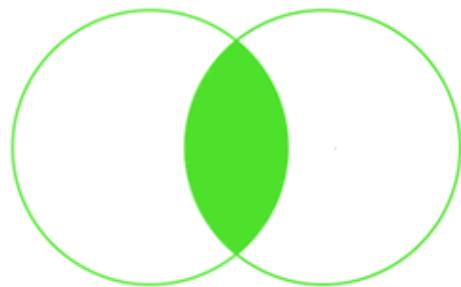
Full joins keep all rows in both table

NAs are added where there are no matches

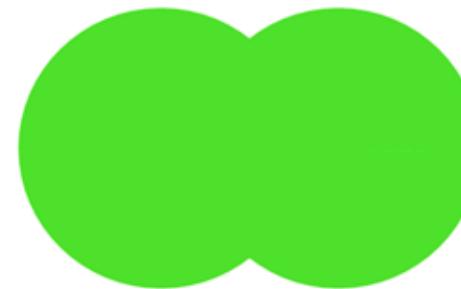


```
x_df.join(y_df, how = "outer")
```

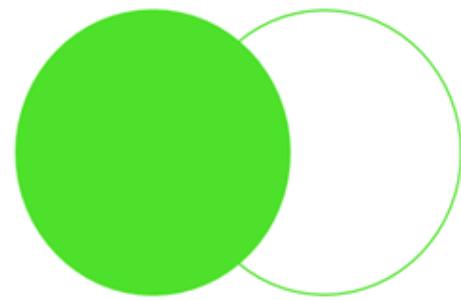
Summary



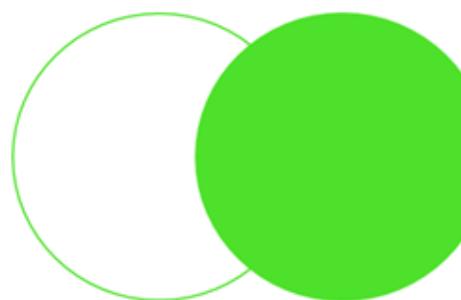
INNER JOIN



FULL OUTER JOIN



LEFT JOIN



RIGHT JOIN

“Merging” data frames

We can also join DataFrames based on values in **columns** rather than based on the DataFrames Index values

To do this we can use the merge method which has the form:

- `x_df.merge(y_df, how = "left", left_on = "x_col", right_on = "y_col")`

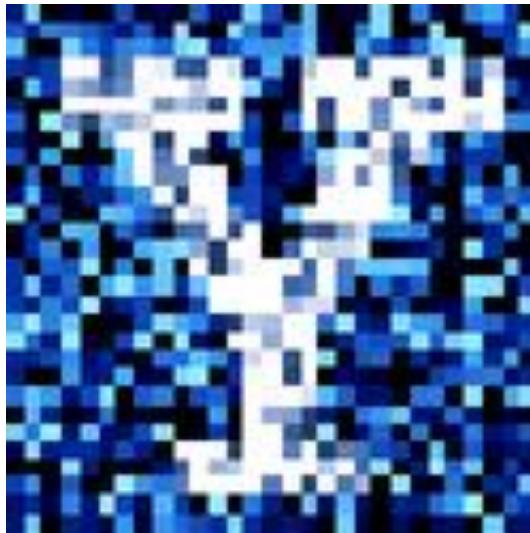
All the same types of joins still work

- i.e., we can do: left, right, inner and outer joins

Let's explore this in Jupyter!

Questions?

YData: Introduction to Data Science



Class 09: Pandas continued

Overview

Quick review of pandas:

- Tuples and dictionaries
- Series and DataFrames methods

Continuation of pandas:

- Calculating aggregate statistics for separate groups
- Joining DataFrames

If there is time:

- Additional practice!



Announcement: Homework 4

Homework 2 is due on Gradescope on **Sunday September 29th at 11pm**

- **Be sure to mark each question on Gradescope!**



Quick review: tuples and dictionaries

Tuples are like lists but they are immutable

- `my_tuple = (10, 20, 30) # Creating a tuple`
- `my_tuple[1] # accessing items`
- `my_tuple[1] = 50 # Error! Tuples are immutable`
- `val1, val2, val3 = my_tuple # tuple unpacking`



Dictionaries allow you to look up **values** based on a **key**

- `my_dict = { 'key1': 5, 'key2': 20}`
- `my_dict['key2']`

Review: pandas Series and DataFrames

There are two main data structures in pandas:

- **Series**: represent one-dimensional data
 - **DataFrames**: represent data tables
 - i.e., relational data



Example: egg_prices

pandas Series are: One-dimensional ndarray with an Index

- `egg_prices.iloc[0]` # use index location
 - `egg_prices.loc["1980-01-01"]` # use Index names

DATE	
1980-01-01	0.879
1980-02-01	0.774
1980-03-01	0.812

Index

values (ndarray)

Review: pandas DataFrames

Pandas DataFrame hold Table data

Extracting columns:

- `my_df["my_col"]` # returns a Series!
- `my_df[["my_col"]]` # returns a DataFrame
- `my_df[["col1", "col2"]]` # multiple columns

Extracting rows

- `my_df.iloc[0:3]` # by position
- `my_df.loc["index_name"]` # by index name

Cases

tailnum	arr_delay	dep_delay	carrier
N25201	205.0	203.0	UA
N830DN	53.0	78.0	DL
N807JB	34.0	47.0	B6
N265JB	166.0	173.0	B6
N17730	211.0	228.0	UA

Sorting rows

`my_df.sort_values("col_name")`
`my_df.sort_values("col_name", ascending = True)`

Variables

Warm-up exercises

Warm-up 1: tuples and dictionaries

Warm-up 2: DataFrame operations

- Motivation: [The Dow Jones Industrial Average \(DJIA or Dow\)](#), is a stock market index of 30 prominent companies listed on stock exchanges in the United States.



Date	Year	Month	Day	Open	High	Low	Close	Volume	
1992-01-02	1992		1	Thursday	3152.100098	3172.629883	3139.310059	3172.399902	23550000
1992-01-03	1992		1	Friday	3172.399902	3210.639893	3165.919922	3201.500000	23620000
1992-01-06	1992		1	Monday	3201.500000	3213.330078	3191.860107	3200.100098	27280000
1992-01-07	1992		1	Tuesday	3200.100098	3210.199951	3184.479980	3204.800049	25510000
1992-01-08	1992		1	Wednesday	3204.800049	3229.199951	3185.820068	3203.899902	29040000

Let's try it in Jupyter!

Pandas continued

Adding new columns and renaming columns

We can add a column to a data frame using square brackets. For example:

```
my_df["new_col"] = values_array  
my_df["new col"] = my_df["col1"] + my_df["col2"]
```

We can rename columns by passing a dictionary to the `.rename()` method.

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

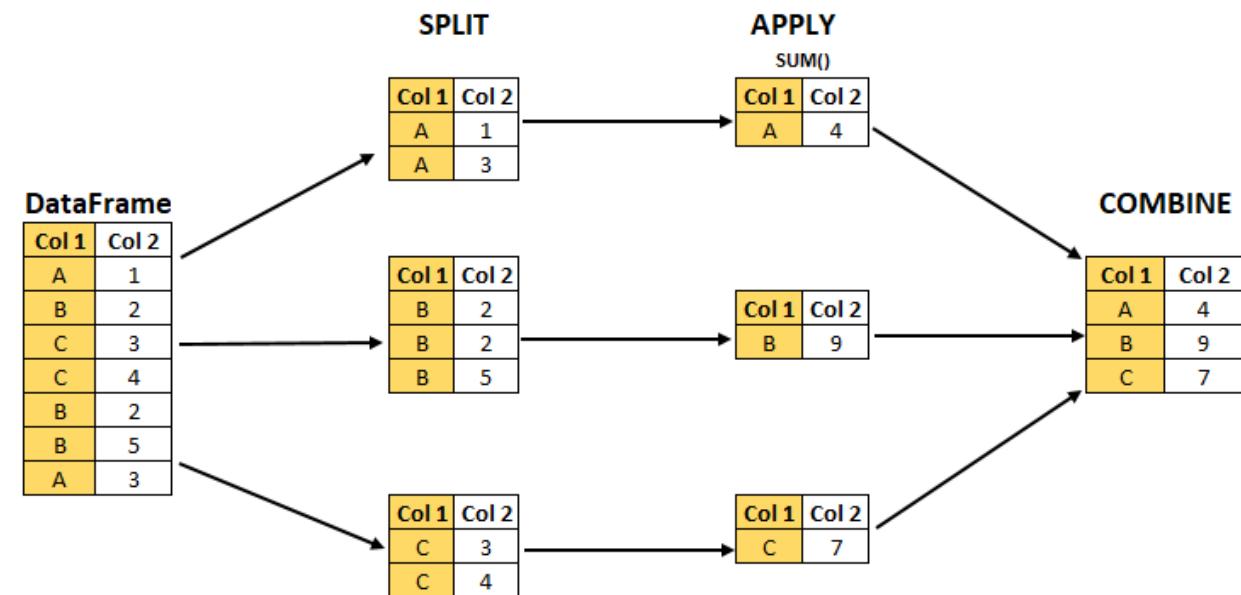
Let's explore this in Jupyter!

Creating aggregate statistics by group

We can get statistics separately by group using the `.groupby()` and `.agg()` methods

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

This implements:
“Split-apply-combine”



Creating aggregate statistics by group

There are several ways to get multiple statistics by group

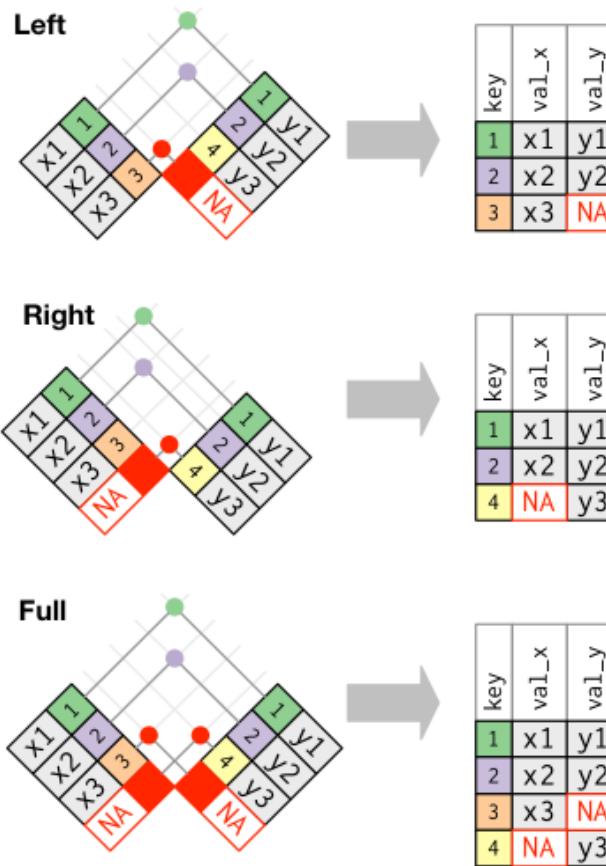
Perhaps the most useful way is to use the syntax:

```
my_df.groupby("group_col_name").agg(  
    new_col1 = ('col_name1', 'statistic_name1'),  
    new_col2 = ('col_name2', 'statistic_name2'),  
    new_col3 = ('col_name3', 'statistic_name3')  
)
```

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

Let's explore this in Jupyter!

Joining data frames



Left and right tables

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

DataFame: x_df

Index `y_vals`

1	y1
2	y2
3	y3

DataFrame: y_df

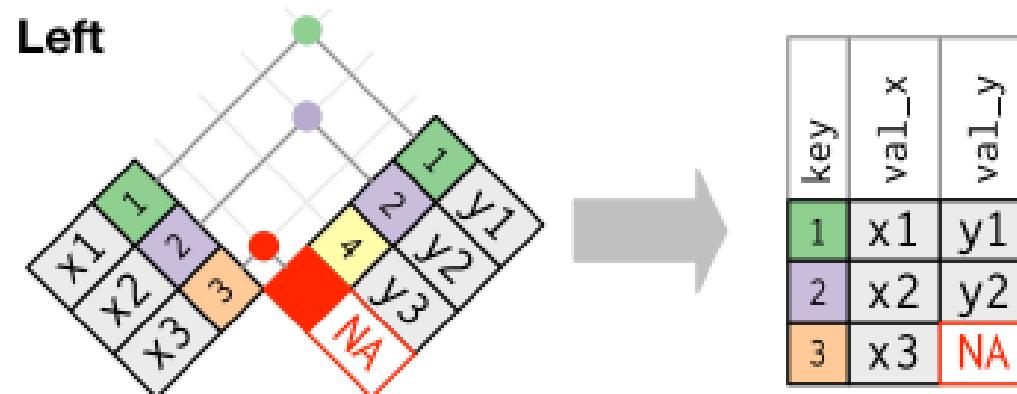
We can join these two DataFrames into a single DataFrame by aligning rows with the same Index value using the general syntax: `x_df.join(y_df)`

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

Left joins

Left joins keep all rows in the left table.

Data from right table is added when there is a matching Index value, otherwise NA as added

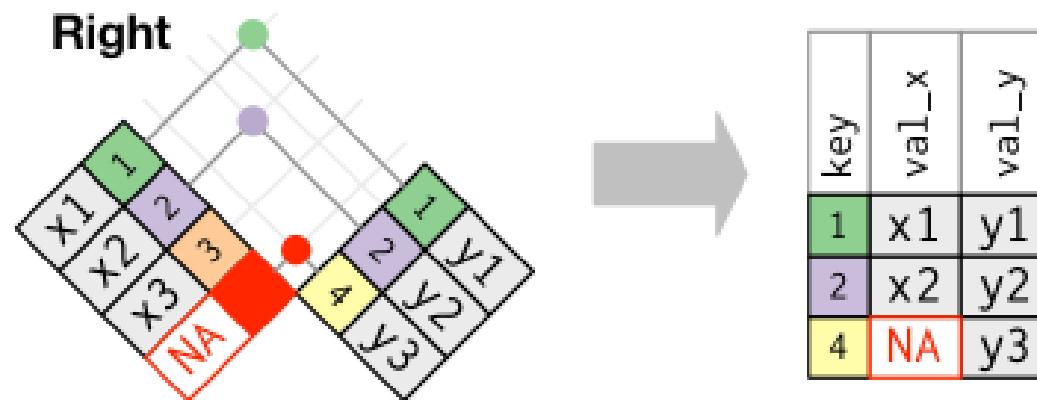


```
x_df.join(y_df, how = "left")
```

Right joins

Right joins keep all rows in the right table.

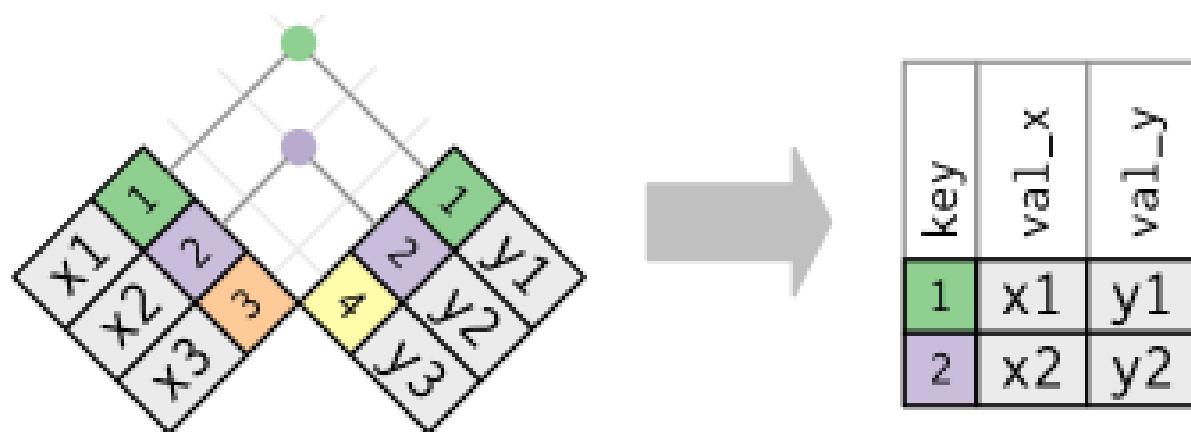
Data from left table added when there is a matching Index value
otherwise NA as added



```
x_df.join(y_df, how = "right")
```

Inner joins

Inner joins only keep rows in which there are matches between the Index values in both tables.

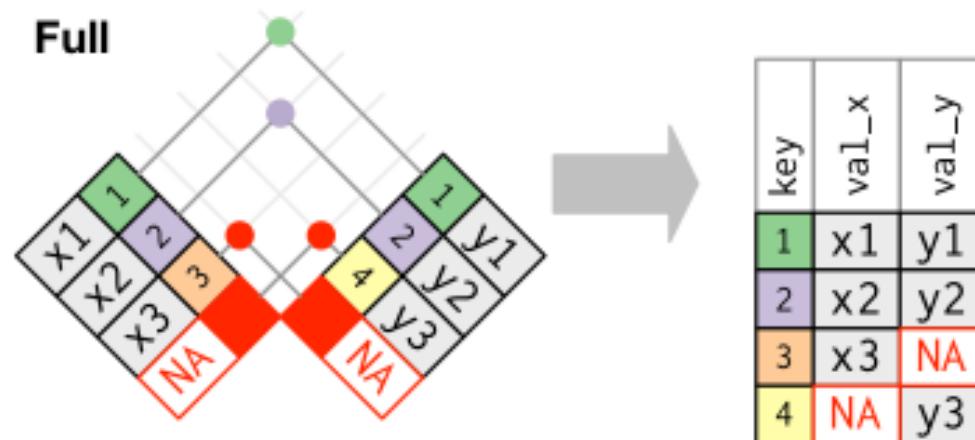


```
x_df.join(y_df, how = "inner")
```

Full (outer) joins

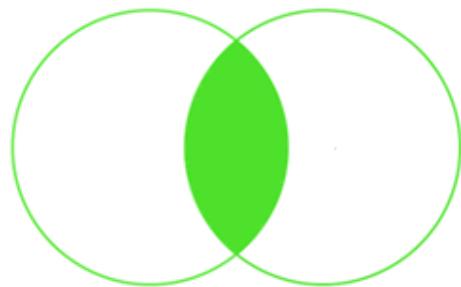
Full joins keep all rows in both table

NAs are added where there are no matches



```
x_df.join(y_df, how = "outer")
```

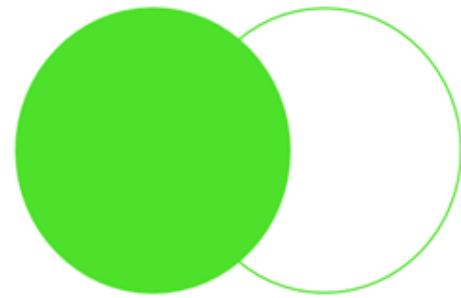
Summary



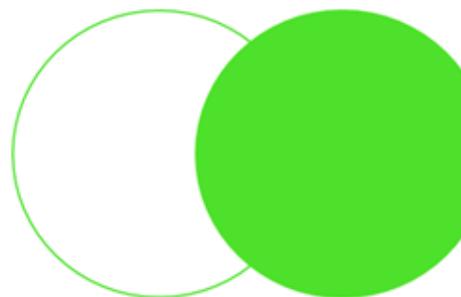
INNER JOIN



FULL OUTER JOIN



LEFT JOIN



RIGHT JOIN

“Merging” data frames

We can also join DataFrames based on values in **columns** rather than based on the DataFrames Index values

To do this we can use the merge method which has the form:

- `x_df.merge(y_df, how = "left", left_on = "x_col", right_on = "y_col")`

All the same types of joins still work

- i.e., we can do: left, right, inner and outer joins

Let's explore this in Jupyter!

Let's do a few more practice exercises!

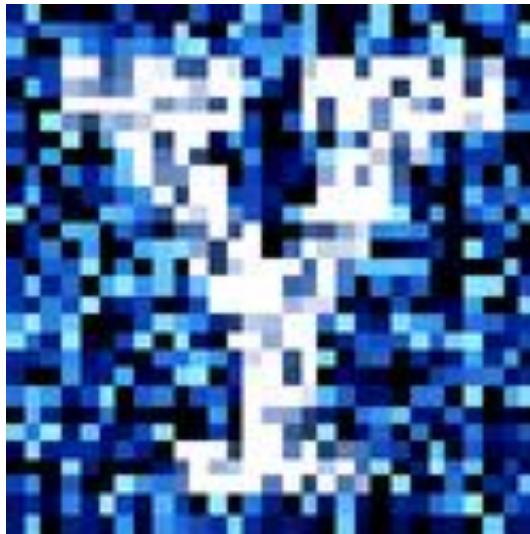
Work in pairs to see if you can calculate and visualize how the mean delay differs for:

1. Different hours of the day
2. Months of the year
3. Airport flight left from

If you solve these, see if you can calculate how the mean delay differs by wind speed

- You will need the *nyc23_weather.csv* to solve this

YData: Introduction to Data Science



Class 10: Data visualization

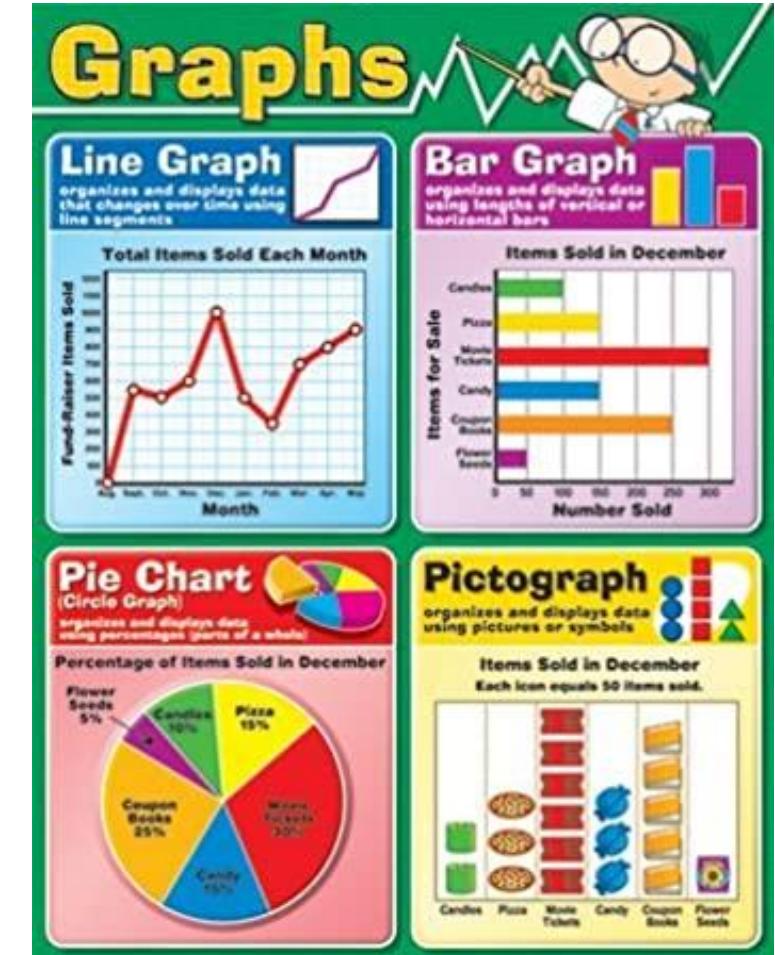
Overview

Quick review and warm-up exercise of pandas
DataFrames

History of Data Visualization

Review and additional features of visualizing
data with matplotlib

If there is time: Visualizing data with seaborn



Much of what we cover is a review and will be good practice for the midterm

Announcement: Homework 5

Homework 5 has been posted!

It is due on Gradescope on **Sunday October 6th 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 for the midterm exam

Also, please fill out a survey to help Mark adjust his office hours

- https://docs.google.com/forms/d/e/1FAIpQLScxpt3uO_rS4fn0HJMEpXBkvgQEU5FapG8aeyAPNksRtWPp8Q/formrestricted

Midterm exam

Thursday October 10th **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](#)

- 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., `sns.catplot(data = , x = , y = , hue = , kind = "strip"/"swarm")`

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

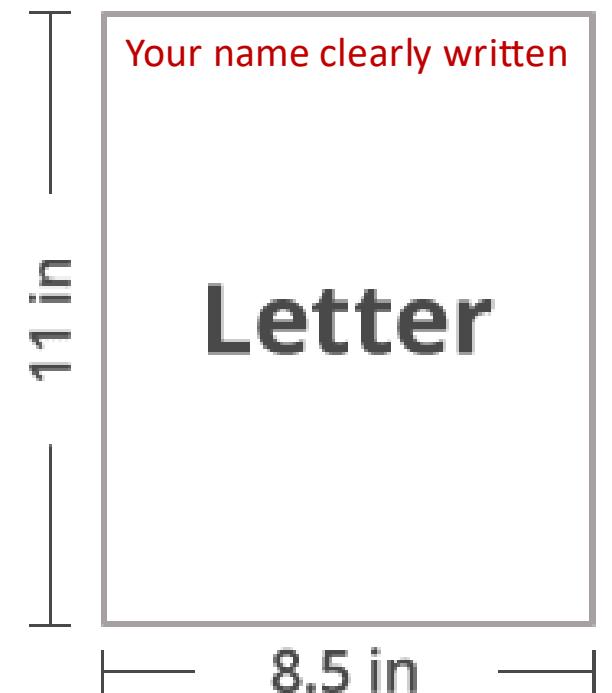
- Your name on the upper left of both sides of the paper

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



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:

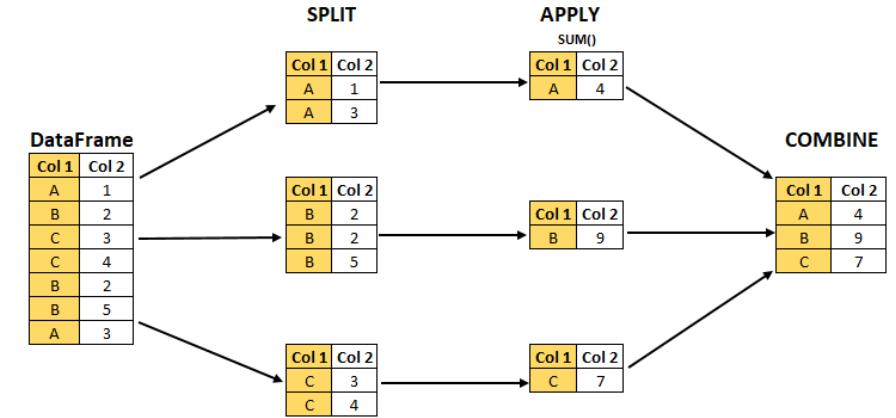
- `rename_dictionary = {"old_col_name": "new_col_name"}`
- `my_df.rename(columns = rename_dictionary)`

Quick review: 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`

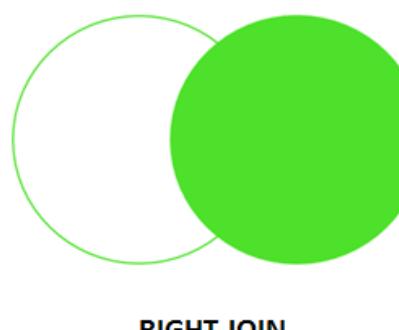
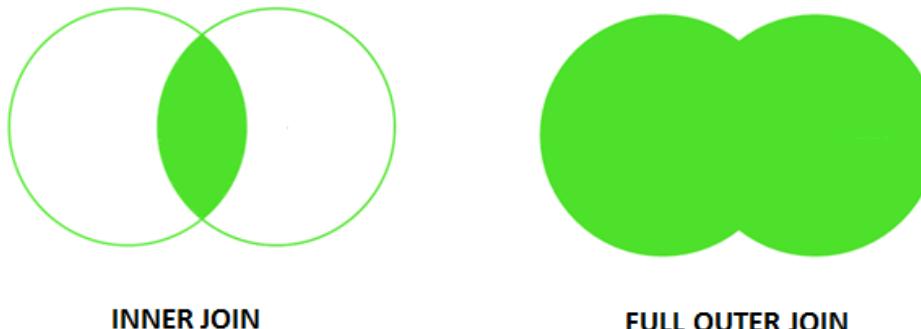
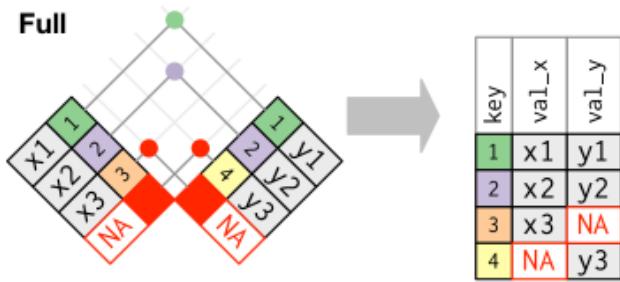
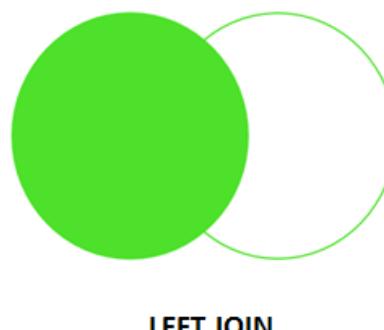
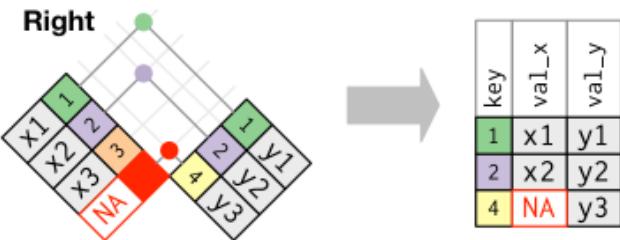
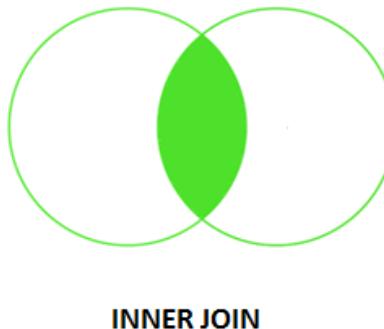
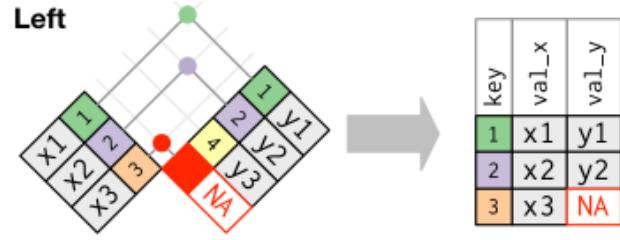
1	y1
2	y2
3	y3

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?



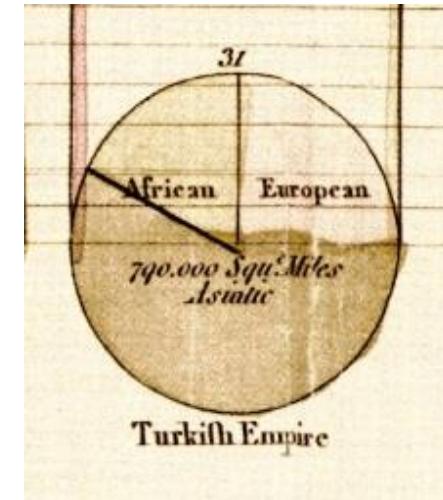
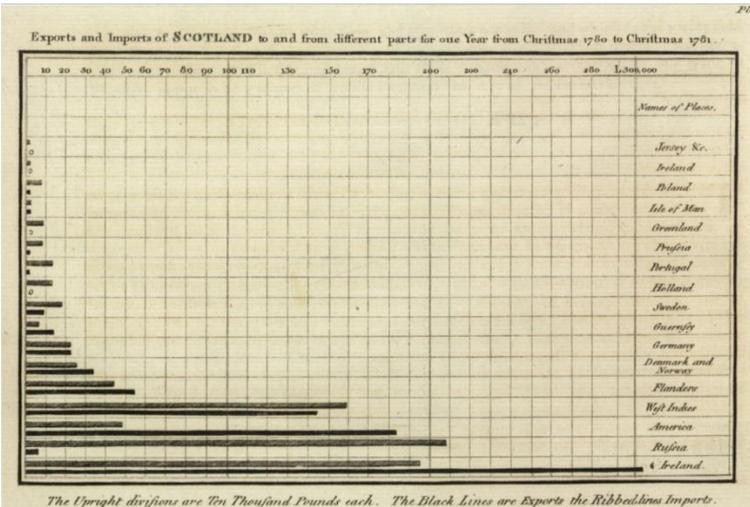
*Whatever relates to extent and quantity may be represented by geometrical figures. Statistical projections **which speak to the senses without fatiguing the mind**, possess the advantage of fixing the attention on a great number of important facts.*

—Alexander von Humboldt, 1811

A very brief history of data visualization

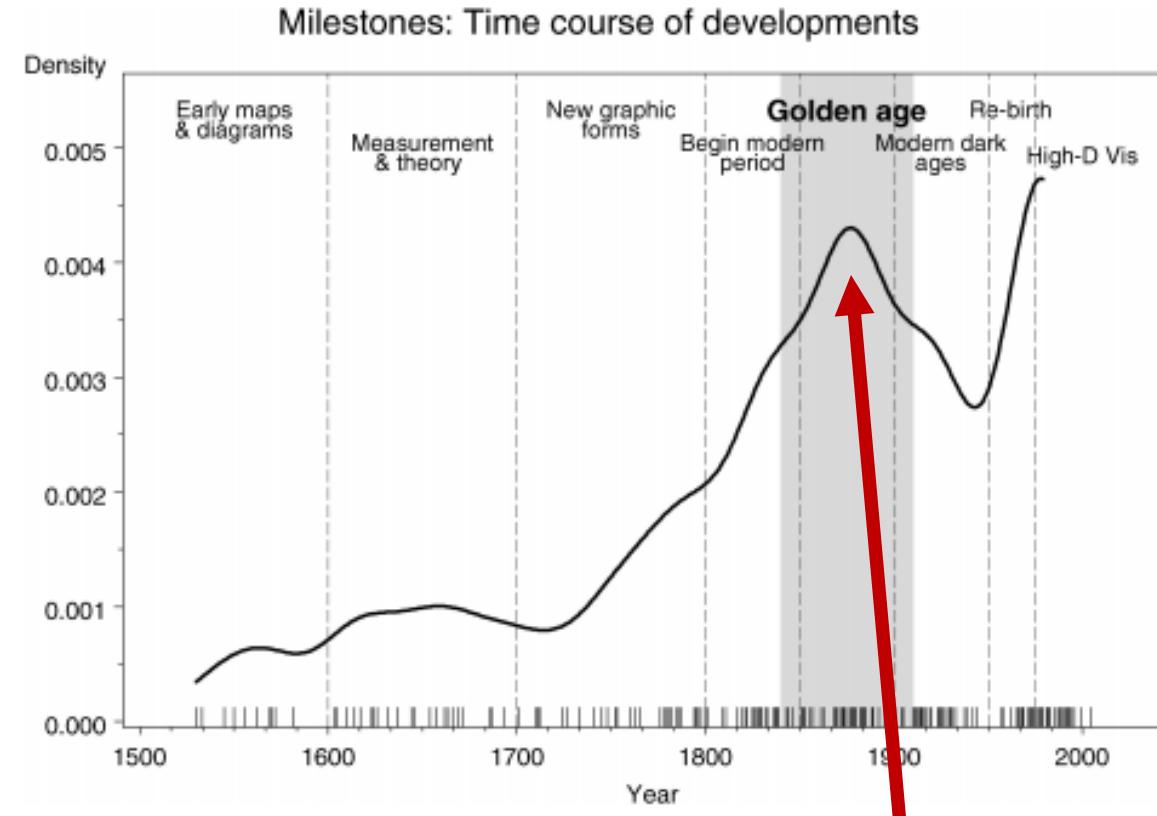
The age of modern statistical graphs began around the beginning of the 19th century

[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

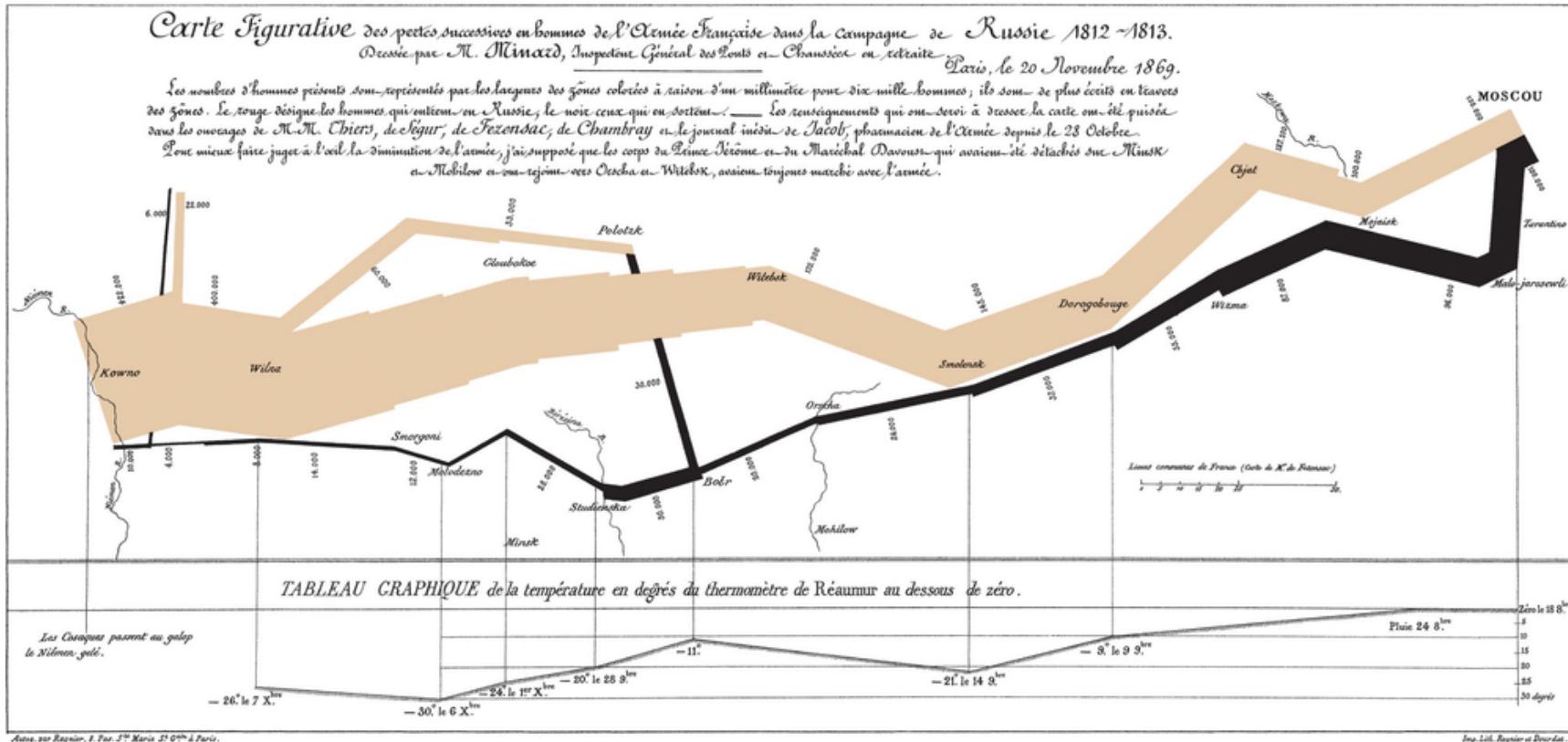


A very brief history of data visualization

Joseph Minard (1781-1870)

Map of Napoleon's march on Russia

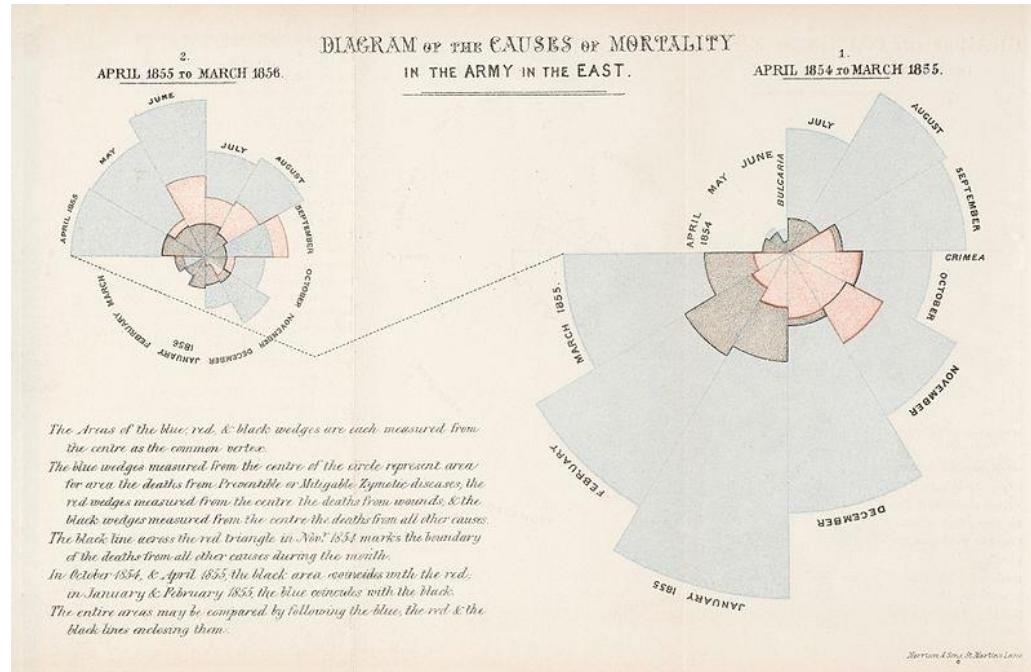
"It may well be the best statistical graphic ever drawn" – E. Tufte



A very brief history of data visualization

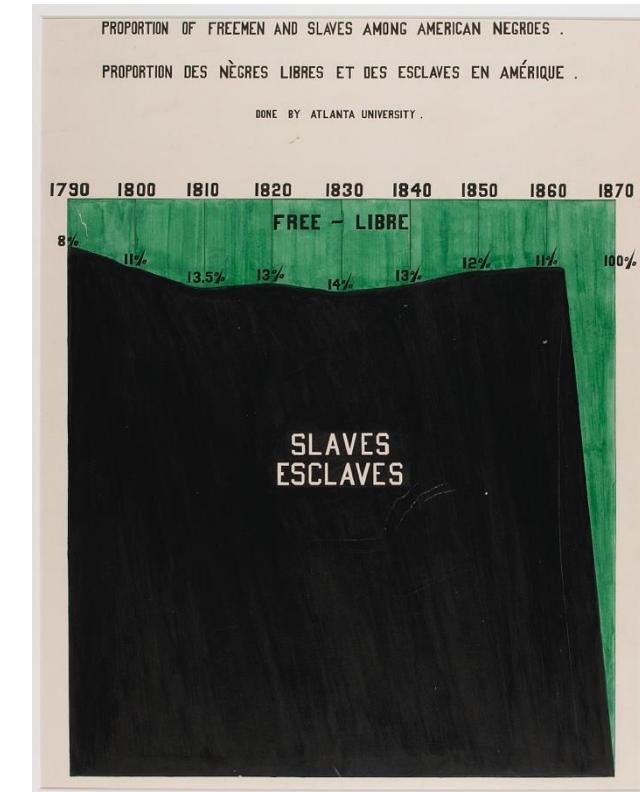
Florence Nightingale (1820-1910)

Causes of mortality in the army in the east



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

Percent of African Americans who were slaves

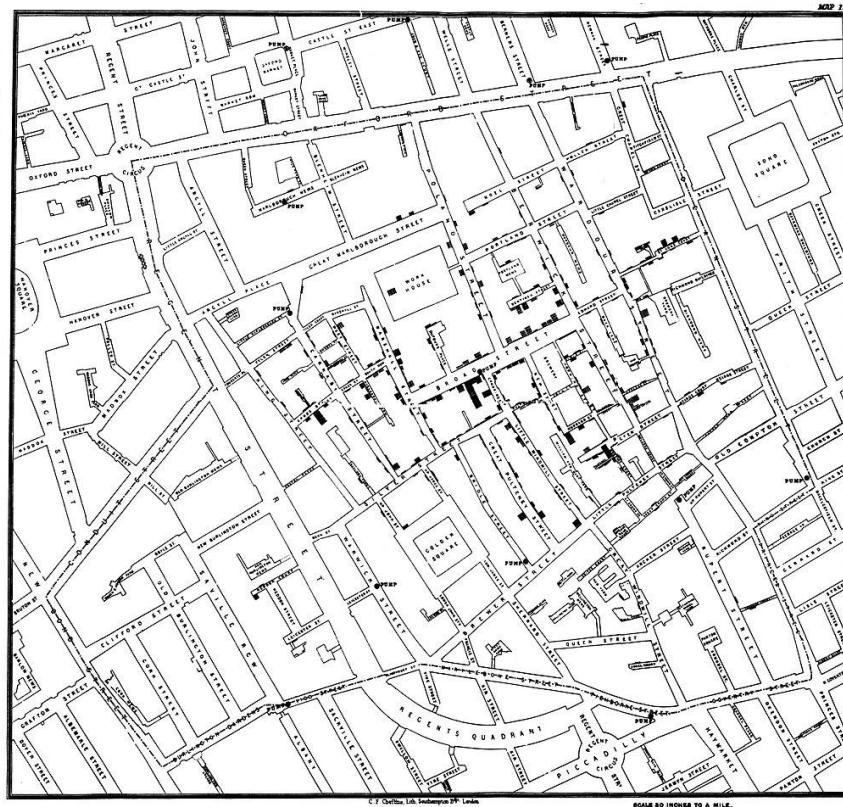


[See Du Bois Visualization Challenge](#)

A very brief history of data visualization

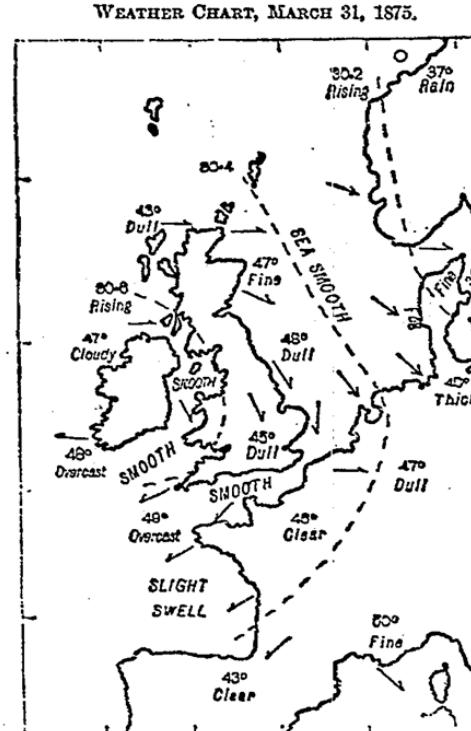
John Snow (1813-1858)

Clusters of cholera cases in London epidemic of 1854



Francis Galton (1822-1911)

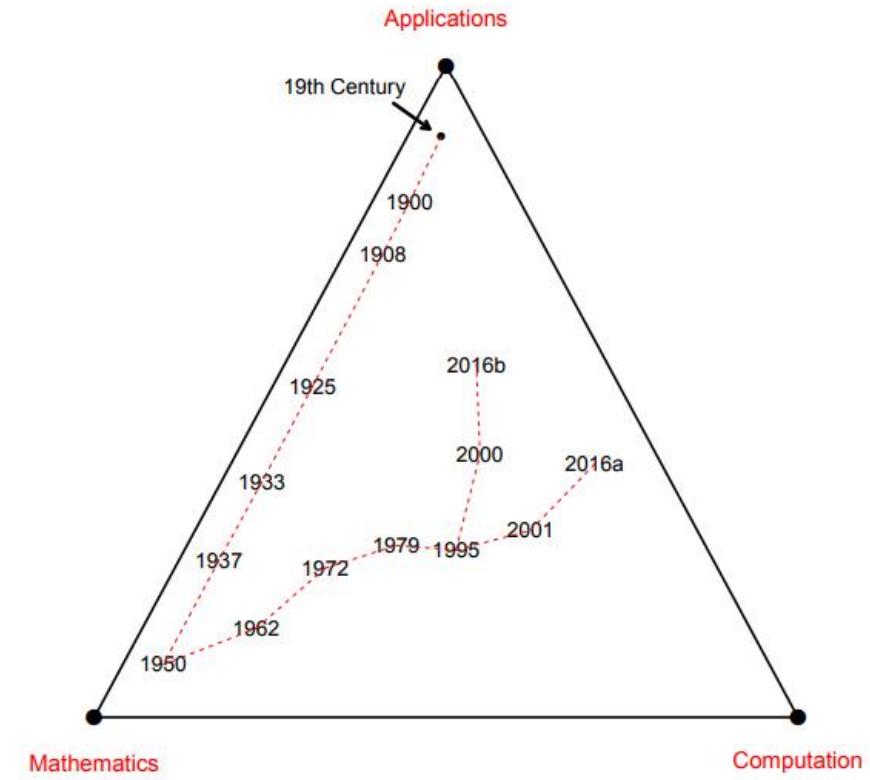
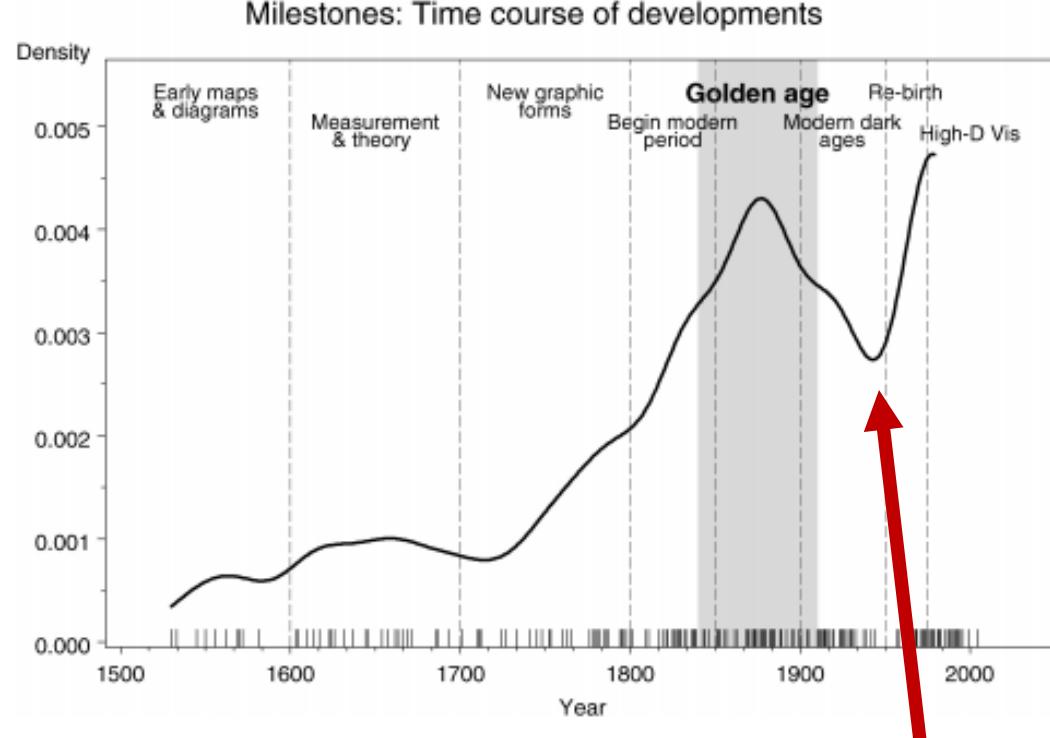
First weather map published in a newspaper (1875)



The dotted lines indicate the gradations of barometric pressure
The variations of the temperature are marked by figures, the state
of the sea and sky by descriptive words, and the direction of the wind
by arrows—barbed and feathered according to its force. © denotes
calm.

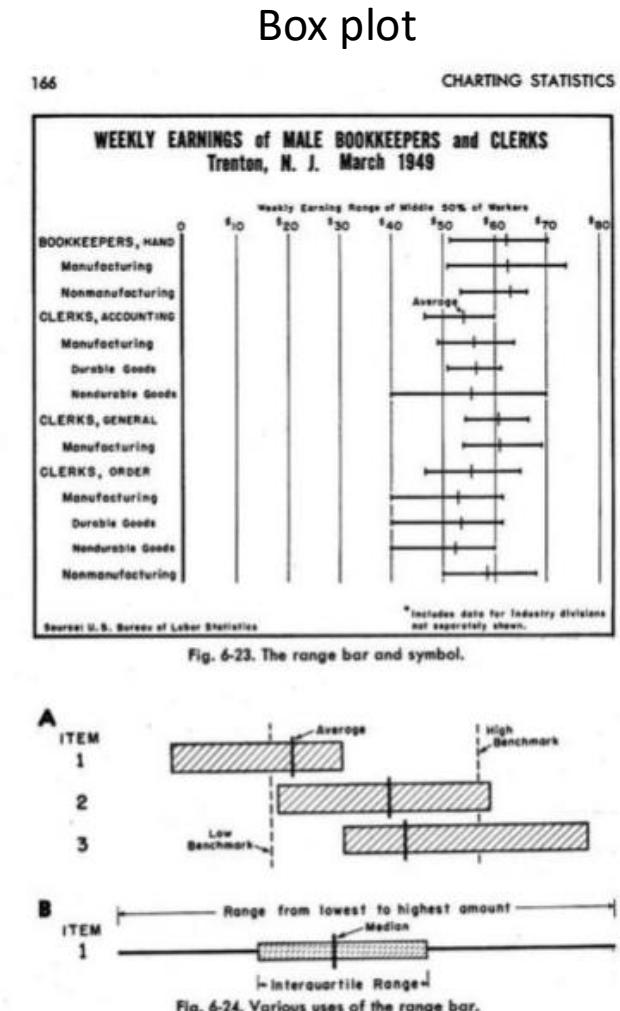
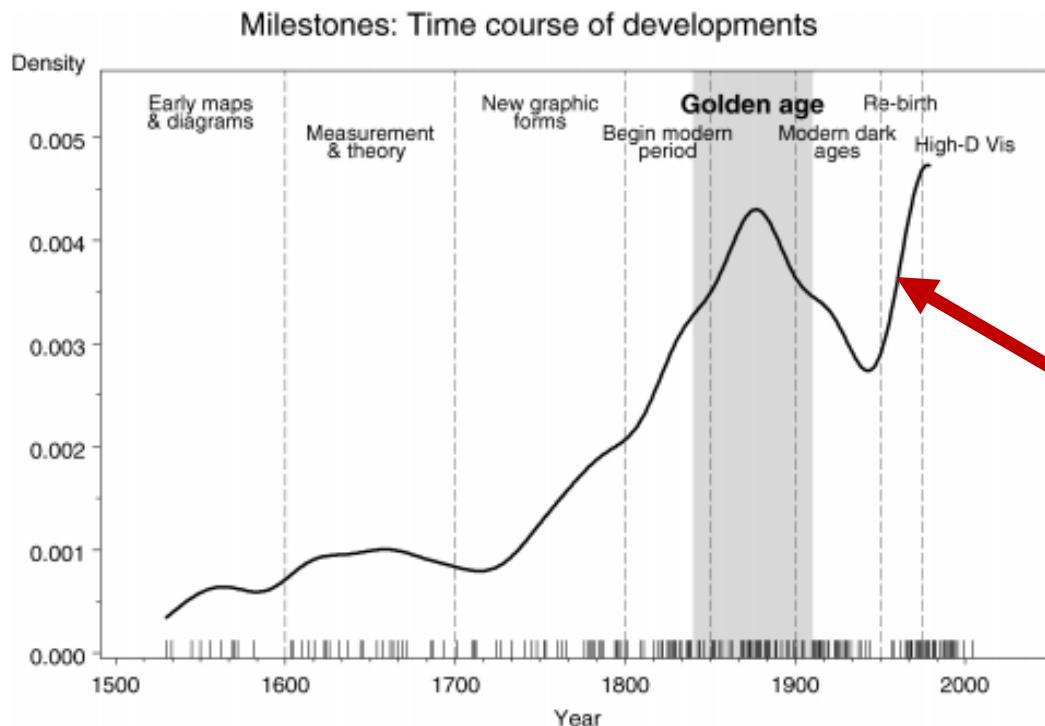
A very brief history of data visualization

“Graphical dark ages” around 1950



A very brief history of data visualization

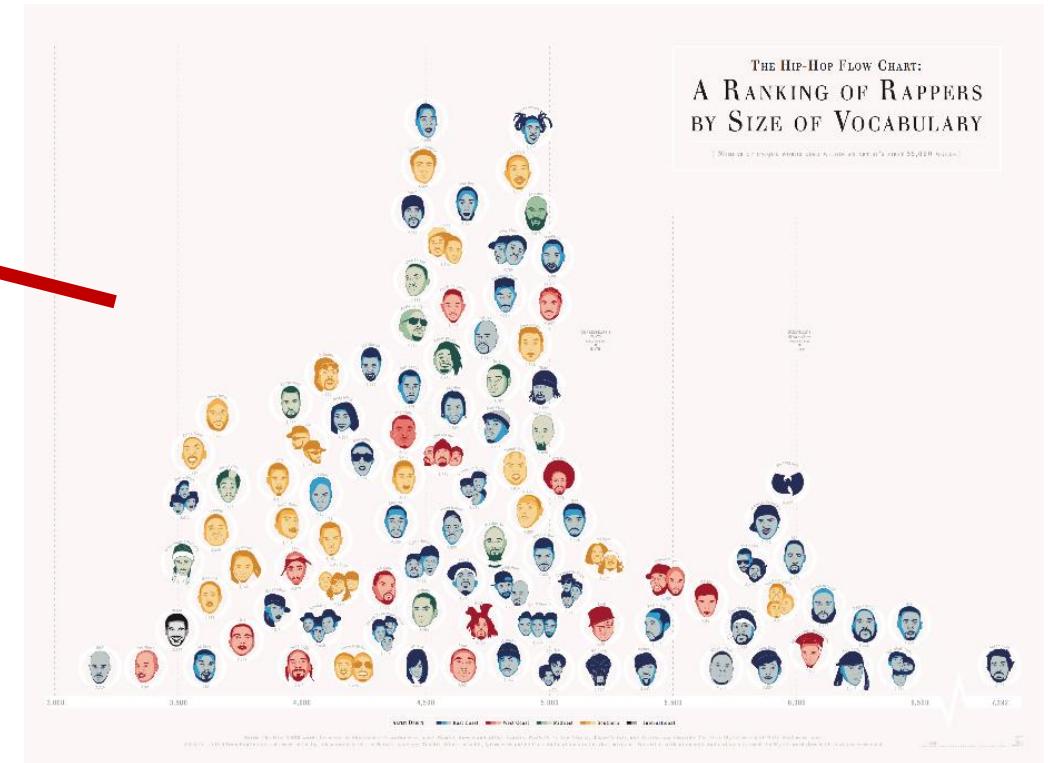
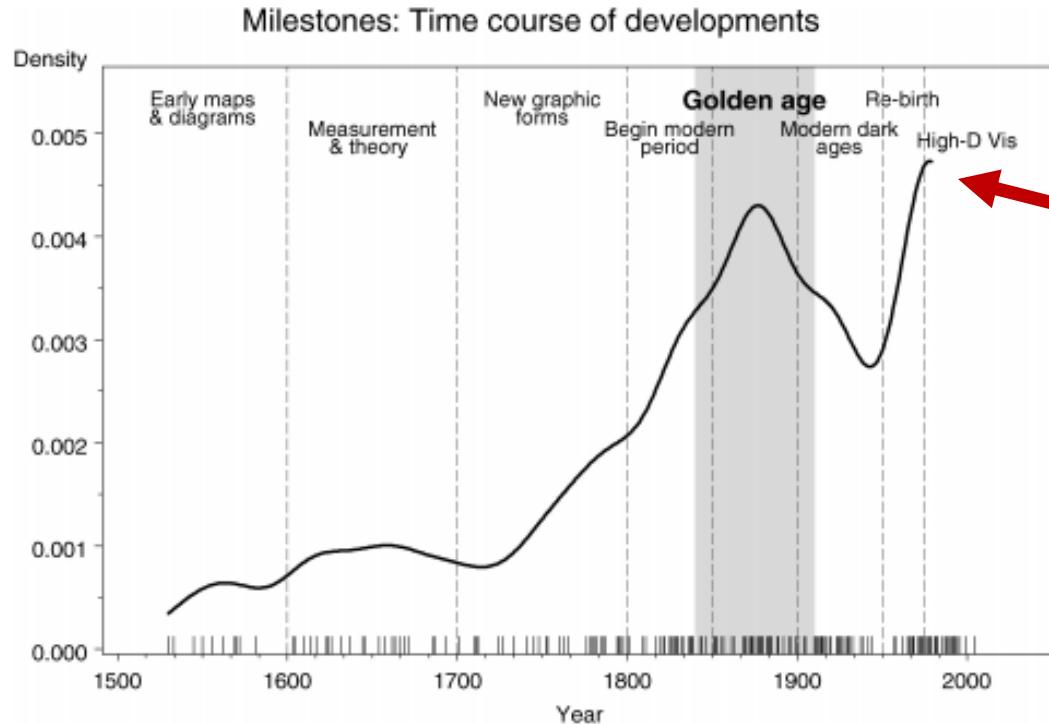
Currently undergoing a “Graphical re-birth”



[Spear 1952, Tukey 1970](#)

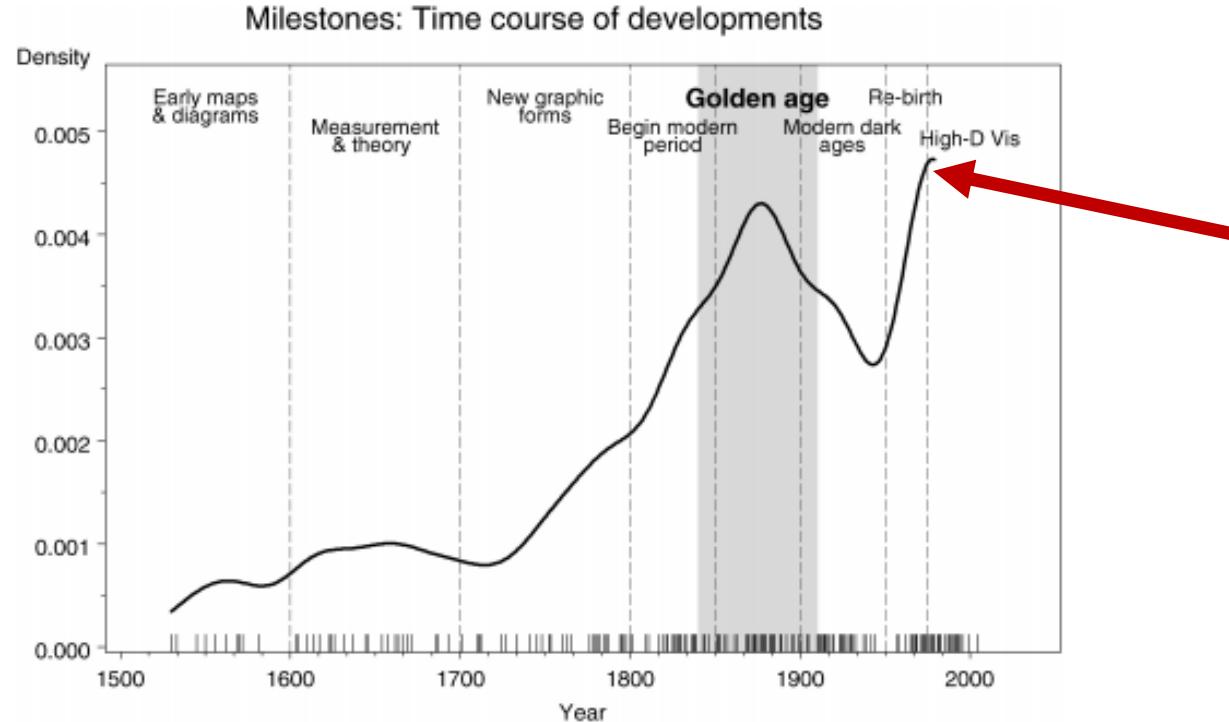
A very brief history of data visualization

Currently undergoing a “Graphical re-birth”



A very brief history of data visualization

Currently undergoing a “Graphical re-birth”

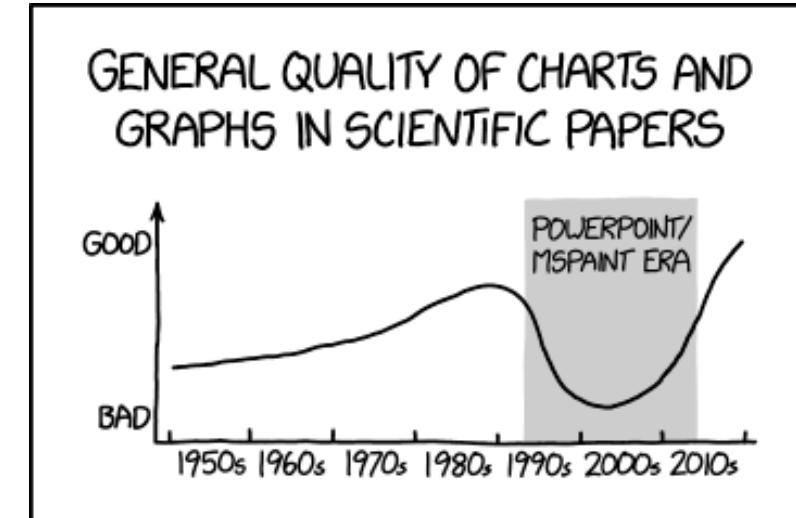
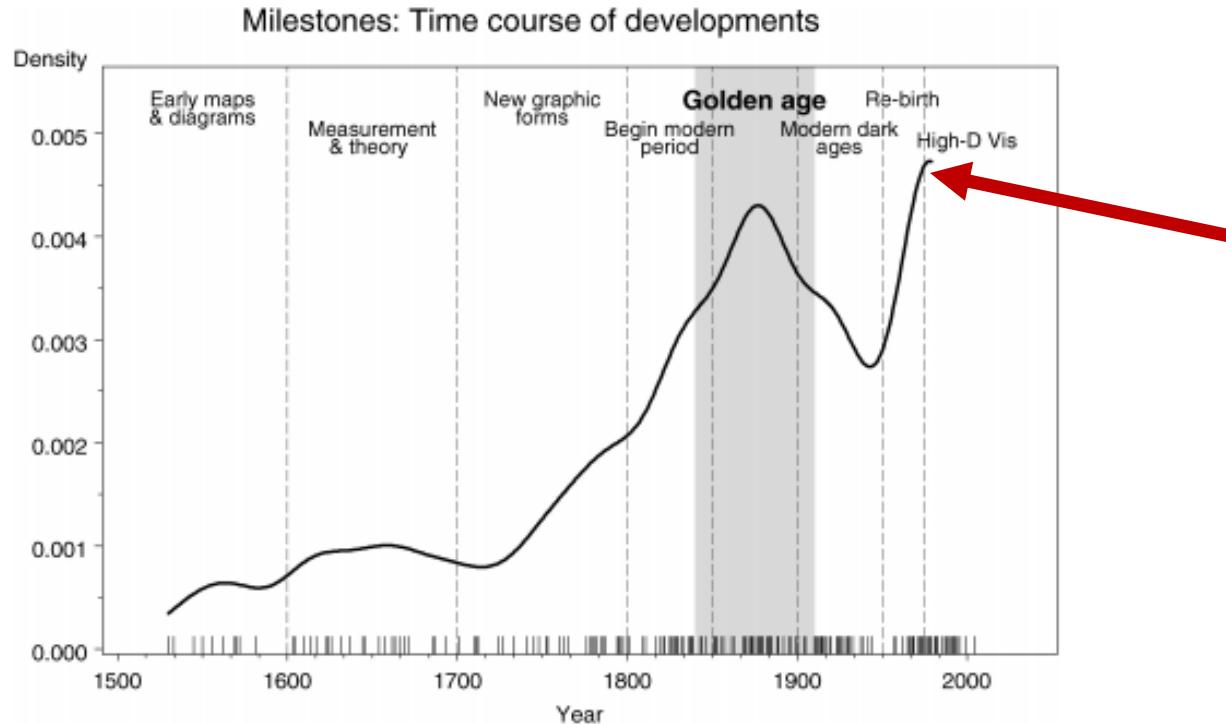


Hans Rosling's gapminder

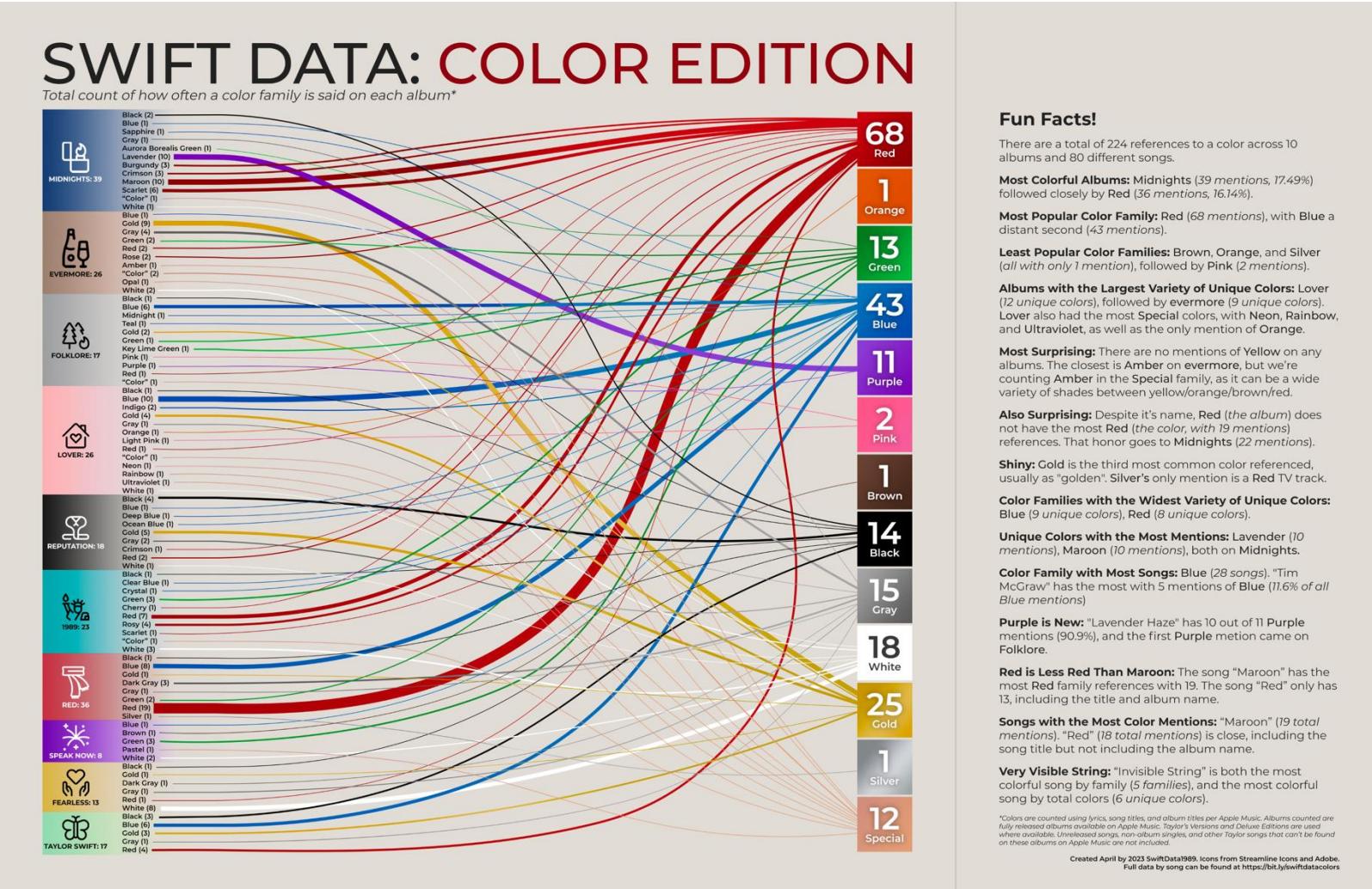
- [Simple version](#)
- [TV special effects](#)
- [Ted Talk](#)

A very brief history of data visualization

Currently undergoing a “Graphical re-birth”



A very brief history of data visualization

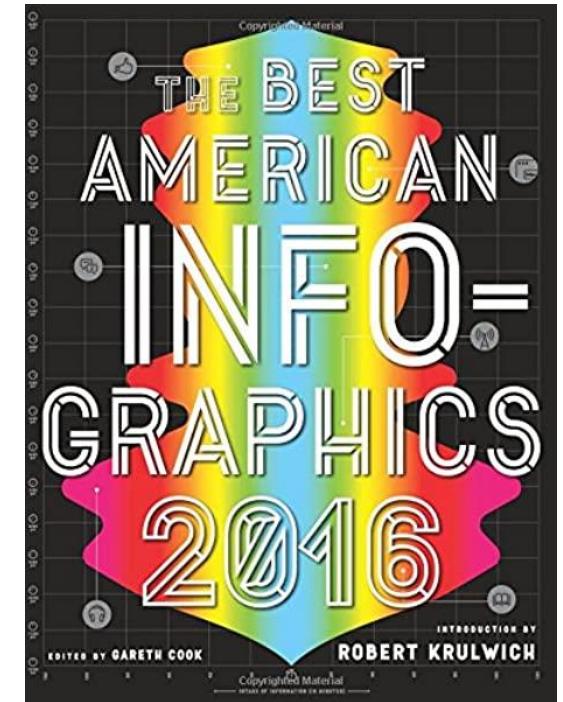


Coming up on homework 5: find an interesting data visualization...

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 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 one quantitative variable

Q: How can we visualize one quantitative variable?

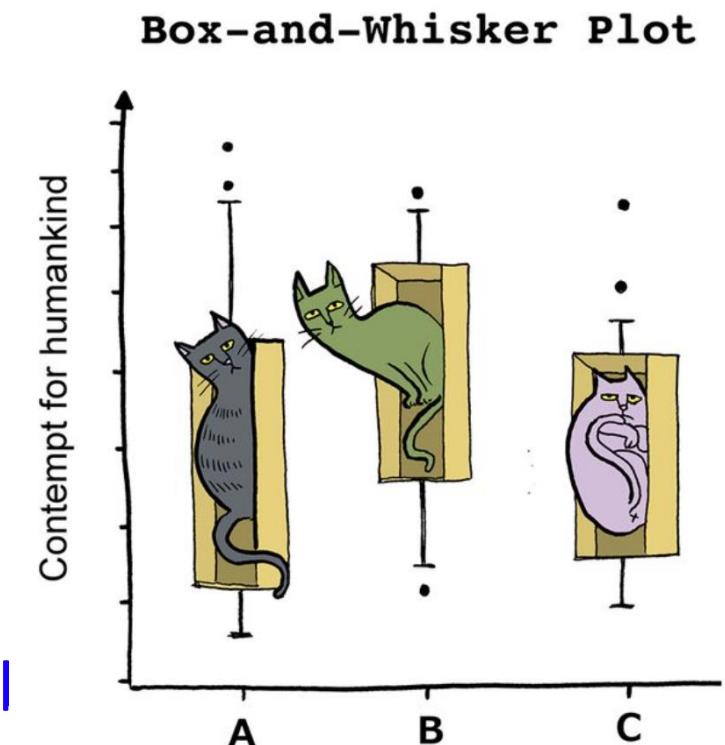
A: Histograms!

A: Box plots!

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

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

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



Visualizing time series

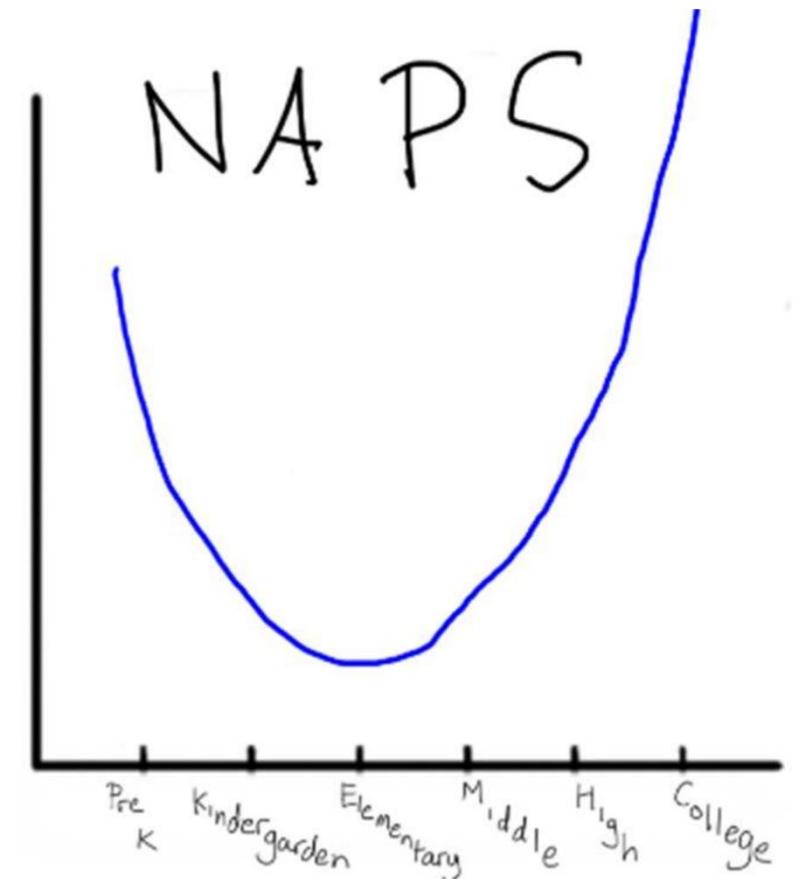
Q: How can we visualize a time series?

A: Line plot

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

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

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



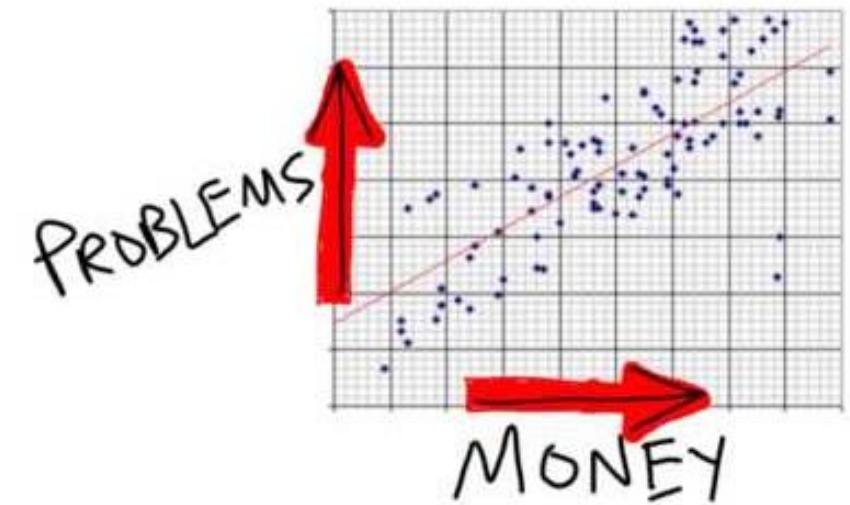
Let's review this in Jupyter!

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`: specifies 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 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

Subplots: pyplot interface

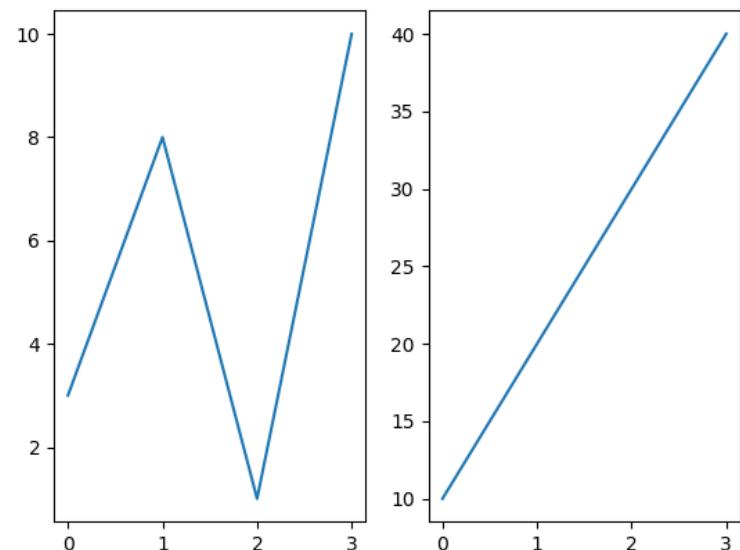
Matplotlib makes it easy to create multiple subplots within a larger figure

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

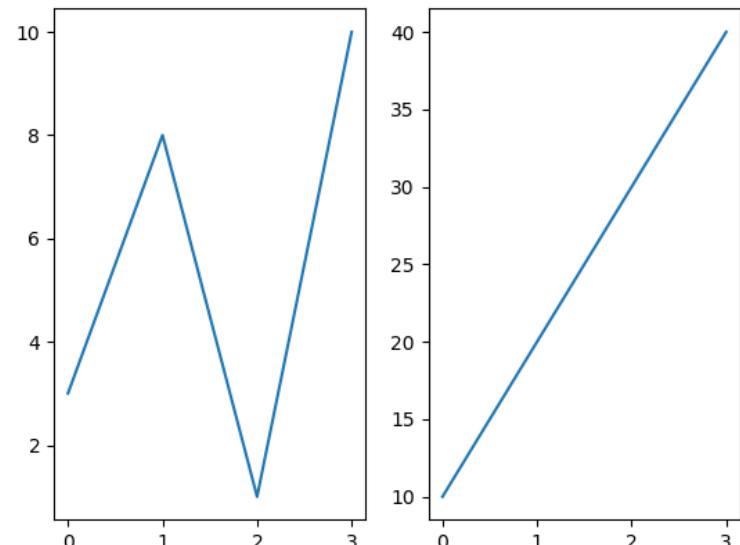
plot on the second subplot



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
```

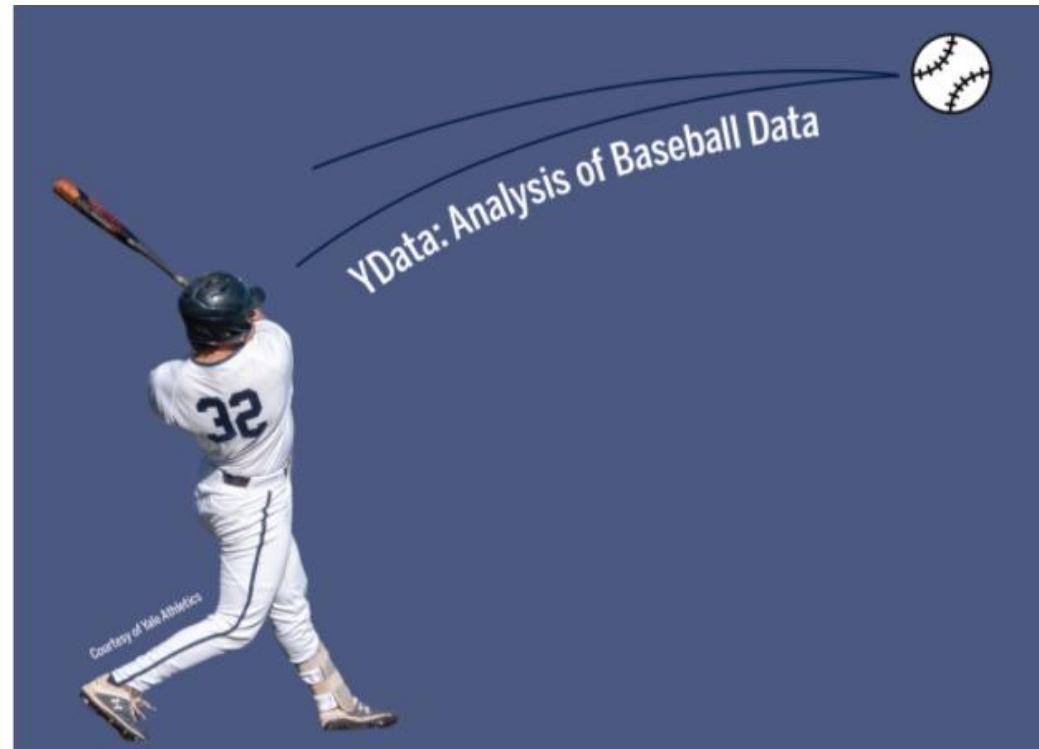


Let's explore this in Jupyter!

Using matplotlib as a canvas

We can also use 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



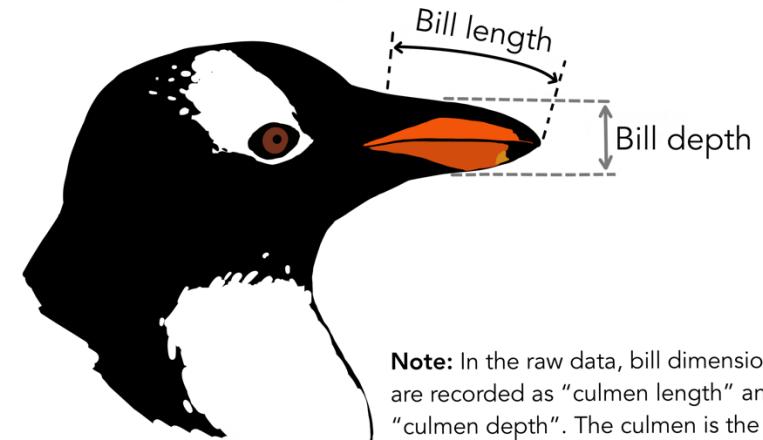
To make plots better looking we can set a theme

```
import seaborn as sns
```

```
sns.set_theme()
```

Inspiration: Palmer penguins

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



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

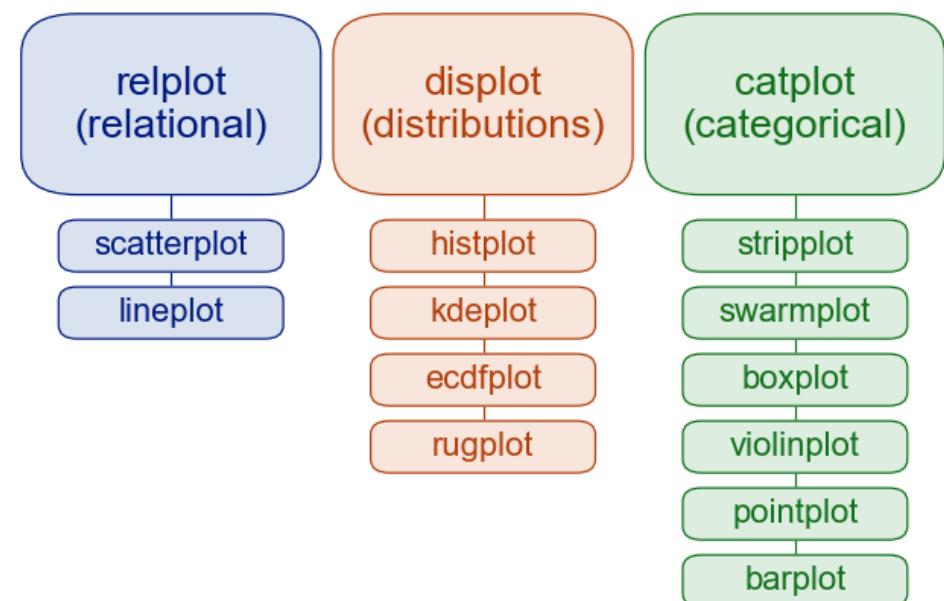
Seaborn figure level plots

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()`

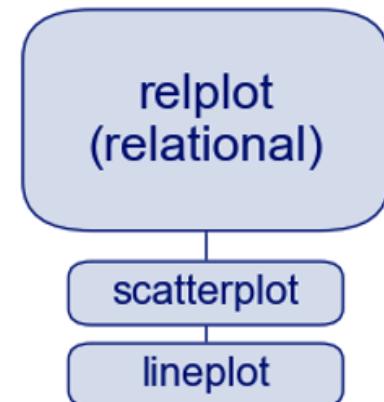
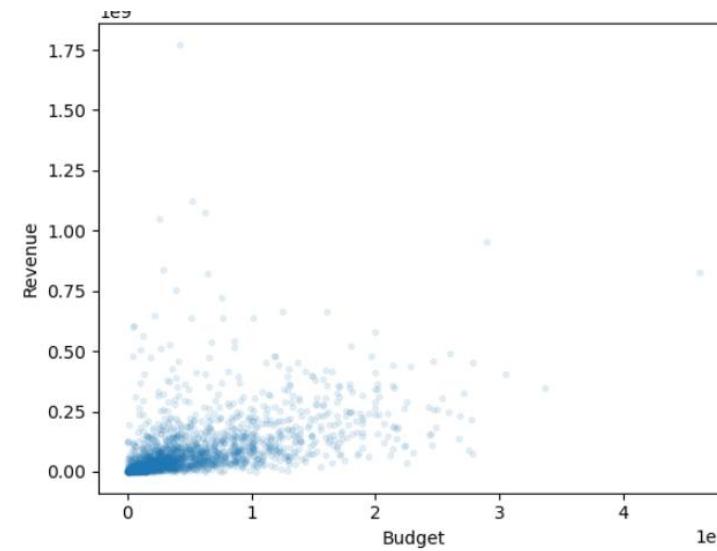
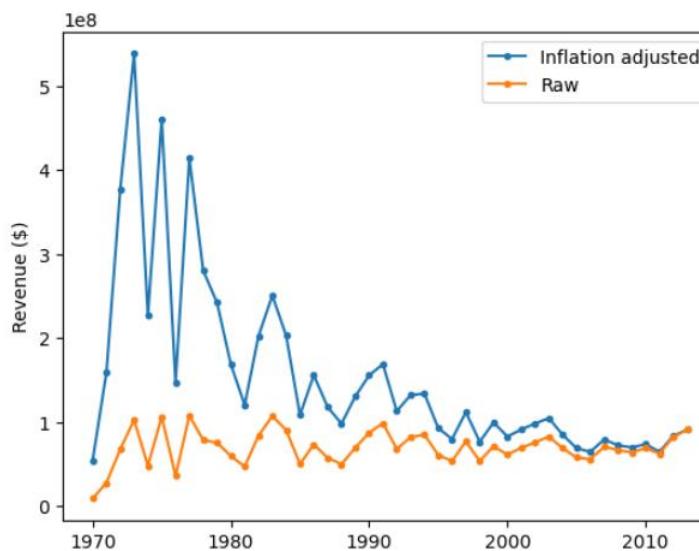
Figure level plots



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!

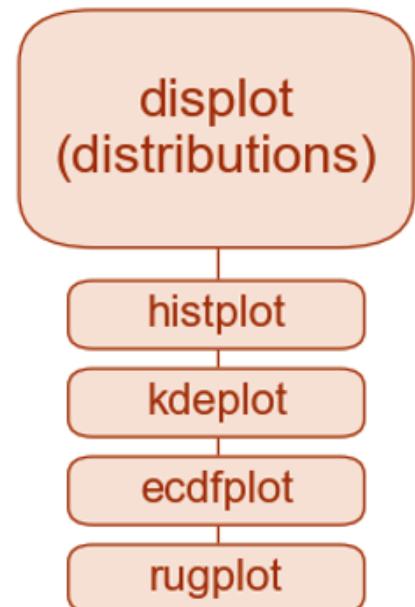
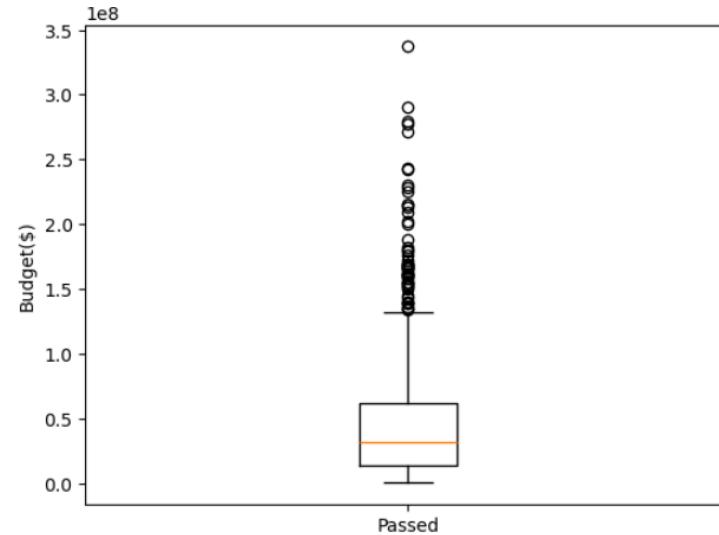
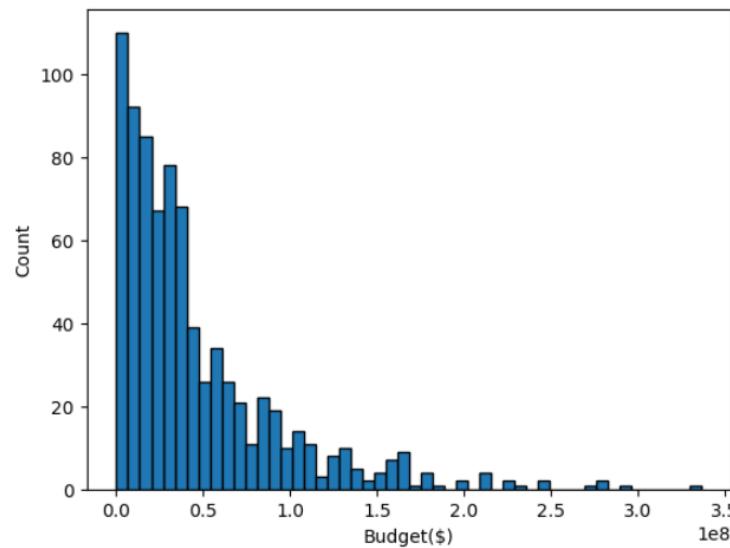


Let's explore this in Jupyter!

Plots for a single quantitative variable

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

- Histograms and boxplots

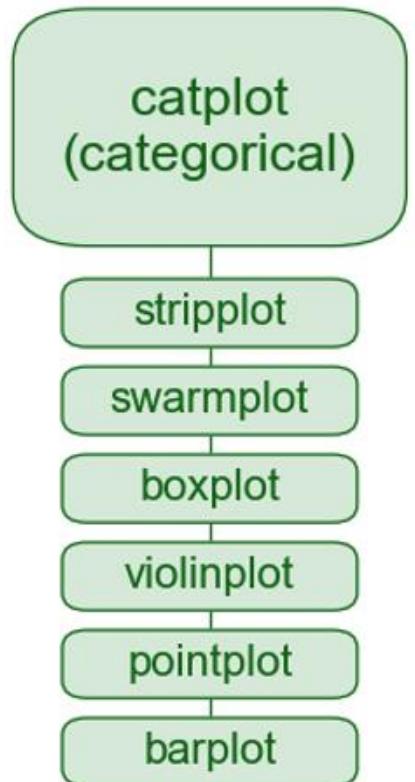
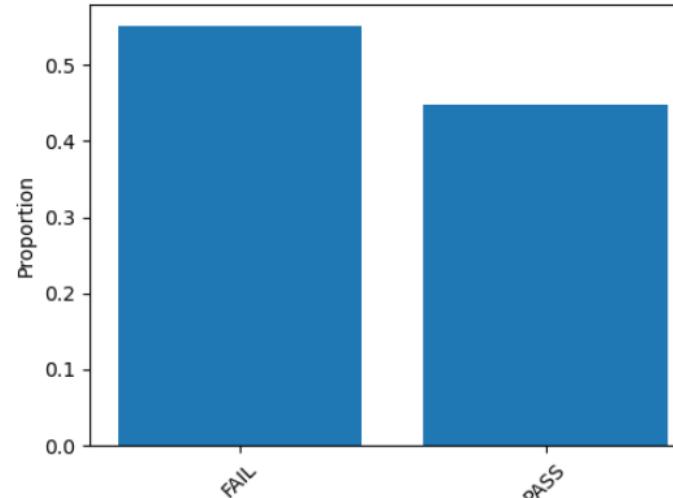
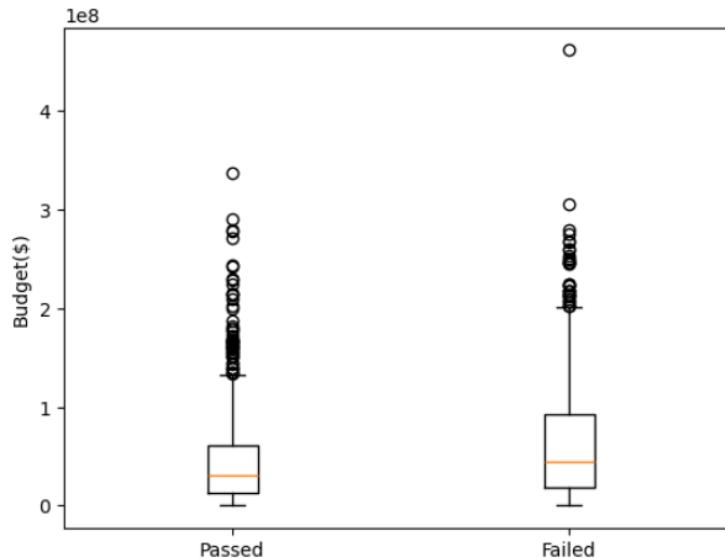


Let's explore this in Jupyter!

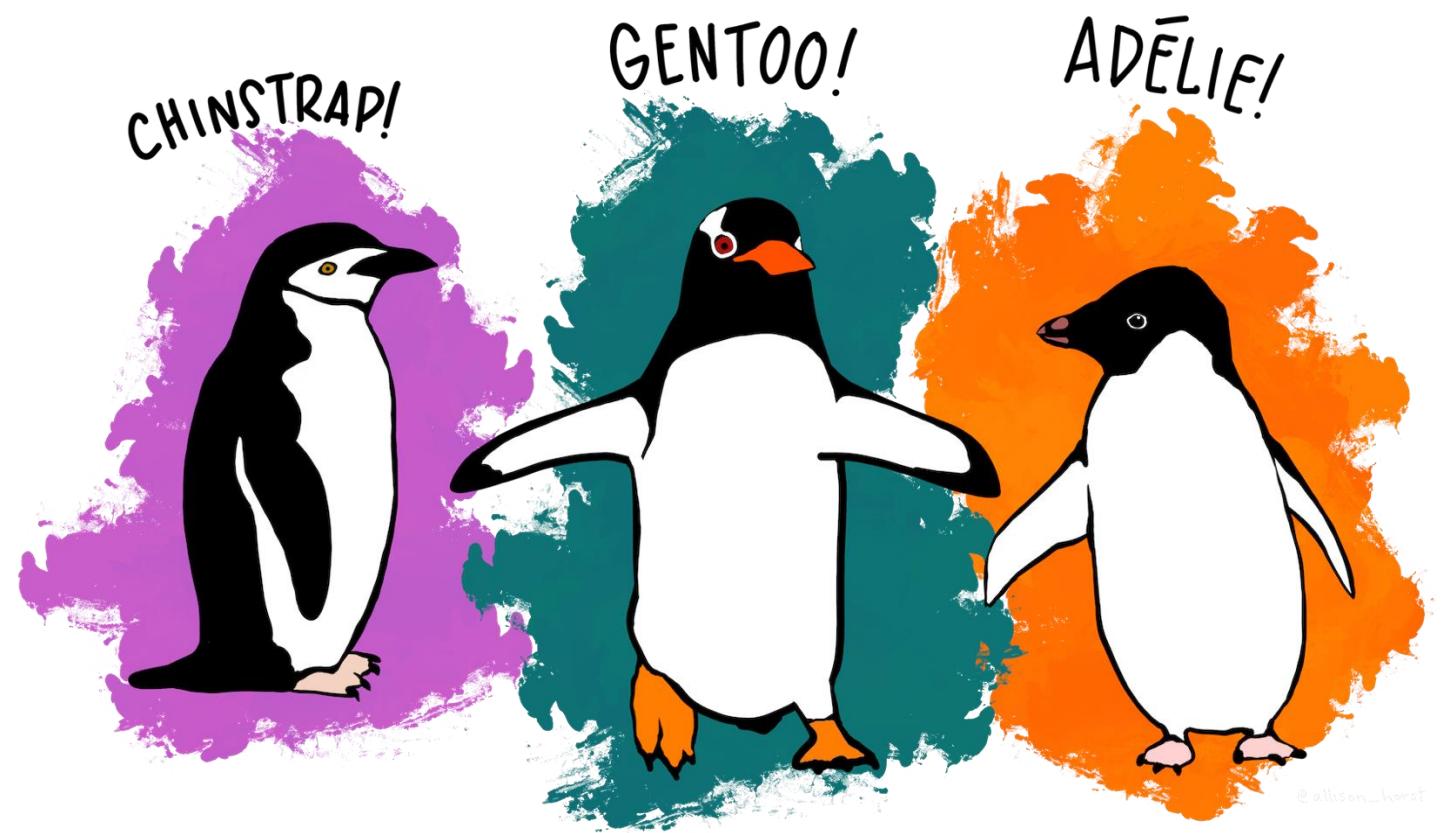
Plots for quantitative data comparing across different categorical levels

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

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

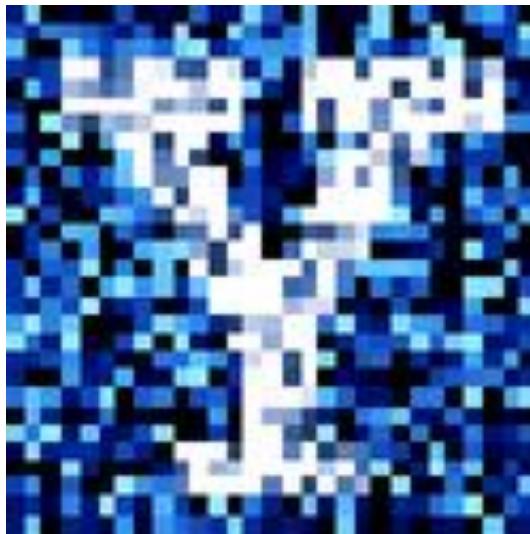


Let's explore this in Jupyter!



@allison_horst

YData: Introduction to Data Science



Class 11: Data visualization continued

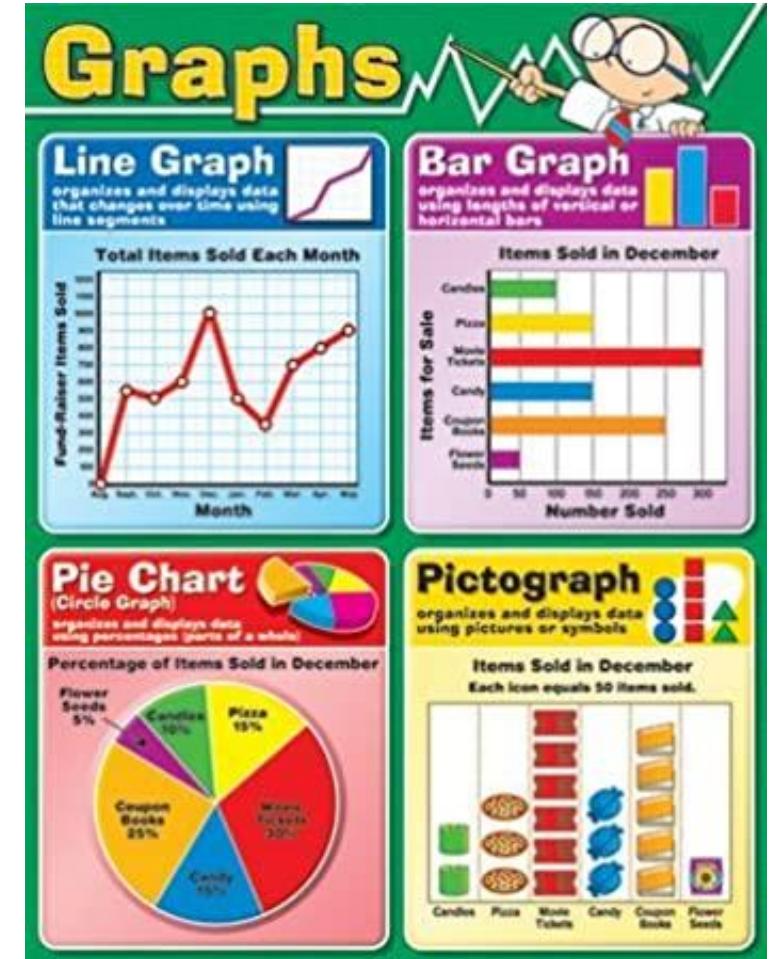
Overview

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: Start on interactive graphics



Midterm exam

Homework 5 has been posted!

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

Midterm: Thursday October 10th **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., `sns.catplot(data = , x = , y = , hue = , kind = "strip"/"swarm")`

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

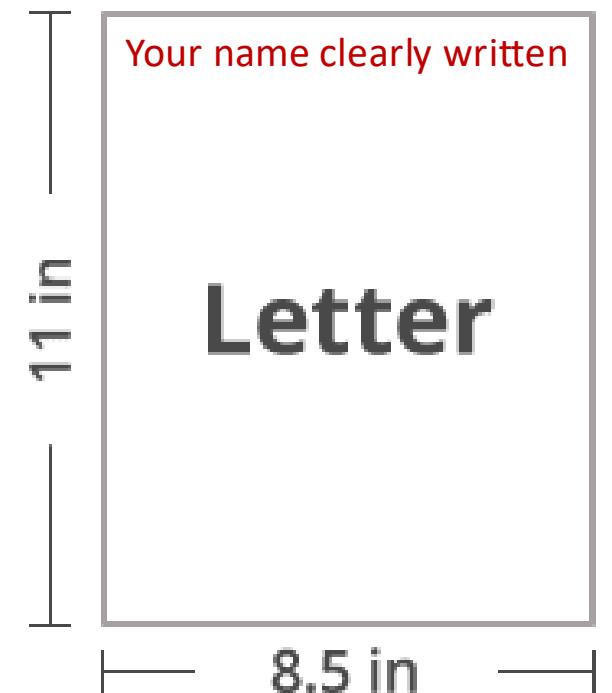
- Your name on the upper left of both sides of the paper

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.*

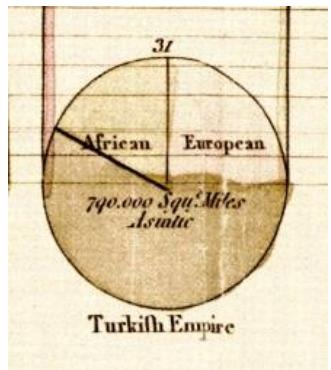
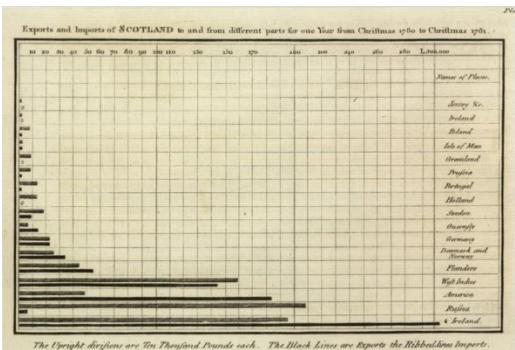
—Alexander von Humboldt, 1811



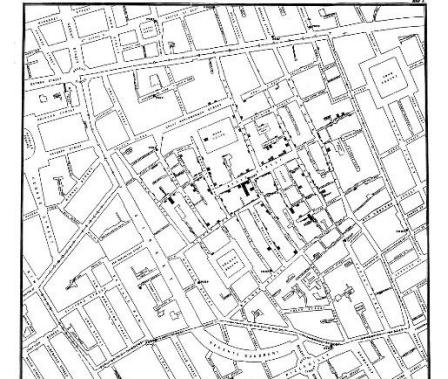
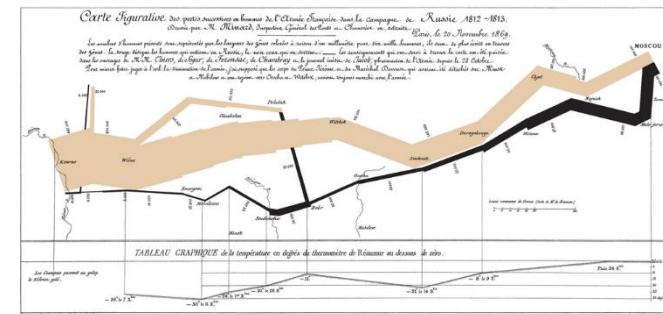
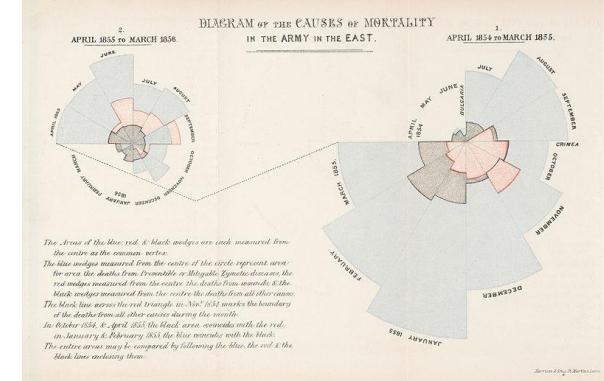
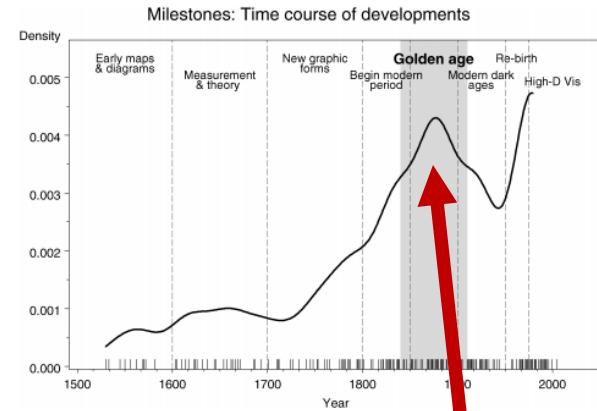
A very very brief history of data visualization

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

William Playfair (1759-1823)

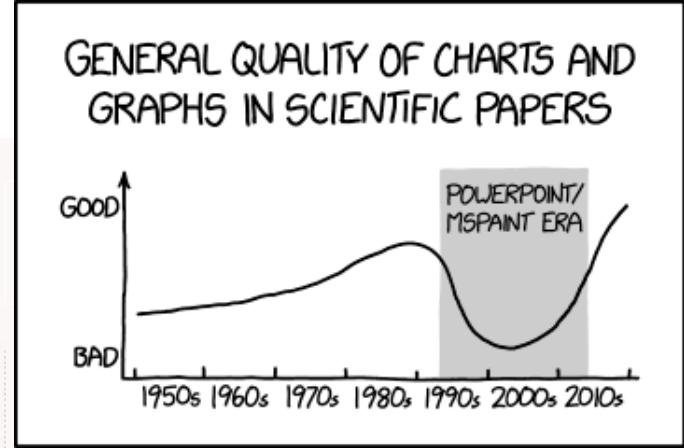
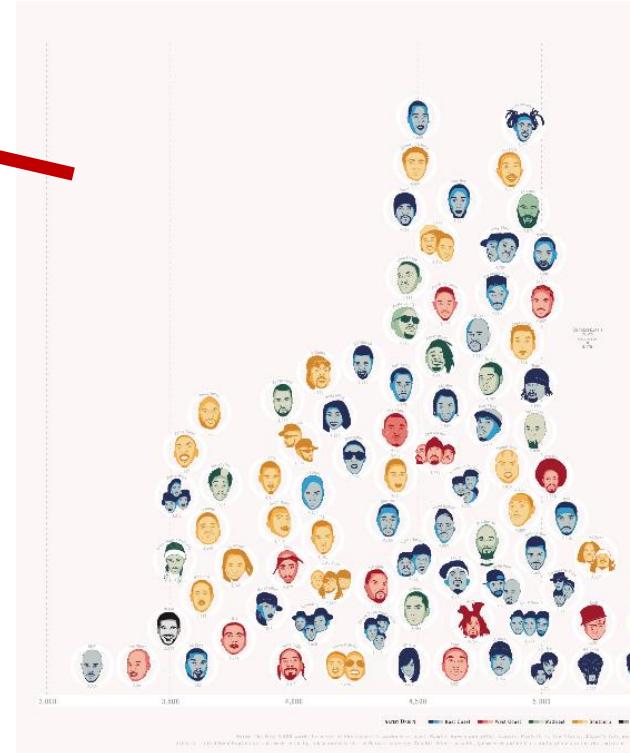
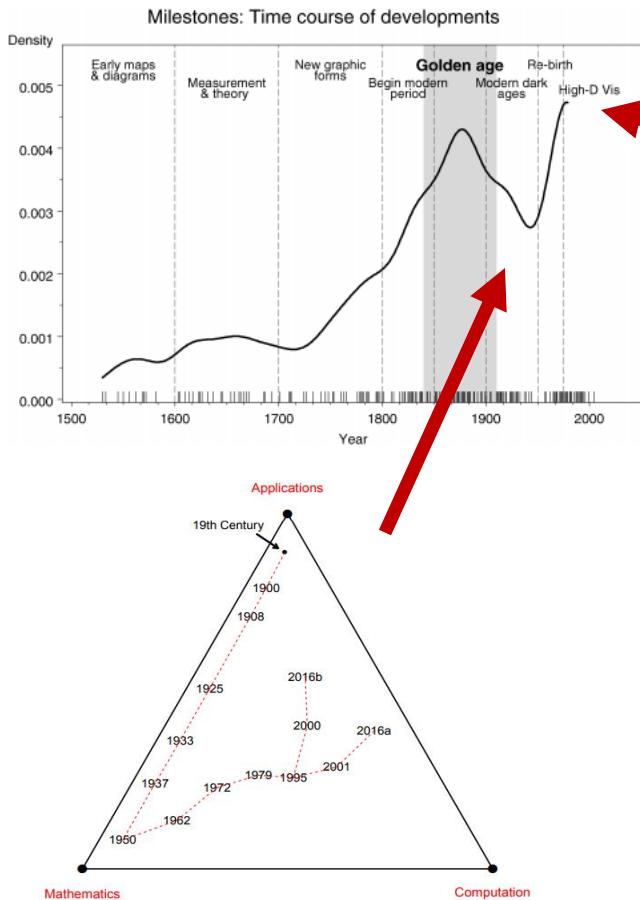


According to Friendly, statistical graphics researched its golden age between 1850-1900



A very very brief history of data visualization

“Graphical dark ages” around 1950



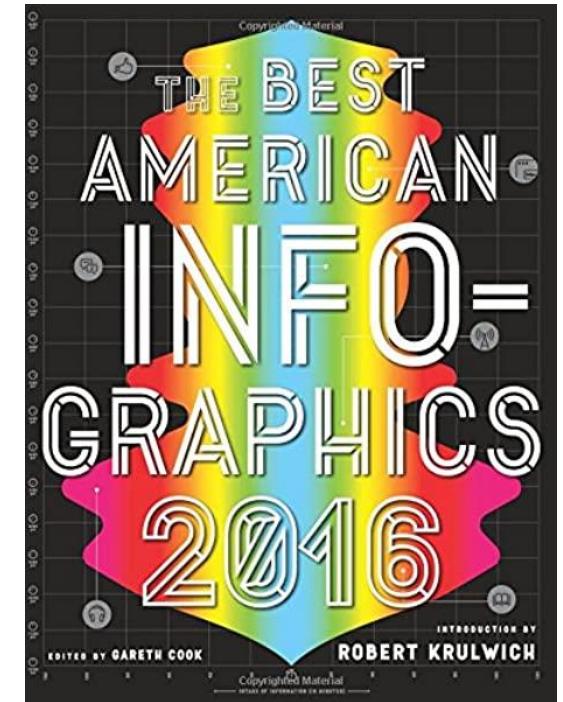
Currently undergoing a “Graphical re-birth”

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/>

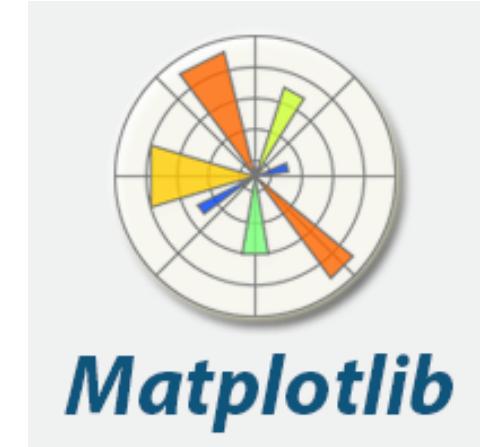
We will do a little show and tell in class



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

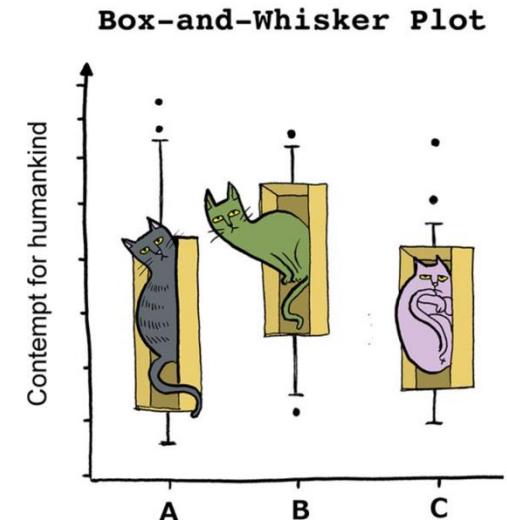
We will then discuss another visualization library called "seaborn" which makes it even easier to create beautiful looking graphics



Quick review: matplotlib

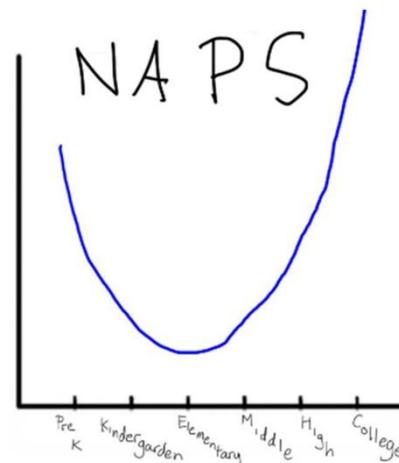
Single quantitative variable

```
plt.hist(data1, edgecolor = "k", alpha = .5);  
plt.boxplot( [data1, data2], labels = ["lab1", "lab2"])
```



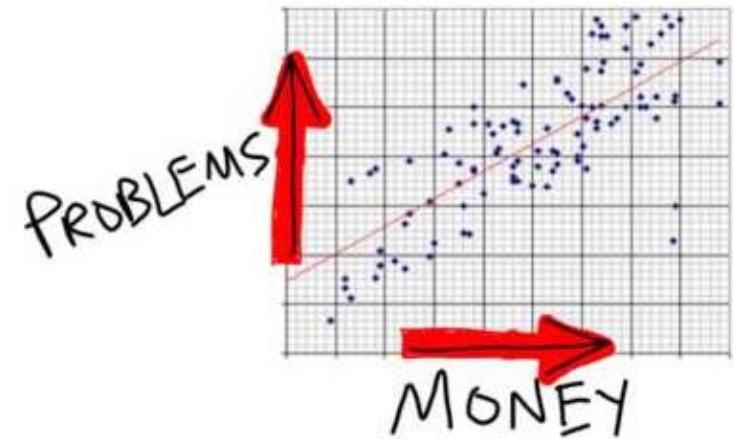
Line plot

- `plt.plot(x, y, '-')`



Two quantitative variables

- `plt.plot(x, y, '.')`
- `plt.scatter(x, y)`



Review/continuation of visualizing categorical data

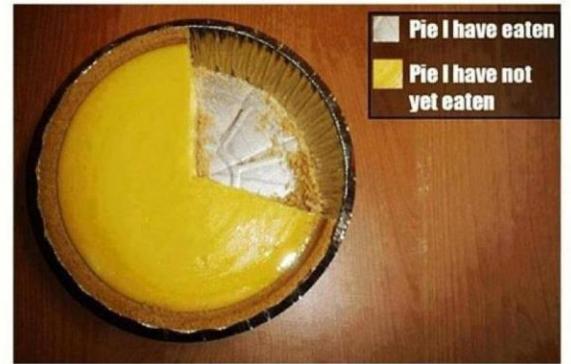
Q: How can we visualize categorical data?

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")
```

**World's Most Accurate
Pie Chart**



REAL Bar Chart

Subplots: pyplot interface

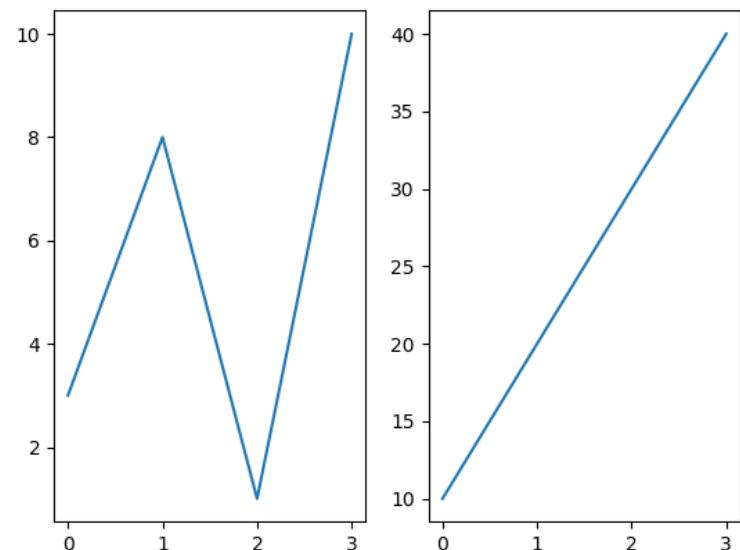
Matplotlib makes it easy to create multiple subplots within a larger figure

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

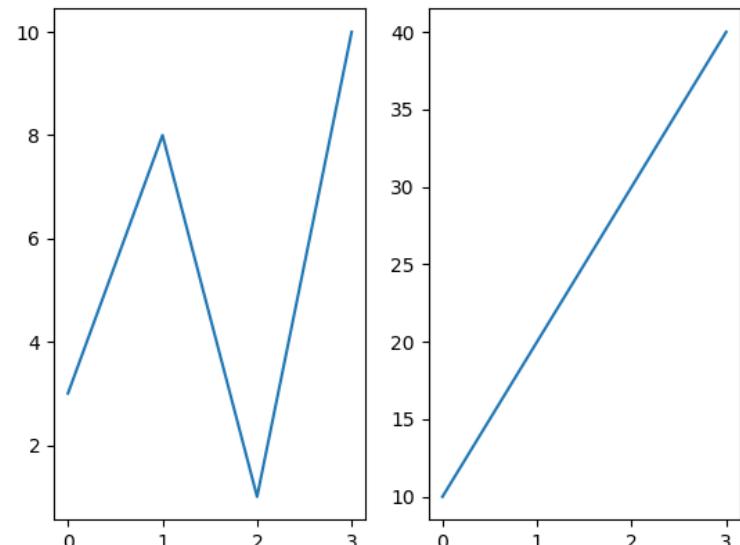
plot on the second subplot



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
```

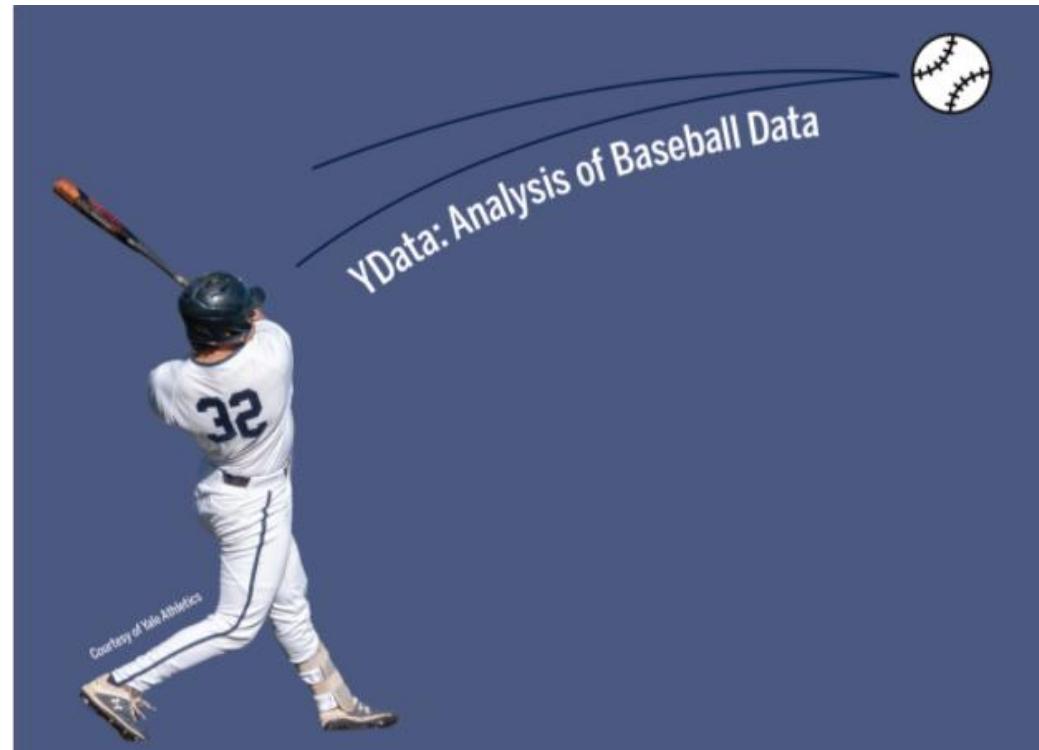


Let's explore this in Jupyter!

Using matplotlib as a canvas

We can also use 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



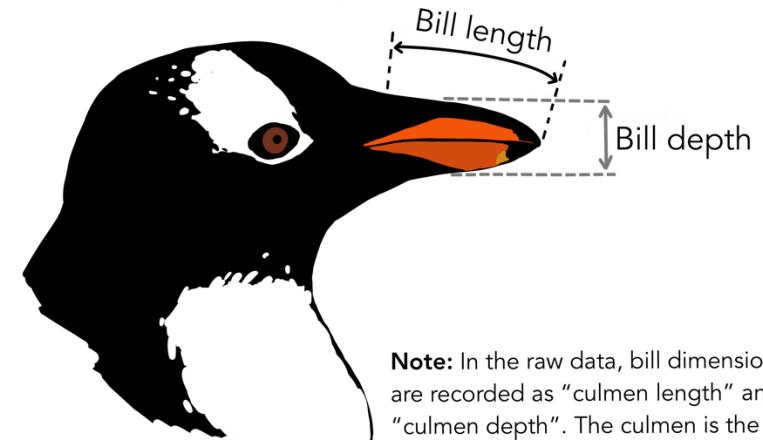
To make plots better looking we can set a theme

```
import seaborn as sns
```

```
sns.set_theme()
```

Inspiration: Palmer penguins

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



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

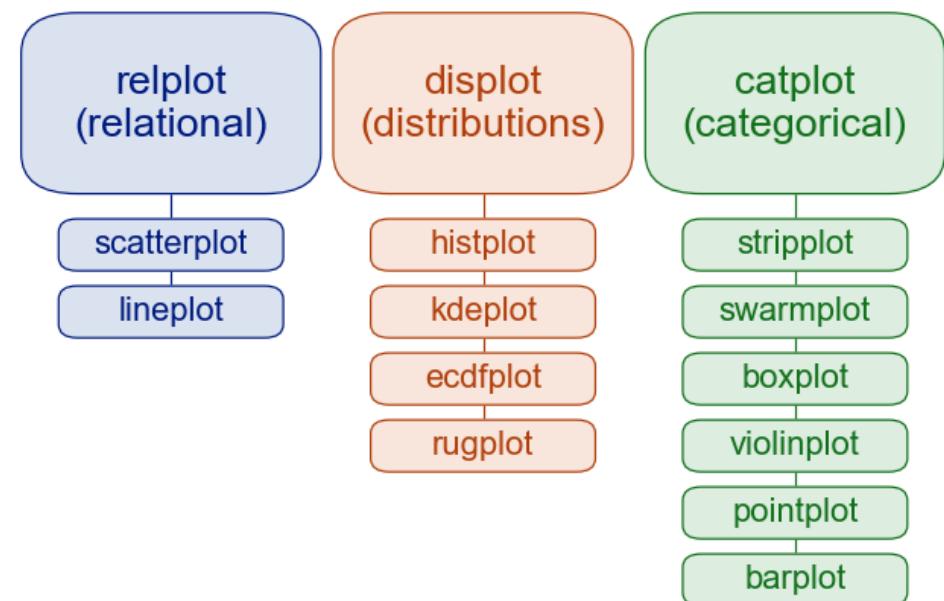
Seaborn figure level plots

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()`

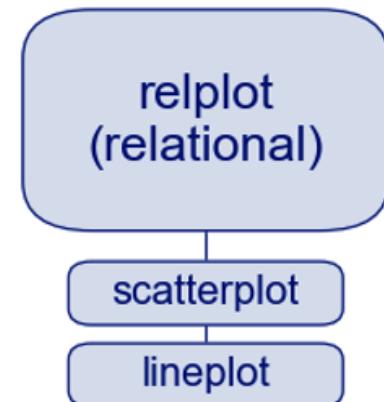
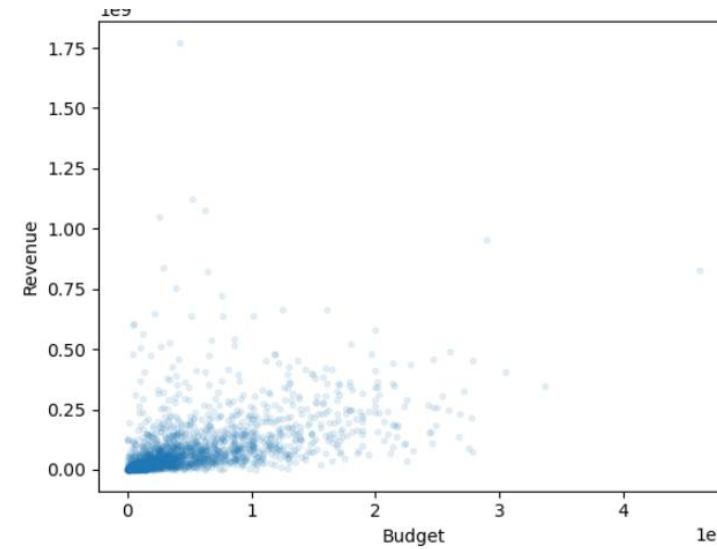
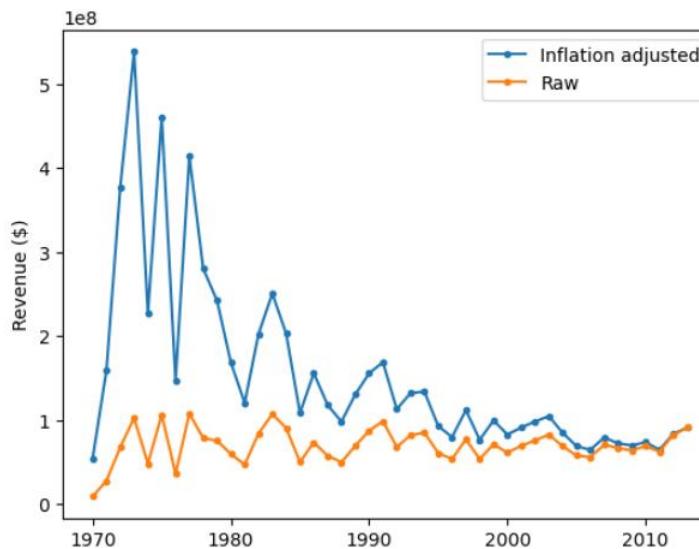
Figure level plots



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!

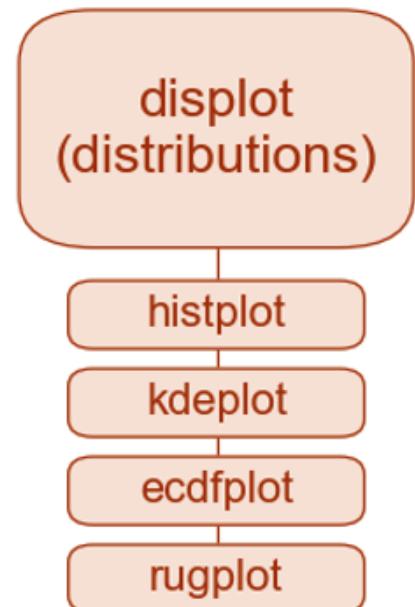
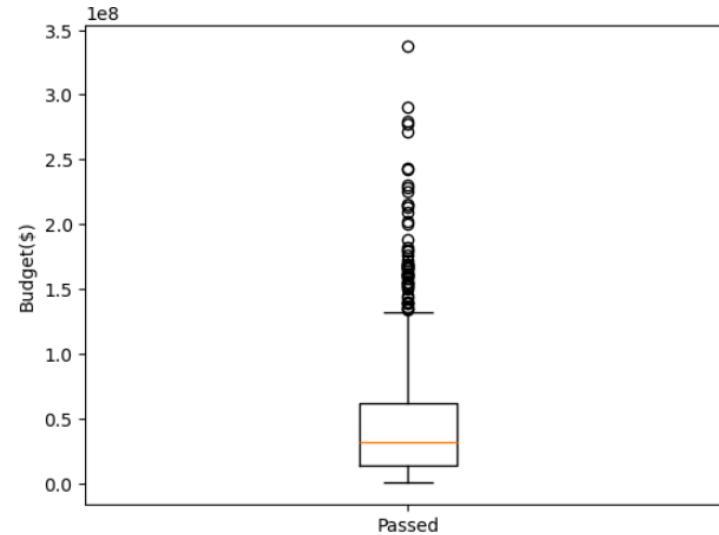
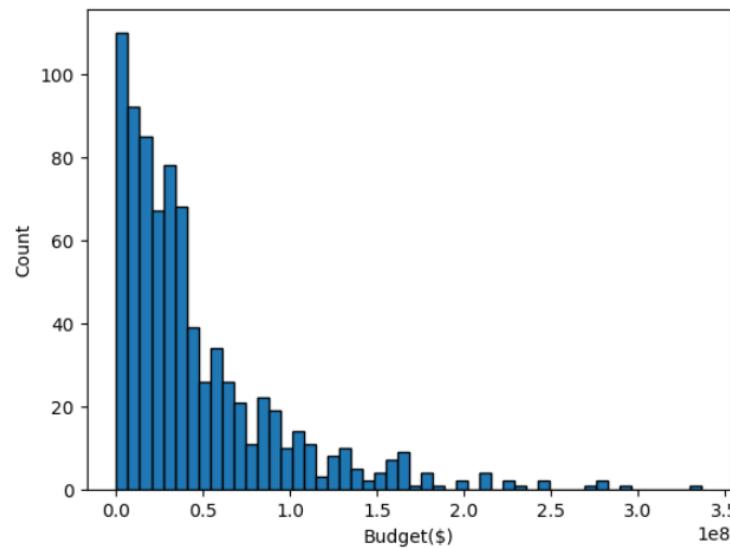


Let's explore this in Jupyter!

Plots for a single quantitative variable

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

- Histograms and boxplots

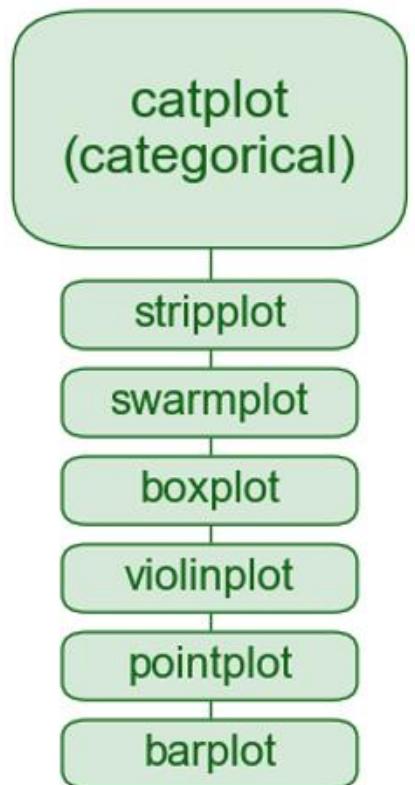
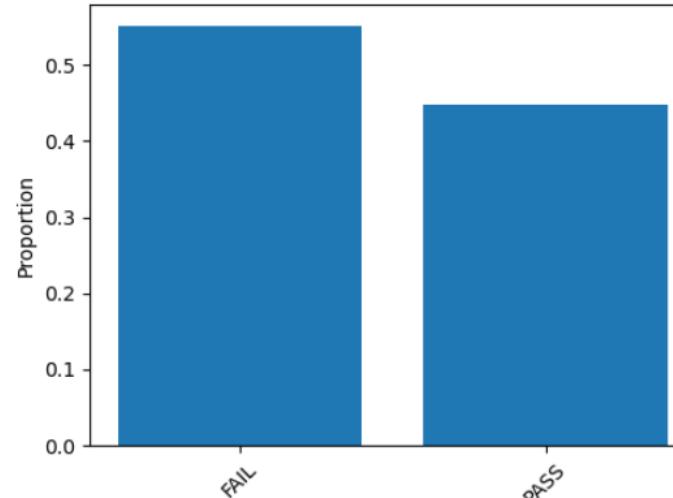
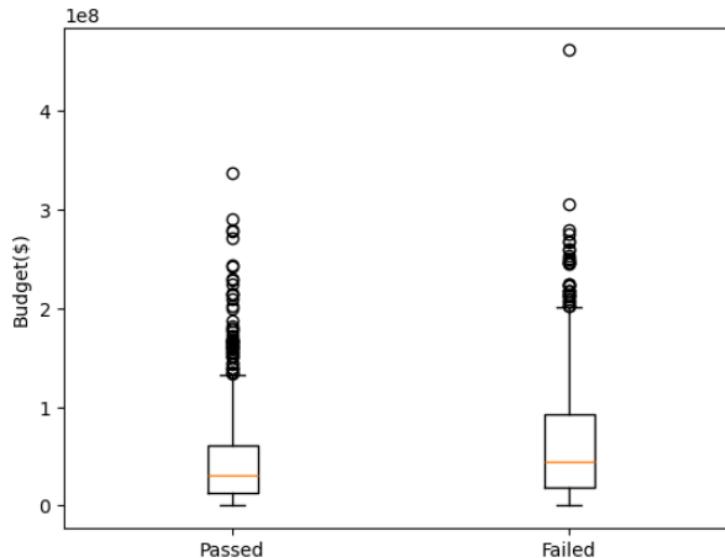


Let's explore this in Jupyter!

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!

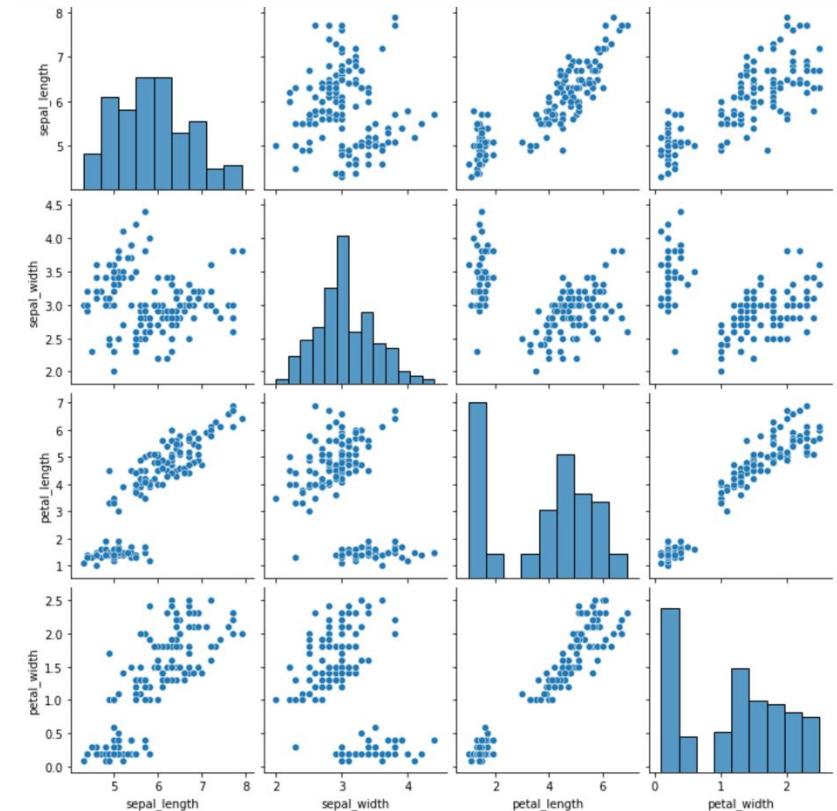
Pairs plot

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

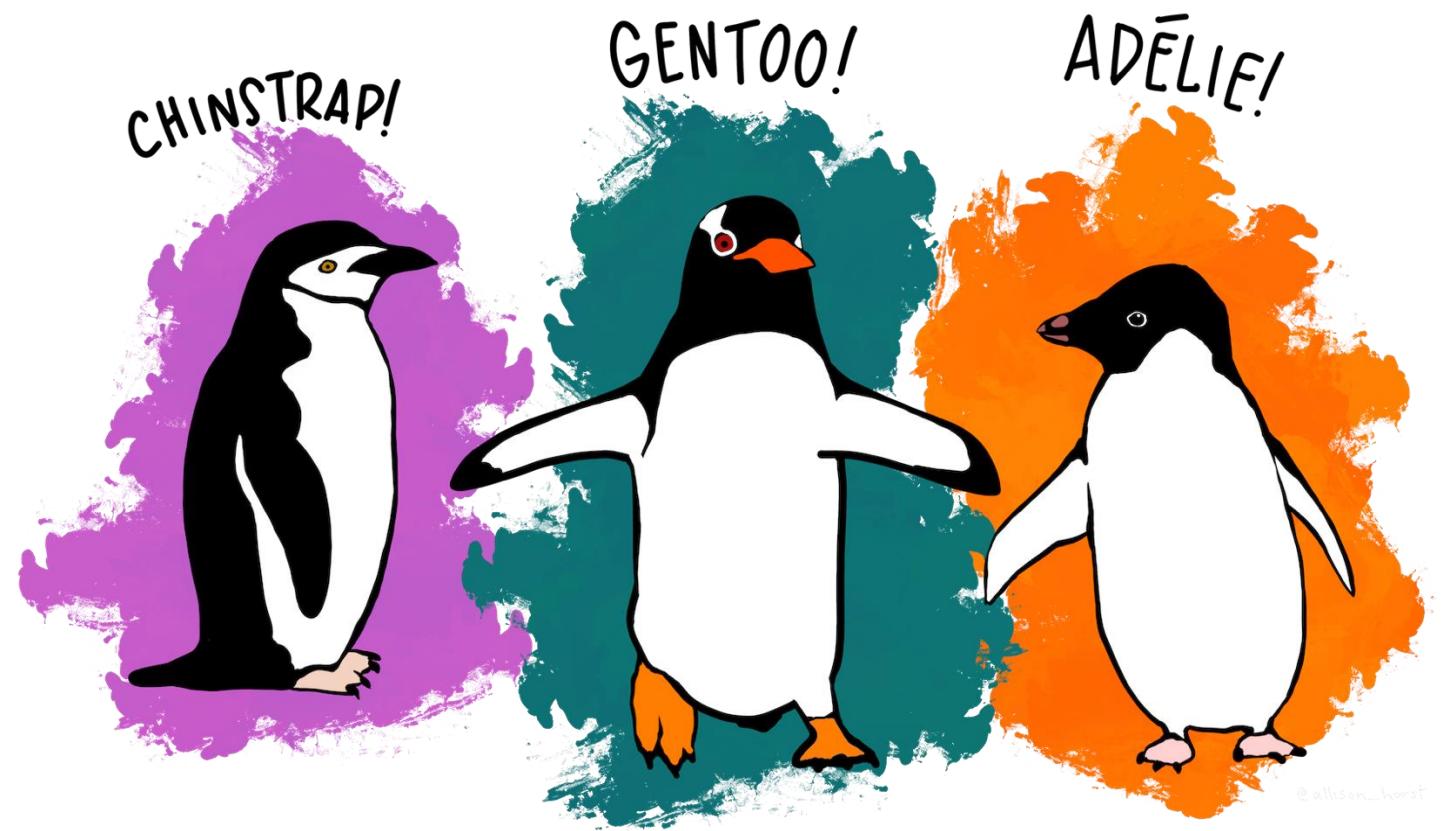
It can be one of the most useful ways to see relationships between multiple quantitative variables

We can create pairs plots in seaborn using:

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



Let's explore this in Jupyter!



@allison_horst

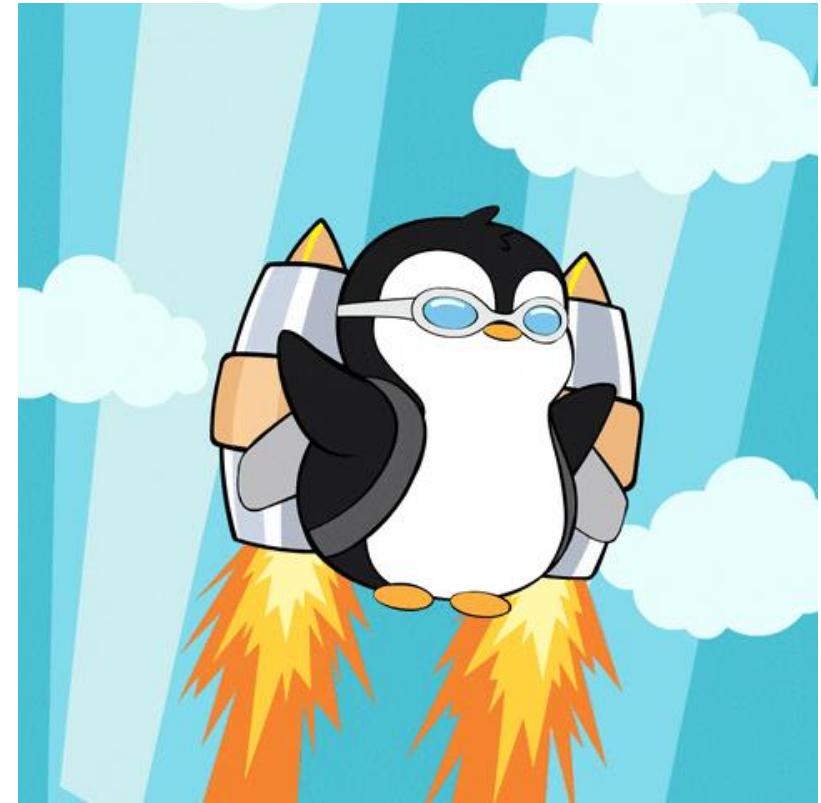
Interactive visualizations for data exploration

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

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

Add axis labels

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

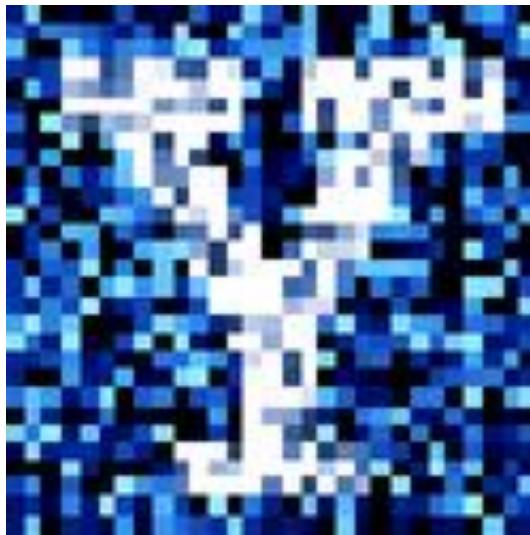
Let's explore this in Jupyter!

Coming soon...

The exam, more
interactive plots,
and mapping!



YData: Introduction to Data Science



Class 12: review

Overview

Very quick overview over topics we have covered

Answering your questions

If there is time: Practice problems

Midterm exam

Midterm: Thursday October 10th **in person**
during regular class time

- Exam is on paper

If you have accommodations, schedule to take
the exam with SAS

A practice exam has been posted



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., `sns.catplot(data = , x = , y = , hue = , kind = "strip"/"swarm")`

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

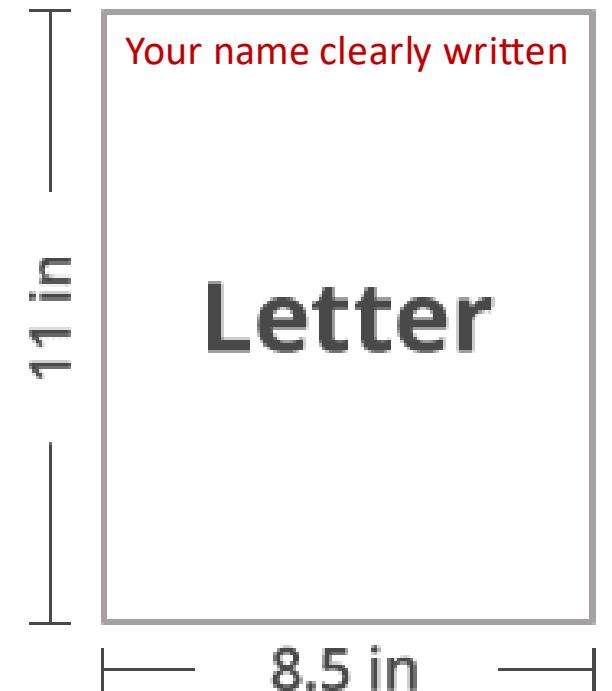
- Your name on the upper left of both sides of the paper

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



Quick review of what is Data Science?

Data Science is a broadening of data analyses beyond what traditional Statistical mathematical/inferential analyses to use more computation

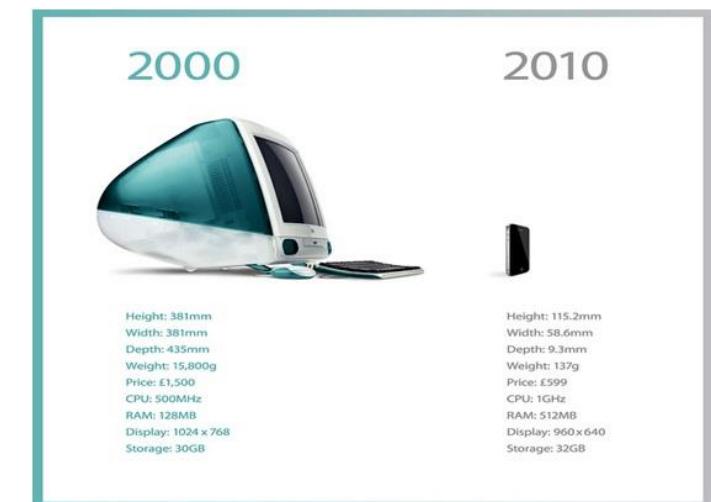
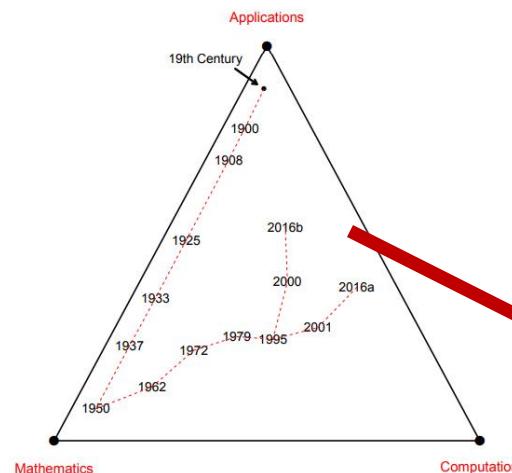
Many fields impacted by 'Data Science'

- Making business decisions
- Predictive medicine
- Fraud detection
- Etc.

Examples:

- [NYC city bike visualization](#)
- [Wind map visualization](#)

Ethical concerns around privacy, fairness and other issues



Quick review of the history of Data Science

(a very incomplete list)

Data



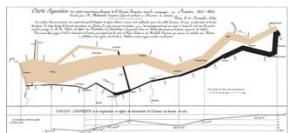
Ishango bone (20,000 BCE)



Cuneiform tablets (4,000 BCE)



Quipus in South America (1100-1500)



Probability

Key Take Away

Probability models
dominated data analysis
prior to using
computational methods

Computers



Abacus (2400 BCE)



Antikythera mechanism (100 BCE)



Analytical Engine (1800's)



Hollerith Tabulating Machine (1890)

Demographics (1600's)

Golden age of data visualization

(1850-1900)

Big data (now)

"Small data"

Probability in Statistics

(1820's – 1950's)

Math Stats dominates (1900-1960's)

"Big data"

Mainframes, PCs, Internet, etc. (1950-present)

Quick review of Python basics

Expressions and types

```
my_num = 2 * 3
```

```
my_string = 'ja' * 5
```

```
type(my_num)
```

TO DO LIST

1. make lists
2. look at lists
3. PANIC!

List, tuples, and dictionaries

```
my_list = [1, 2, 3 , 4, 5, 'six']      # create a list
```

```
my_list2 = my_list[0:3]      # get the first 3 elements
```

```
my_tuple = (10, 20, 30)      # immutable
```

```
my_dict = { 'a': 7, 'b': 20}    # create a dictionary
```



Review: String methods

Suppose we have a string `my_string = "Hello Yale"`

String methods

`my_string.upper()`

`my_string.replace("Yale", "Whale")`

`string_list = my_string.split(" ")`

`", ".join(string_list)`

`my_string.count("Yale")`

`f"my string {value_to_fill} will be filled in"`



NO STRINGS ATTACHED
(LITERALLY)

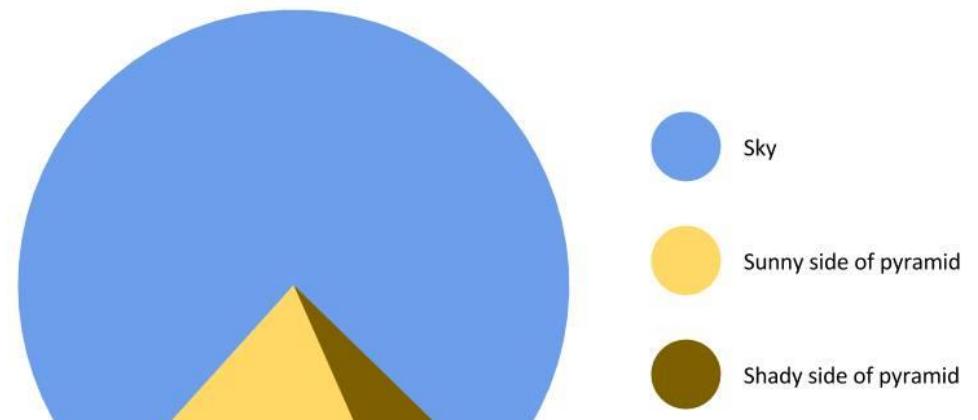
Quick review: categorical data

Categorical data

Proportion = $\frac{\text{number in category}}{\text{total number}}$

bechdel.`count("PASS")/len(bechdel)`

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



Quick review: one quantitative variable data

Basics statistics and plots:

`plt.hist(data)`

$$\text{statistics.mean(data)} = \frac{1}{n} \sum_{i=1}^n x_i$$

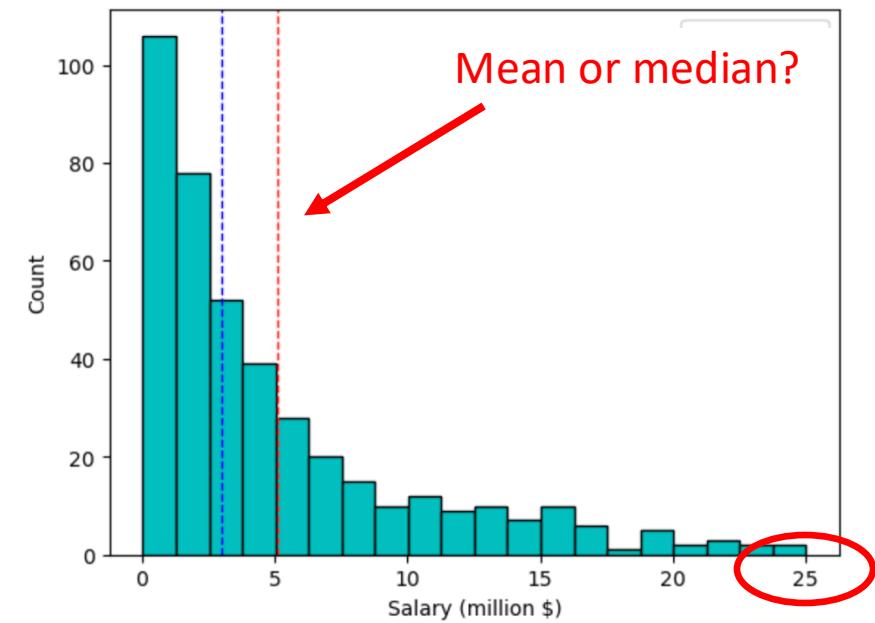
`statistics.median(data)`

Quantitative data spread

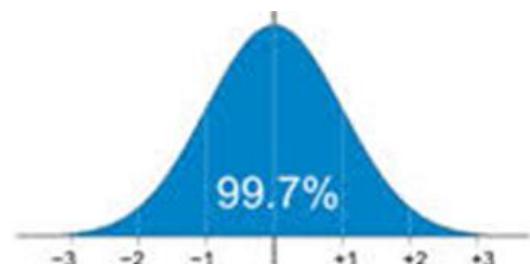
`statistics.stdev(data)`

$$s = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\text{z-score}(x_i) = \frac{x_i - \bar{x}}{s}$$



Outliers



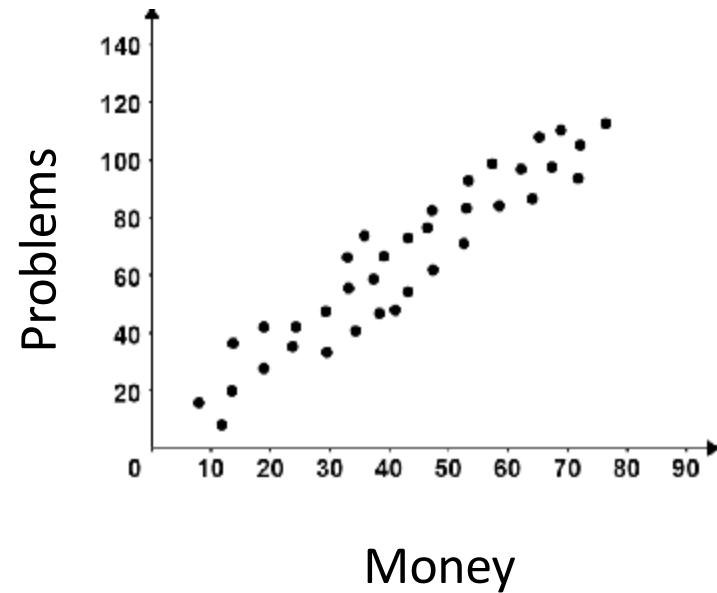
Quick review: two quantitative variables data

Basics statistics and plots:

```
plt.plot(x, y, '.')
```

```
statistics.correlation(x, y)
```

$$r = \frac{1}{(n-1)} \sum_{i=1}^n \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right)$$



- Correlation is always between -1 and 1: $-1 \leq r \leq 1$
- The sign of r indicates the direction of the association
- Values close to ± 1 show strong linear relationships, values close to 0 show no linear relationship
- Correlation is symmetric: $r = \text{cor}(x, y) = \text{cor}(y, x)$

Quick review of NumPy arrays and functions

Hopefully we are comfortable with:

- Creating arrays and accessing elements: `np.array()`
- Getting their type and size: `.shape`, `.dtype`
- Using numeric functions: `np.sum()`, `np.mean()`, `np.diff()`
- Functions that return ndarrays: `np.diff()`, `np.cumsum()`
- Using broadcasting: `my_array * 2`, `my_array1 - my_array2`
- Creating Boolean arrays: `my_array < 5`, `my_array == "C"`
- Using Boolean masks to get elements: `my_array[my_array < 5]`



Quick review of pandas DataFrames

Pandas DataFrame hold Table data

Selecting columns:

```
my_df[["col1", "col2"]]
```

getting multiple columns using a list

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 method

PLAYER	POSITION	TEAM	SALARY
str	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

Quick review of 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 )
```

Quick review of pandas DataFrames

We can get statistics separately by group:

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

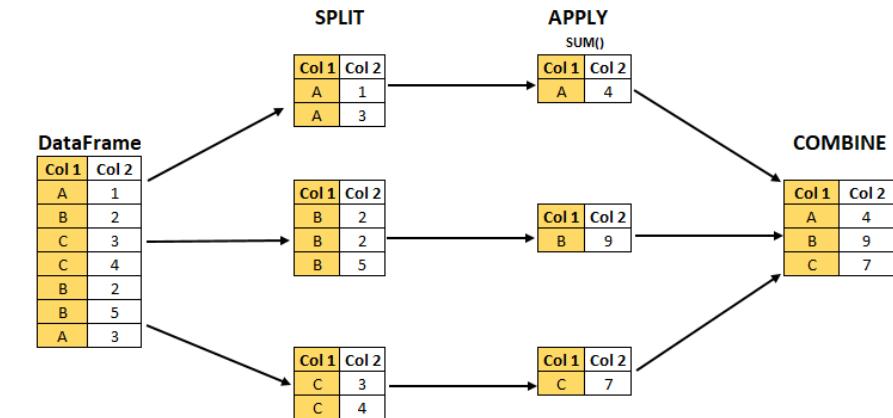
```
my_df.groupby("group_col_name").agg(
```

```
    new_col1 = ('col_name', 'statistic_name1'),
```

```
    new_col2 = ('col_name', 'statistic_name2'),
```

```
    new_col3 = ('col_name', 'statistic_name3')
```

```
)
```



Review of joining data frames by Index values

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

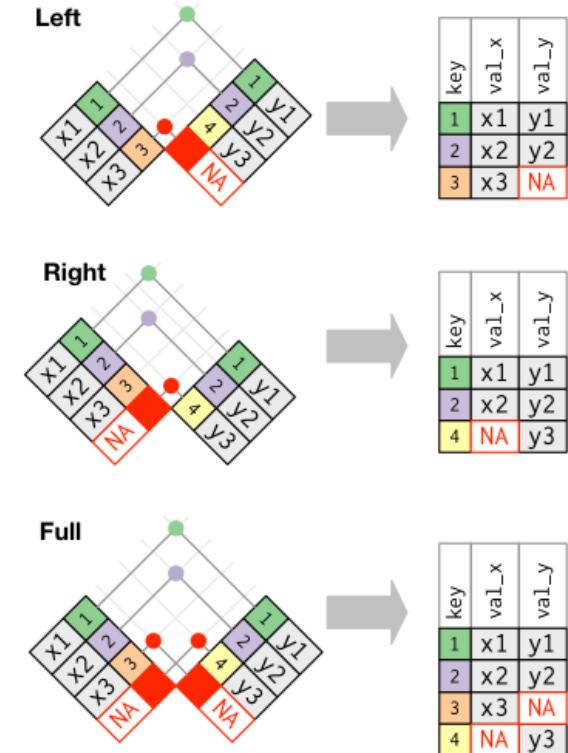
- `x_df` have one column called `x_vals`
- `y_df` has one column called `y_vals`

Index	x_vals
1	x1
2	x2
3	x3

DataFrame: `x_df`

Index	y_vals
1	y1
2	y2
4	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, how = "left")`

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

Review of **merging** data frames **by columns**

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

- **x_df** have two columns called **col1_x**, and **col2_x**
- **y_df** has two columns called **col1_y** and **col2_y**



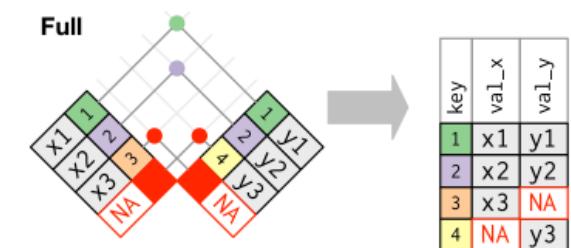
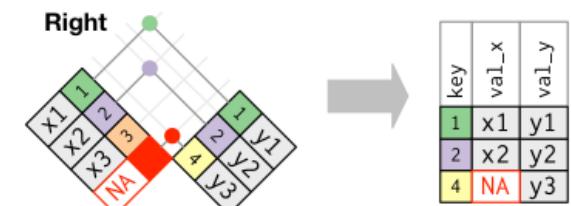
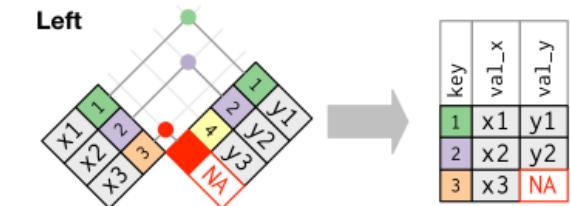
DataFame: x_df



DataFrame y_df

Joins have the general form:

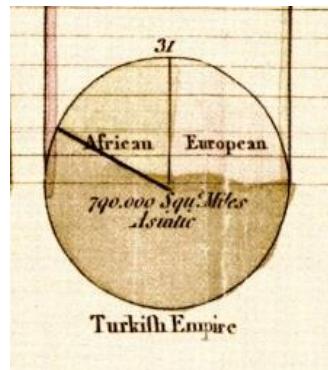
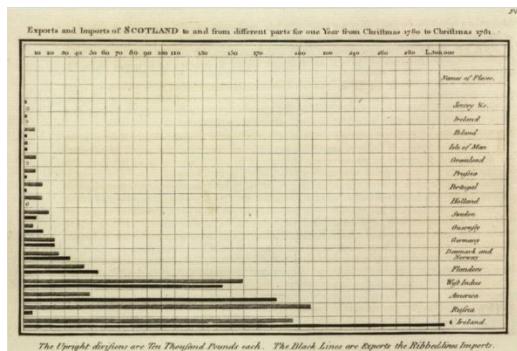
```
x_df.merge(y_df, how = "left", left_on = "col1_x", right_on = "col1_y")
```



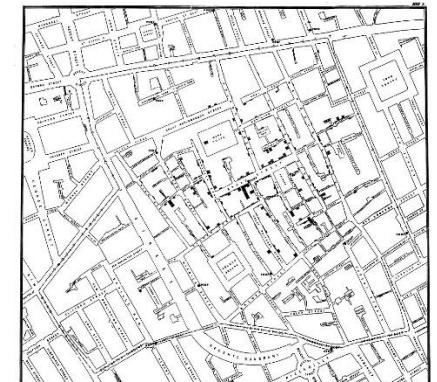
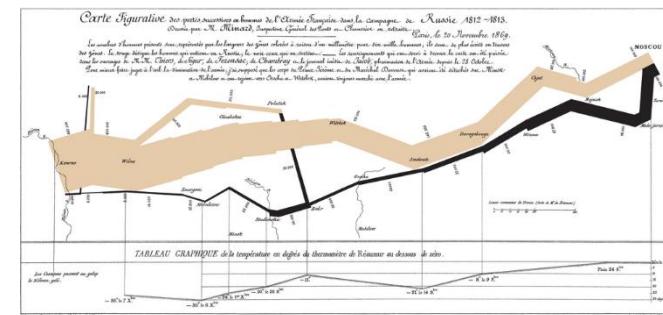
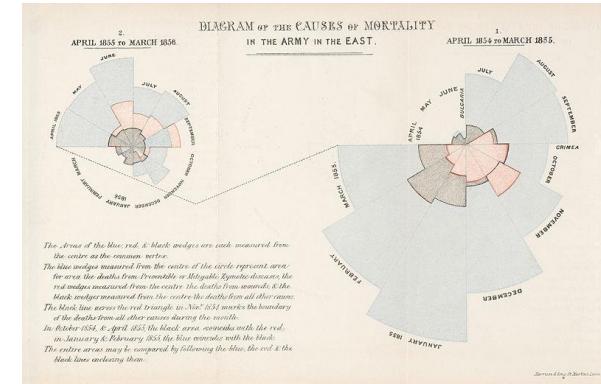
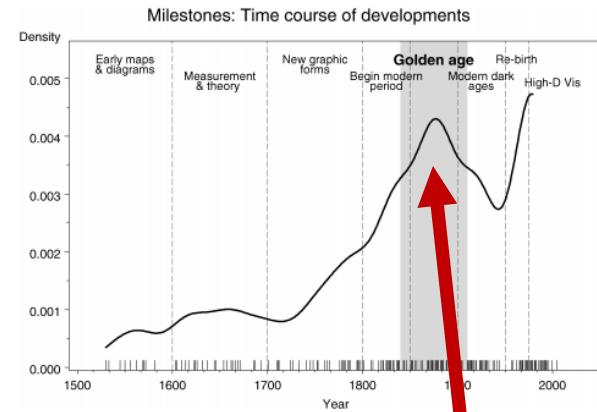
Quick review of the history of data visualization

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

William Playfair (1759-1823)

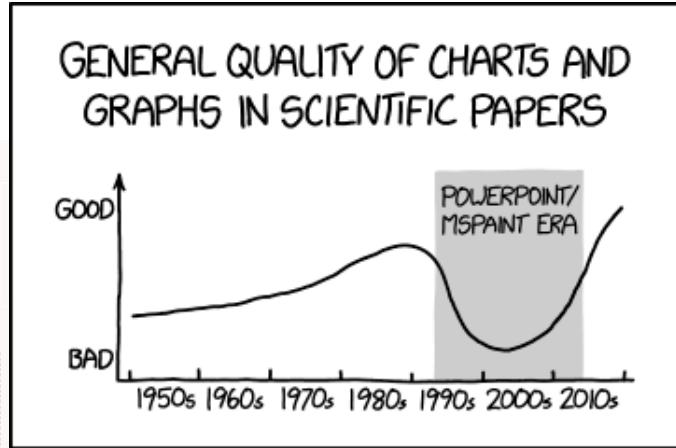
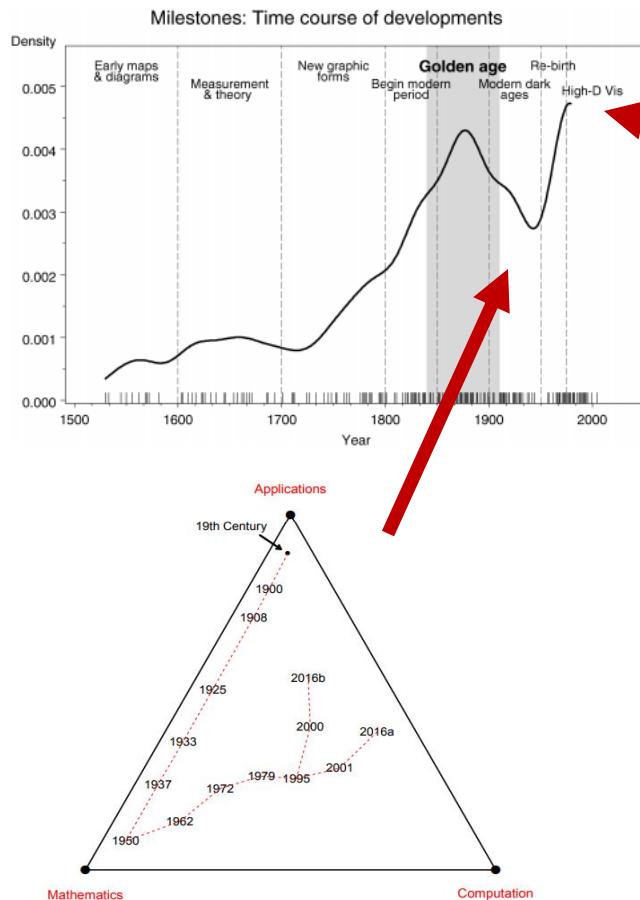


According to Friendly, statistical graphics researched its golden age between 1850-1900



Quick review of the history of data visualization

“Graphical dark ages” around 1950



Currently undergoing a “Graphical re-birth”

Quick review of visualizing data with matplotlib

[Matplotlib](#) is a comprehensive library for creating static, animated, and interactive visualizations.

- `import matplotlib.pyplot as plt`

Types of plots we have created

```
plt.plot(x, y, '-o') # line plot/scatter plot
```

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

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

```
plot.scatter(x, y, s = , color = , marker = )
```



Quick review of visualizing data with matplotlib

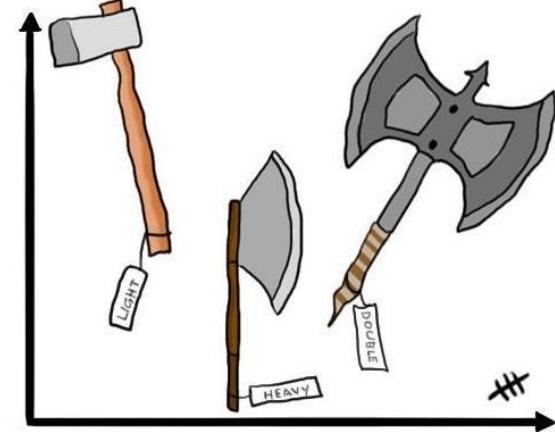
Make sure always label your axes:

```
plt.ylabel("y label")  
plt.xlabel("x label")  
plt.title("my title")  
plt.plot(x, y, label = "blah")  
plt.legend()
```

We can create subplots:

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

Always label your axes



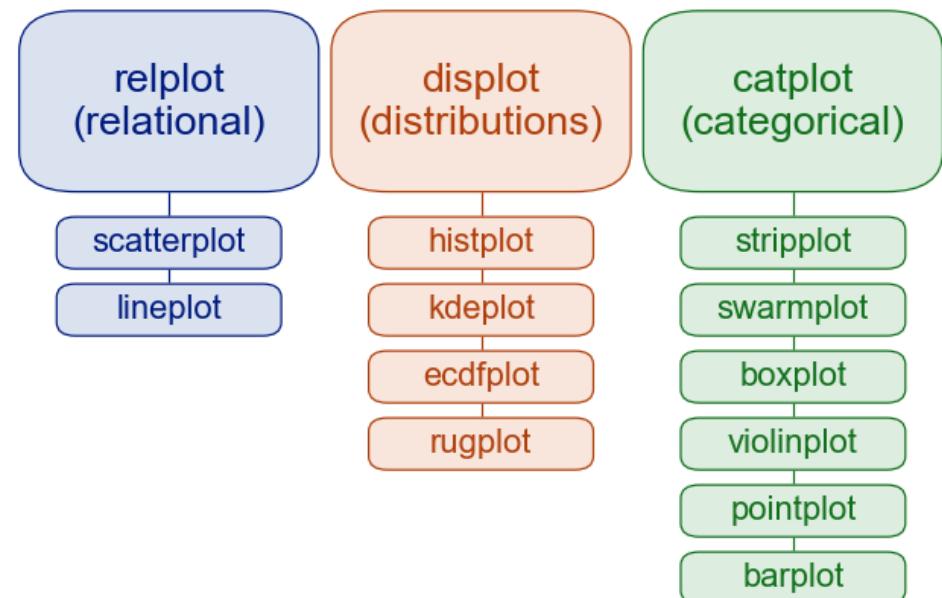
Quick review of seaborn

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



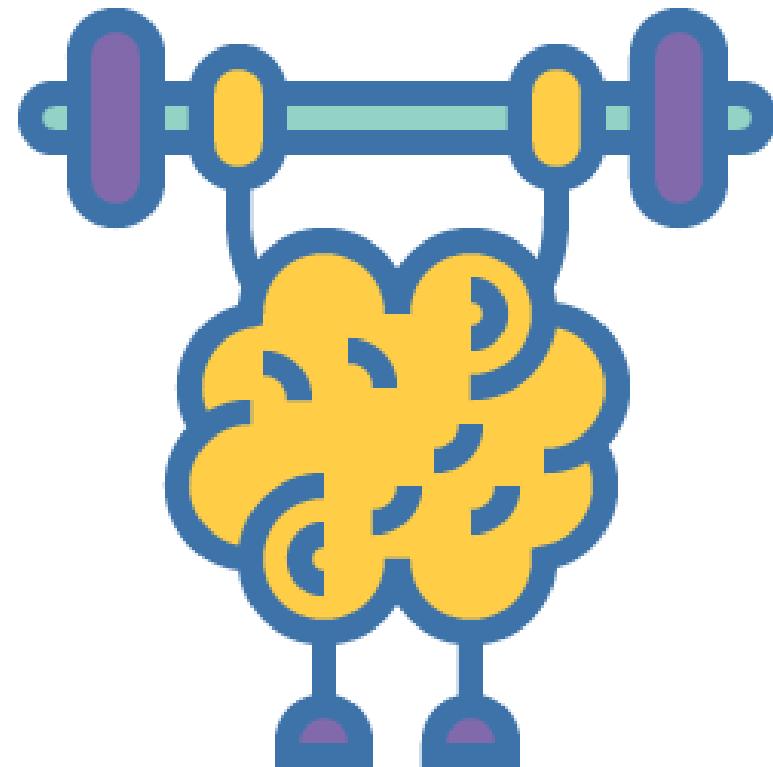
WHEN DOES THIS HAPPEN IN THE MOVIE



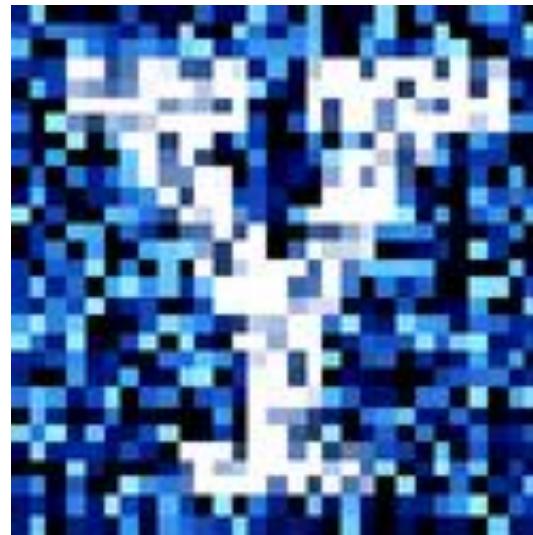
Questions???



PRACTICE QUESTIONS



YData: Introduction to Data Science



Class 14: Interactive data visualizations

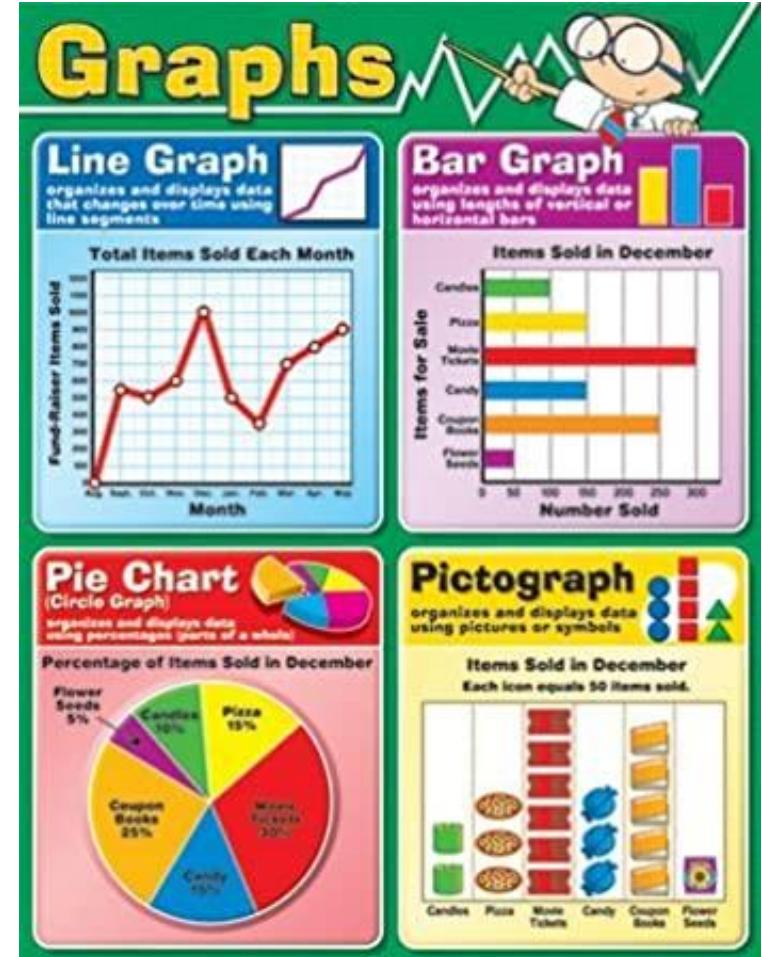
Overview

Show and tell of your visualizations from homework 5

Very quick review seaborn

Interactive graphics with plotly

If there is time: maps!



Announcement: class 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

- It's a chance to practice everything you've learned in class!

The goal of is to present a clear and compelling data analysis showing a few interesting results!

- Ideally something you are proud of that you could show off to potential future employers, etc.

A few sources for data sets are listed on Canvas

- You can use data you collect as well. If you use data from another class, your work must be unique for each class

Announcement: class project

A draft of the project is due on November 10th

- I'm telling you about this early so:
 - If you want to think/work on the project over break you can
 - If don't want to think about it over break you don't have to

There will be a “peer review” period where you will give and receive feedback on three other projects

- Instructions for how to do the review will be given

The final version will be due on December 8th

A Jupyter notebook “template” project will be available soon

Review of data visualization!

*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*



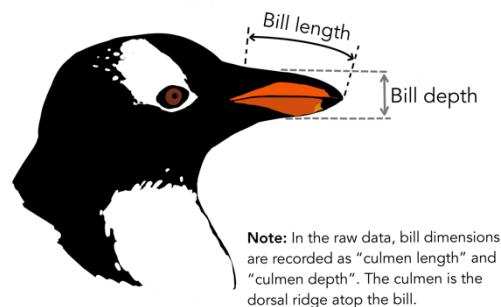
Let's take a few minutes to explore the data visualizations you found and that you created as part of homework 5!

Quick review: Seaborn

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



To explore seaborn, we looked at data on penguins!



```
import seaborn as sns  
  
sns.set_theme()
```

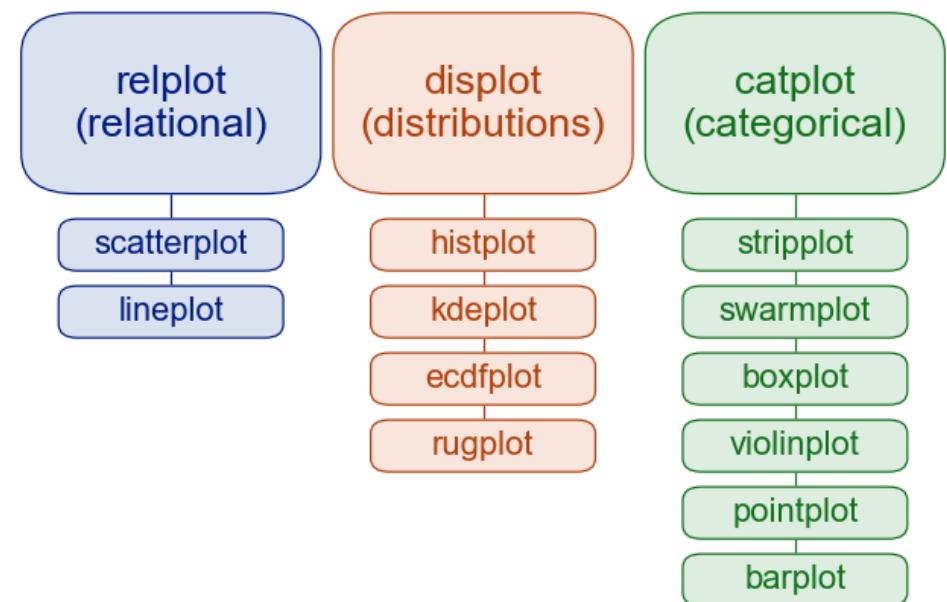
Seaborn figure level plots

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



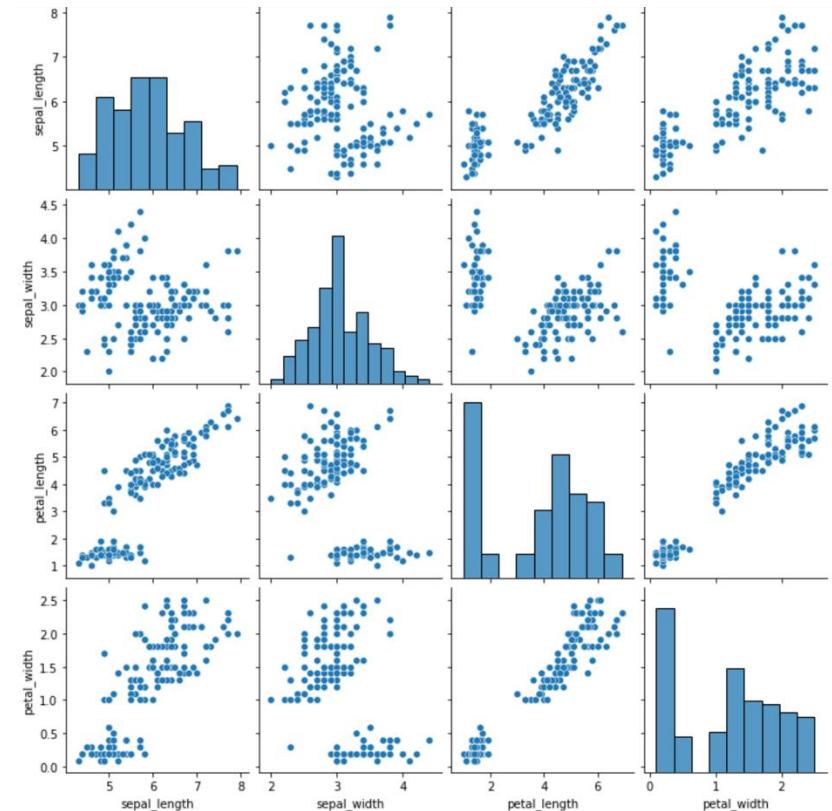
Pairs plot

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

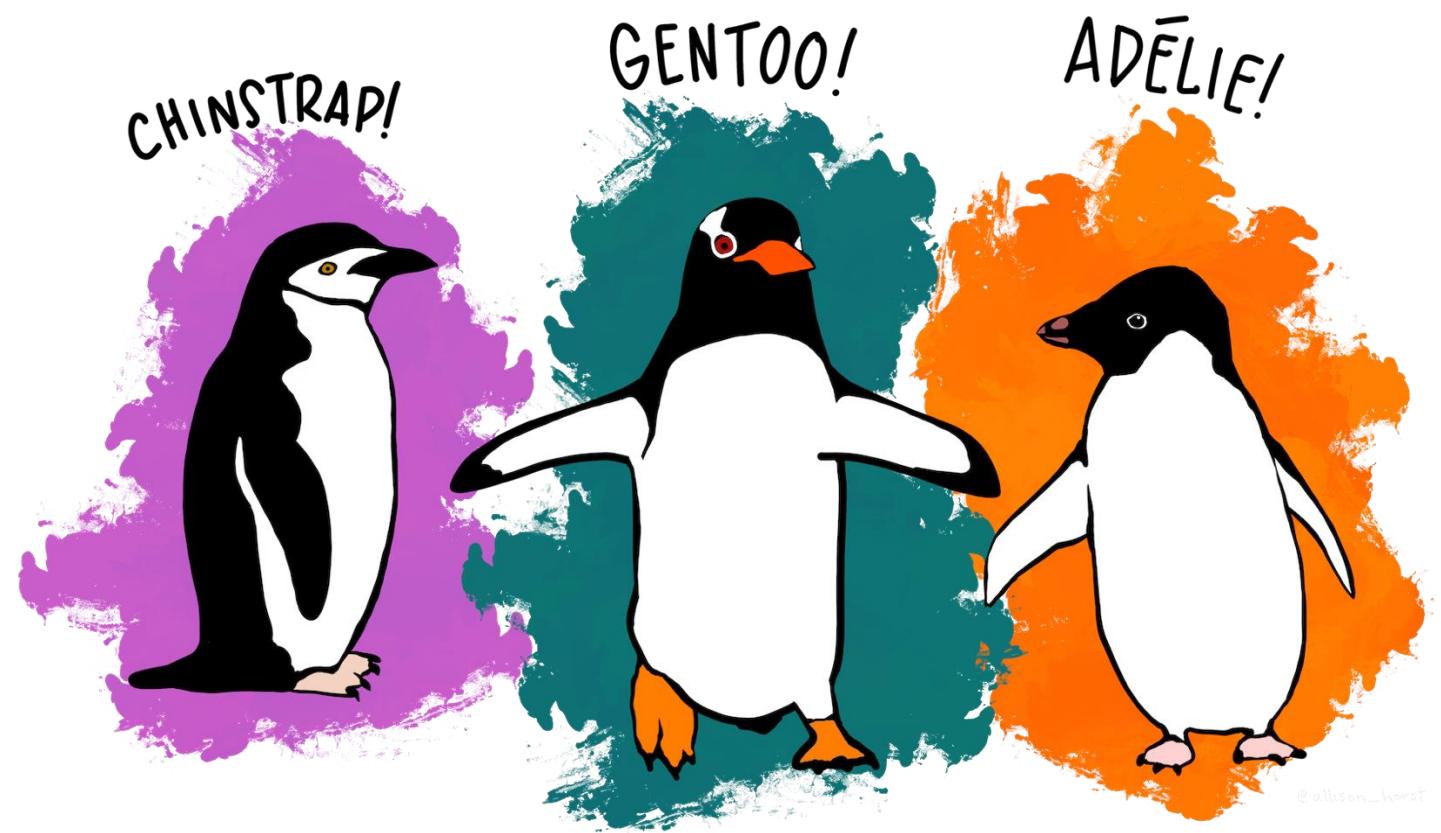
It can be one of the most useful ways to see relationships between multiple quantitative variables

We can create pairs plots in seaborn using:

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



Let's do a quick warm-up exercise!



@allison_horst

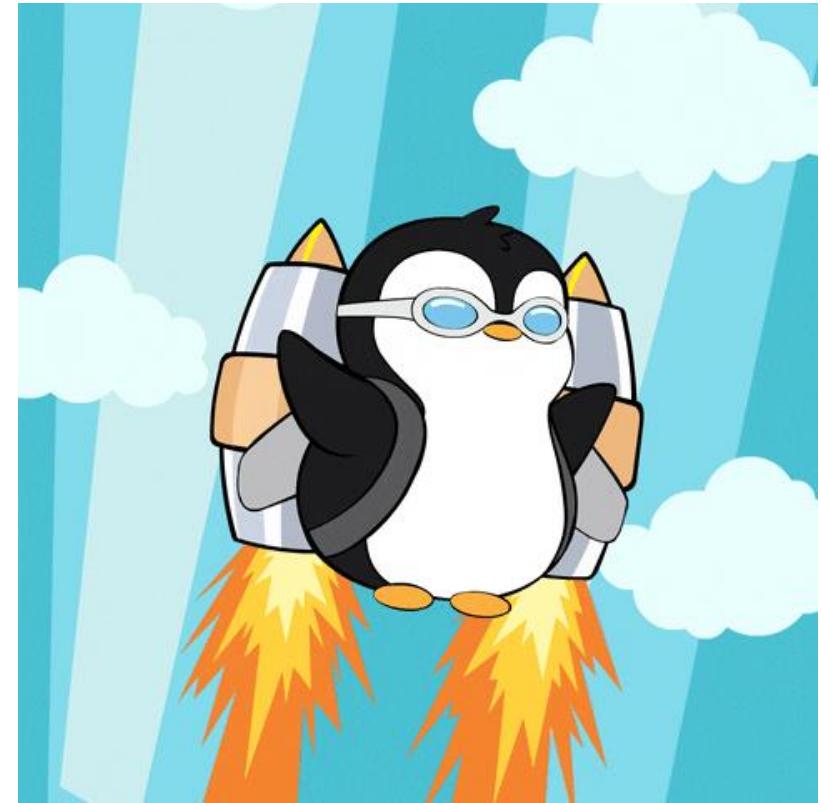
Interactive visualizations for data exploration

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

```
fig = 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!

Plotly interactive plots

Sunburst plots

```
px.sunburst(data_frame = , path = , values = , color = )
```

Treemap

```
px.treemap(data_frame = , path = , values = , color = )
```

Let's explore this in Jupyter!

Pivot Tables and heatmaps

Pivot tables aggregate values based on two grouping variables, and create a table where:

- The rows are the levels of one cat. variable
- The columns are the levels of the second cat. variable
- The values are aggregated over a third quant. variable

```
df2 = df.pivot_table(index = "col1", columns = "col2",
                      values = "col3", aggfunc = "max")
```

```
nba2 = nba.pivot_table(index = "TEAM", columns = "POSITION",
                        values = "SALARY", aggfunc = "max")
```

	PLAYER	TEAM	POSITION	SALARY
0	De'Andre Hunter	Atlanta Hawks	SF	9.835881
1	Jalen Johnson	Atlanta Hawks	SF	2.792640
2	AJ Griffin	Atlanta Hawks	SF	3.536160
3	Trent Forrest	Atlanta Hawks	SG	0.508891
4	John Collins	Atlanta Hawks	PF	23.500000

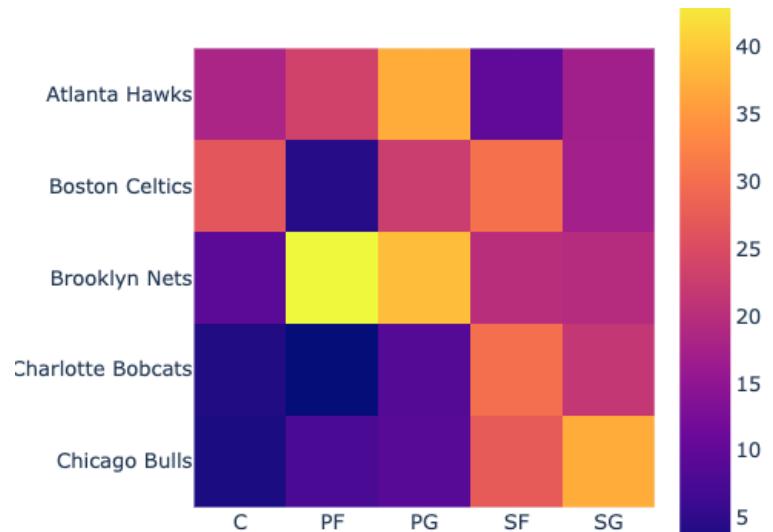
POSITION	C	PF	PG	SF	SG
TEAM					
Atlanta Hawks	18.206896	23.500000	37.096500	9.835881	17.071120
Boston Celtics	26.500000	4.306281	22.000000	30.351780	17.142857
Brooklyn Nets	9.391069	44.119845	38.917057	20.100000	19.500000
Charlotte Bobcats	3.722040	1.563518	8.623920	30.075000	21.486316
Chicago Bulls	3.200000	7.775400	9.030000	27.300000	37.096500

Pivot Tables and heatmaps

One can then visualize the data as a heatmap using plotly or seaborn

```
px.imshow(nba2)      # plotly
```

```
sns.heatmap(nba2)  # seaborn
```



rows → columns → values ↓

	PLAYER	TEAM	POSITION	SALARY
0	De'Andre Hunter	Atlanta Hawks	SF	9.835881
1	Jalen Johnson	Atlanta Hawks	SF	2.792640
2	AJ Griffin	Atlanta Hawks	SF	3.536160
3	Trent Forrest	Atlanta Hawks	SG	0.508891
4	John Collins	Atlanta Hawks	PF	23.500000

	POSITION	C	PF	PG	SF	SG
	TEAM					
Atlanta Hawks	18.206896	23.500000	37.096500	9.835881	17.071120	
Boston Celtics	26.500000	4.306281	22.000000	30.351780	17.142857	
Brooklyn Nets	9.391069	44.119845	38.917057	20.100000	19.500000	
Charlotte Bobcats	3.722040	1.563518	8.623920	30.075000	21.486316	
Chicago Bulls	3.200000	7.775400	9.030000	27.300000	37.096500	

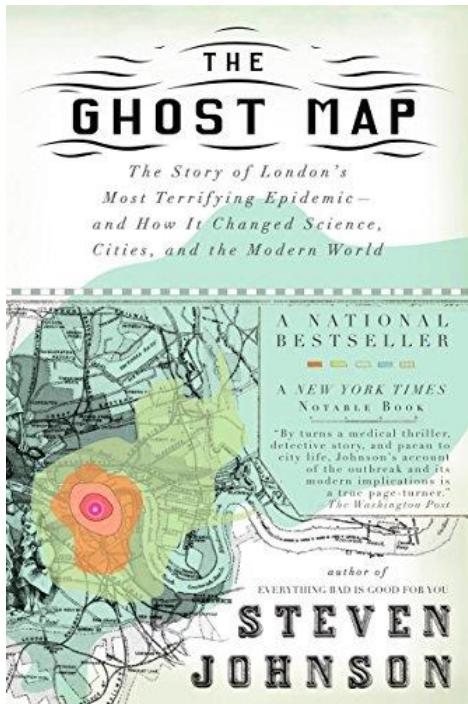
Let's explore this in Jupyter!

Maps

Maps to determine the causes of cholera

Visualizing data on a map can be a powerful way to see spatial trends

- One of the first maps used to show spatial trends was created by John Snow to further his case that cholera was a water born illness



Cholera in London in the 19th century

Cholera reached London in early 1830s

It was greatly feared as it was often deadly

- An outbreak in 1849 killed over 14,000 people in London



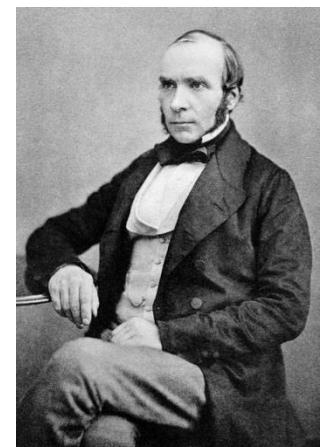
Cause of cholera was unknown. Several theories:

1. Miasmas theory: caused by bad air/smells

- Florence Nightingale, Edwin Chadwick (board of health)

2. Water born disease

- John Snow (anesthesiologist)

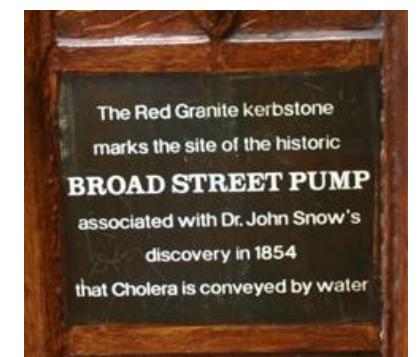
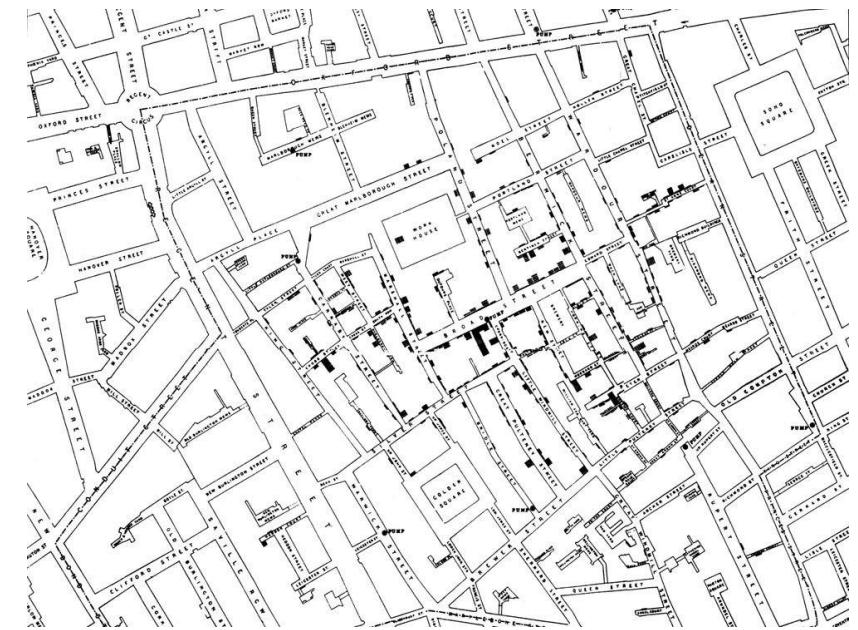


John Snow and spatial mapping

To try to understand the cause of the cholera outbreak of 1854, John Snow plotted a map of cholera deaths

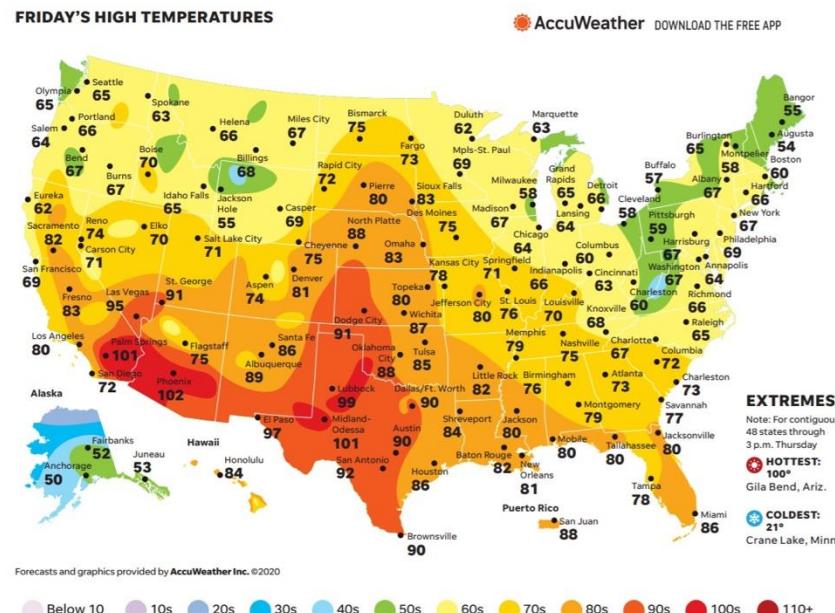
Based on this map and interviews, he concluded that the source of cholera was the Broad Street well

- He famously removed the handle of the well to prevent the spread of disease
- Now he is considered the founder of epidemiology

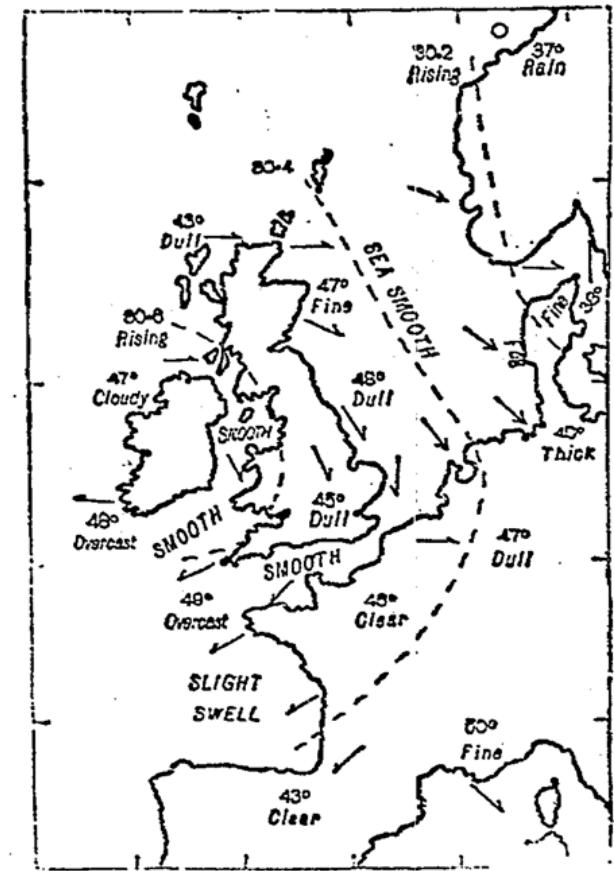


Maps

Another early use where a map gave insight was the mapping of weather by John Galton in 1875



WEATHER CHART, MARCH 31, 1875.



The dotted lines indicate the gradations of barometric pressure
The variations of the temperature are marked by figures, the state
of the sea and sky by descriptive words, and the direction of the wind
by arrows—barbed and feathered according to its force. ◊ denotes
calm.

Galton's first weather map (1875)

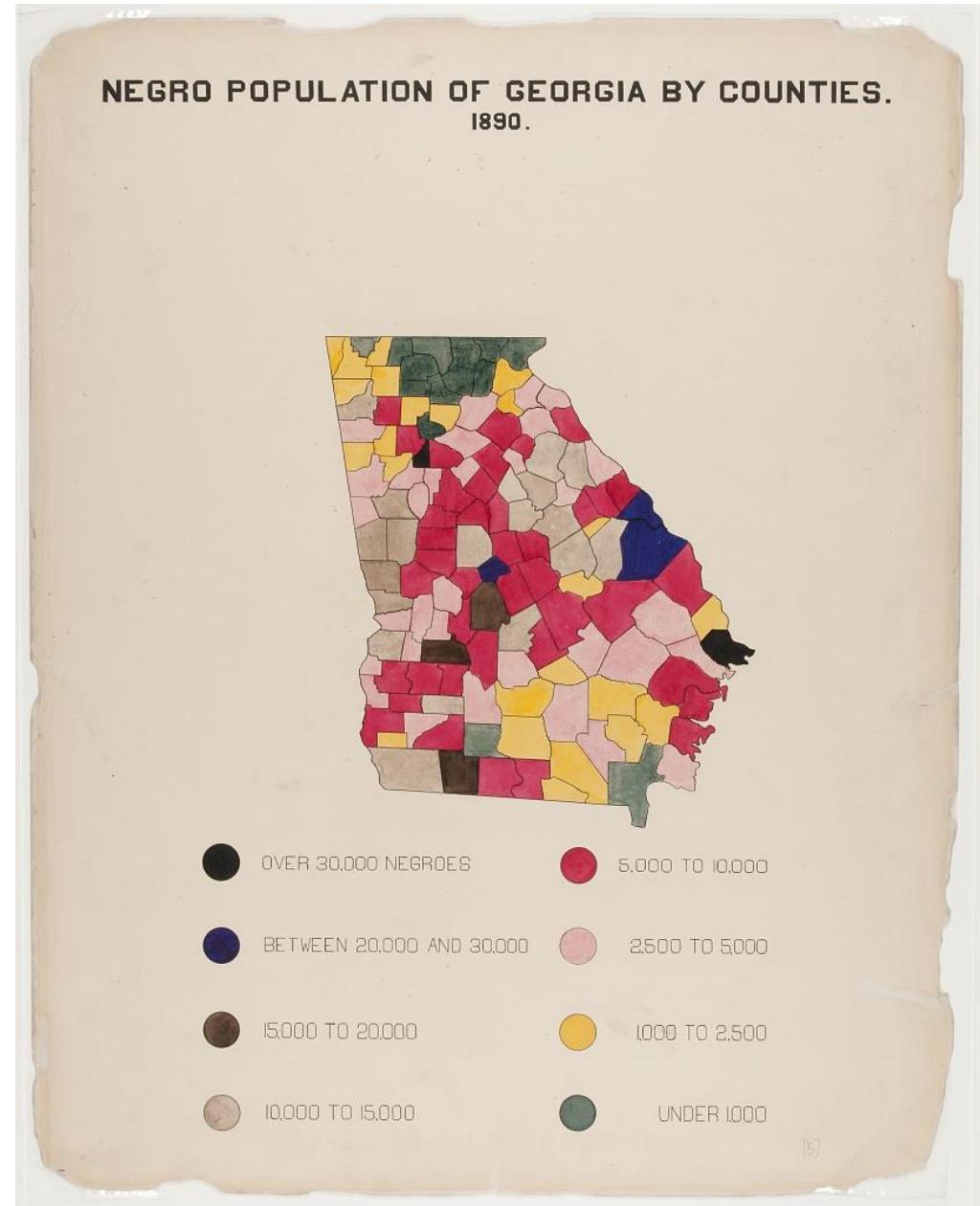
W. E. B. Du Bois

W.E.B. Du Bois was a social scientist and prominent African-American rights activist

Took on the complex task of gathering and manually visualizing the lives of Black Americans in the 1890s

Presented 58 visualizations in the 1900 World's Fair in Paris

- Won a gold award



geopandas

To create maps in Python we will use the geopandas package

```
import geopandas as gpd
```

The key object of interest is the geopandas DataFrame

- It is the same as a regular data frame but it has an extra column called “geometry” that contains geospatial shape features
- The geometry column contains “Shapely” objects used to represent geometric shapes

	key_comb_drvr	geometry
0	M11551	POINT (117.525391 34.008926)
1	M17307	POINT (86.51248 30.474344)
2	M19584	POINT (89.537415 37.157627)
3	M21761	POINT (117.526871 34.00647)
4	M22374	POINT (117.525345 34.008915)
5	U01997A	POINT (84.80533 33.719654)
6	U153601	POINT (78.24838 39.986454)
7	U159393	POINT (98.49438499999999 40.801544)
8	U722222	POINT (84.23309 33.9386)
9	U723030	POINT (83.86456 34.08479)
10	U723333	POINT (85.67151 42.83093)
11	U753333	POINT (117.498535 34.069157)
12	U760505	POINT (90.61252 41.456993)

geopandas

We can read in data as a geopandas DataFrame using

```
map = gpd.read_file('my_file.geojson')
```

We can plot maps using the `gpd.plot()` function

Let's explore this in Jupyter!

Coordinate reference systems

A coordinate reference system (CRS) is a framework used to precisely measure locations on the surface of the Earth as coordinates



The goal of any coordinate reference system is to create a common reference frame in which locations can be measured precisely as coordinates, so that any recipient can identify the same location that was originally intended

- Needed for aligning different layers on maps

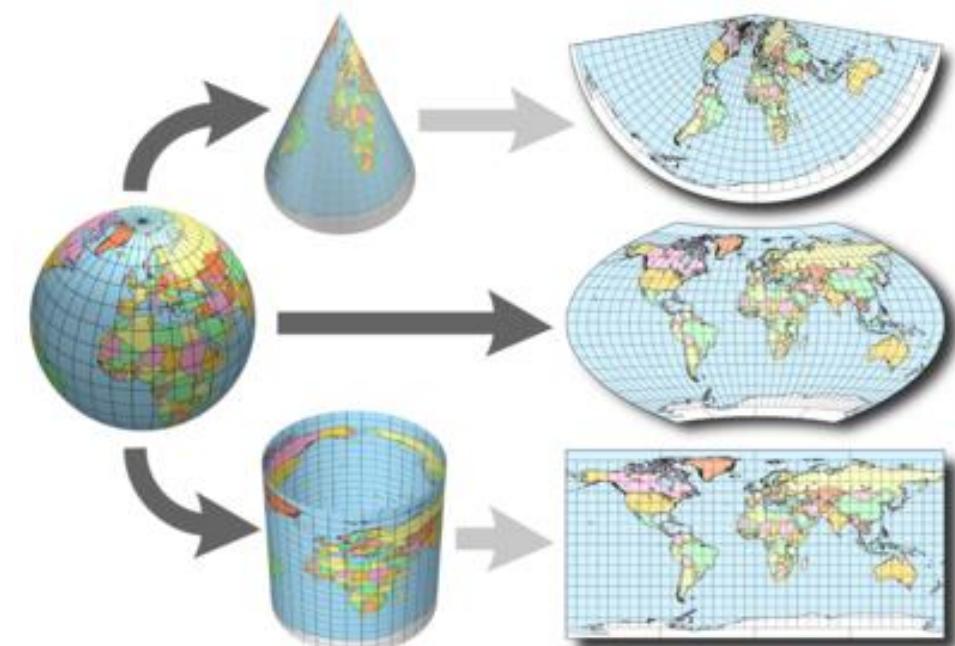


Map projections

Since the earth is a 3D structure, coordinate systems have to project their data onto a 2D maps

Different projects preserve different properties

- **Mercator projection** keeps angles intact
 - Useful for navigation
- **Eckert IV projection** keeps the size of land areas intact



Let's explore this in Jupyter!

WHAT YOUR FAVORITE

MAP PROJECTION SAYS ABOUT YOU

MERCATOR



YOU'RE NOT REALLY INTO MAPS.

VAN DER Grinten



YOU'RE NOT A COMPLICATED PERSON. YOU LOVE THE MERCATOR PROJECTION; YOU JUST WISH IT WEREN'T SQUARE. THE EARTH'S NOT A SQUARE, IT'S A CIRCLE. YOU LIKE CIRCLES. TODAY IS GONNA BE A GOOD DAY!

ROBINSON



YOU HAVE A COMFORTABLE PAIR OF RUNNING SHOES THAT YOU WEAR EVERYWHERE. YOU LIKE COFFEE AND ENJOY THE BEATLES. YOU THINK THE ROBINSON IS THE BEST-LOOKING PROJECTION, HANDS DOWN.

WINKEL-TRIPEL



NATIONAL GEOGRAPHIC ADOPTED THE WINKEL-TRIPEL IN 1998, BUT YOU'VE BEEN A WT FAN SINCE LONG BEFORE "Nat Geo" SHOWED UP. YOU'RE WORRIED IT'S GETTING PLAYED OUT, AND ARE THINKING OF SWITCHING TO THE KAVRAYSKY. YOU ONCE LEFT A PARTY IN DISGUST WHEN A GUEST SHOWED UP WEARING SHOES WITH TOES. YOUR FAVORITE MUSICAL GENRE IS "POST-".

Dymaxion



YOU LIKE ISAAC ASIMOV, XML, AND SHOES WITH TOES. YOU THINK THE SEGWAT GOT A BAD RAP. YOU OWN 3D GOGGLES, WHICH YOU USE TO VIEW ROTATING MODELS OF BETTER 3D GOGGLES. YOU TYPE IN DVORAK.

GOODE HOMOLOSINE



THEY SAY MAPPING THE EARTH ON A 2D SURFACE IS LIKE FLATTENING AN ORANGE PEEL, WHICH SEEMS EASY ENOUGH TO YOU. YOU LIKE EASY SOLUTIONS. YOU THINK WE WOULDN'T HAVE SO MANY PROBLEMS IF WE JUST ELECT MORPHE PEOPLE TO CONGRESS INSTEAD OF POLITICIANS. YOU THINK AIRLINES SHOULD JUST BUY FOOD FROM THE RESTAURANTS NEAR THE GATES AND SERVE THAT ON BOARD. YOU CHANGE YOUR CHS OIL, BUT SECRETLY WONDER IF YOU REALLY NEED TO.

Hobo-Dyer



YOU WANT TO AVOID CULTURAL IMPERIALISM, BUT YOU'VE HEARD BAD THINGS ABOUT GALL-PETERS. YOU'RE CONFUT-AVERSE AND BUY ORGANIC. YOU USE A RECENTLY-INVENTED SET OF GENDER-NEUTRAL PRONOUNS AND THINK THAT WHAT THE WORLD NEEDS IS A REVOLUTION IN CONSCIOUSNESS.

A GLOBE!



YES, YOU'RE VERY CLEVER.

PEIRCE QUINCUNCIAL



YOU THINK THAT WHEN WE LOOK AT A MAP, WHAT WE REALLY SEE IS OURSELVES. AFTER YOU FIRST SAW INCEPTION, YOU SAT SILENT IN THE THEATER FOR SIX HOURS. IT FREAKS YOU OUT TO REALIZE THAT EVERYONE AROUND YOU HAS A SKELETON INSIDE THEM. YOU HAVE REALLY LOOKED AT YOUR HANDS.

PLATE CARRÉE
(EQURECTANGULAR)



YOU THINK THIS ONE IS FINE. YOU LIKE HOW X AND Y MAP TO LATITUDE AND LONGITUDE. THE OTHER PROJECTIONS OVERCOMPLICATE THINGS. YOU WANT ME TO STOP ASKING ABOUT MAPS SO YOU CAN ENJOY DINNER.

WATERMAN BUTTERFLY



REALLY? YOU KNOW THE WATERMAN? HAVE YOU SEEN THE 1909 CHILL MAP IT'S BASED—... YOU HAVE A FRAMED REPRODUCTION AT HOME?! WHOA... LISTEN, FORGET THESE QUESTIONS. ARE YOU DOING ANYTHING TONIGHT?

GALL-PETERS



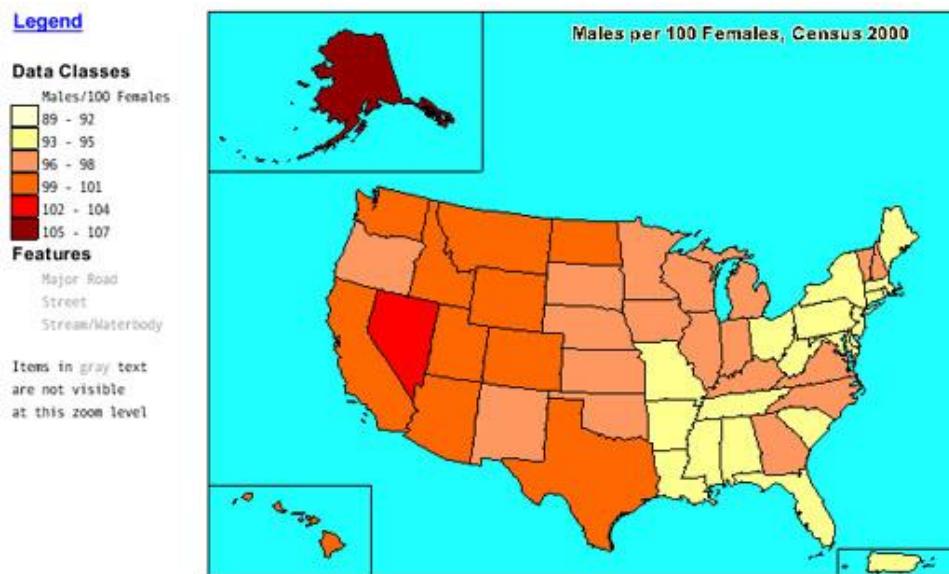
I HATE YOU.

Maps

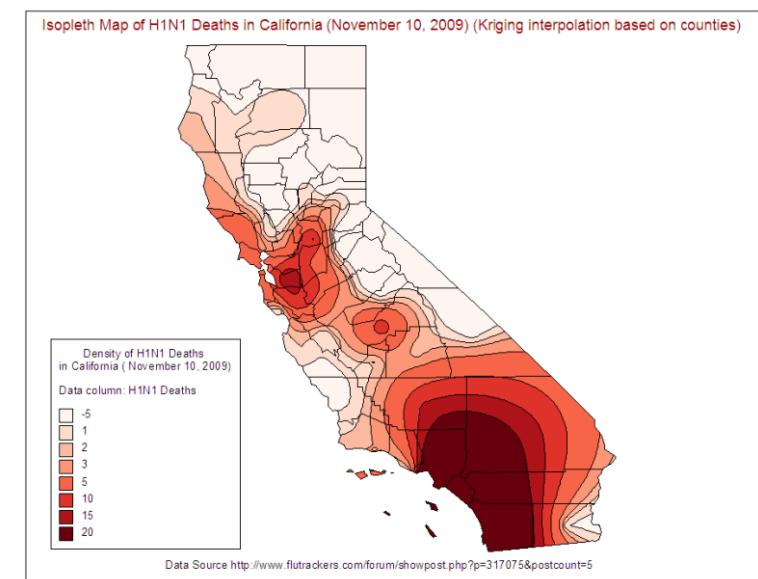
Choropleth maps: shades/colors in predefined areas based on properties of a variable

Isopleth maps: creates regions based on constant values

Choropleth map



Isopleth map



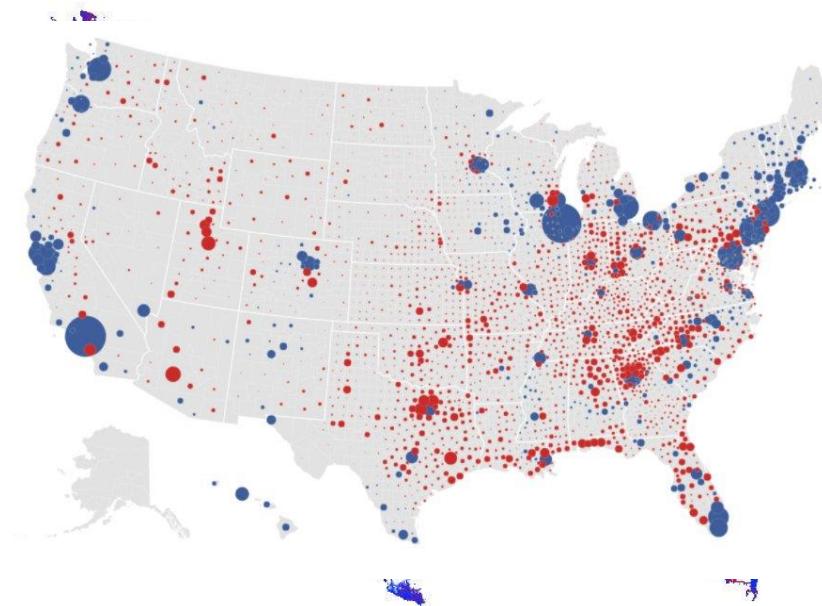
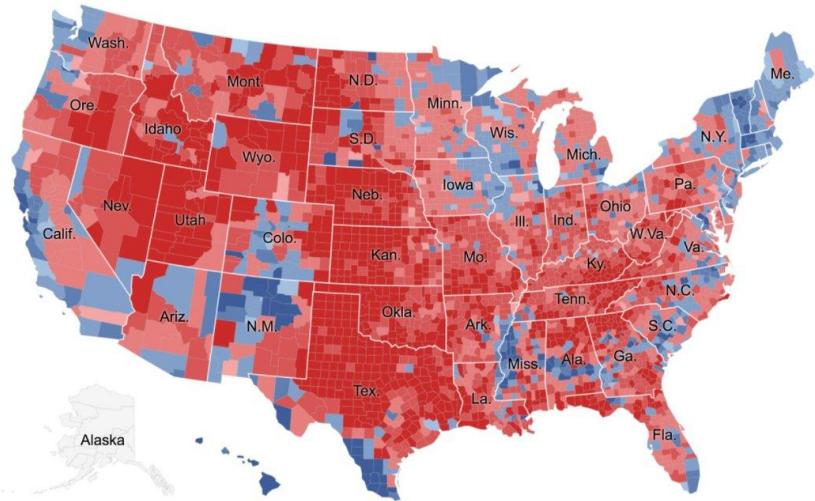
Choropleth maps

We can create choropleth maps using geopandas by joining region information on to a geopandas DataFrame that has a map

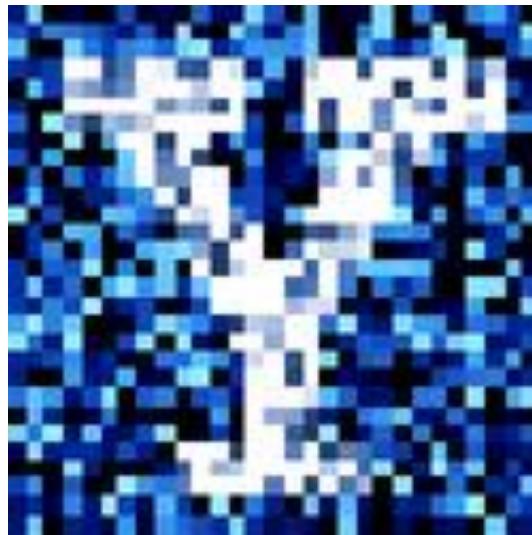
We can then use the `gpd.plot(column =)` method to visualize the map

Let's explore this in Jupyter!

Choropleth maps can sometimes be misleading



YData: Introduction to Data Science



Class 15: Mapping

Overview

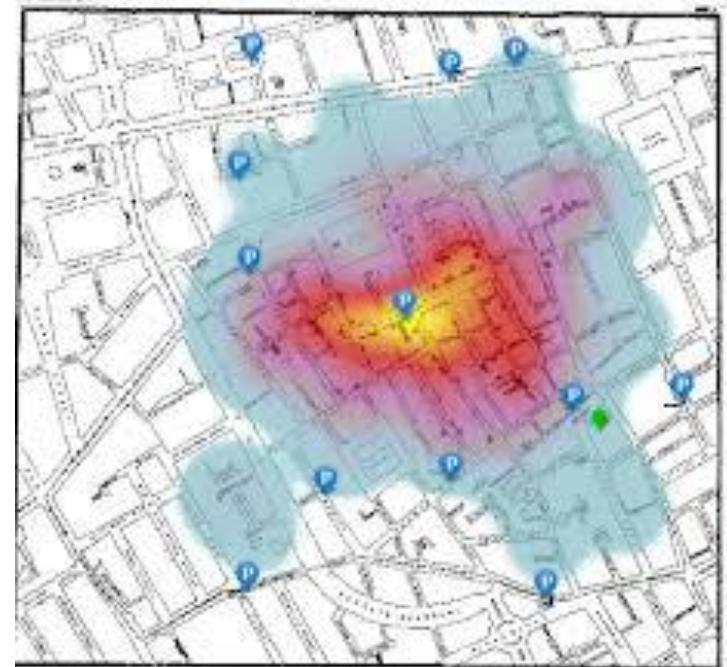
Very quick review of interactive graphics with plotly

Maps continued

- geopandas
- Coordinate reference systems and projections
- Choropleth maps

If there is time

- For loops
- Writing functions



John Snow's ghost map

Reminder: class project

The class project is a **6-10 page** Jupyter notebook report where you analyze data you find interesting.

Think about what questions you want to examine, find data, and load it into Python

- A few sources for data sets are listed on Canvas

You can download a project template Jupyter notebook using:

```
import YData  
YData.download_class_file('project_template.ipynb', 'homework')
```

A **polished** draft of the project is due on **November 10th**



Collaborative projects?

Note: you can submit a collaborative project with one other person

- Project should be twice as long and twice as impressive!
 - i.e., 10-16 pages, more in depth analyses, etc.

Homework 6 is due on Sunday

- We will cover all material you need to complete the homework after today's class, so start early so you have some time to also work on your project

Finally, I encourage everyone to continue to attend practice sessions

- Particularly if you found the midterm difficult

Where we are and where we're going...

What we have covered:

- What is Data Science
- Basics of Python (data types, lists, etc.)
- Numerical computations (numpy)
- Data tables (pandas)
- Data visualization (matplotlib and seaborn)
- Interactive graphics

Today: Mapping

The rest of the semester:

- Functions and for loops
- Statistical analysis
- Machine Learning
- Ethics and conclusions



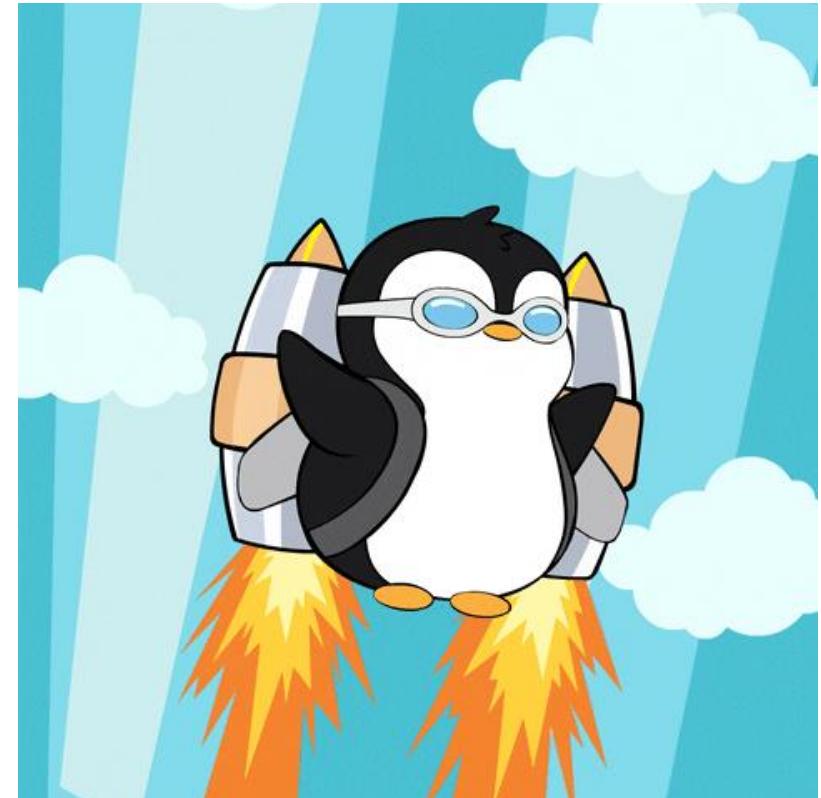
Interactive visualizations for data exploration

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 used plotly to create interactive graphics

```
import plotly.express as px
```



Plotly interactive plots

Interactive plots:

- px.line()
- px.scatter()
- px.sunburst()
- px.treemap()

	PLAYER	TEAM	POSITION	SALARY
0	De'Andre Hunter	Atlanta Hawks	SF	9.835881
1	Jalen Johnson	Atlanta Hawks	SF	2.792640
2	AJ Griffin	Atlanta Hawks	SF	3.536160
3	Trent Forrest	Atlanta Hawks	SG	0.508891
4	John Collins	Atlanta Hawks	PF	23.500000



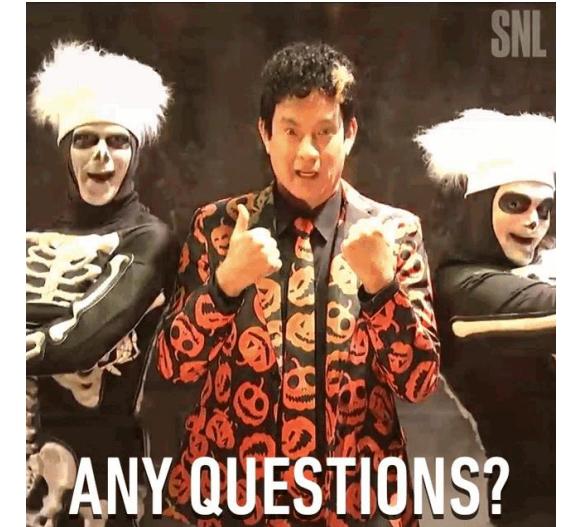
POSITION	C	PF	PG	SF	SG
TEAM					
Atlanta Hawks	18.206896	23.500000	37.096500	9.835881	17.071120
Boston Celtics	26.500000	4.306281	22.600000	30.351780	17.142857
Brooklyn Nets	9.391069	44.119845	38.917057	20.100000	19.500000
Charlotte Bobcats	3.722040	1.563518	8.623920	30.075000	21.486316
Chicago Bulls	3.200000	7.775400	9.030000	27.300000	37.096500

Pivot tables:

```
df2 = df.pivot_table(index = "col1", columns = "col2",
                      values = "col3", aggfunc = "mean")
```

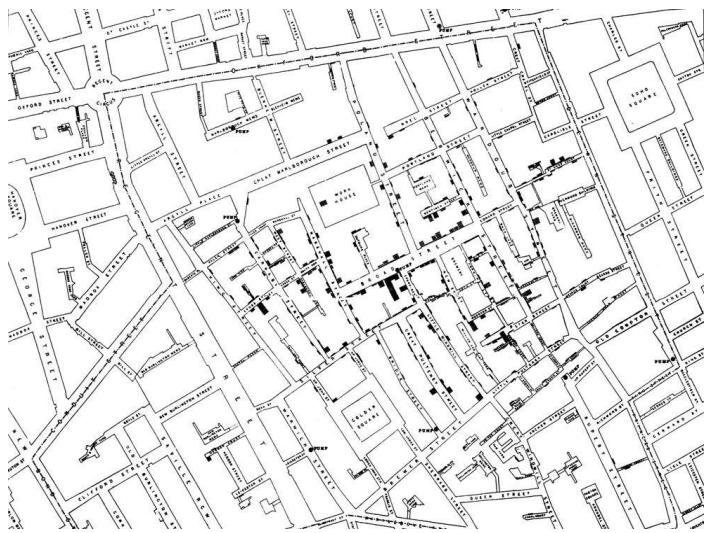
Once we have a 2D table, we can visualize it using:

- px.imshow(df2) # create a heatmap using plotly
- sns.heatmap(df2) # create a heatmap using seaborn

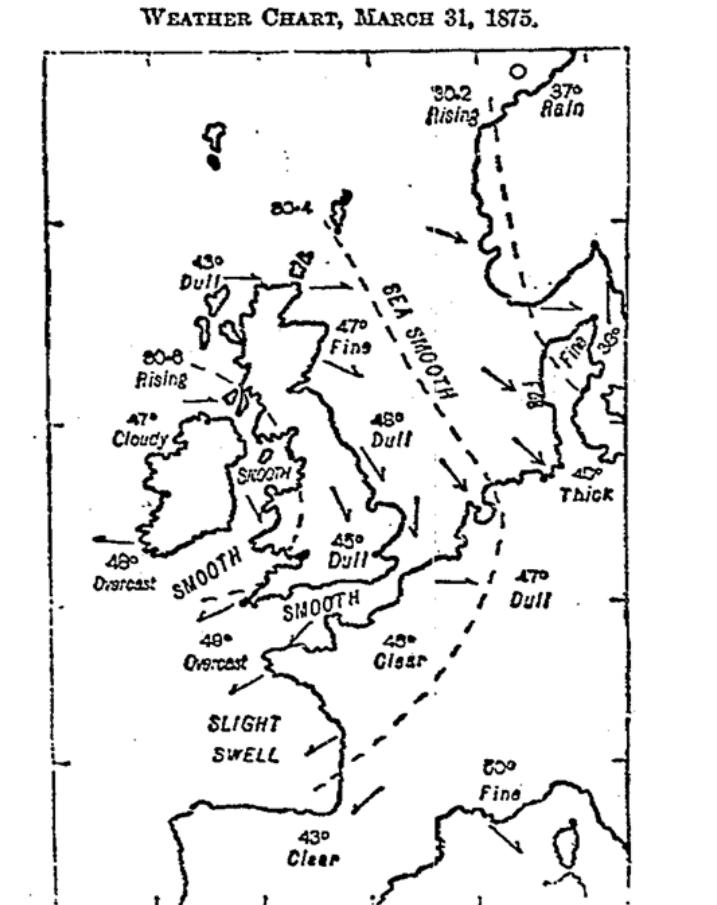


Maps

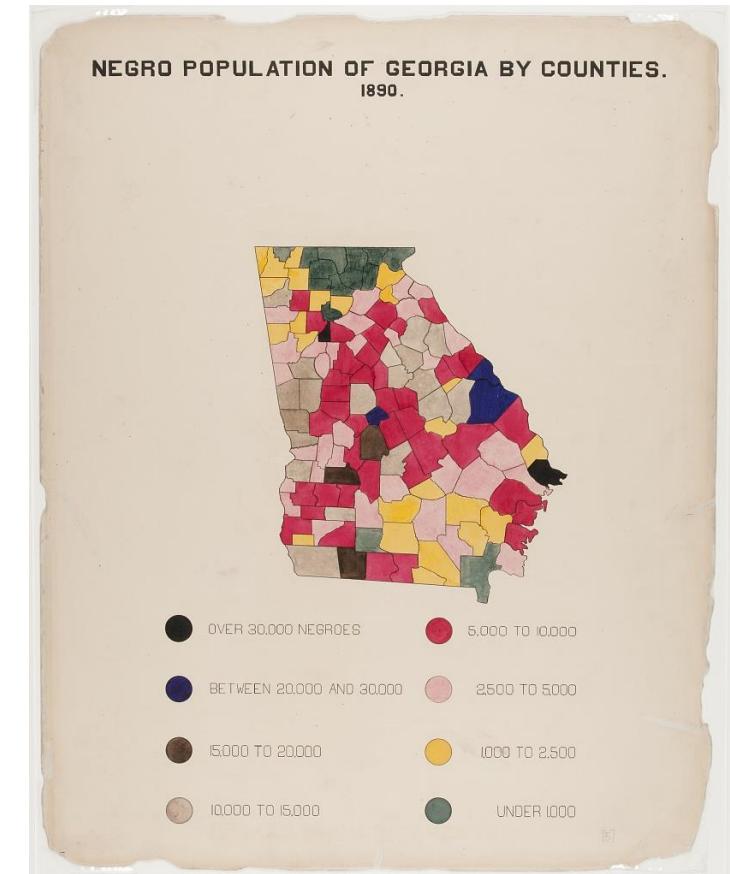
Review Maps



Snow's map of cholera deaths (1854)



Galton's weather map (1875)



Du Bois maps of African-Americans demographics (1900)

geopandas

To create maps in Python we will use the geopandas package

```
import geopandas as gpd
```

The key object of interest is the geopandas DataFrame

- It is the same as a regular data frame but it has an extra column called “geometry” that contains geospatial shape features
- The geometry column contains “Shapely” objects used to represent geometric shapes

	key_comb_drvr	geometry
0	M11551	POINT (117.525391 34.008926)
1	M17307	POINT (86.51248 30.474344)
2	M19584	POINT (89.537415 37.157627)
3	M21761	POINT (117.526871 34.00647)
4	M22374	POINT (117.525345 34.008915)
5	U01997A	POINT (84.80533 33.719654)
6	U153601	POINT (78.24838 39.986454)
7	U159393	POINT (98.49438499999999 40.801544)
8	U722222	POINT (84.23309 33.9386)
9	U723030	POINT (83.86456 34.08479)
10	U723333	POINT (85.67151 42.83093)
11	U753333	POINT (117.498535 34.069157)
12	U760505	POINT (90.61252 41.456993)

geopandas

We can read in data as a geopandas DataFrame using

```
map = gpd.read_file('my_file.geojson')
```

We can plot maps using the `gpd.plot()` function

Let's explore this in Jupyter!

Coordinate reference systems

A coordinate reference system (CRS) is a framework used to precisely measure locations on the surface of the Earth as coordinates



The goal of any coordinate reference system is to create a common reference frame in which locations can be measured precisely as coordinates, so that any recipient can identify the same location that was originally intended.

- Needed for aligning different layers on maps

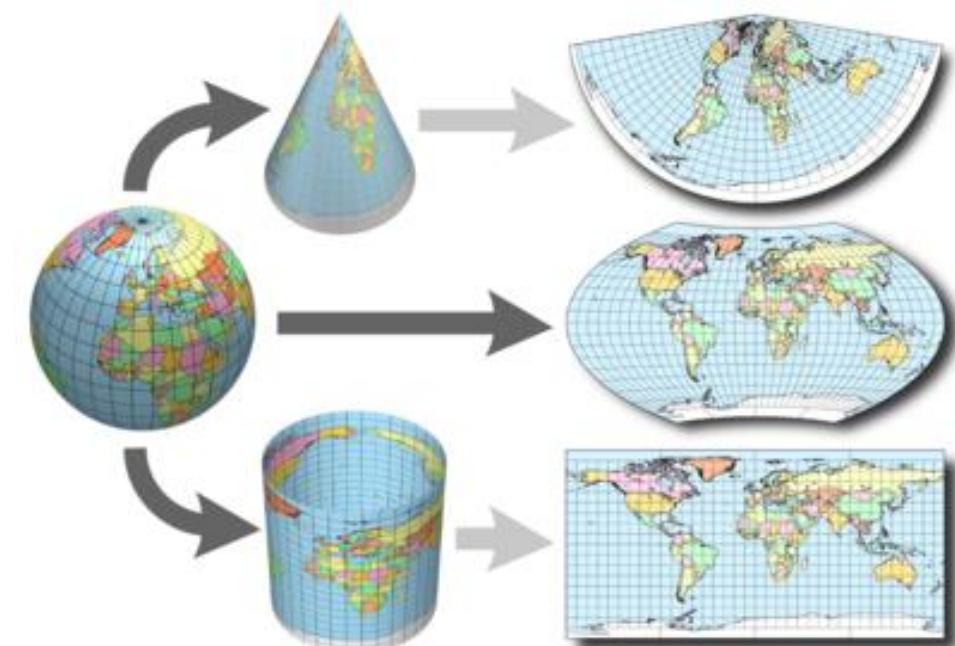


Map projections

Since the earth is a 3D structure, coordinate systems have to project their data onto a 2D maps

Different projects preserve different properties

- **Mercator projection** keeps angles intact
 - Useful for navigation
- **Eckert IV projection** keeps the size of land areas intact



Let's explore this in Jupyter!

WHAT YOUR FAVORITE

MAP PROJECTION SAYS ABOUT YOU

MERCATOR



YOU'RE NOT REALLY INTO MAPS.

VAN DER Grinten



YOU'RE NOT A COMPLICATED PERSON. YOU LOVE THE MERCATOR PROJECTION; YOU JUST WISH IT WEREN'T SQUARE. THE EARTH'S NOT A SQUARE, IT'S A CIRCLE. YOU LIKE CIRCLES. TODAY IS GONNA BE A GOOD DAY!

ROBINSON



YOU HAVE A COMFORTABLE PAIR OF RUNNING SHOES THAT YOU WEAR EVERYWHERE. YOU LIKE COFFEE AND ENJOY THE BEATLES. YOU THINK THE ROBINSON IS THE BEST-LOOKING PROJECTION, HANDS DOWN.

WINKEL-TRIPEL



NATIONAL GEOGRAPHIC ADOPTED THE WINKEL-TRIPEL IN 1998, BUT YOU'VE BEEN A WT FAN SINCE LONG BEFORE "Nat Geo" SHOWED UP. YOU'RE WORRIED IT'S GETTING PLAYED OUT, AND ARE THINKING OF SWITCHING TO THE KAVRAYSKY. YOU ONCE LEFT A PARTY IN DISGUST WHEN A GUEST SHOWED UP WEARING SHOES WITH TOES. YOUR FAVORITE MUSICAL GENRE IS "POST-".

Dymaxion



YOU LIKE ISAAC ASIMOV, XML, AND SHOES WITH TOES. YOU THINK THE SEGWAF GOT A BAD RAP. YOU OWN 3D GOGGLES, WHICH YOU USE TO VIEW ROTATING MODELS OF BETTER 3D GOGGLES. YOU TYPE IN DVORAK.

GOODE HOMOLOSINE



THEY SAY MAPPING THE EARTH ON A 2D SURFACE IS LIKE FLATTENING AN ORANGE PEEL, WHICH SEEMS EASY ENOUGH TO YOU. YOU LIKE EASY SOLUTIONS. YOU THINK WE WOULDN'T HAVE SO MANY PROBLEMS IF WE JUST ELECT MORPHE PEOPLE TO CONGRESS INSTEAD OF POLITICIANS. YOU THINK AIRLINES SHOULD JUST BUY FOOD FROM THE RESTAURANTS NEAR THE GATES AND SERVE THAT ON BOARD. YOU CHANGE YOUR CHS OIL, BUT SECRETLY WONDER IF YOU REALLY NEED TO.

Hobo-Dyer



YOU WANT TO AVOID CULTURAL IMPERIALISM, BUT YOU'VE HEARD BAD THINGS ABOUT GALL-PETERS. YOU'RE CONFUT-AVERSE AND BUY ORGANIC. YOU USE A RECENTLY-INVENTED SET OF GENDER-NEUTRAL PRONOUNS AND THINK THAT WHAT THE WORLD NEEDS IS A REVOLUTION IN CONSCIOUSNESS.

A GLOBE!



YES, YOU'RE VERY CLEVER.

PEIRCE QUINCUNCIAL



YOU THINK THAT WHEN WE LOOK AT A MAP, WHAT WE REALLY SEE IS OURSELVES. AFTER YOU FIRST SAW INCEPTION, YOU SAT SILENT IN THE THEATER FOR SIX HOURS. IT FREAKS YOU OUT TO REALIZE THAT EVERYONE AROUND YOU HAS A SKELETON INSIDE THEM. YOU HAVE REALLY LOOKED AT YOUR HANDS.

PLATE CARRÉE
(EQURECTANGULAR)



YOU THINK THIS ONE IS FINE. YOU LIKE HOW X AND Y MAP TO LATITUDE AND LONGITUDE. THE OTHER PROJECTIONS OVERCOMPLICATE THINGS. YOU WANT ME TO STOP ASKING ABOUT MAPS SO YOU CAN ENJOY DINNER.

WATERMAN BUTTERFLY



REALLY? YOU KNOW THE WATERMAN? HAVE YOU SEEN THE 1909 CHILL MAP IT'S BASED—... YOU HAVE A FRAMED REPRODUCTION AT HOME?! WHOA... LISTEN, FORGET THESE QUESTIONS. ARE YOU DOING ANYTHING TONIGHT?

GALL-PETERS



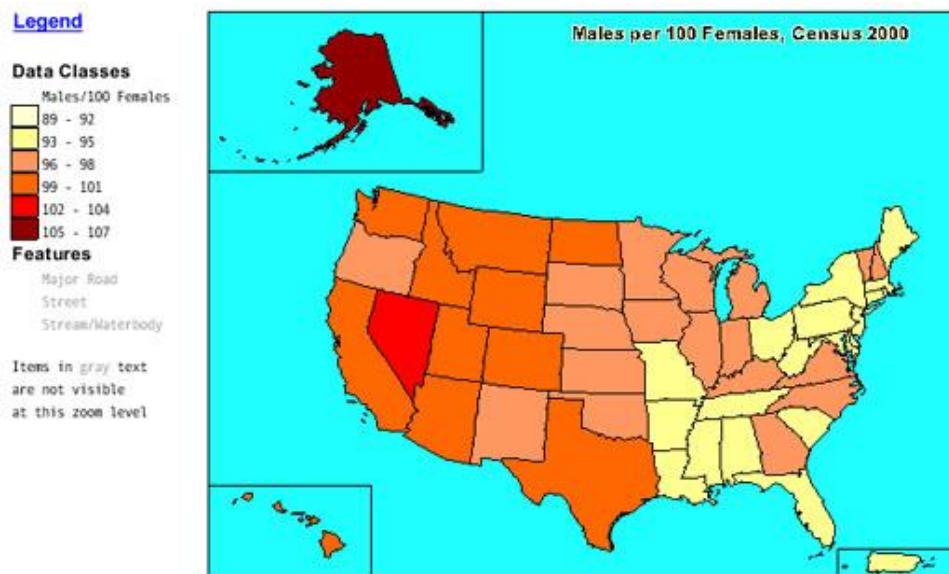
I HATE YOU.

Maps

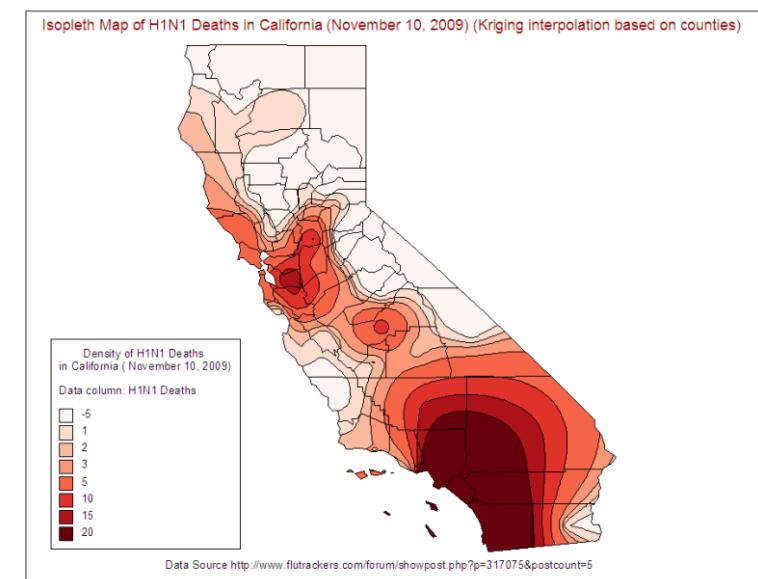
Choropleth maps: shades/colors in predefined areas based on properties of a variable

Isopleth maps: creates regions based on constant values

Choropleth map



Isopleth map



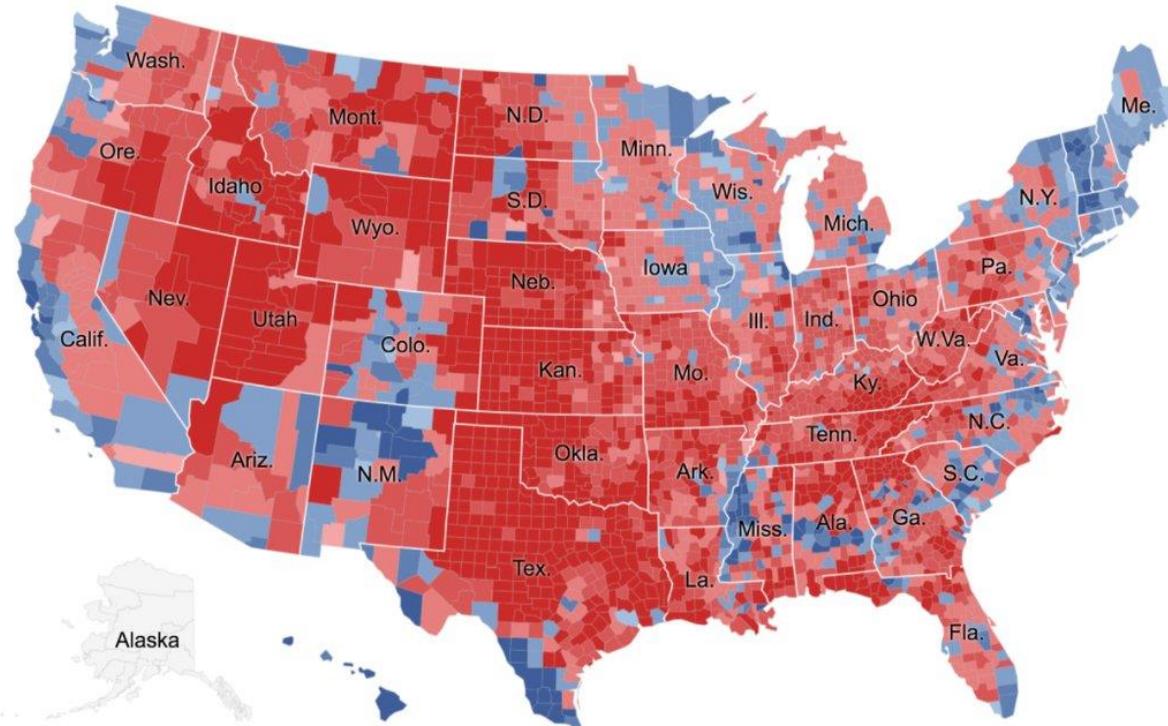
Choropleth maps

We can create choropleth maps using geopandas by joining region information on to a geopandas DataFrame that has a map

We can then use the `gpd.plot(column =)` method to visualize the map

Let's explore this in Jupyter!

Question: in what way could this map be misleading?



Darker red: county had higher % Trump vote
Darker blue: county had higher % Clinton vote

Loops

For loops

For loops repeat a process many times, iterating over a sequence of items

- Often we are iterating over an array of sequential numbers

```
animals = ["cat", "dog", "bat"]
```

```
for creature in animals:  
    print(creature)
```

```
for i in np.arange(4):  
    print(i**2)
```



Review: ranges

A range gives us a sequence of consecutive numbers

An sequence of increasing integers from 0 up to *end* - 1

- `range(end)`

An sequence of increasing integers from *start* up to *end* - 1

- `range(start, end)`

A sequence with step between consecutive values

- `range(start, end, step)`

The range always includes start but excludes end



Let's explore this in Jupyter!

Enumerate and zip

We can use the `enumerate()` function to both items in a list, and sequential integers:

```
animals = ["cat", "dog", "bat"]
for i, creature in enumerate(animals):
    print(i, creature)
```

cat -> feline, dog -> canine, bat -> ?



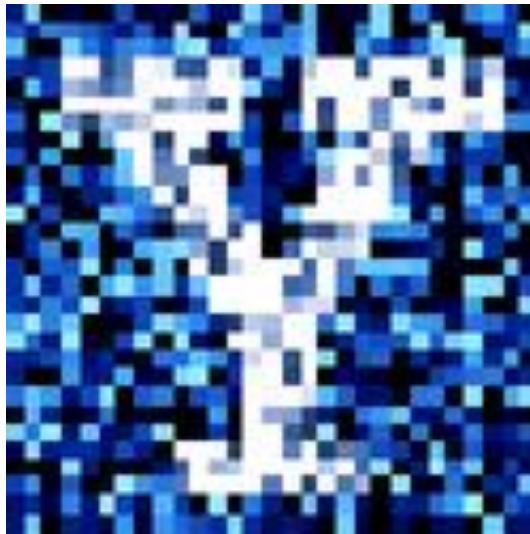
ChatGPT can make mistakes. Check important info.

We can use the `zip()` function to get items for two lists:

```
animal_order = ["feline", "canine", "chiropteran"]
for curr_order, curr_animal in zip(animal_order, animals):
    print(curr_order, curr_animal)
```

Let's explore this in Jupyter!

YData: Introduction to Data Science



Class 16: For loops and writing functions

Overview

Very quick review of mapping

For loops

Conditional statements

Writing functions



Reminder: class project

The class project is a **6-10 page** Jupyter notebook report where you analyze data you find interesting.

Think about what questions you want to examine, find data, and load it into Python

- A few sources for data sets are listed on Canvas

You can download a project template Jupyter notebook using:

```
import YData  
YData.download_class_file('project_template.ipynb', 'homework')
```

A **polished** draft of the project is due on **November 10th**



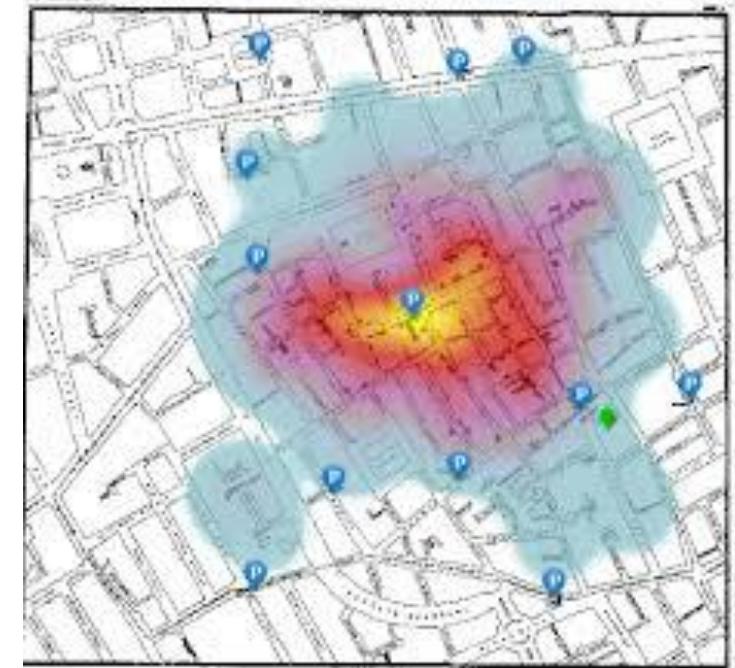
Quick review of mapping

Quick review of maps

Visualizing data on a map can be a powerful way to see spatial trends

We can create maps in Python using geopandas DataFrames

- Like regular DataFrames with an additional geometry column that has Shapely objects



John Snow's ghost map (1854)

	key_comb_drvr	geometry
0	M11551	POINT (117.525391 34.008926)
1	M17307	POINT (86.51248 30.474344)
2	M19584	POINT (89.537415 37.157627)

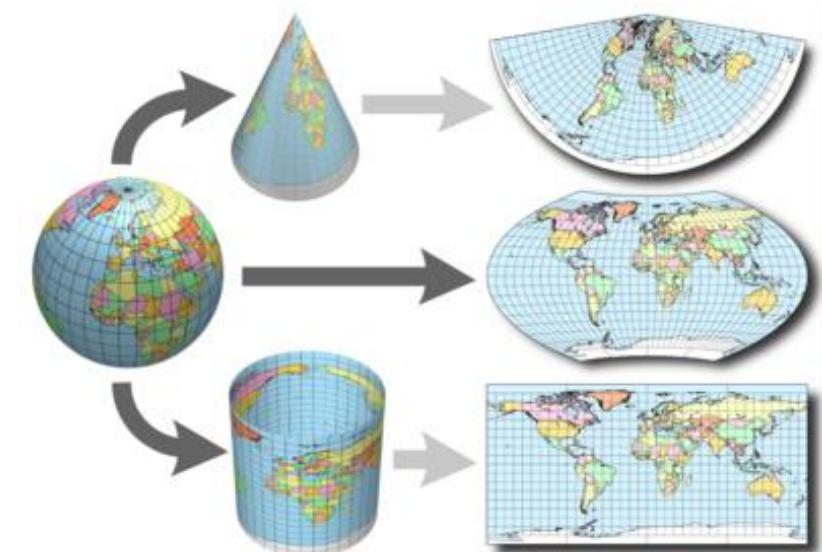
Review: CRSs and map projections

A coordinate reference system (CRS) is a framework used to precisely measure locations on the surface of the Earth as coordinates

- Needed for aligning different layers on maps

There are many map projections to display Earth's 3D structure on a 2D map surface.

- **Mercator projection** keeps angles intact
- **Eckert IV projection** keeps the size of land areas intact



WHAT YOUR FAVORITE

MAP PROJECTION SAYS ABOUT YOU

MERCATOR



YOU'RE NOT REALLY INTO MAPS.

VAN DER Grinten



YOU'RE NOT A COMPLICATED PERSON. YOU LOVE THE MERCATOR PROJECTION; YOU JUST WISH IT WEREN'T SQUARE. THE EARTH'S NOT A SQUARE, IT'S A CIRCLE. YOU LIKE CIRCLES. TODAY IS GONNA BE A GOOD DAY!

ROBINSON



YOU HAVE A COMFORTABLE PAIR OF RUNNING SHOES THAT YOU WEAR EVERYWHERE. YOU LIKE COFFEE AND ENJOY THE BEATLES. YOU THINK THE ROBINSON IS THE BEST-LOOKING PROJECTION, HANDS DOWN.

WINKEL-TRIPEL



NATIONAL GEOGRAPHIC ADOPTED THE WINKEL-TRIPEL IN 1998, BUT YOU'VE BEEN A WT FAN SINCE LONG BEFORE "Nat Geo" SHOWED UP. YOU'RE WORRIED IT'S GETTING PLAYED OUT, AND ARE THINKING OF SWITCHING TO THE KAVRAYSKY. YOU ONCE LEFT A PARTY IN DISGUST WHEN A GUEST SHOWED UP WEARING SHOES WITH TOES. YOUR FAVORITE MUSICAL GENRE IS "POST-".

Dymaxion



YOU LIKE ISAAC ASIMOV, XML, AND SHOES WITH TOES. YOU THINK THE SEGWAT GOT A BAD RAP. YOU OWN 3D GOGGLES, WHICH YOU USE TO VIEW ROTATING MODELS OF BETTER 3D GOGGLES. YOU TYPE IN DVORAK.

GOODE HOMOLOSINE



THEY SAY MAPPING THE EARTH ON A 2D SURFACE IS LIKE FLATTENING AN ORANGE PEEL, WHICH SEEMS EASY ENOUGH TO YOU. YOU LIKE EASY SOLUTIONS. YOU THINK WE WOULDN'T HAVE SO MANY PROBLEMS IF WE JUST ELECT MORPHE PEOPLE TO CONGRESS INSTEAD OF POLITICIANS. YOU THINK AIRLINES SHOULD JUST BUY FOOD FROM THE RESTAURANTS NEAR THE GATES AND SERVE THAT ON BOARD. YOU CHANGE YOUR CHS OIL, BUT SECRETLY WONDER IF YOU REALLY NEED TO.

Hobo-Dyer



YOU WANT TO AVOID CULTURAL IMPERIALISM, BUT YOU'VE HEARD BAD THINGS ABOUT GALL-PETERS. YOU'RE CONFUT-AVERSE AND BUY ORGANIC. YOU USE A RECENTLY-INVENTED SET OF GENDER-NEUTRAL PRONOUNS AND THINK THAT WHAT THE WORLD NEEDS IS A REVOLUTION IN CONSCIOUSNESS.

A GLOBE!



YES, YOU'RE VERY CLEVER.

PEIRCE QUINCUNCIAL



YOU THINK THAT WHEN WE LOOK AT A MAP, WHAT WE REALLY SEE IS OURSELVES. AFTER YOU FIRST SAW INCEPTION, YOU SAT SILENT IN THE THEATER FOR SIX HOURS. IT FREAKS YOU OUT TO REALIZE THAT EVERYONE AROUND YOU HAS A SKELETON INSIDE THEM. YOU HAVE REALLY LOOKED AT YOUR HANDS.

PLATE CARRÉE
(EQURECTANGULAR)



YOU THINK THIS ONE IS FINE. YOU LIKE HOW X AND Y MAP TO LATITUDE AND LONGITUDE. THE OTHER PROJECTIONS OVERCOMPLICATE THINGS. YOU WANT ME TO STOP ASKING ABOUT MAPS SO YOU CAN ENJOY DINNER.

WATERMAN BUTTERFLY



REALLY? YOU KNOW THE WATERMAN? HAVE YOU SEEN THE 1909 CHILL MAP IT'S BASED—... YOU HAVE A FRAMED REPRODUCTION AT HOME?! WHOA... LISTEN, FORGET THESE QUESTIONS. ARE YOU DOING ANYTHING TONIGHT?

GALL-PETERS



I HATE YOU.

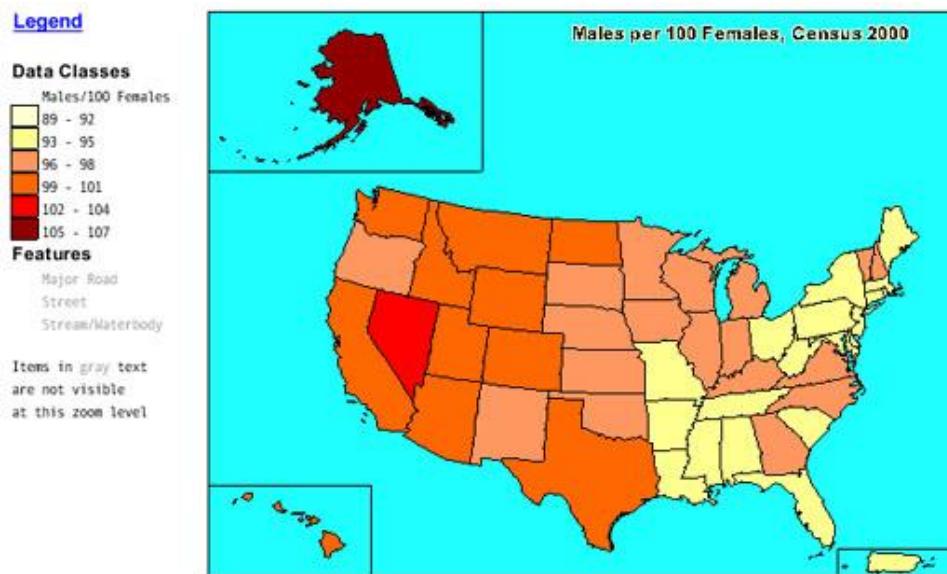
Review: Choropleth and Isopleth maps

Choropleth maps: shades/colors in predefined areas based on properties of a variable

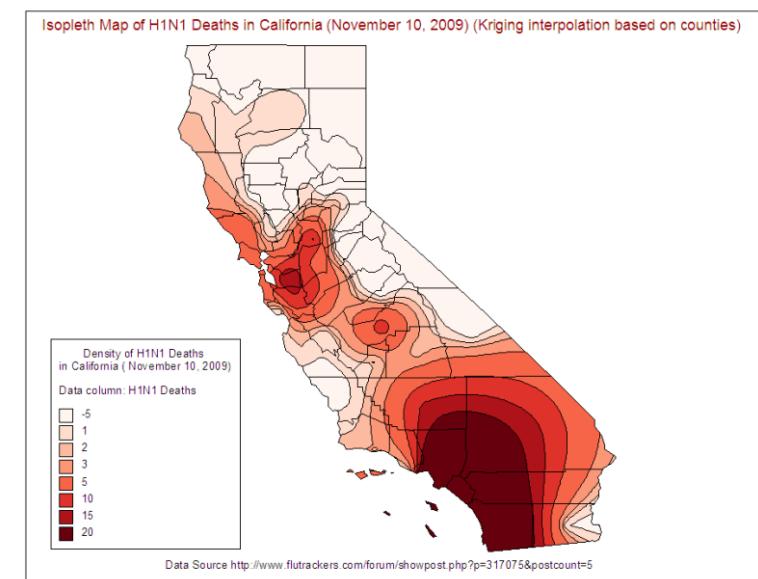
- We can then use the `gpd.plot(column =)` method to create choropleth maps

Isopleth maps: creates regions based on constant values

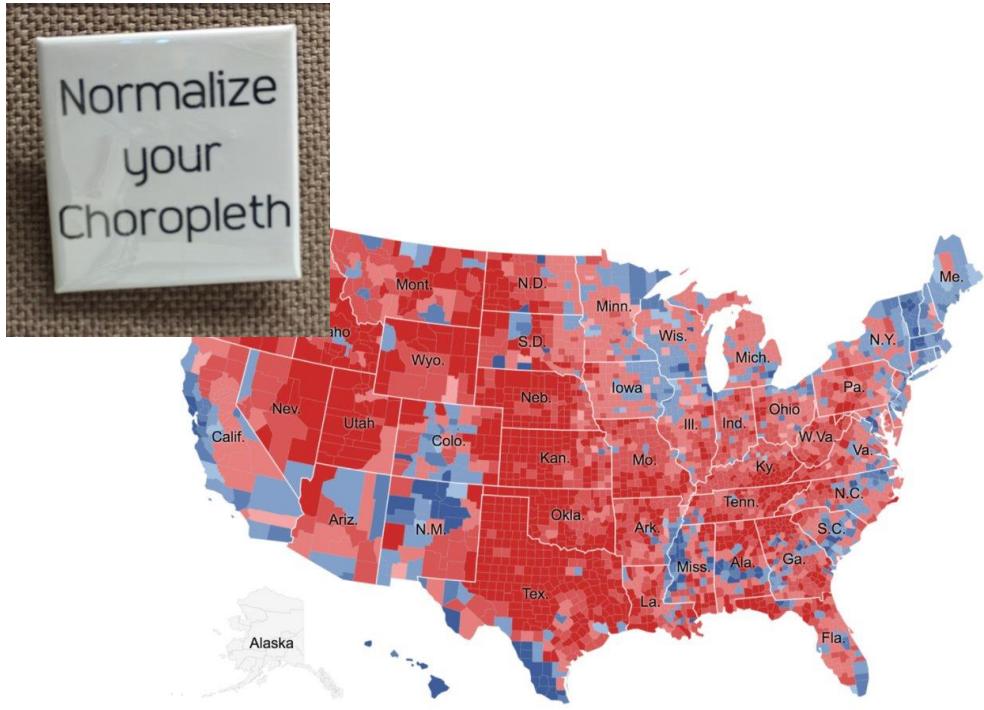
Choropleth map



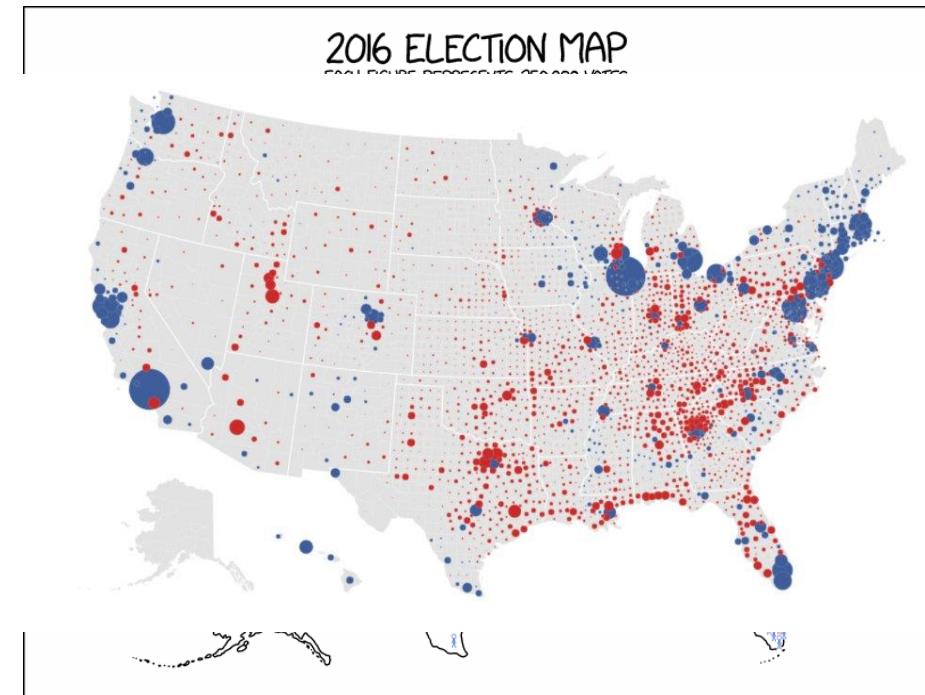
Isopleth map



Choropleth maps can be misleading



Looks like most of the country
voted republican



Let's try a quick warm-up exercise in Jupyter!

Loops

For loops

For loops repeat a process many times, iterating over a sequence of items

- Often we are iterating over an array of sequential numbers

```
animals = ["cat", "dog", "bat"]
```

```
for creature in animals:  
    print(creature)
```

```
for i in np.arange(4):  
    print(i**2)
```

Review: ranges

A range gives us a sequence of consecutive numbers

An sequence of increasing integers from 0 up to *end* - 1
`range(end)`

An sequence of increasing integers from *start* up to *end* - 1
`range(start, end)`

A sequence with step between consecutive values
`range(start, end, step)`

The range always includes start but excludes end



Let's explore this in Jupyter!

Enumerate and zip

We can use the `enumerate()` function to both items in a list, and sequential integers:

```
animals = ["cat", "dog", "bat"]
for i, creature in enumerate(animals):
    print(i, creature)
```

cat -> feline, dog -> canine, bat -> ?



ChatGPT can make mistakes. Check important info.

We can use the `zip()` function to get items for two lists:

```
animal_order = ["feline", "canine", "chiropteran"]
for curr_order, curr_animal in zip(animal_order, animals):
    print(curr_order, curr_animal)
```

Let's explore this in Jupyter!

Conditional statements

Review: comparisons

We can use mathematical operators to compare numbers and strings

- Results return Boolean values **True** and **False**

Comparison	Operator	True example	False Example
Less than	<	$2 < 3$	$2 < 2$
Greater than	>	$3 > 2$	$3 > 3$
Less than or equal	\leq	$2 \leq 2$	$3 \leq 2$
Greater or equal	\geq	$3 \geq 3$	$2 \geq 3$
Equal	\equiv	$3 \equiv 3$	$3 \equiv 2$
Not equal	\neq	$3 \neq 2$	$2 \neq 2$

We can also make comparisons across elements in an array

Conditional statements

Conditional statements control the sequence of computations that are performed in a program

We use the keyword **if** to begin a conditional statement to only execute lines of code if a particular condition is met.

We can use **elif** to test additional conditions

We can use an **else** statement to run code if none of the if or elif conditions have been met.

```
num = 5
if num == 1:
    print("Monday")
elif num == 2:
    print("Tuesday")
elif num == 3:
    print("Wednesday")
elif num == 4:
    print("Thursday")
elif num == 5:
    print("Friday")
elif num == 6:
    print("Saturday")
elif num == 7:
    print("Sunday")
else:
    print("Invalid input")
```

Let's explore this in Jupyter!

Defining functions

Writing functions

We have already used many functions that are built into Python or are imported from different modules/packages.

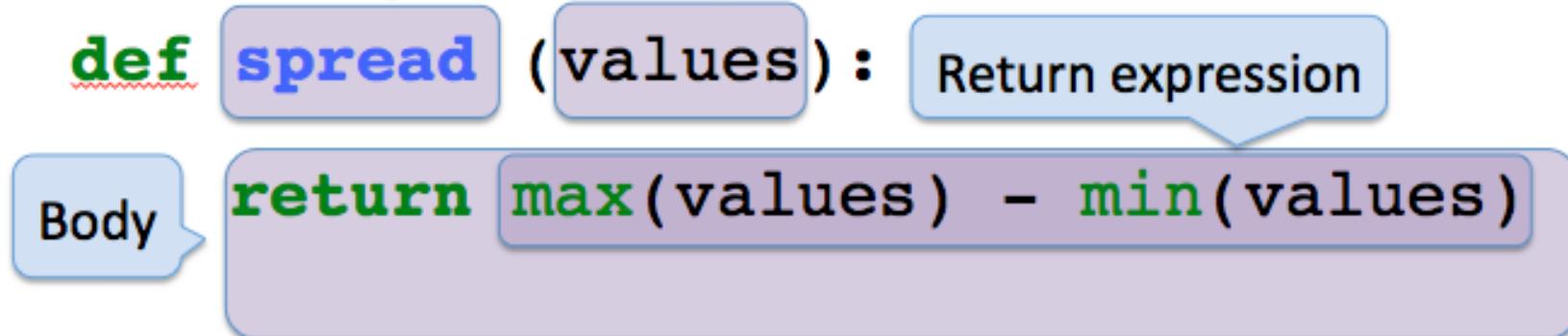
Examples...???

- `sum()`
- `statistics.mean()`
- `np.diff()`
- etc.

Let's now write our own functions!

Def statements

User-defined functions give names to blocks of code



Let's explore this in Jupyter!

Practice: simulating flipping coins

Simulating flipping a coin

Let's practice writing functions by writing a function that can simulate flipping coins, where each coin has π probability of being heads

- Where π is a number between 0 and 1; e.g., $\pi = 0.5$ is a fair coin

We can do this using the following procedure:

1. Generate a random number between 0 and 1

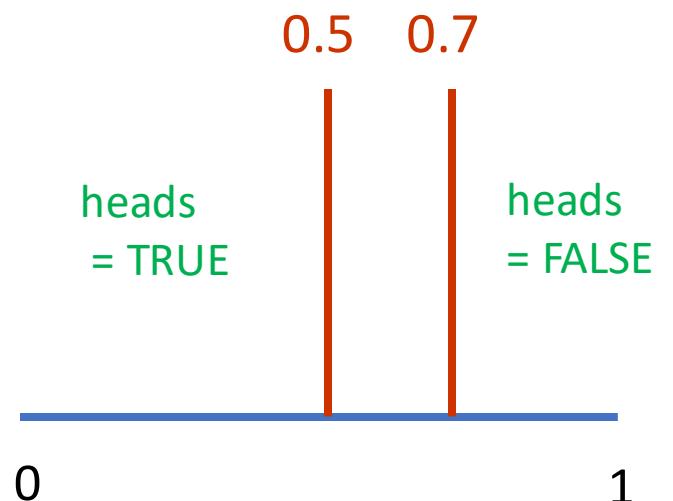
```
rand_num = np.random.rand(1)
```

2. Simulate a fair coin (.5) by mark values less than .5 as heads (**True**)

```
heads = rand_num <= .5
```

3. We can simulate a biased coin that will come up with heads 70% of the time ($\pi = 0.7$) using:

```
rand_num = np.random.rand(1)  
heads = rand_num <= .7
```



Simulating n random coin flips

We can simulate the number of heads we would get flipping a coin n times using:

1. Generate n random numbers uniformly distributed between 0 and 1

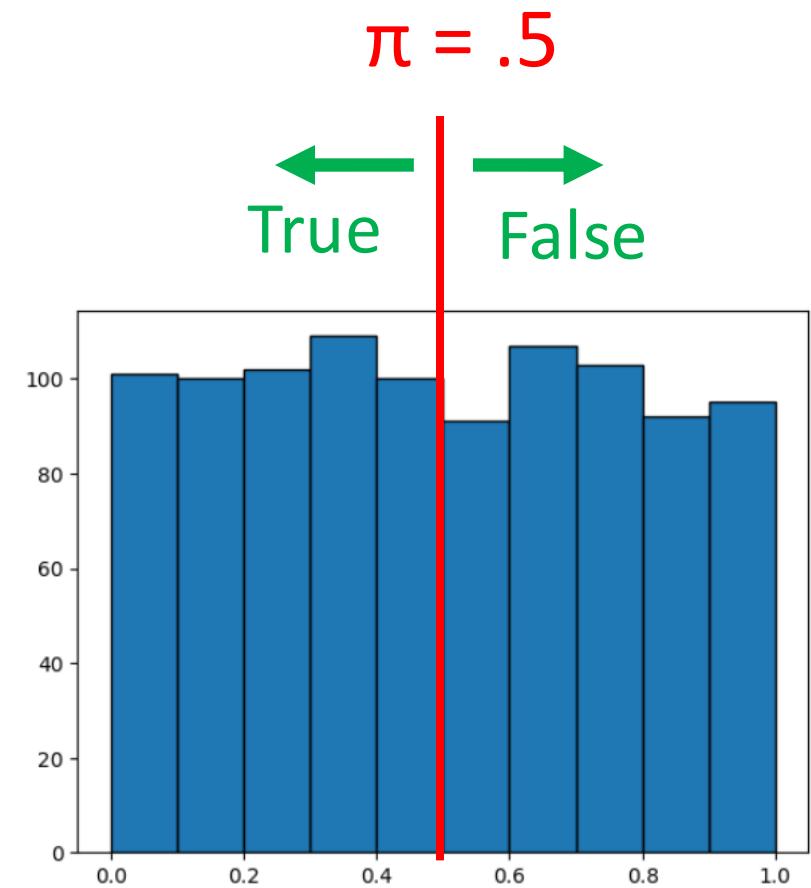
```
rand_nums = np.random.rand(n)
```

2. Mark points less than π as being **True**, and greater π than as being **False**

```
rand_binary = rand_nums <= prob_value
```

3. Sum the number of heads (**True's**) we get

```
num_heads = np.sum(rand_binary)
```



Let's explore this in Jupyter!

YData: Introduction to Data Science



Class 17: Introduction to Statistical Inference

Overview

Review of for loops and functions

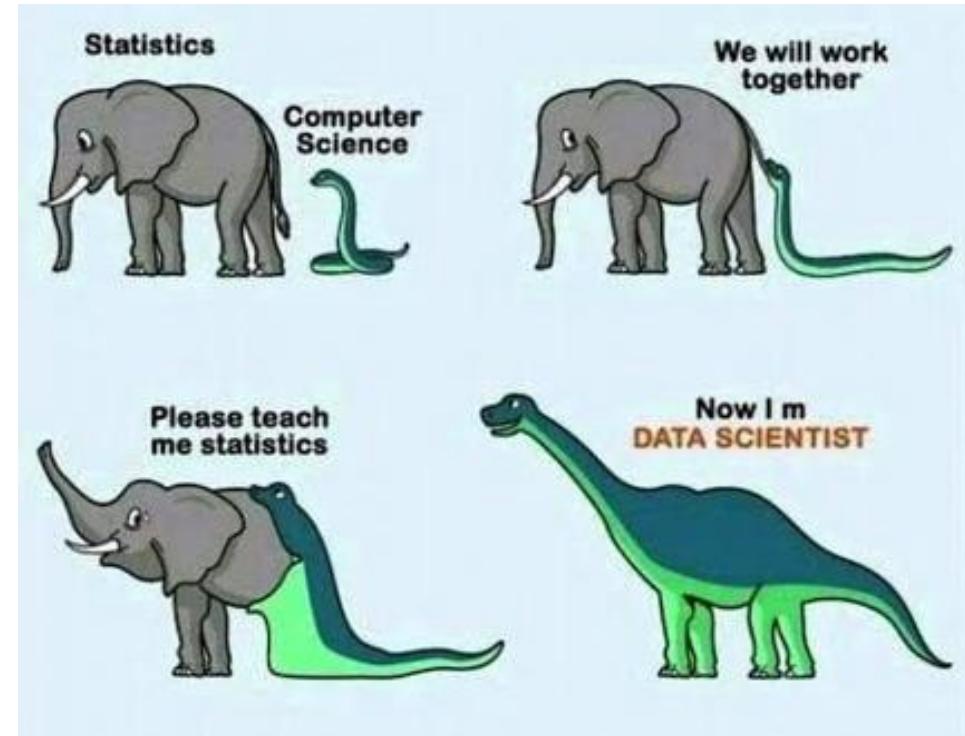
Central concepts in Statistical Inference

- Parameters and statistics

Sampling distributions

Hypothesis tests

- If there is time: Hypothesis tests for a single proportion



Reminder: keep working on your class project

A polished draft of the project is due on November 10th

Also, homework 7 is due on Sunday November 3rd

- I recommend finishing it early and then starting on your project by coming up with a topic and getting the relevant data.



Quick review of for loops and
conditional statements

Review: for loops

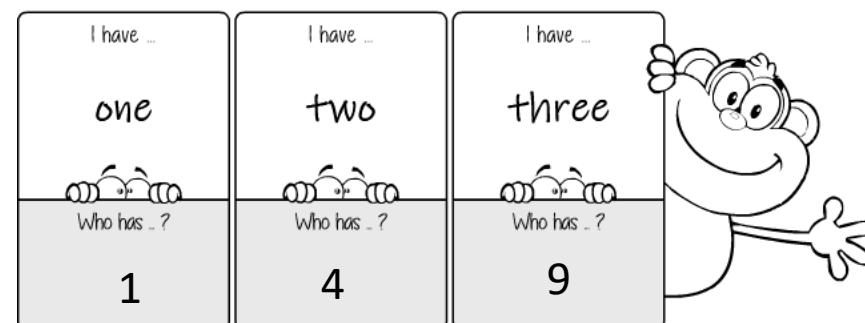
For loops repeat a process many times, iterating over a sequence of items

- Often we are iterating over an array of sequential numbers

```
animals = ["cat", "dog", "bat"]
```

```
for creature in animals:  
    print(creature)
```

```
for i in range(10):  
    print(i**2)
```



Review: conditional statements

Conditional statements control the sequence of computations that are performed in a program

We use the keyword **if** to begin a conditional statement to only execute lines of code if a particular condition is met.

We can use **elif** to test additional conditions

We can use an **else** statement to run code if none of the if or elif conditions have been met.

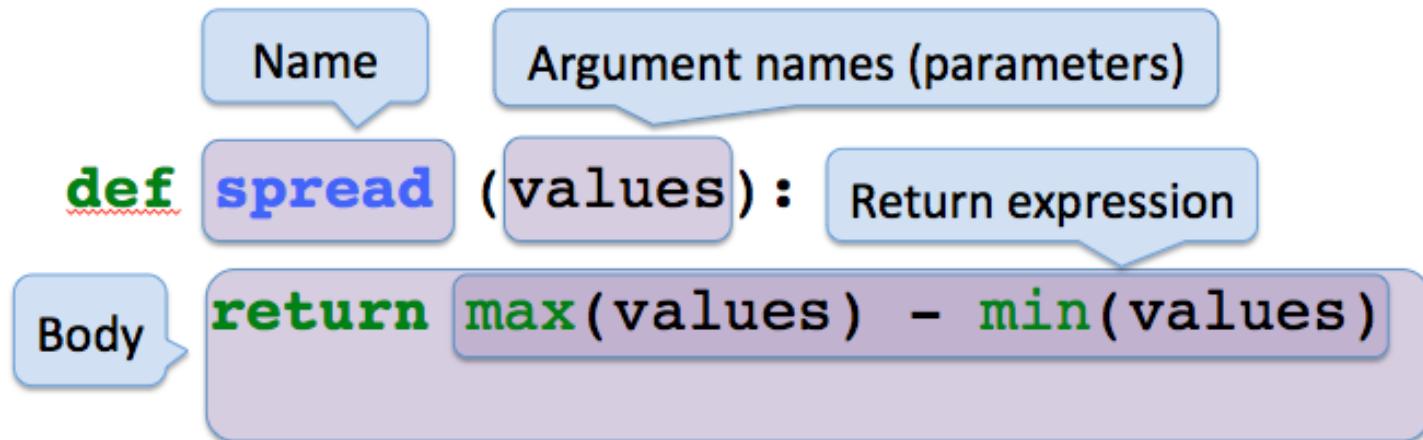
```
num = 5
if num == 1:
    print("Monday")
elif num == 2:
    print("Tuesday")
elif num == 3:
    print("Wednesday")
elif num == 4:
    print("Thursday")
elif num == 5:
    print("Friday")
elif num == 6:
    print("Saturday")
elif num == 7:
    print("Sunday")
else:
    print("Invalid input")
```

Let's do some warm up exercises in Jupyter!

Review of writing functions

Review of writing your own functions

User-defined functions give names to blocks of code



Functions can return tuples which allow us to return multiple names

```
val1, val2 = my_function()
```

Simulating flipping a coin

Let's practice writing functions by writing a function that can simulate flipping coins, where each coin has π probability of being heads

- Where π is a number between 0 and 1; e.g., $\pi = 0.5$ is a fair coin

We can do this using the following procedure:

1. Generate a random number between 0 and 1

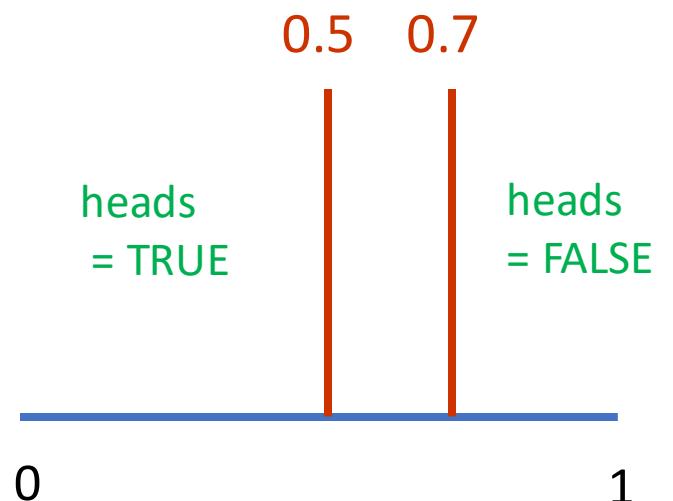
```
rand_num = np.random.rand(1)
```

2. Simulate a fair coin (.5) by mark values less than .5 as heads (**True**)

```
heads = rand_num <= .5
```

3. We can simulate a biased coin that will come up with heads 70% of the time ($\pi = 0.7$) using:

```
rand_num = np.random.rand(1)  
heads = rand_num <= .7
```



Simulating n random coin flips

We can simulate the number of heads we would get flipping a coin n times using:

1. Generate n random numbers uniformly distributed between 0 and 1

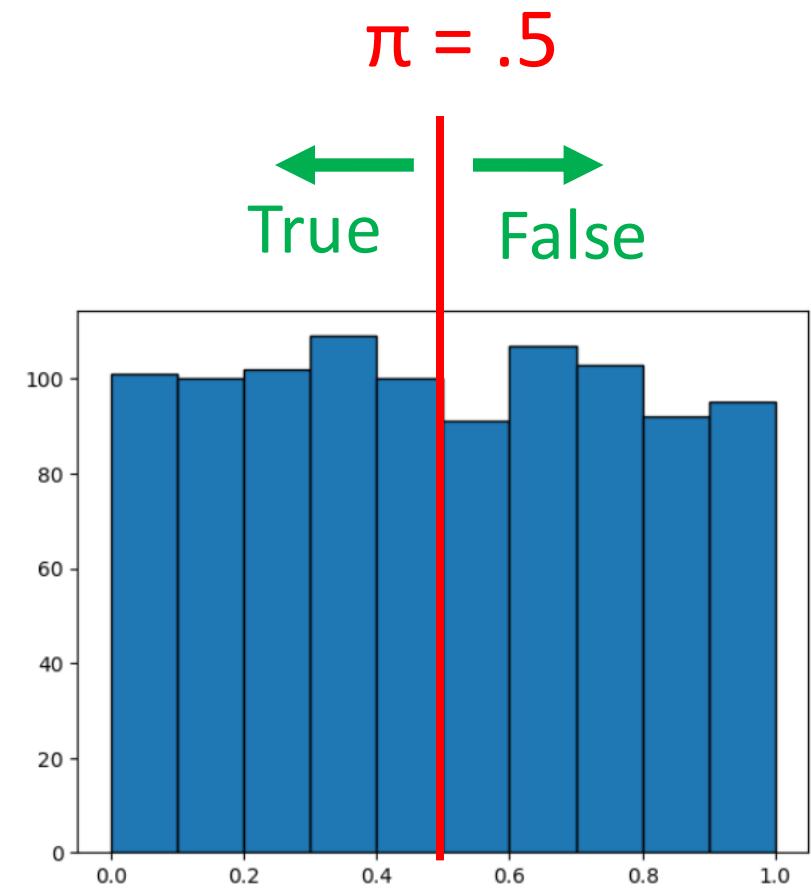
```
rand_nums = np.random.rand(n)
```

2. Mark points less than π as being **True**, and greater π than as being **False**

```
rand_binary = rand_nums <= prob_value
```

3. Sum the number of heads (**True's**) we get

```
num_heads = np.sum(rand_binary)
```



Let's explore this in Jupyter!

Statistical Inference

Statistical Inference

In **statistical inference** we use a sample of data to make claims about a larger population (or process)

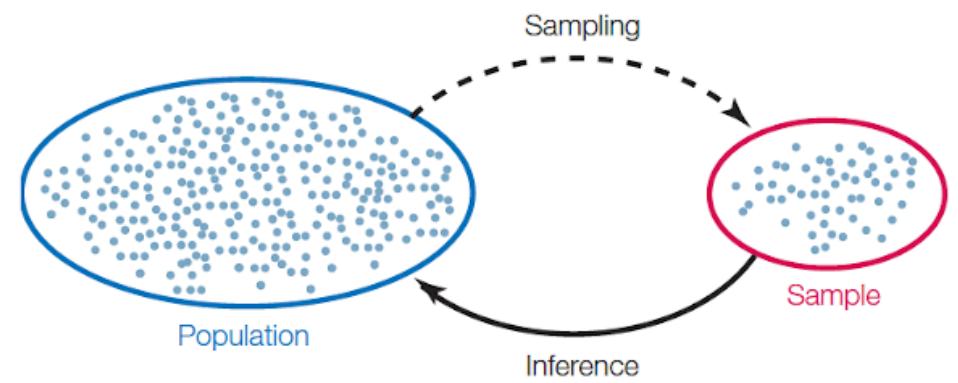
Examples:

Voting data

- **Population:** everyone who will vote
- **Sample:** survey of 1,000 randomly selected voters

Bechdel data:

- **Population:** All movies with budgets > \$10,000,000
- **Sample:** 1794 movies randomly selected



Population: all individuals/objects of interest

Sample: A subset of the population

Terminology

A **parameter** is number associated with the population

- e.g., population proportion π
- e.g., the proportion of voters who will vote for Trump

A **statistic** is number calculated from the sample

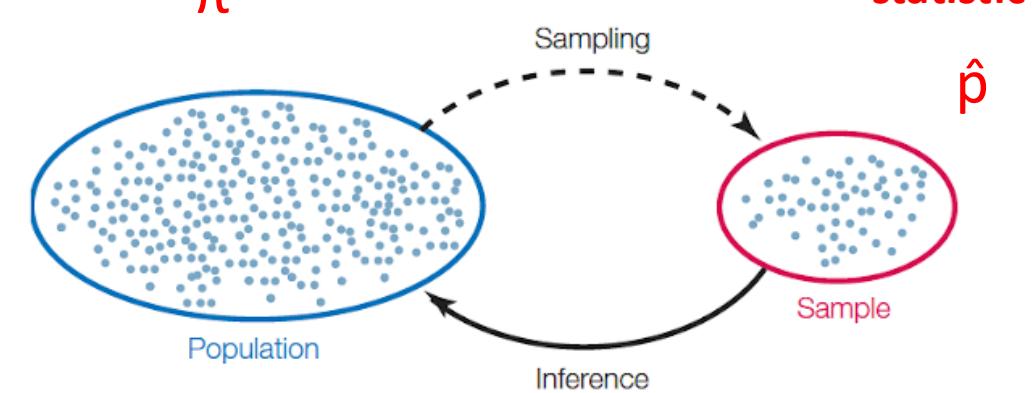
- e.g., sample proportion \hat{p}
- e.g., proportion of 1,000 people in our sample

A statistic can be used as an estimate of a parameter

- A parameter is a single fixed value
- Statistics tend to vary from sample to sample

parameter

π



Example:

- Using the proportion of 1,000 voters (\hat{p}_{Trump}) to estimate the proportion of all voters who will vote for Trump (π_{Trump})

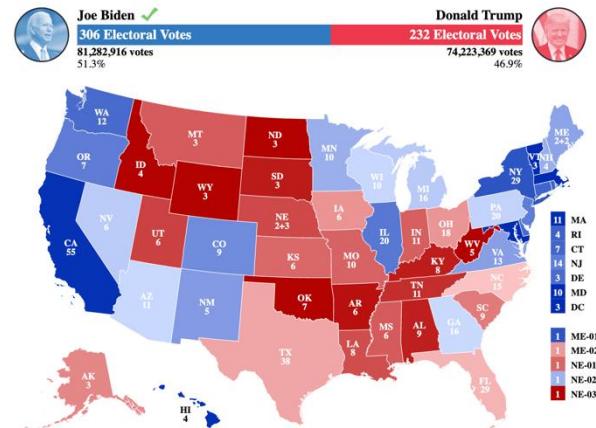
Examples of parameters and statistics

	Sample statistic	Population parameter	Bechdel example
Mean	\bar{x}	μ	<code>statistics.mean(domgross_2013)</code> $\bar{x} = \$95,174,783$
Proportion	\hat{p}	π	<code>bechdel.count("PASS")/len(bechdel)</code> $\hat{p} = 0.45$
Correlation	r	ρ	<code>statistics.correlation(budget_2013, domgross_2013)</code> $r = 0.46$

Sampling

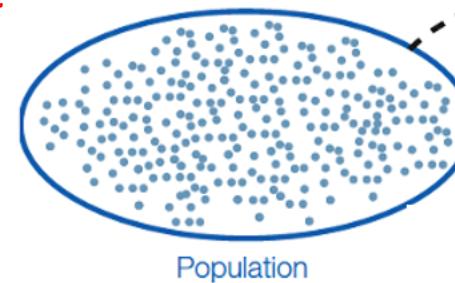
Simple random sample: each member in the population is equally likely to be in the sample

- Allows for generalizations to the population



parameter

π



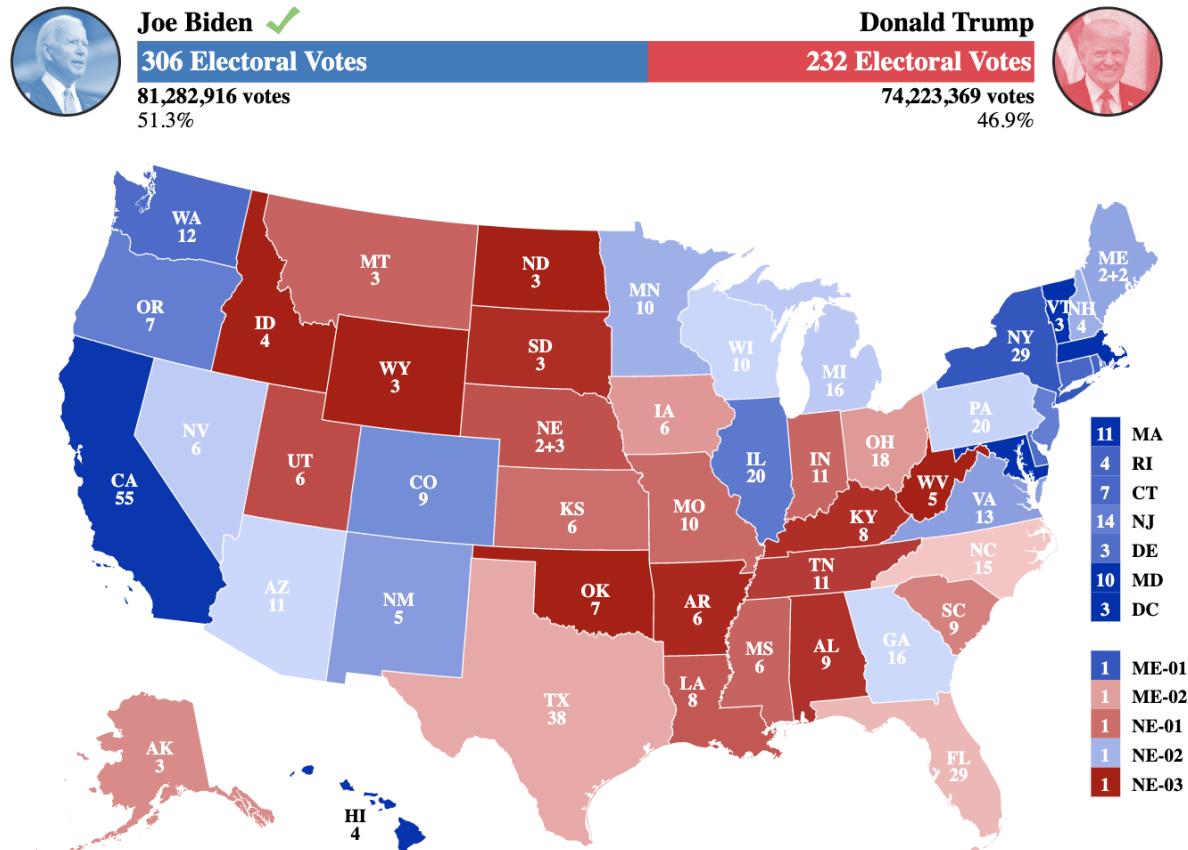
statistic

\hat{p}

Polls of 1,000 voters: \hat{p}_{Trump}

Vote on election day: π_{Trump}

Example: The 2020 US Presidential Election



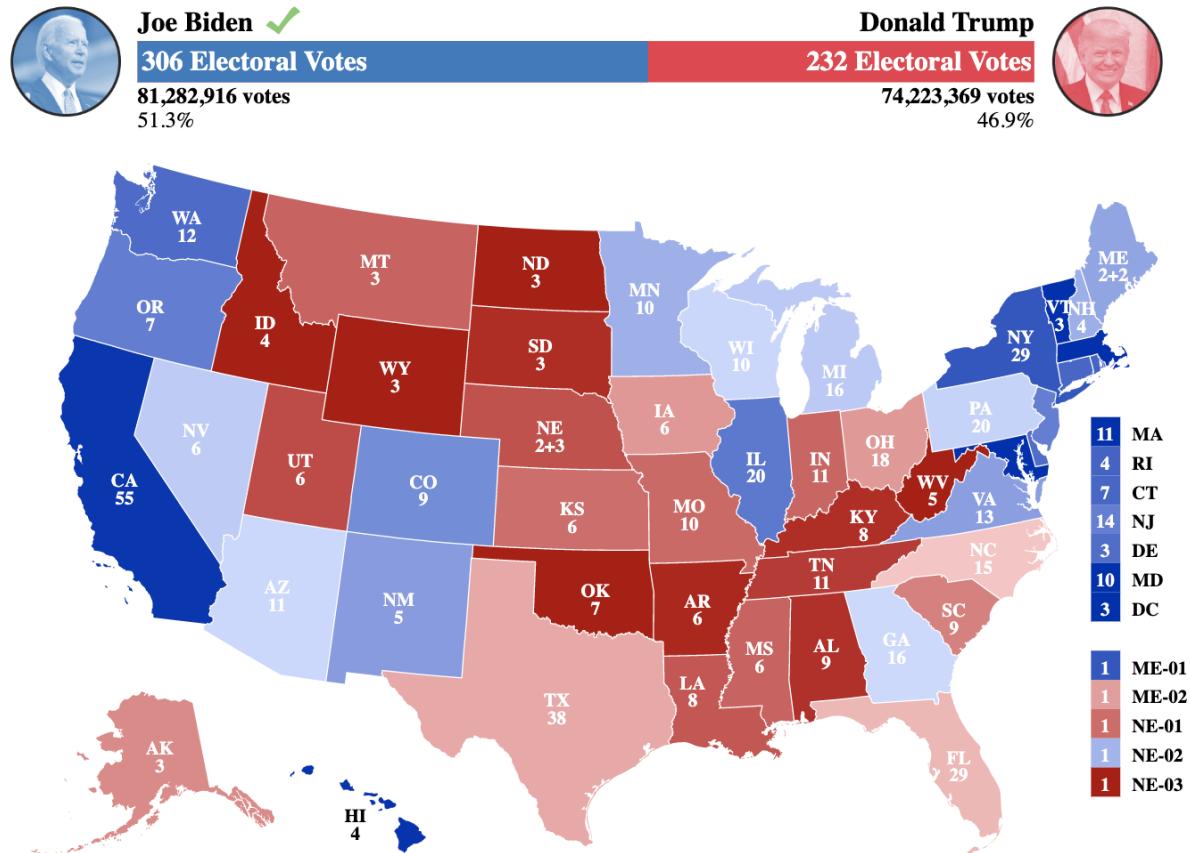
According to The Cook Political Report, the voting outcome in Georgia in 2020 was

- Trump = 2,461,854
- Biden = 2,473,633

We can denote the proportion of the vote that Trump got using π_{Trump}

- Q: what is the value of π_{Biden} ?

Example: The 2020 US Presidential Election



If 1,000 voters were randomly sampled, we could denote the proportion in the sample that voted for Biden using: \hat{p}_{Biden}

Would we expect \hat{p}_{Biden} to be equal to π_{Biden} ?

If we repeated the process of sampling another 1,000 random voters, would we expect to get the same \hat{p}_{Biden} ?

Let's explore this in Jupyter!

Sampling distributions

Probability distribution of a statistic

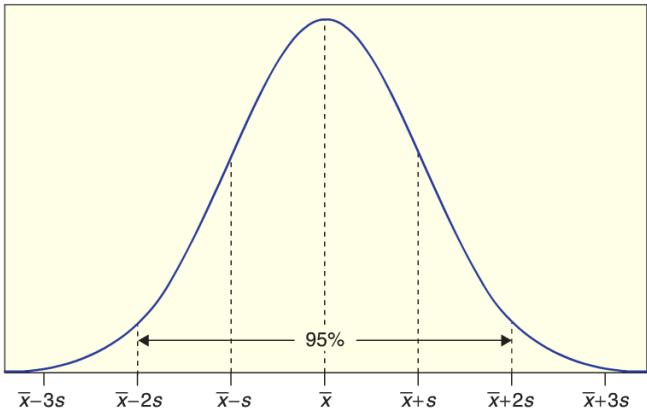
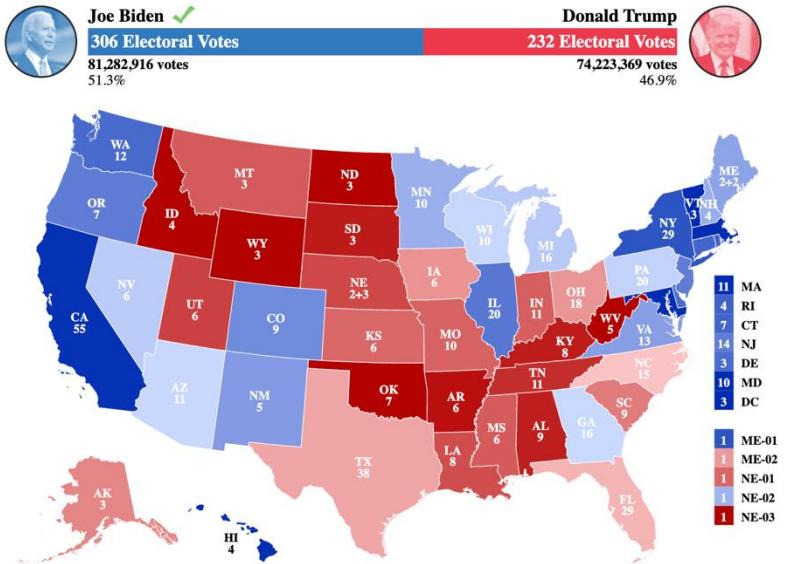
Values of a statistic vary because random samples vary

A **sampling distribution** is a probability distribution of *statistics*

- All possible values of the statistic and all the corresponding probabilities
- We can approximate a sampling distribution by a simulated statistics

π_{Trump}

$n = 1,000$



Sampling distribution!



\hat{p}_{Trump}



\hat{p}_{Trump}



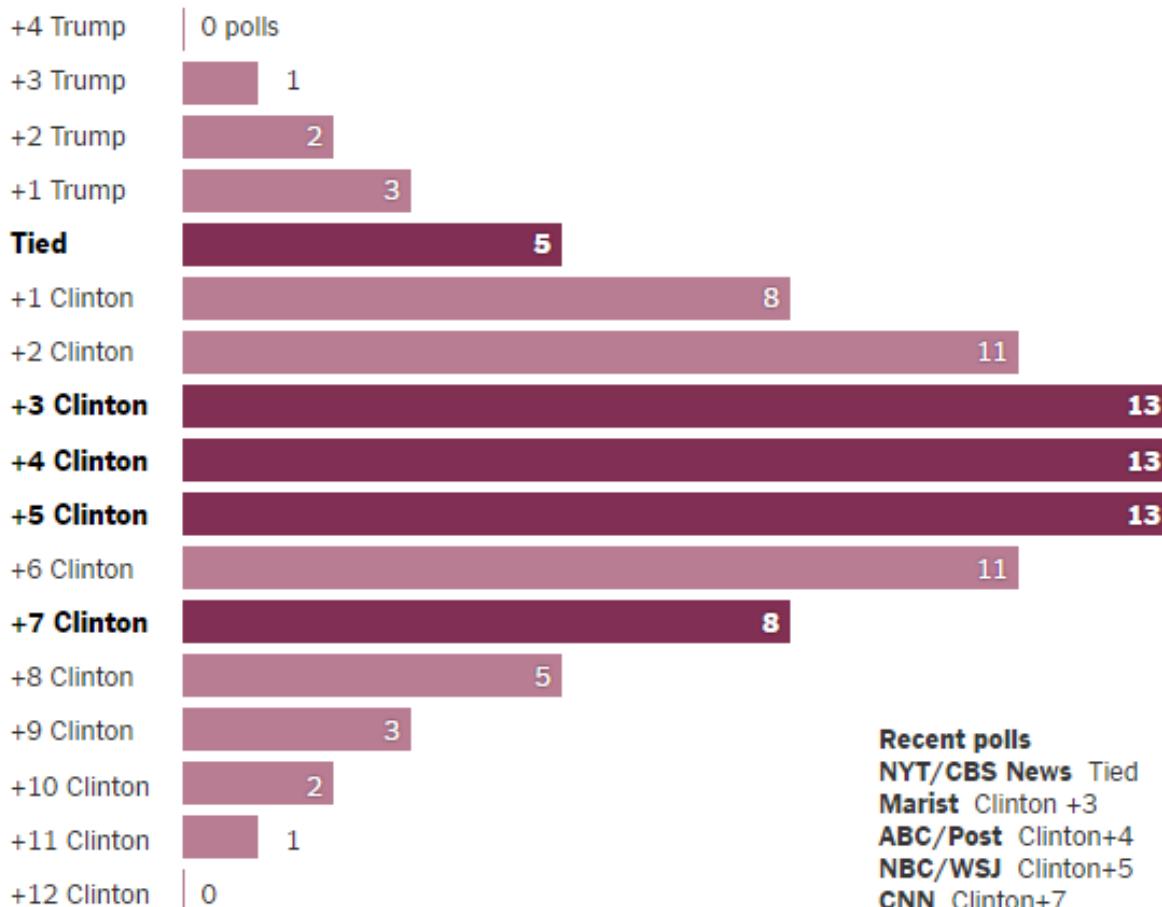
\hat{p}_{Trump}

Confused by Contradictory Polls? Take a Step Back

Noisy Polls Are to Be Expected

If Hillary Clinton were up by a modest margin, there would be plenty of polls showing a very close race — or even a Trump lead.

A simulation of 100 surveys, if Mrs. Clinton were really up 4 points nationally.



What is this called?



What parameter are they trying to estimate?

Let's explore this in Jupyter!

Simulating flipping a coin

We can simulate flipping a fair coin using the following procedure

1. Generated a random number between 0 and 1

```
rand_num = np.random.rand(1)
```

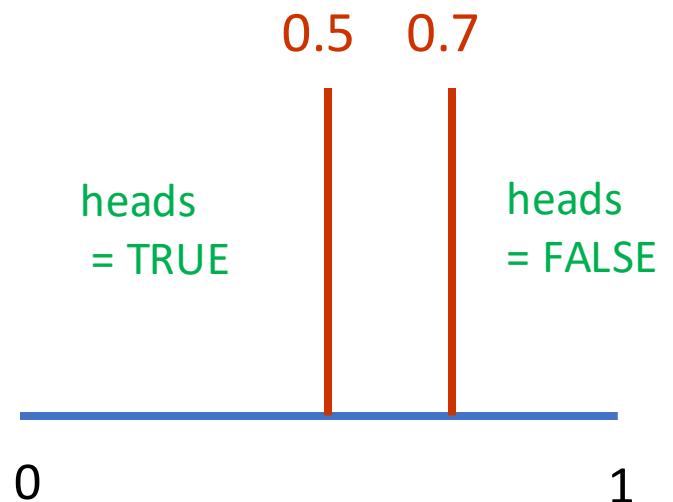
2. We mark values less than .5 has heads (**True**)

```
heads = rand_num <= .5
```

We can simulate a biased coin that will come up with heads 70% of the time using

```
rand_num = np.random.rand(1)
```

```
heads = rand_num <= .7
```



Simulating a random proportion (\hat{p})

We can simulate a random proportions \hat{p} (from a sample of size n) consistent with a population proportion π by:

1. Generated n random numbers uniformly distributed between 0 and 1

```
rand_nums = np.random.rand(1000)
```

2. Marking points less than π as being **True**, and greater π than as being **False**

```
rand_binary = rand_nums <= pi_value
```

3. Calculating the proportion of points to get a \hat{p}

```
rand_phat = np.mean(rand_binary)
```



Let's explore this in Jupyter!

Hypothesis tests

A quick note on probability

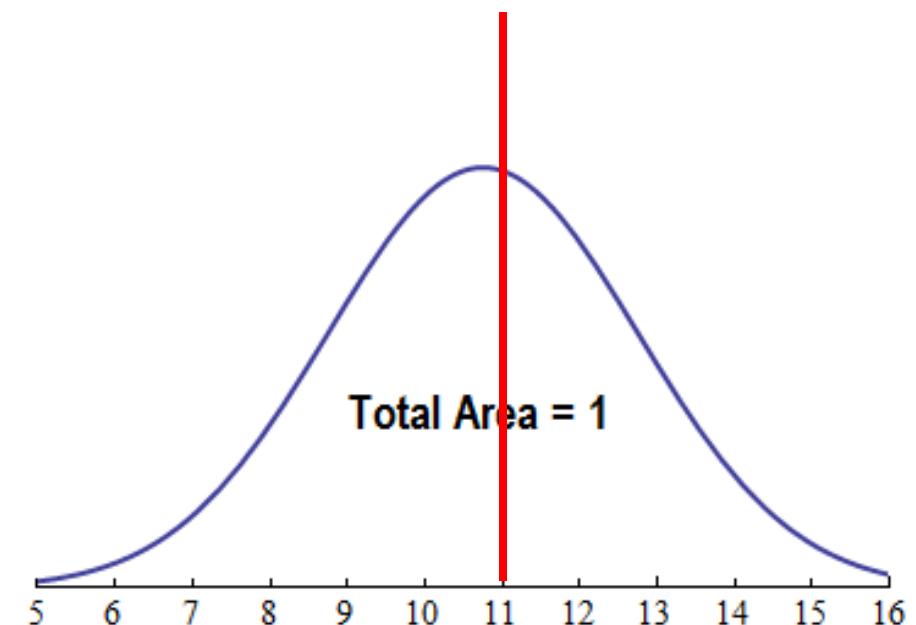
Probability is a way of measuring the likelihood that an event will occur

Probability models assigns a number between 0 and 1 to the outcome of an event (outcome) occurring

We can use a probability model to calculate the probability of an event

For example:

- $P(X < 11) = 0.55$
- $P(X > 20) = 0$



Statistical tests (hypothesis test)

A **statistical test** uses data from a sample to assess a claim about a population (parameter)

Example 1: The average body temperature of humans is 98.6°

How can we write this using symbols?

- $\mu = 98.6$

Statistical tests (hypothesis test)

A **statistical test** uses data from a sample to assess a claim about a population (parameter)

Example 2: A higher proportion of voters will vote for Trump compared to Harris

How can we write this using symbols?

- $\pi_{\text{Trump}} > \pi_{\text{Harris}}$ or $\pi_{\text{Trump}} - \pi_{\text{Harris}} > 0$

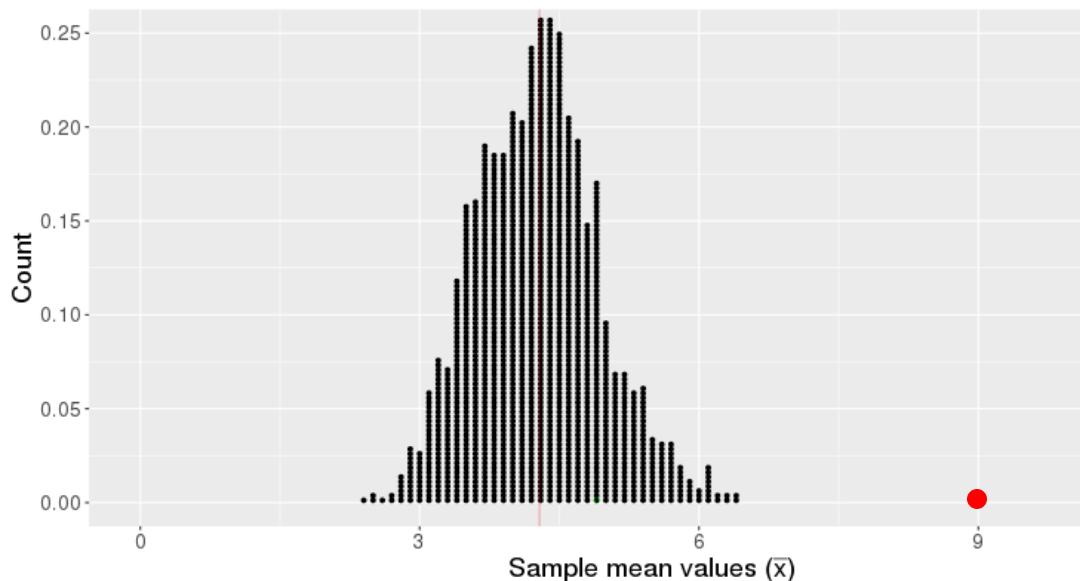
Basic hypothesis test logic

We start with a claim about a population parameter

- E.g., $\mu = 4$



This claim implies we should get a certain distribution of statistics



If our observed statistic is highly unlikely, we reject the claim

Example claims (hypotheses)

Let's see if we can write the following claims (hypotheses) using symbols

Claim: 88% of Yale students graduate within four years

- $H: \pi = 0.88$

Claim: The average age of a Yale undergraduate is 20

- $H: \mu = 20$

Claim: 70.7% of Yale classrooms have fewer than 20 students in attendance

- $H: \pi = 0.707$

Testing claims (hypotheses)

Claim: 88% of Yale students graduate within four years

- $H_0: \pi = 0.88$
- To test this claim, we could randomly selected $n = 100$ Yale graduates.
- If we found the proportion that graduated in 4 years is $\hat{p} = .80$, would we believe the claim?

Testing claims (hypotheses)

Claim: The average age of a Yale undergraduate is 20

- $H_0: \mu = 20$
- To test this claim, we could randomly selected $n = 50$ Yale graduates.
- If we found the average age of in our sample of students was $\bar{x} = 20.2$, would we believe the claim?

Motivating example: The Bechdel Test



Question: Do less than 50% of movies pass the Bechdel test?

Questions:

- What is the population/process?
- What is our parameter of interest?
 - What symbol should we use to denote it?
- What is our statistic of interest?
 - What symbol should we use to denote it?

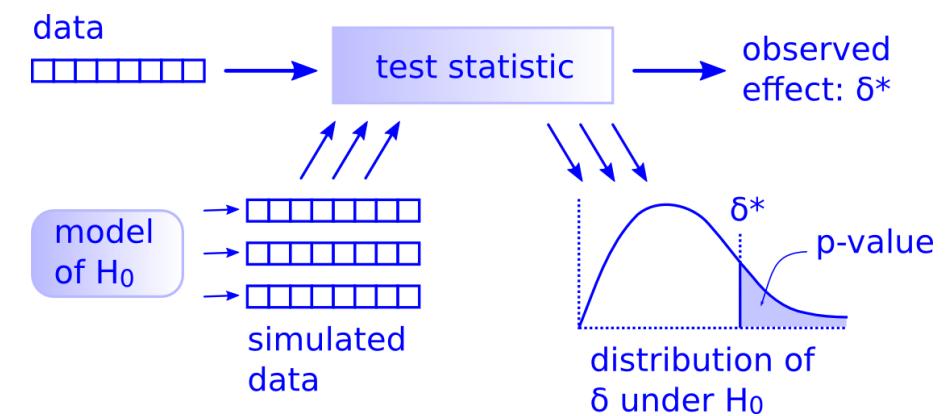
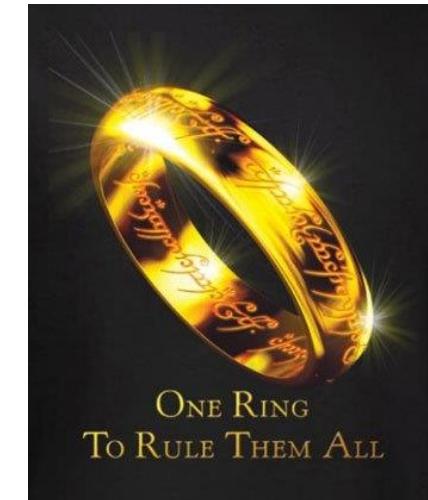
	title	binary
1	Dredd 3D	PASS
2	12 Years a Slave	FAIL
3	2 Guns	FAIL
4	42	FAIL
5	47 Ronin	FAIL
6	A Good Day to Die Hard	FAIL
7	About Time	PASS
8	Admission	PASS
9	After Earth	FAIL
10	American Hustle	PASS
11	August: Osage County	PASS
12	Beautiful Creatures	PASS
13	Blue Jasmine	PASS
14	Captain Phillips	FAIL

Steps needed to run a hypothesis test

To run a hypothesis test, we can use 5 steps:

1. State the null and alternative hypothesis
2. Calculate the observed statistic of interest
3. Create the null distribution
4. Calculate the p-value
5. Make a decision

Let's go through these steps now...



Do less than 50% of movies pass the Bechdel test?

Step 1: state the null and alternative hypotheses

If only 50% of the movies passed the Bechdel test, what would we expect the value of the parameter to be?

$$H_0: \pi = 0.5$$

If fewer than 50% of movies passed the Bechdel test, what would we expect the value of the parameter to be?

$$H_A: \pi < 0.5$$

Observed statistic value

Step 2: calculate the observed statistic

There are 1794 movies in our data set

Of these, 803 passed the Bechdel test

What is our observed statistic value and what symbol should we use to denote this value?

A: $\hat{p} = 803/1794 = 0.448$

Step 3: Create a null distribution

How can we assess whether 803 out of 1794 movies passing the Bechdel test ($\hat{p} = 0.448$) is consistent with what we would expect if 50% (or more) movies passed the Bechdel test?

- i.e., is $\hat{p} = 0.448$ a likely value if $\pi = 0.5$?

If 50% of movies passed the Bechdel test, we can model movies passing the as a fair coin flip:

Heads (True) = passed the Bechdel test

Tails (False) = failed to pass the Bechdel test

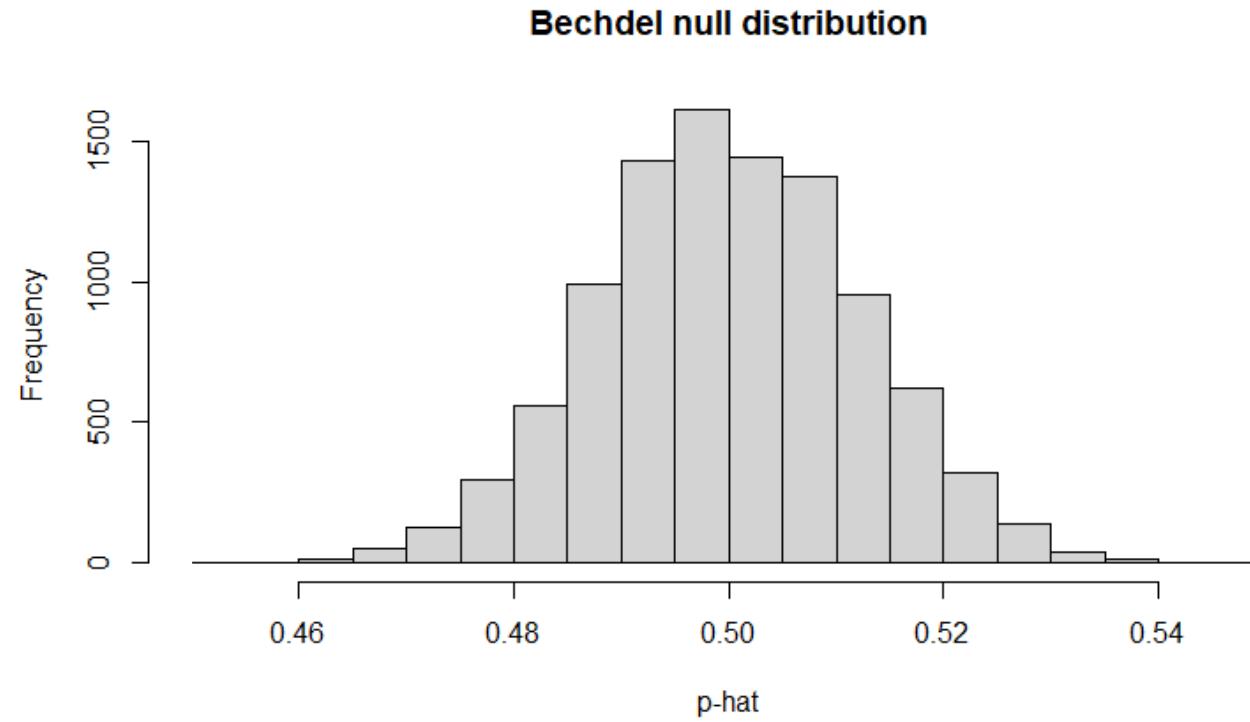
Let's simulate flipping a coin 1794 times and see how many times we get 803 or fewer heads

Chance models

To really be sure, how many repetitions of flipping a coin 1794 times should we do?

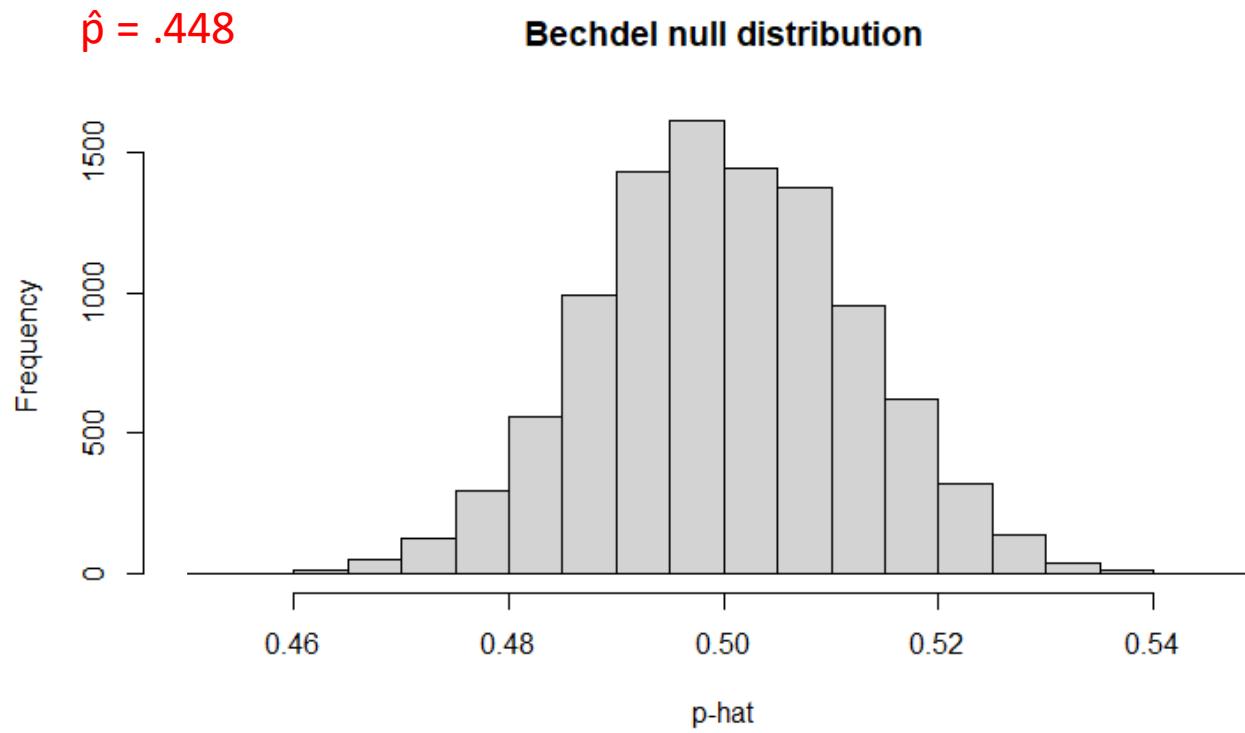
Any ideas how to do this?

Simulating Flipping 1794 coins 10,000 times



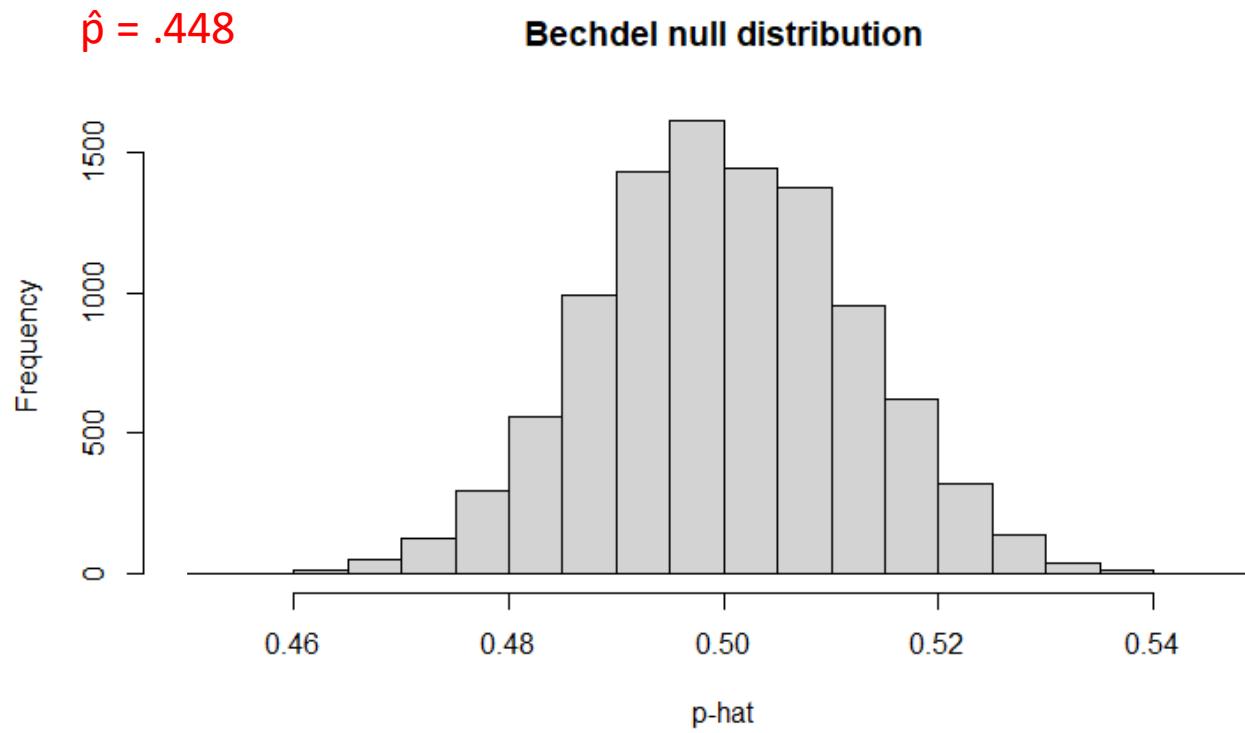
Assuming the null hypothesis is true, the distribution of statistics we get is called the **null distribution**

Step 4: calculate the p-value



Q: Is it likely that 50% of movies pass the Bechdel test?
• i.e., is it likely that $\pi = .5$?

Step 4: calculate the p-value

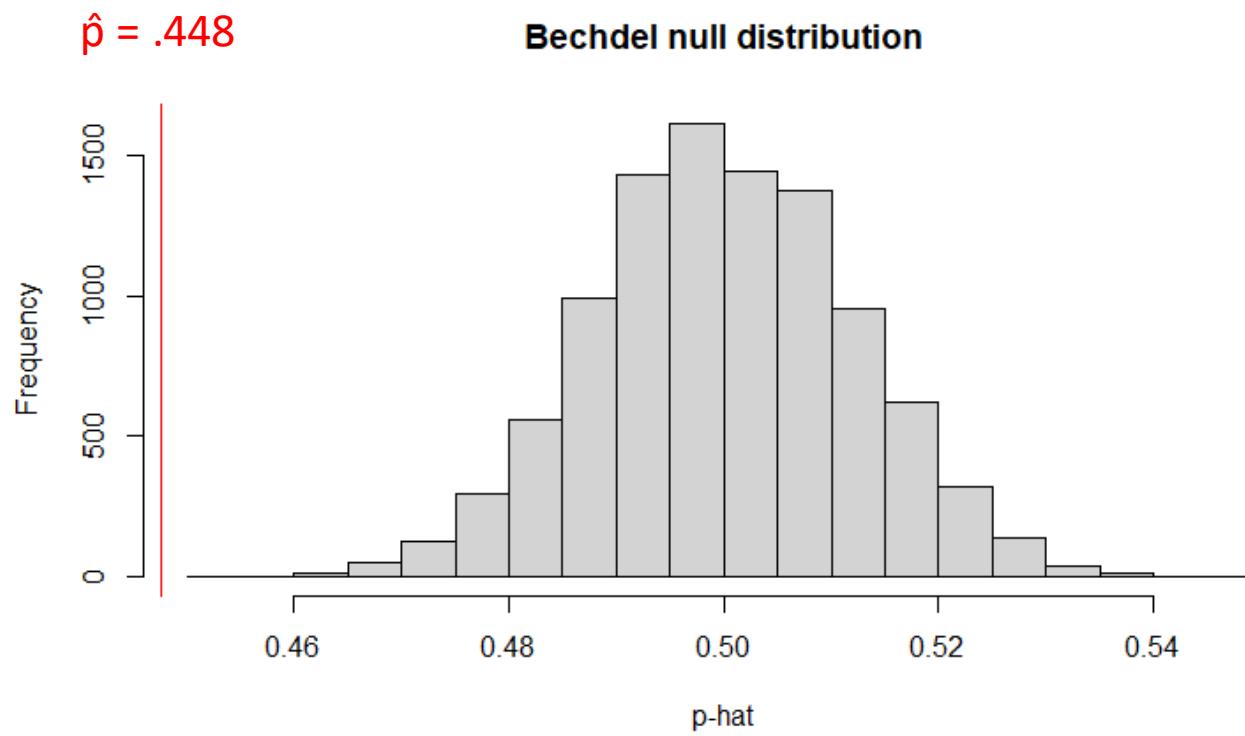


The **p-value** is the probability we will get a statistic as or more extreme than the observed statistic, if the null hypothesis was true

Q: What is the p-value here?

A: the p-value is 0

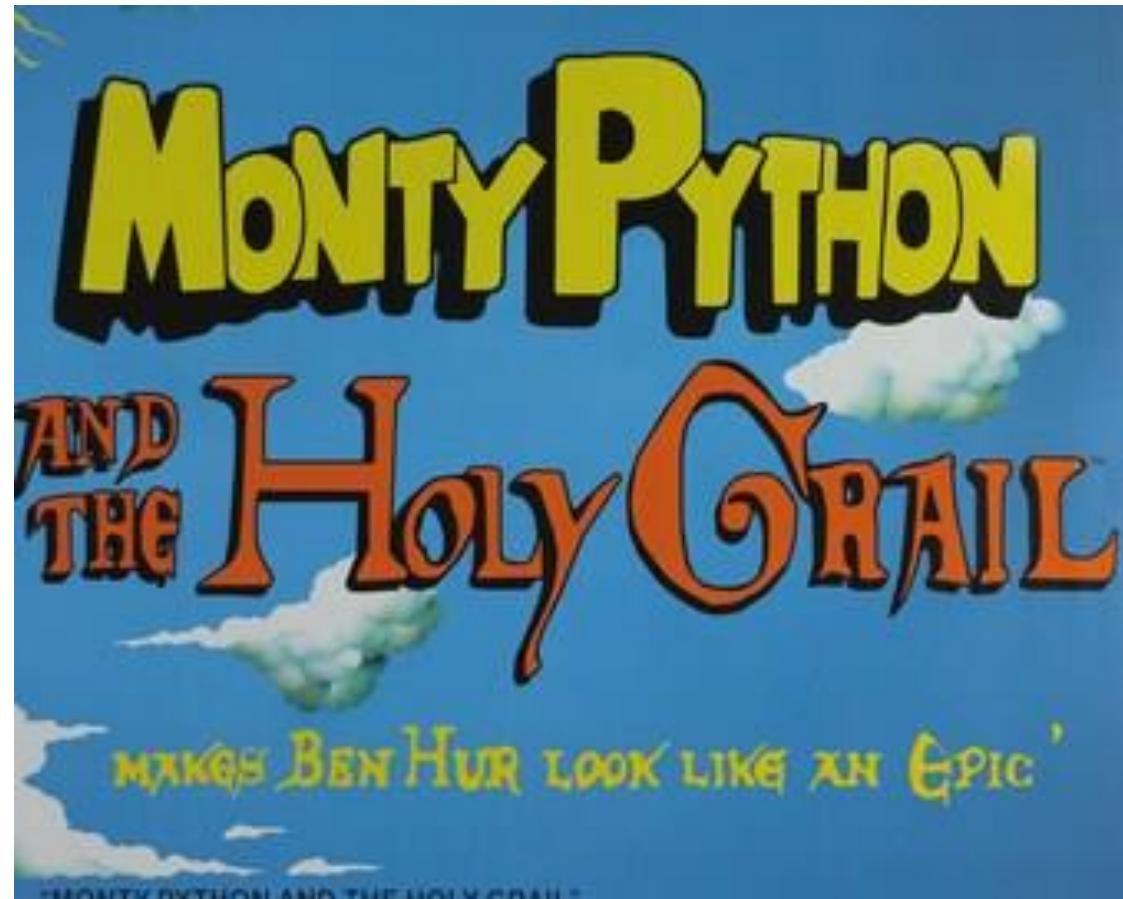
Step 5: Make a decision



If the observed data is very unlikely if the null hypothesis is true, we can reject the null hypothesis

- i.e., if p-value is very small we can reject the null hypothesis

Let's try it in Python



Bechdel (hypothesis) test

1. State the null hypothesis and the alternative hypothesis

- 50% of the movies pass the Bechdel test: $H_0: \pi = 0.5$
- Less than 50% of movies pass the: $H_A: \pi < 0.5$

2. Calculate the observed statistic

- 803 out of 1794 movies passed the Bechdel test

$$\hat{p} = .448$$

3. Create a null distribution that is consistent with the null hypothesis

- i.e., the statistics we expect if 50% of the movies passed the Bechdel test

4. Examine how likely the observed statistic is to come from the null distribution

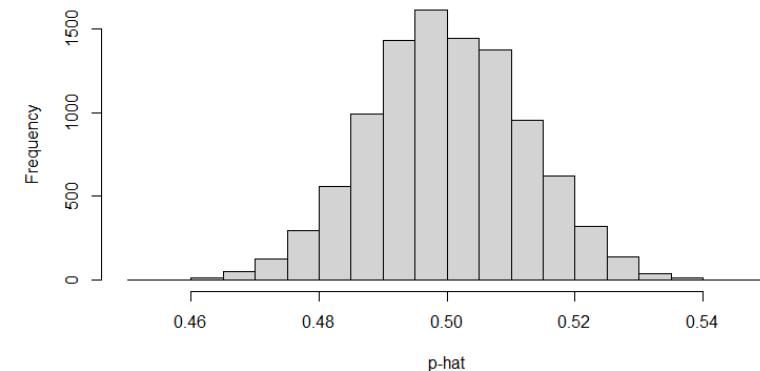
- What is the probability that only 803 of 1794 movies would pass the Bechdel test ($\hat{p} = .448$) if the null hypothesis was true?
- i.e., what is the p-value?

5. Make a judgement

- A small p-value this means that $\pi = .5$ is unlikely, and so it is likely $\pi < .5$
- i.e., we say our results are ‘statistically significant’



Bechdel null distribution





Class 18: Hypothesis test for proportions

Halloween edition...



Overview

Hypothesis tests

- Hypothesis tests for a single proportion

If there is time

- Assessing causal relationships
- Hypothesis tests for two proportions

Reminder: keep working on your class project

Change in plan!

A polished draft of the project is due on **November 17th**

Homework 7 is due on **Sunday November 3rd**

Homework 8 will be due on **Sunday November 10th**



Statistical Inference

Inference

Population: all individuals/objects of interest

- E.g., all voters

A parameter is number associated with the population

- E.g., The proportion of all voters who voted for Biden: π_{Trump}

Sample: A subset of the population

- E.g., 1000 randomly sampled voters

A statistic is number calculated from the sample

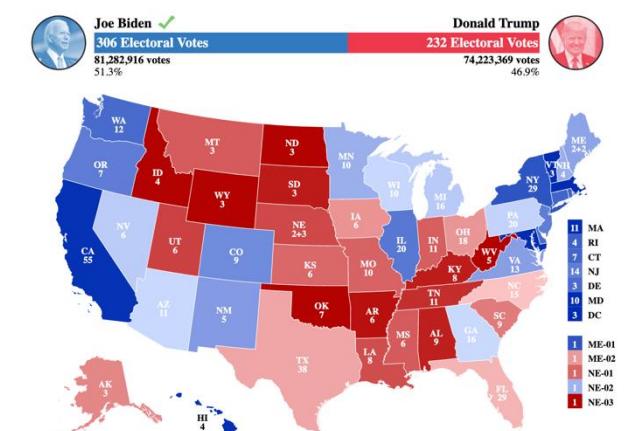
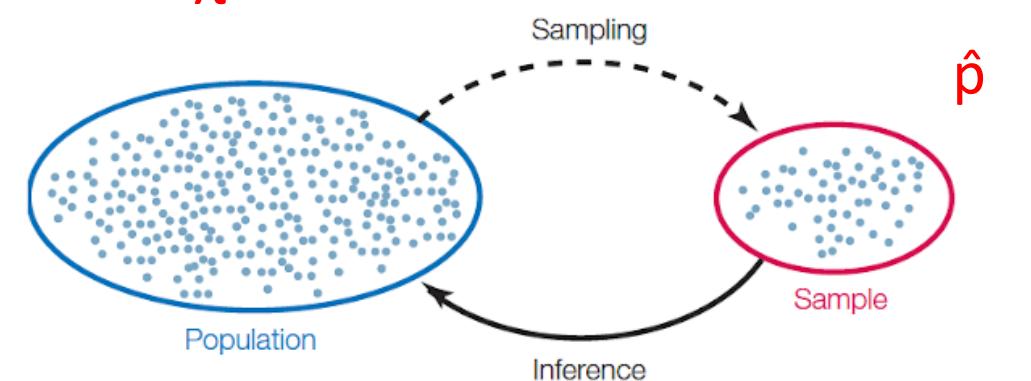
- e.g., The proportion in the sample who voted for Biden: \hat{p}_{Trump}

Statistical Inference: Making conclusions about a population based on data in a sample

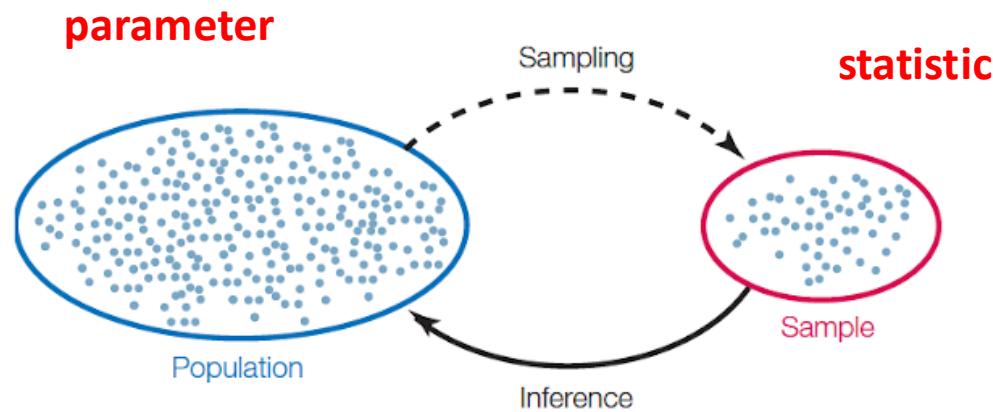
- E.g., using \hat{p}_{Trump} as an estimate of π_{Trump}

parameter

π



Examples of parameters and statistics



	Population Parameter	Sample Statistic
Mean	μ	\bar{x}
Proportion	π	\hat{p}
Standard deviation	σ	s
Correlation	ρ	r

Probability distribution of a statistic

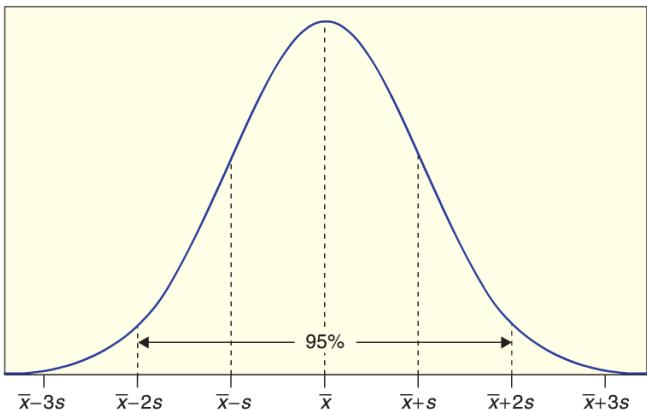
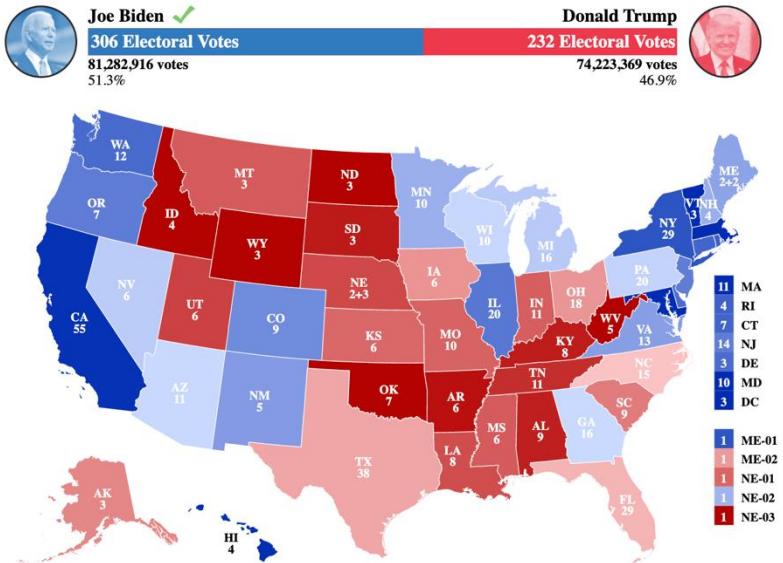
Values of a statistic vary because random samples vary

A **sampling distribution** is a probability distribution of *statistics*

- All possible values of the statistic and all the corresponding probabilities
- We can approximate a sampling distribution by a simulated statistics

π_{Trump}

$n = 1,000$



Sampling distribution!



\hat{p}_{Trump}



\hat{p}_{Trump}



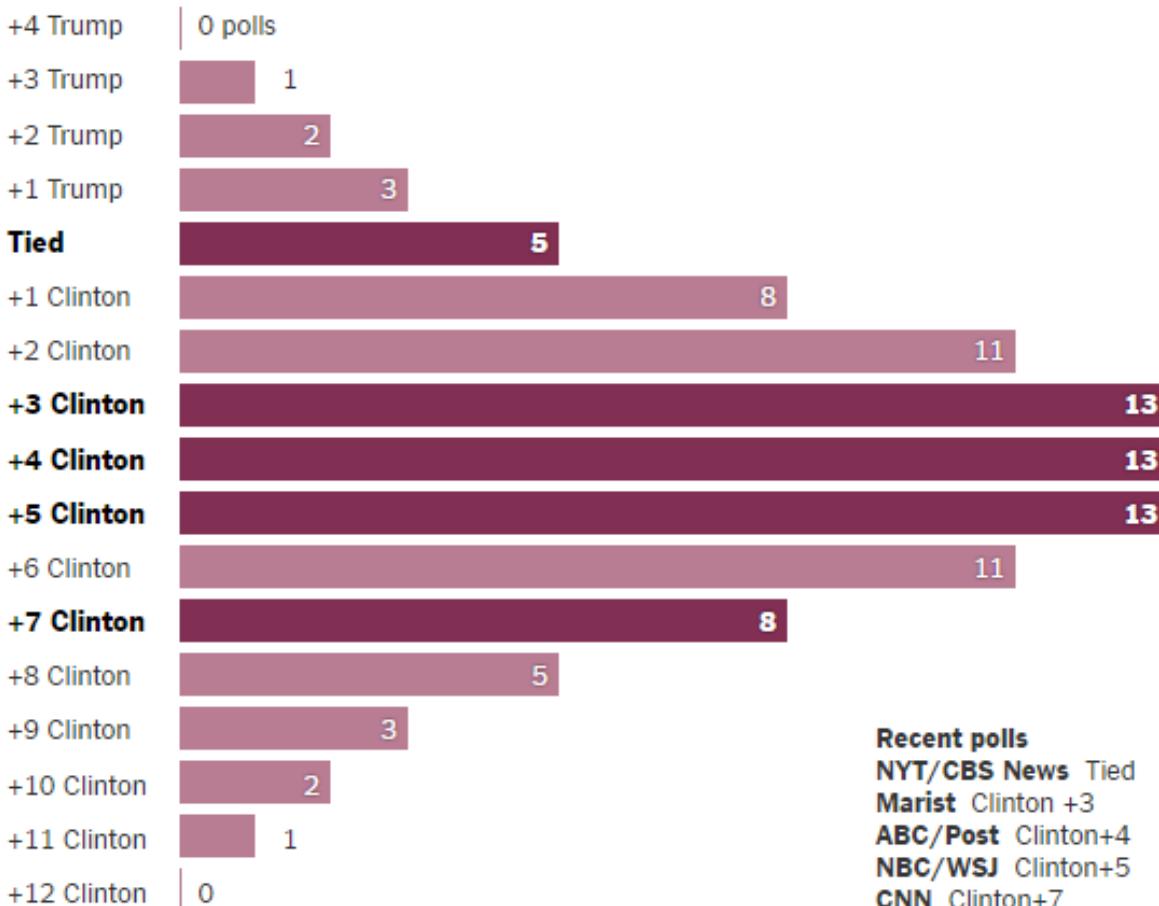
\hat{p}_{Trump}

Confused by Contradictory Polls? Take a Step Back

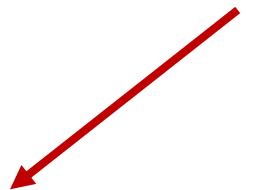
Noisy Polls Are to Be Expected

If Hillary Clinton were up by a modest margin, there would be plenty of polls showing a very close race — or even a Trump lead.

A simulation of 100 surveys, if Mrs. Clinton were really up 4 points nationally.



Sampling distribution of \hat{p}_{Clinton}



What parameter are they trying to estimate?

Questions?

Hypothesis tests

Statistical tests (hypothesis test)

A **statistical test** uses data from a sample to assess a claim about a population (parameter)

Example 1: The average body temperature of humans is 98.6°

How can we write this using symbols?

- $\mu = 98.6$

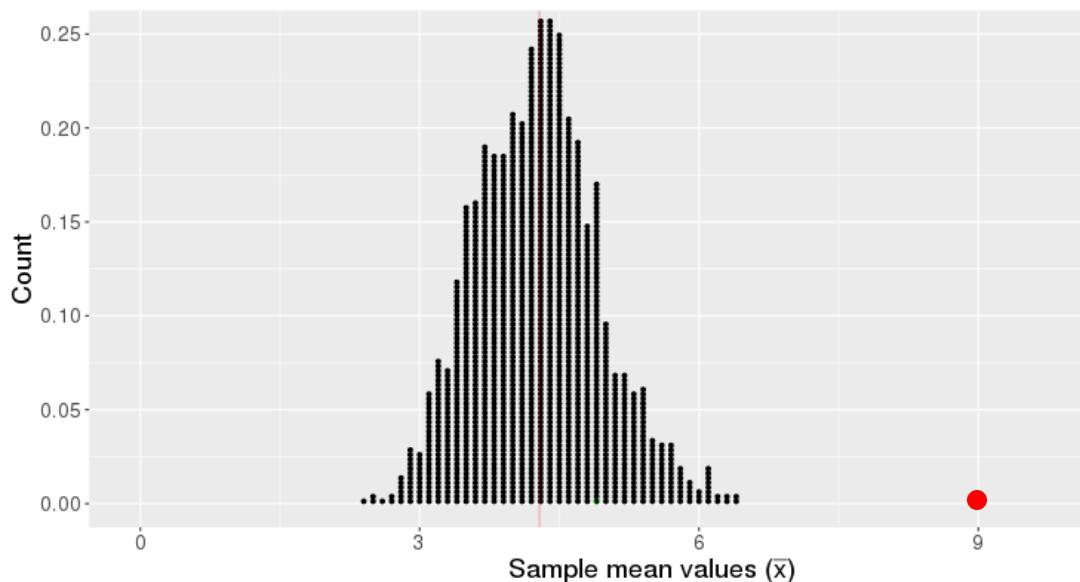
Basic hypothesis test logic

We start with a claim about a population parameter

- E.g., $\mu = 4$



This claim implies we should get a certain distribution of statistics



If our observed statistic is highly unlikely, we reject the claim

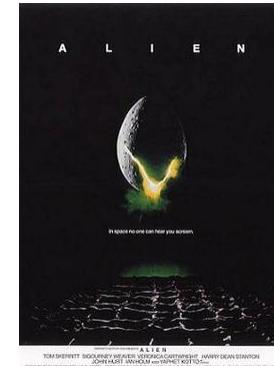
Motivating example: The Bechdel Test

Question: Do less than 50% of movies pass the Bechdel test?



Questions:

- What is the population/process?
- What is our parameter of interest?
 - What symbol should we use to denote it?
- What is our statistic of interest?
 - What symbol should we use to denote it?

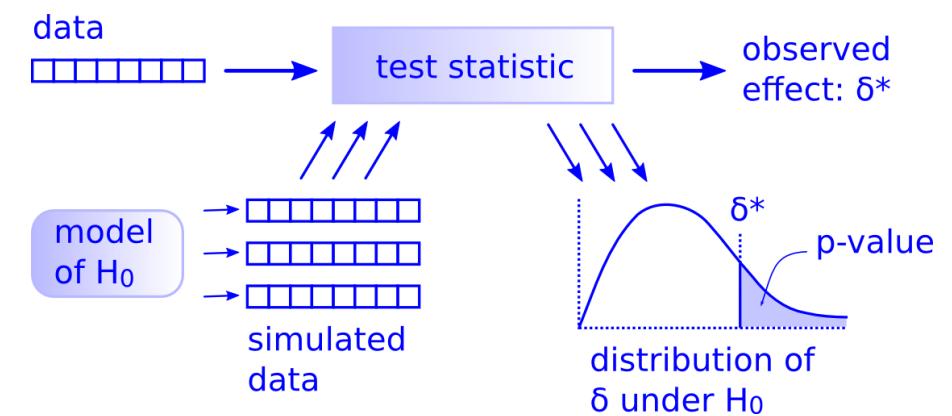
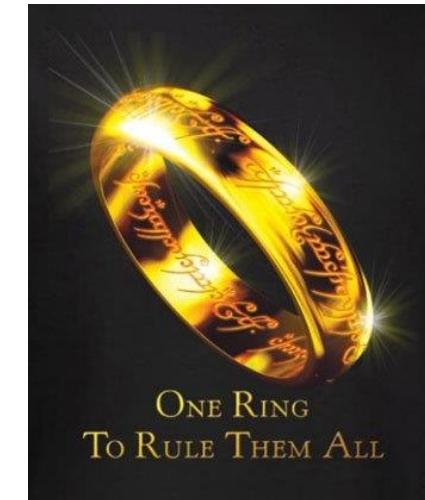


Steps needed to run a hypothesis test

To run a hypothesis test, we can use 5 steps:

1. State the null and alternative hypothesis
2. Calculate the observed statistic of interest
3. Create the null distribution
4. Calculate the p-value
5. Make a decision

Let's go through these steps now...



Do less than 50% of movies pass the Bechdel test?

Step 1: state the null and alternative hypotheses

If only 50% of the movies passed the Bechdel test, what would we expect the value of the parameter to be?

$$H_0: \pi = 0.5$$

If fewer than 50% of movies passed the Bechdel test, what would we expect the value of the parameter to be?

$$H_A: \pi < 0.5$$

Observed statistic value

Step 2: calculate the observed statistic

There are 1794 movies in our data set

Of these, 803 passed the Bechdel test

What is our observed statistic value and what symbol should we use to denote this value?

$$A: \hat{p} = 803/1794 = 0.448$$

	title	binary
1	Dredd 3D	PASS
2	12 Years a Slave	FAIL
3	2 Guns	FAIL
4	42	FAIL
5	47 Ronin	FAIL
6	A Good Day to Die Hard	FAIL
7	About Time	PASS
8	Admission	PASS
9	After Earth	FAIL
10	American Hustle	PASS
11	August: Osage County	PASS
12	Beautiful Creatures	PASS
13	Blue Jasmine	PASS
14	Captain Phillips	FAIL

Step 3: Create a null distribution

How can we assess whether 803 out of 1794 movies passing the Bechdel test ($\hat{p} = 0.448$) is consistent with what we would expect if 50% (or more) movies passed the Bechdel test?

- i.e., is $\hat{p} = 0.448$ a likely value if $\pi = 0.5$?

If 50% of movies passed the Bechdel test, we can model movies passing the as a fair coin flip:

Heads (True) = passed the Bechdel test

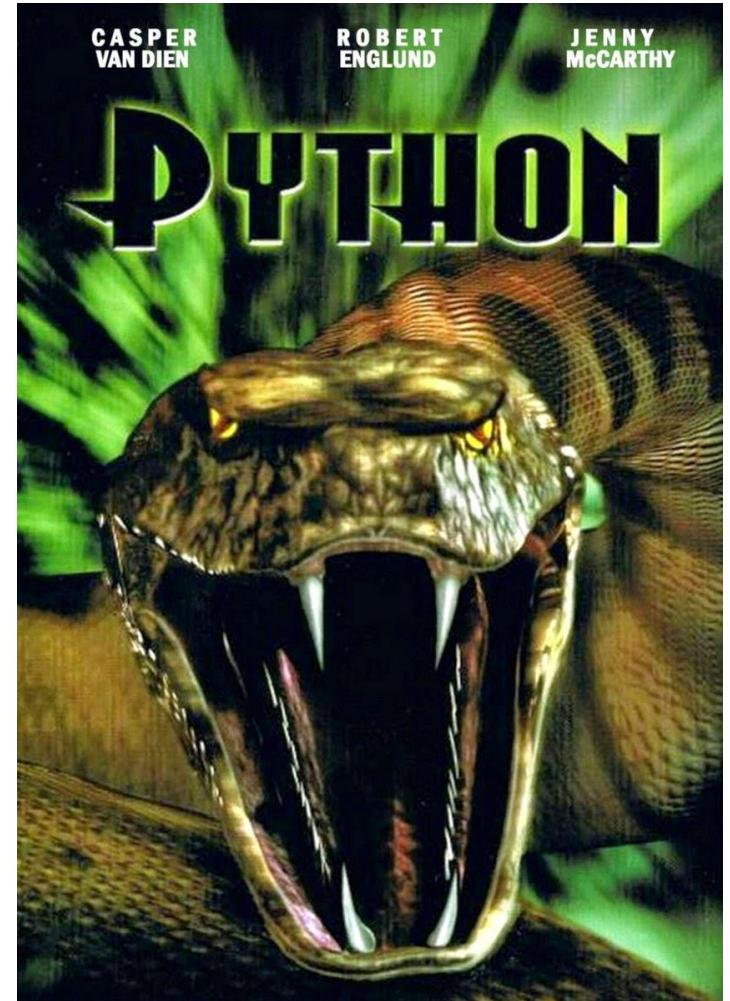
Tails (False) = failed to pass the Bechdel test

Let's simulate flipping a coin 1794 times and see how many times we get 803 or fewer heads

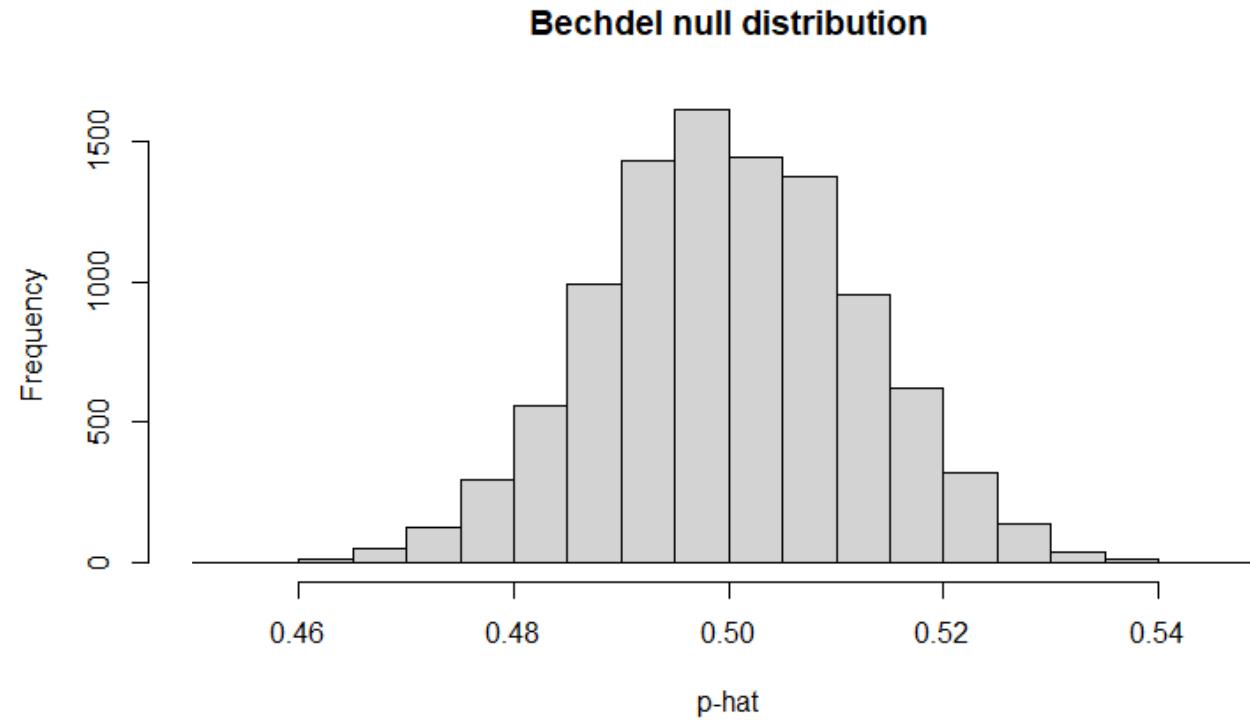
Chance models

To really be sure, how many repetitions of flipping a coin 1794 times should we do?

Any ideas how to do this?

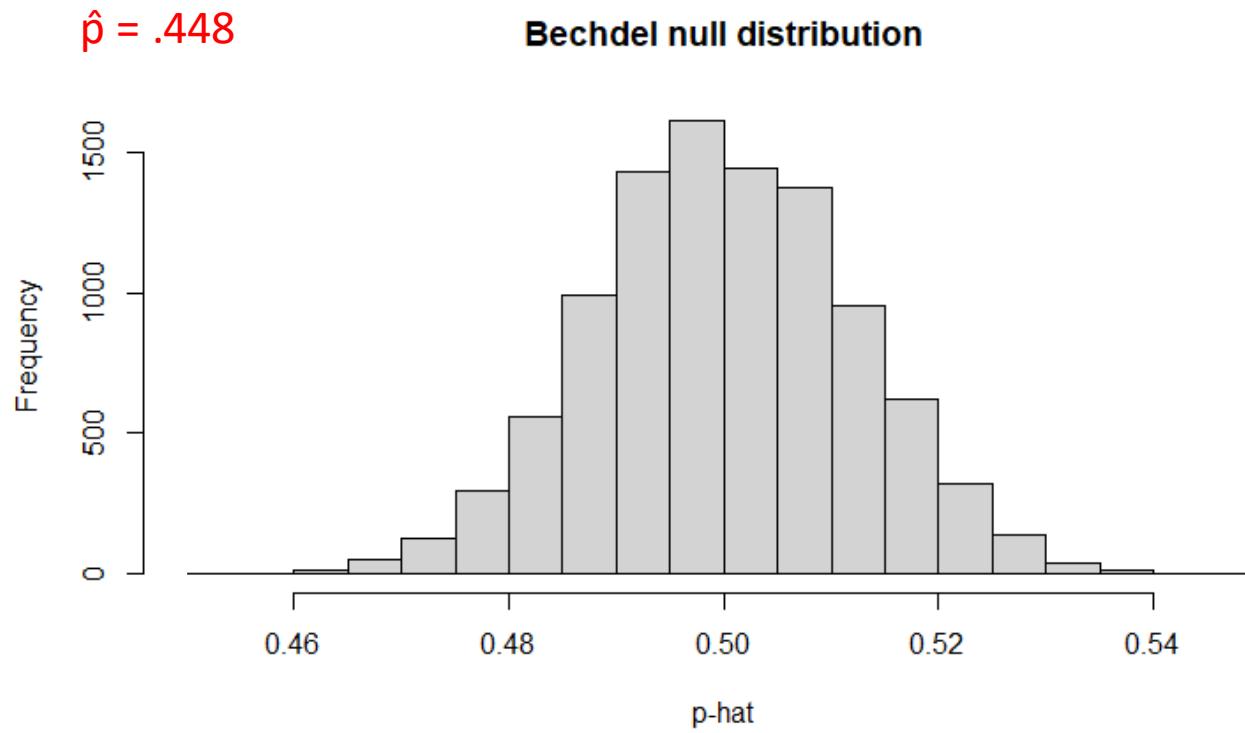


Simulating Flipping 1794 coins 10,000 times



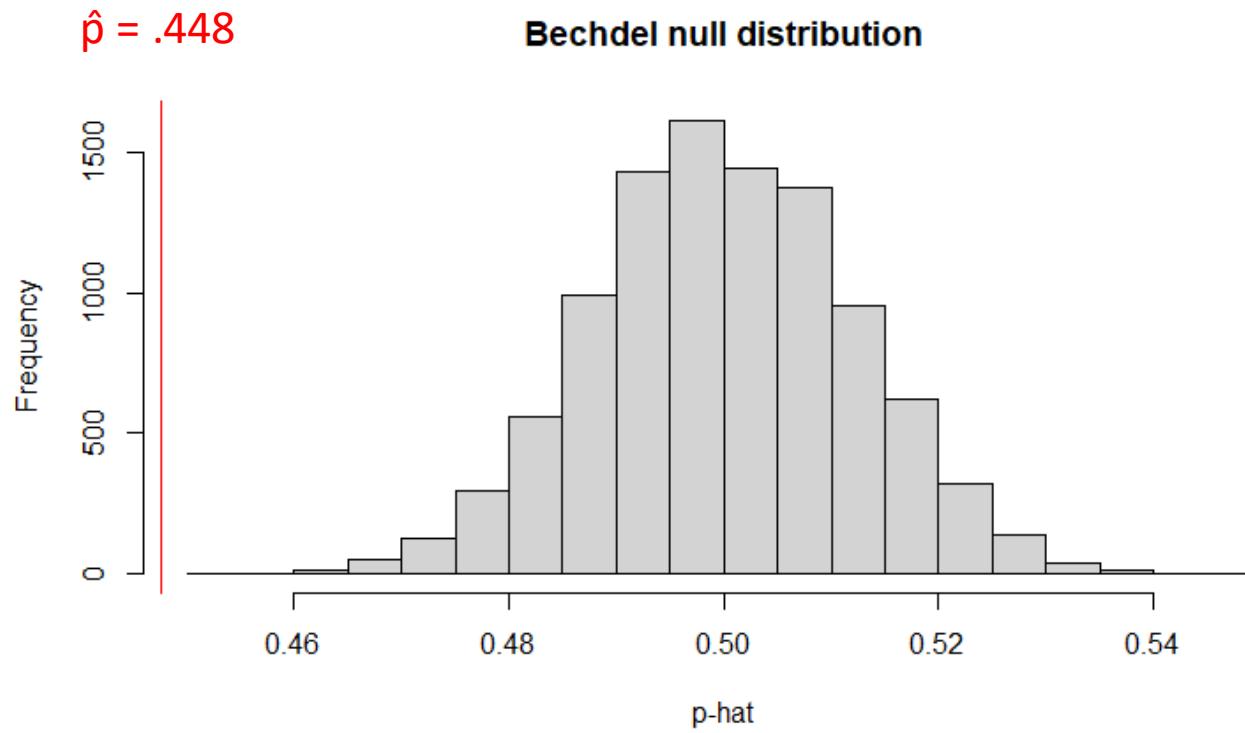
Assuming the null hypothesis is true, the distribution of statistics we get is called the **null distribution**

Step 4: calculate the p-value



Q: Is it likely that 50% of movies pass the Bechdel test?
• i.e., is it likely that $\pi = .5$?

Step 4: calculate the p-value

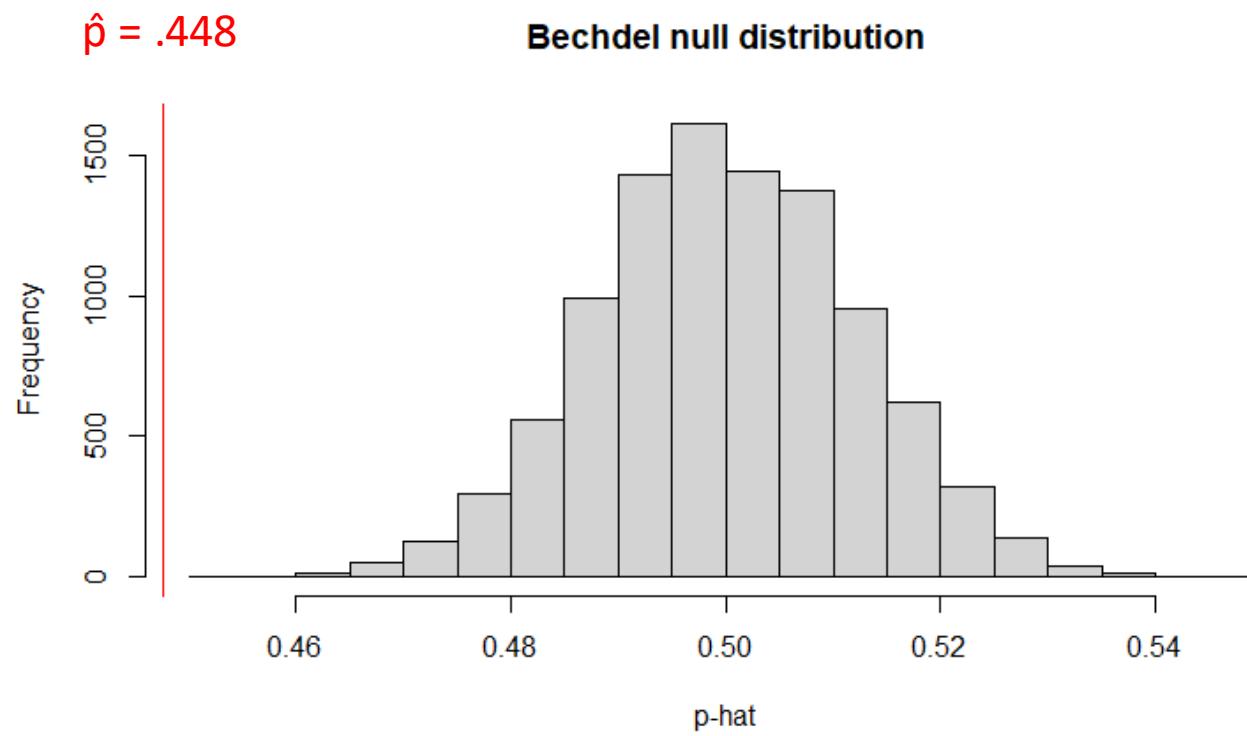


The **p-value** is the probability we will get a statistic as or more extreme than the observed statistic, if the null hypothesis was true

Q: What is the p-value here?

A: the p-value is 0

Step 5: Make a decision



If the observed data is very unlikely if the null hypothesis is true, we can reject the null hypothesis

- i.e., if p-value is very small we can reject the null hypothesis

Bechdel (hypothesis) test

1. State the null hypothesis and the alternative hypothesis

- 50% of the movies pass the Bechdel test: $H_0: \pi = 0.5$
- Less than 50% of movies pass the: $H_A: \pi < 0.5$

2. Calculate the observed statistic

- 803 out of 1794 movies passed the Bechdel test

$$\hat{p} = .448$$

3. Create a null distribution that is consistent with the null hypothesis

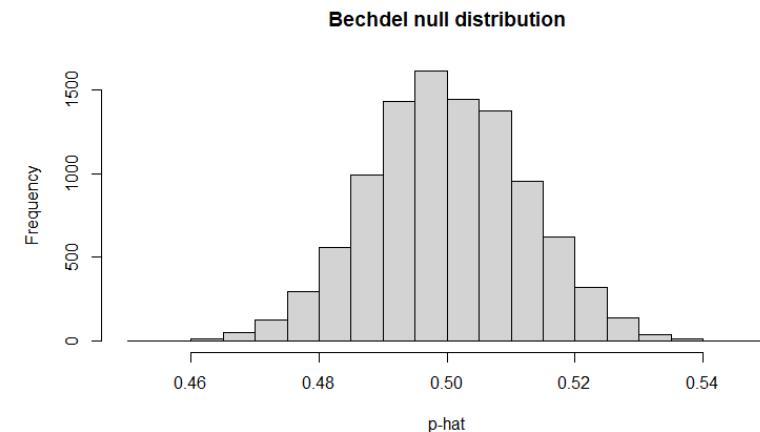
- i.e., the statistics we expect if 50% of the movies passed the Bechdel test

4. Examine how likely the observed statistic is to come from the null distribution

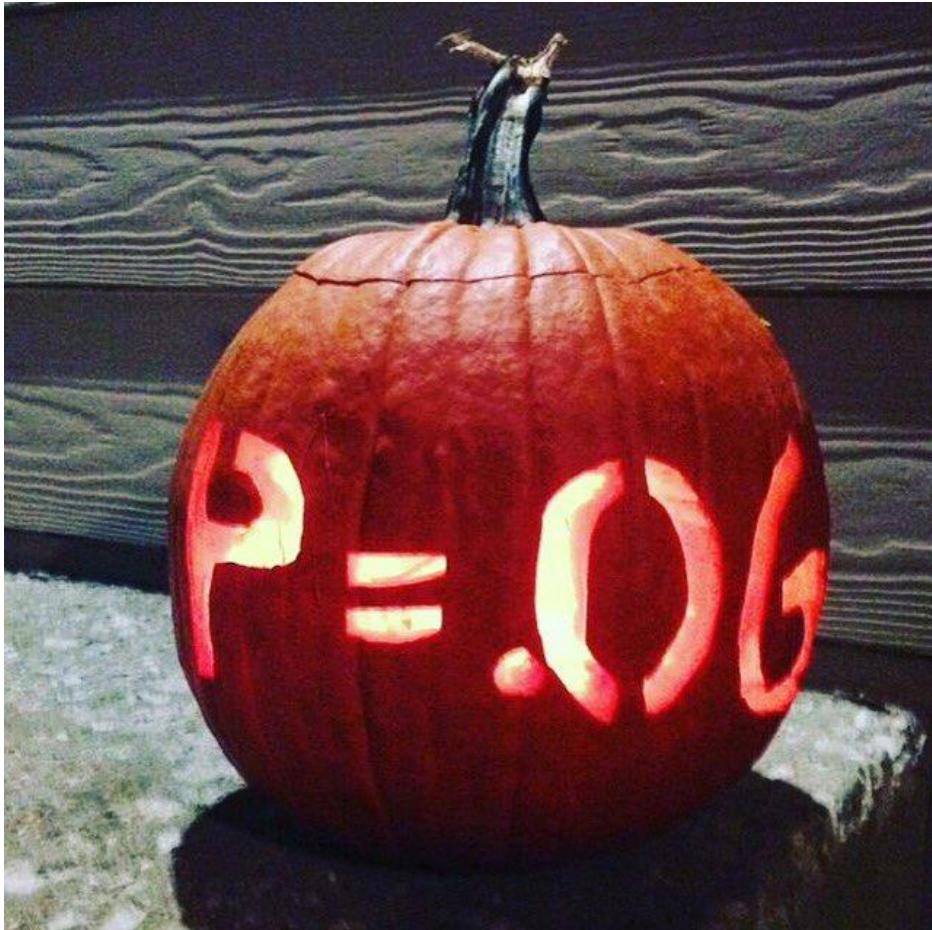
- What is the probability that only 803 of 1794 movies would pass the Bechdel test ($\hat{p} = .448$) if the null hypothesis was true?
- i.e., what is the p-value?

5. Make a judgement

- A small p-value this means that $\pi = .5$ is unlikely, and so it is likely $\pi < .5$
- i.e., we say our results are 'statistically significant'



Let's try it in Python!

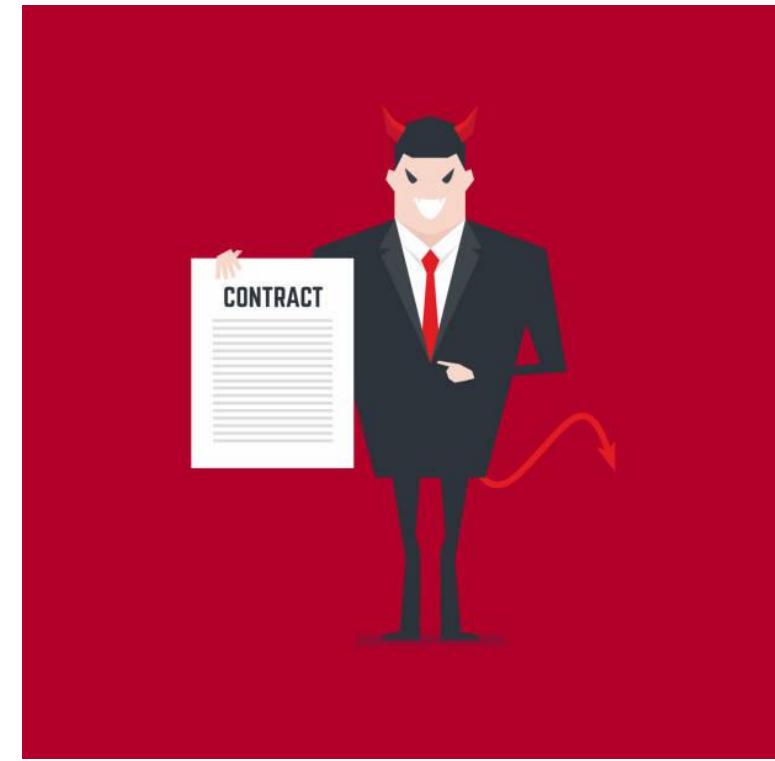


Another example: sinister lawyers

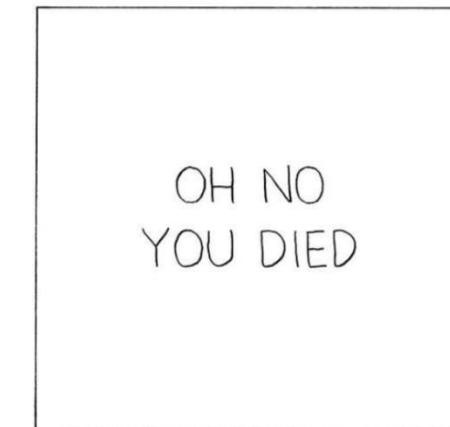
10% of American population on is left-handed

A study found that out of a random sample of 105 lawyers, 16 were left-handed

Use our 5 steps of hypothesis testing to assess whether the proportion of left-handed lawyers is greater than the proportion on found in the American population



Let's try it in Python!



Assessing causal relationships

Causality

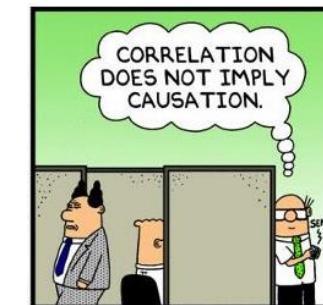
An **association** is the presence of a reliable relationship between the treatments an outcome

A **causal relationship** is when changing the value of a treatment variable influences the value outcome variable

A **confounding variable** (also known as a **lurking variable**) is a third variable that is associated with both the treatment (explanatory) variable and the outcome (response) variable

- A confounding variable can offer a plausible explanation for an association between the other two variables of interest

The screenshot shows a news article from NPR's website. The title is "Chocolate, Chocolate, It's Good For Your Heart, Study Finds". The date is June 19, 2015, at 5:03 AM ET, and it was heard on Morning Edition. The author is Allison Aubrey. The article discusses the health benefits of chocolate, specifically polyphenols found in cocoa beans. There is a small image of dark chocolate bars on the right.



Lurking variable

Randomized Controlled Experiment

Sample A: control group

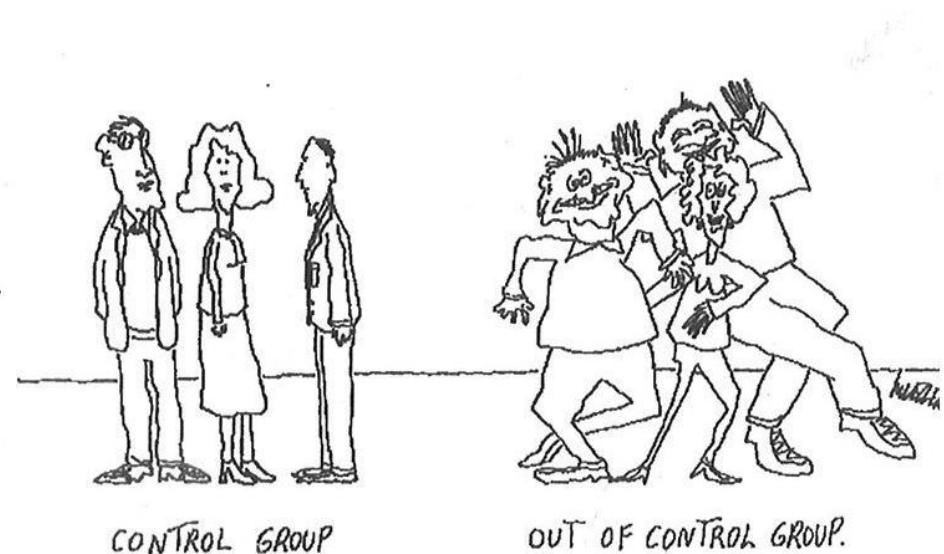
Sample B: treatment group

If members of the treatment and control groups are selected at random; this allows causal conclusions!

In particular, any difference in outcomes between the two groups could be due to:

- Chance
- The treatment

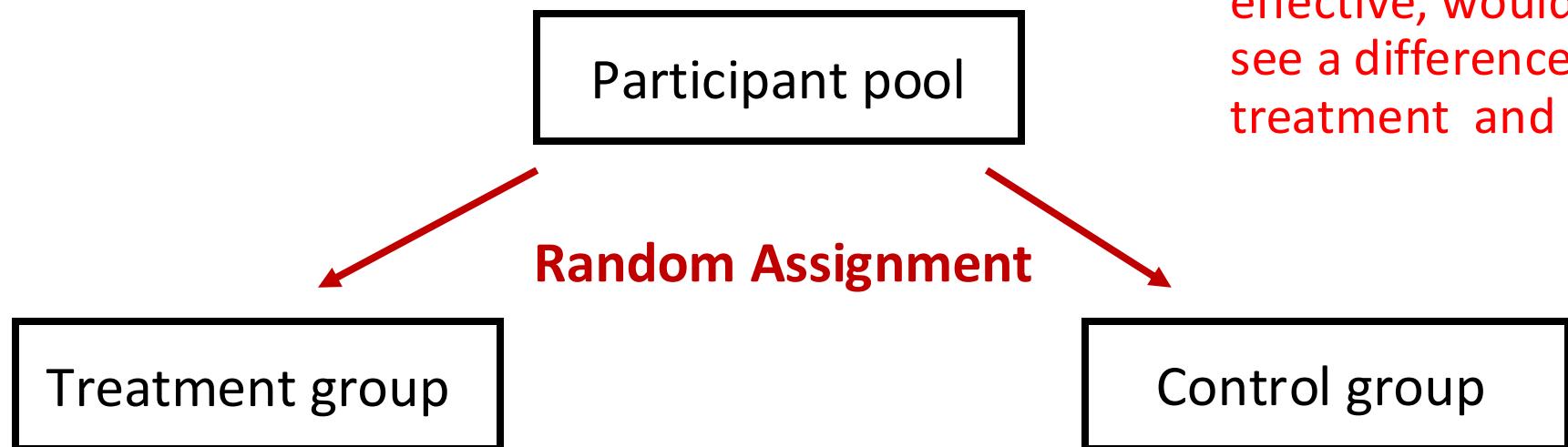
Randomly assigning participants to treatment and control groups allows us to separate what expected by chance and consequently what is due to the treatment



Randomized Controlled Experiment

Take a group of participant and *randomly assign*:

- Half to a *treatment group* where they get chocolate
- Half in a *control group* where they get a fake chocolate (placebo)
- See if there is more improvement in the treatment group compared to the control group



Q: If the treatment was not effective, would we expect to see a difference between the treatment and control groups?

Case study

RCT to study Botulinum Toxin A (BTA) as a treatment to relieve chronic back pain

- 15 patients in the treatment group (received BTA)
- 16 in the control group (normal saline)

Trials were run double-blind: neither doctors nor patients knew which group they were in.

Results

- 2 patients in the control group had relief from pain (outcome=1)
- 9 patients in the treatment group had relief.

Can this difference be just due to chance?

Neurology®

May 22, 2001; 56 (10) ARTICLES

Botulinum toxin A and chronic low back pain

A randomized, double-blind study

Leslie Foster, Larry Clapp, Marleigh Erickson, Bahman Jabbari

First published May 22, 2001, DOI:
<https://doi.org/10.1212/WNL.56.10.1290>

Step 1: The hypotheses

Null:

- BTA does not lead to an increase in pain relief
 - i.e., if many people were to get BTA and saline, the proportion of people who experienced pain relief would be the same in both groups.
 - $H_0: \pi_{\text{treat}} = \pi_{\text{control}}$

Alternative:

- BTA leads to an increase in pain relief
 - i.e., if many people were to get BTA and saline, the proportion of people who experienced pain relief would be higher for those who received BTA
 - $H_A: \pi_{\text{treat}} > \pi_{\text{control}}$

Neurology®

May 22, 2001; 56 (10) ARTICLES

Botulinum toxin A and chronic low back pain

A randomized, double-blind study

Leslie Foster, Larry Clapp, Marleigh Erickson, Bahman Jabbari

First published May 22, 2001, DOI:
<https://doi.org/10.1212/WNL.56.10.1290>

Step 2: The observed statistic

To calculate an observed statistic we need data:

Let's have our observed statistic mirror our hypotheses

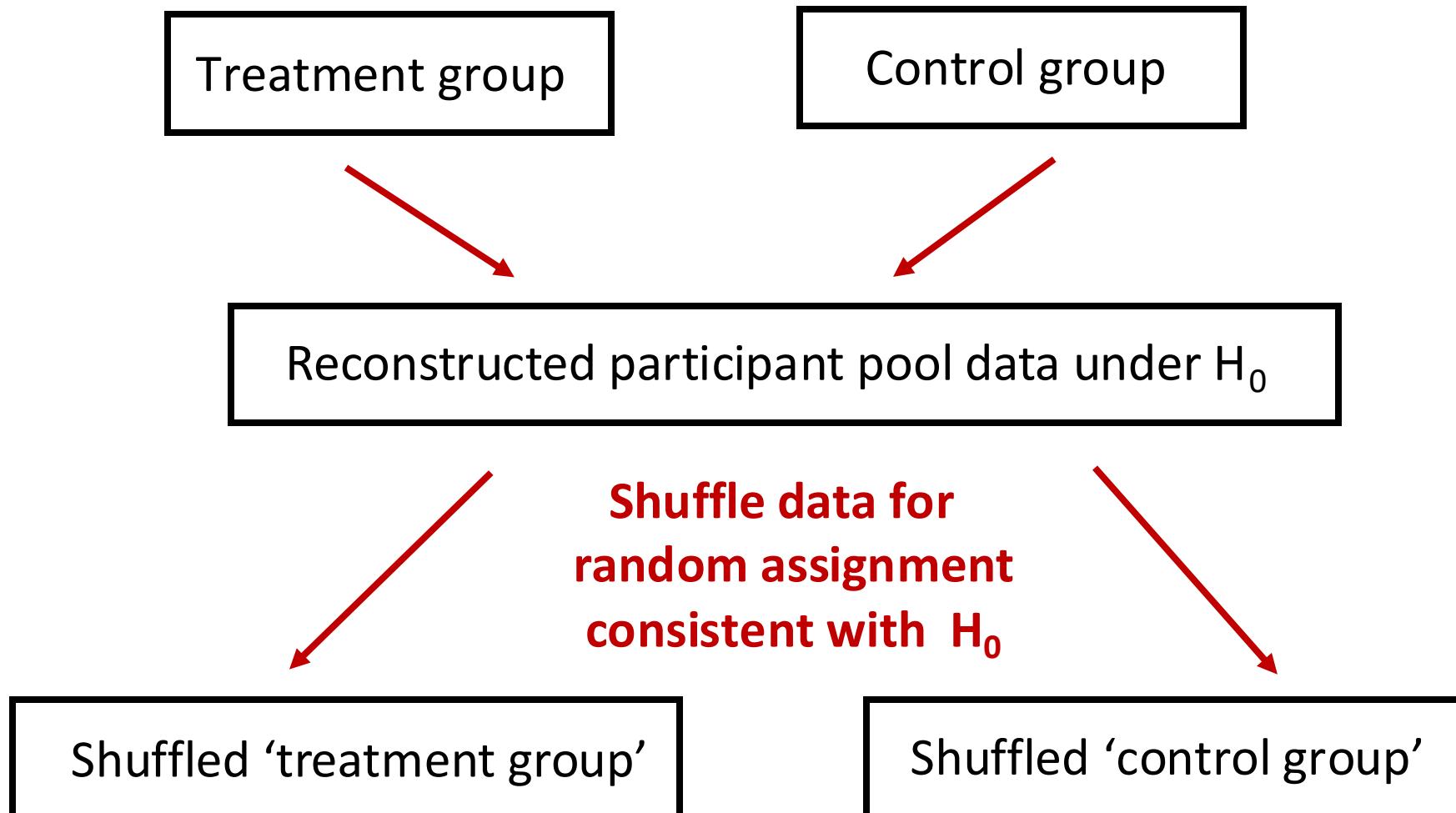
- $H_0: \pi_{\text{treat}} - \pi_{\text{control}} = 0$

Observed statistic is: $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$

$$= 9/15 - 2/16$$
$$= 0.475$$

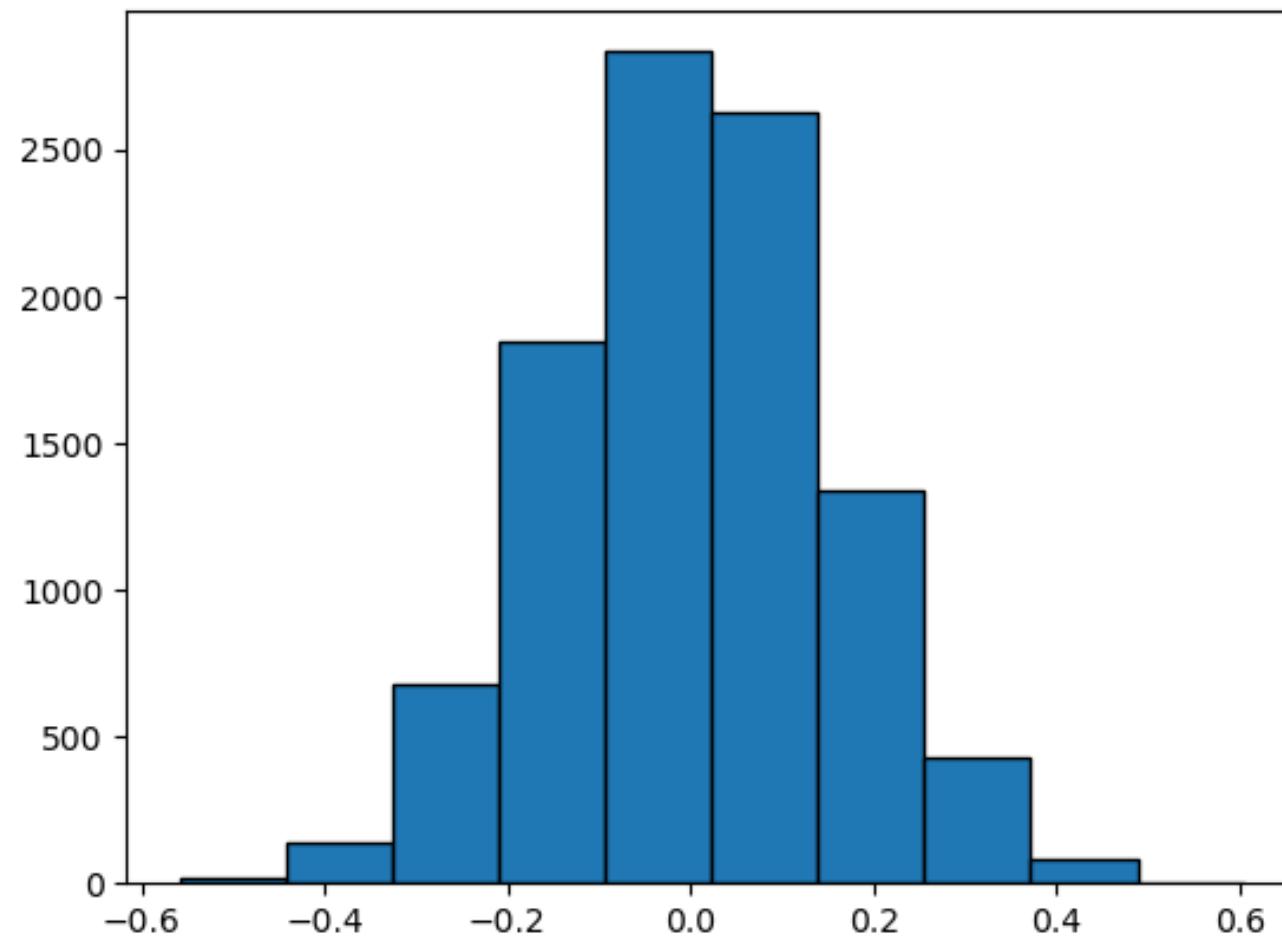
	Group	Result
19	Treatment	1.0
7	Control	0.0
6	Control	0.0
26	Treatment	0.0
17	Treatment	1.0
9	Control	0.0
13	Control	0.0
3	Control	0.0
1	Control	1.0
30	Treatment	0.0
28	Treatment	0.0

3. Create the null distribution!

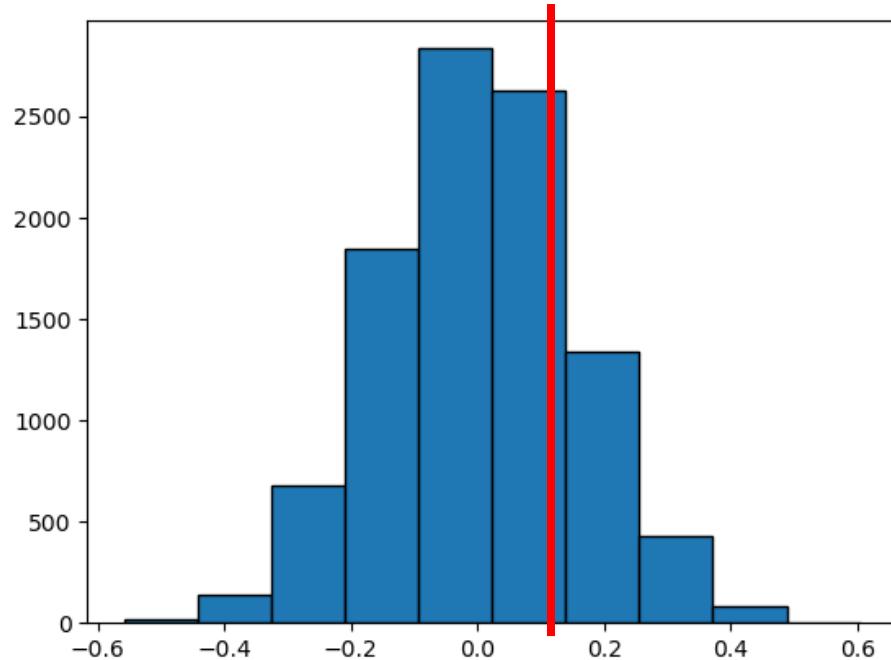


One null distribution statistic: $\hat{p}_{\text{Shuff_Treatment}} - \hat{p}_{\text{Shuff_control}}$

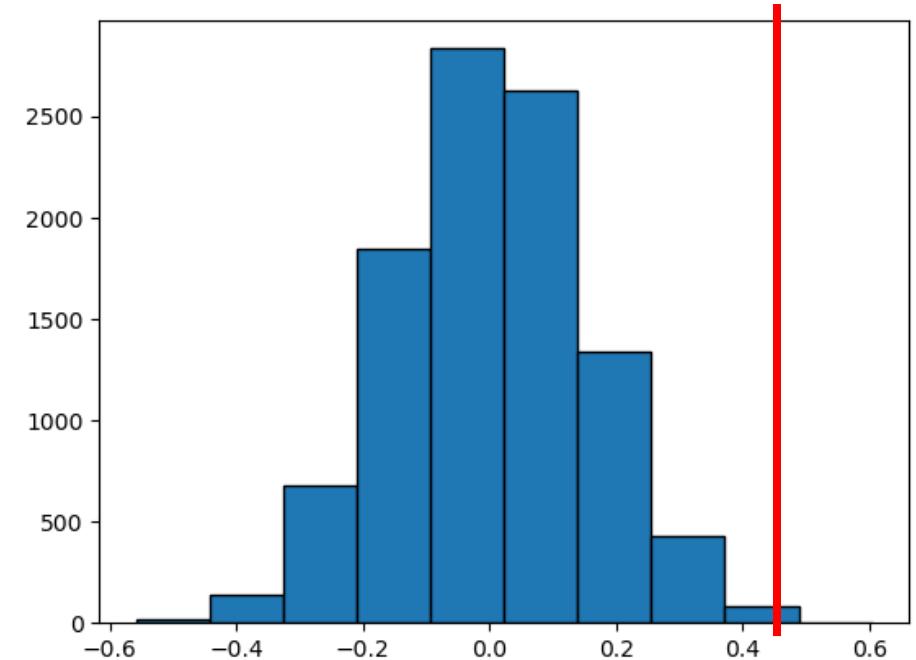
Step 3: Create a null distribution



Step 4: Calculate the p-value

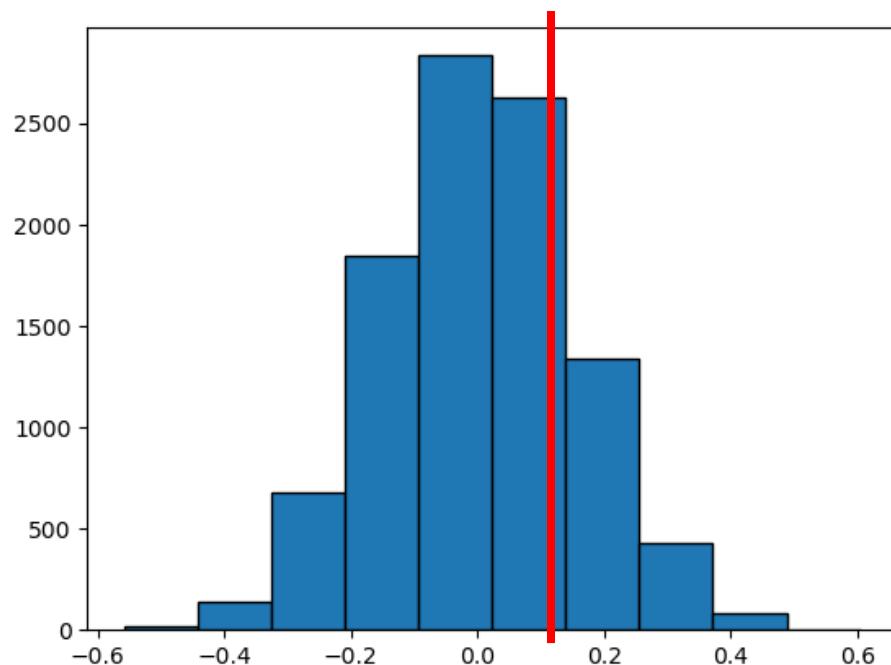


If $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}} = 0.1$ what would the p-value be?

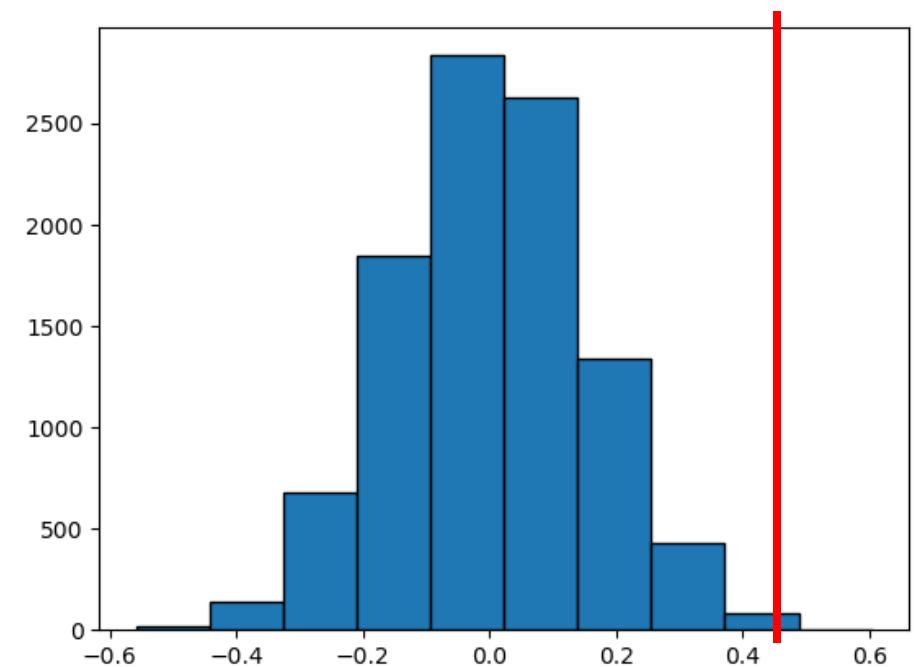


If $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}} = 0.5$ what would the p-value be?

Step 5: Draw a conclusion



If the p-value was 0.19 what would we conclude?



If the p-value was 0.0007 what would we conclude?

Summary: BTA for back pain relief

1. State the null hypothesis and the alternative hypothesis

- BTA does not lead to an increase in pain relief: $H_0: \pi_{\text{treat}} = \pi_{\text{control}}$
- BTA leads to an increase in pain relief: $H_A: \pi_{\text{treat}} > \pi_{\text{control}}$

2. Calculate the observed statistic: $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$

3. Create a null distribution that is consistent with the null hypothesis

- The $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$ statistics we expect if the null hypothesis was true
- i.e., statistics we would expect if there was no difference in pain relief between the two groups

4. Examine how likely the observed statistic is to come from the null distribution

- What is the probability that we would get a $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$ statistic larger than 0.475 if the null hypothesis was true?
- i.e., what is the p-value?

5. Make a judgement

- A small p-value this means that at the proportion of pain relief differed between the two groups
 - i.e., we say our results are 'statistically significant'
- Because our analysis is based on a randomized controlled trial (using random assignment) we can say that BTA causes an increase in pain relief

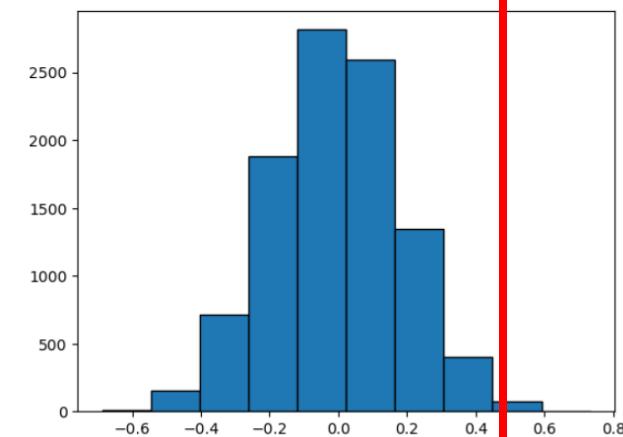
May 22, 2001; 56 (10) ARTICLES

Botulinum toxin A and chronic low back pain
A randomized, double-blind study

Leslie Foster, Larry Clapp, Marleigh Erickson, Bahman Jabbari

First published May 22, 2001, DOI:
<https://doi.org/10.1212/WNL.56.10.1290>

$$\hat{p}_{\text{treat}} - \hat{p}_{\text{control}} = .475$$



Let's try it in Python!

YData: Introduction to Data Science



Class 19: Hypothesis tests for two means

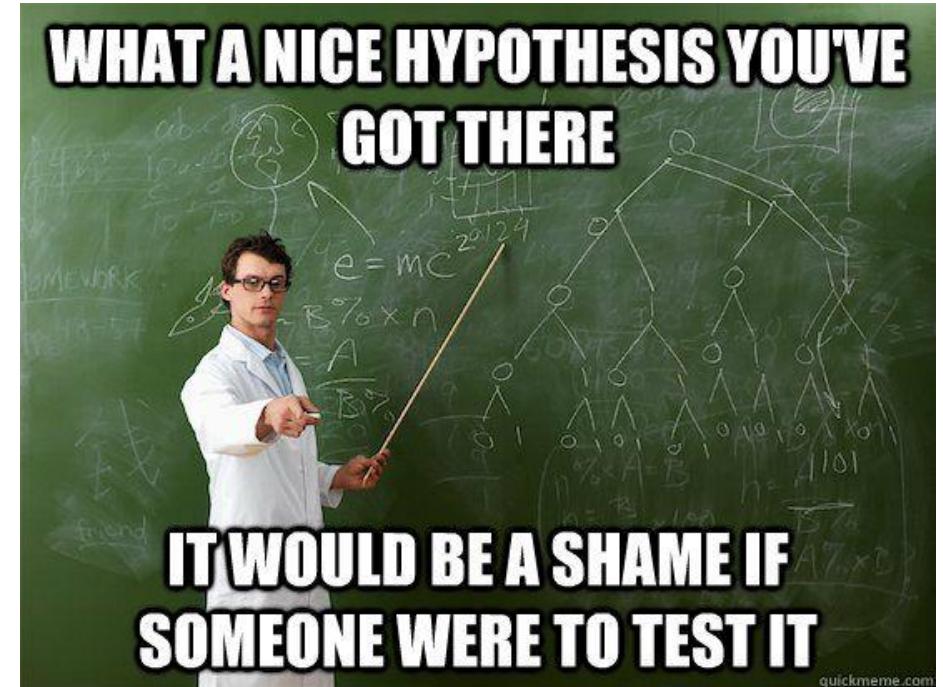
Overview

Quick review of parameters, statistics, sampling, and hypothesis tests for a single proportion

Hypothesis tests for two means and assessing causality

If there is time:

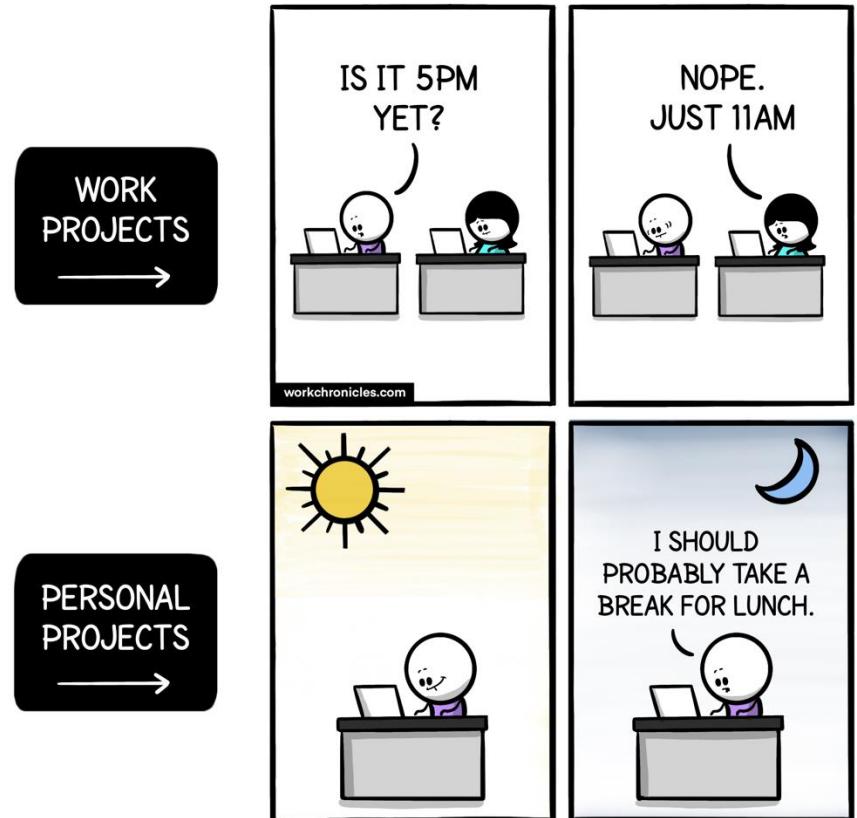
- Hypothesis tests for correlation
- Visual hypothesis tests
- Two-sided hypothesis tests



Reminder: keep working on your class project

Homework 8 is due on **Sunday November 10th**

A **polished** draft of the project is due on **November 17th**



Hello. I make comics about work.
Join [r/workchronicles](#) or follow on [Instagram](#) / [Twitter](#) / [FB](#)

Work Chronicles
[workchronicles.com](#)

Review of Statistical Inference

Review: Statistical Inference

Statistical Inference: Making conclusions about a population based on data in a random sample

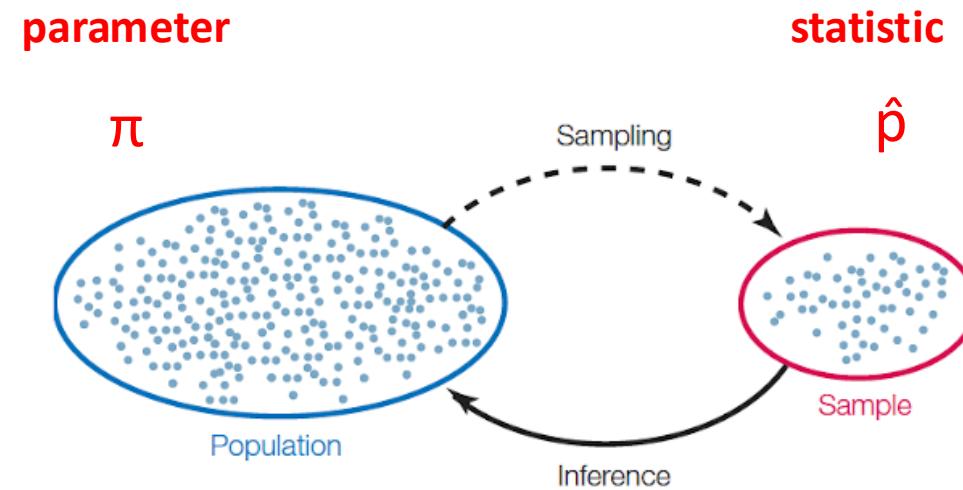
A **parameter** is number associated with the population

- e.g., population proportion π
- e.g., the proportion of all voters who voted for Trump

A **statistic** is number calculated from the sample

- e.g., sample proportion \hat{p}
- e.g., the proportion of Trump's vote out of 1,000 people in a sample

A statistic can be used as an estimate of a parameter



	Sample Statistic	Population Parameter
Mean	\bar{x}	μ
Proportion	\hat{p}	π
Correlation	r	ρ

Probability distribution of a statistic

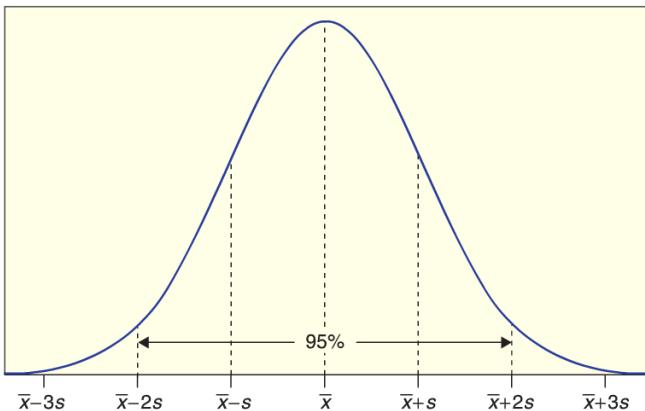
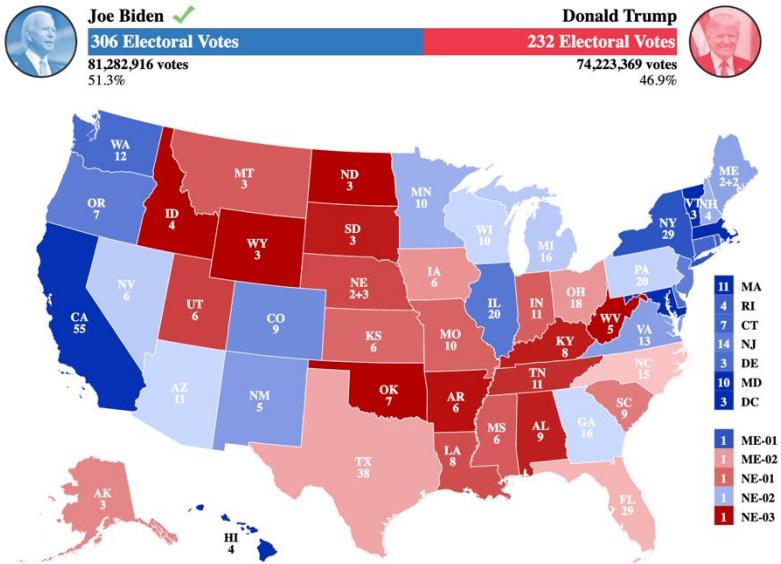
Values of a statistic vary because random samples vary

A **sampling distribution** is a probability distribution of *statistics*

- All possible values of the statistic and all the corresponding probabilities
- We can approximate a sampling distribution by simulating statistics

π_{Trump}

$n = 1,000$



Sampling distribution!



\hat{p}_{Trump}



\hat{p}_{Trump}



\hat{p}_{Trump}

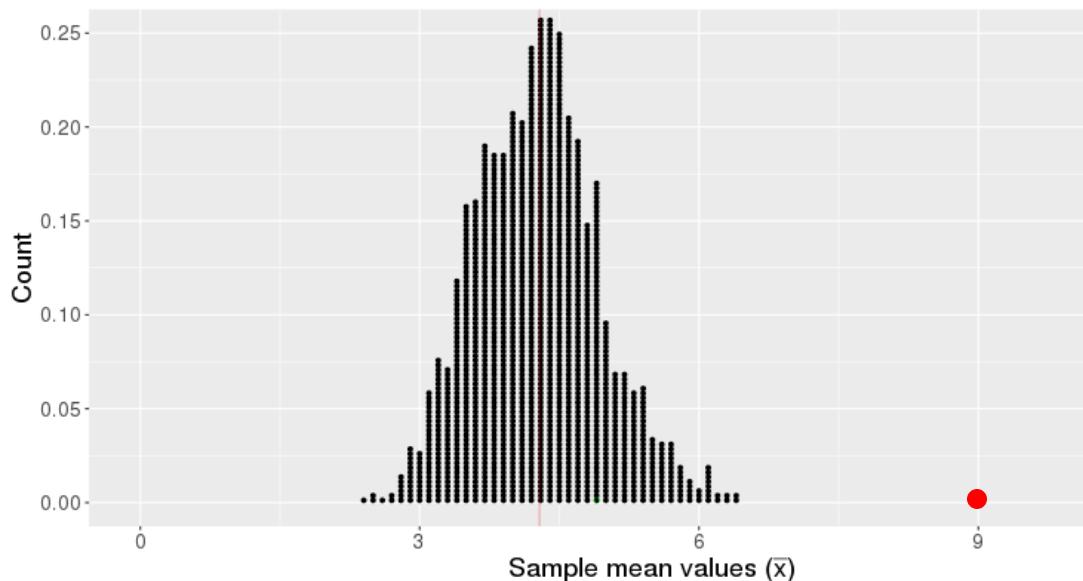
Hypothesis tests

Basic hypothesis test logic

We start with a claim about a population parameter

- E.g., $\mu = 4$

This claim implies we should get a certain distribution of statistics



If our observed statistic is highly unlikely, we reject the claim

Null and Alternative hypotheses

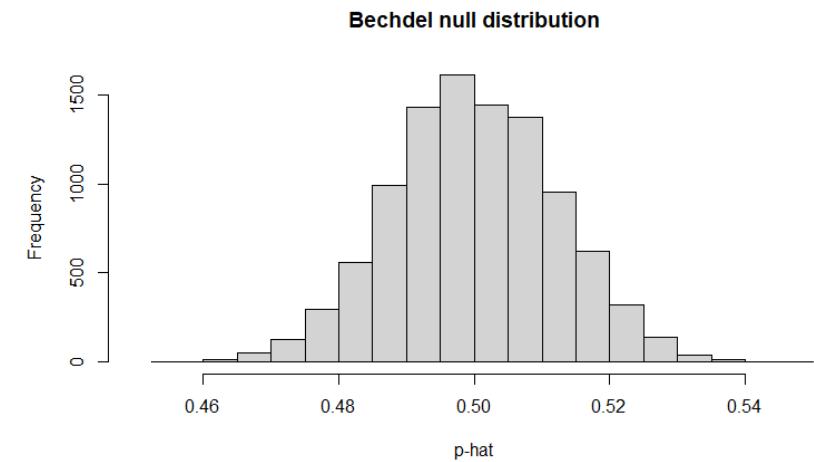
Null hypothesis

- A hypothesis where “nothing interesting” happened
 - E.g., our experiment failed
 - E.g., $H_0: \pi = 0.5$
- We can simulate data under the assumptions of this model to get a "null distribution" of statistics

Alternative hypothesis

- The hypothesis we believe in (would like to see true)
- E.g., $H_A: \pi < 0.5$

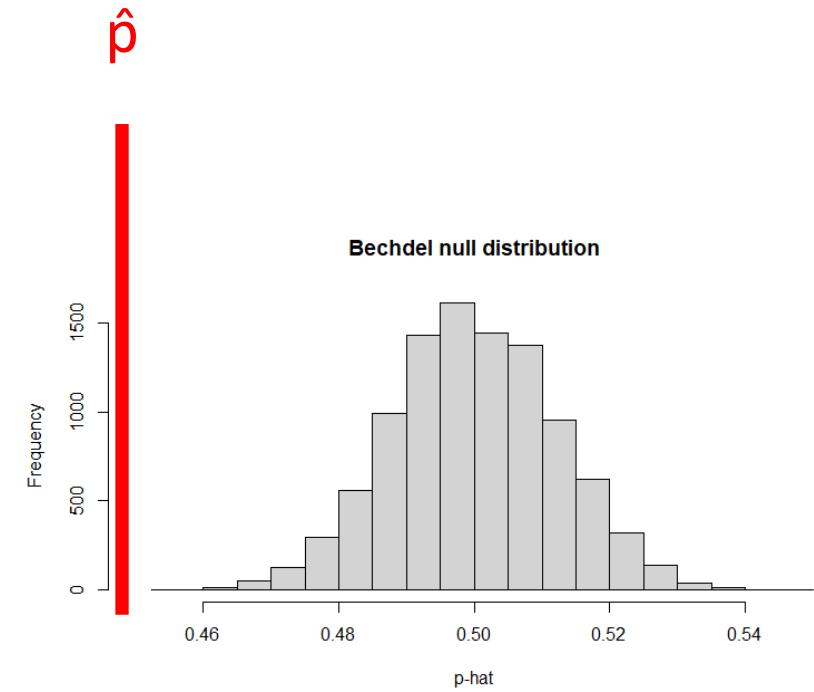
A **test statistic** is the statistic we choose to simulate in order decide between the two hypotheses



Testing the null hypothesis

To resolve choice between null and alternative hypotheses:

- We compare the **observed test statistic** to the statistic values in the null distribution
- If the observed statistic is not consistent with the null distribution, then we can **reject the null hypothesis**
 - E.g., $H_0: \hat{p} \geq 0.5$
 - And we accept the alternative hypothesis
 - E.g., $H_A: \pi < 0.5$



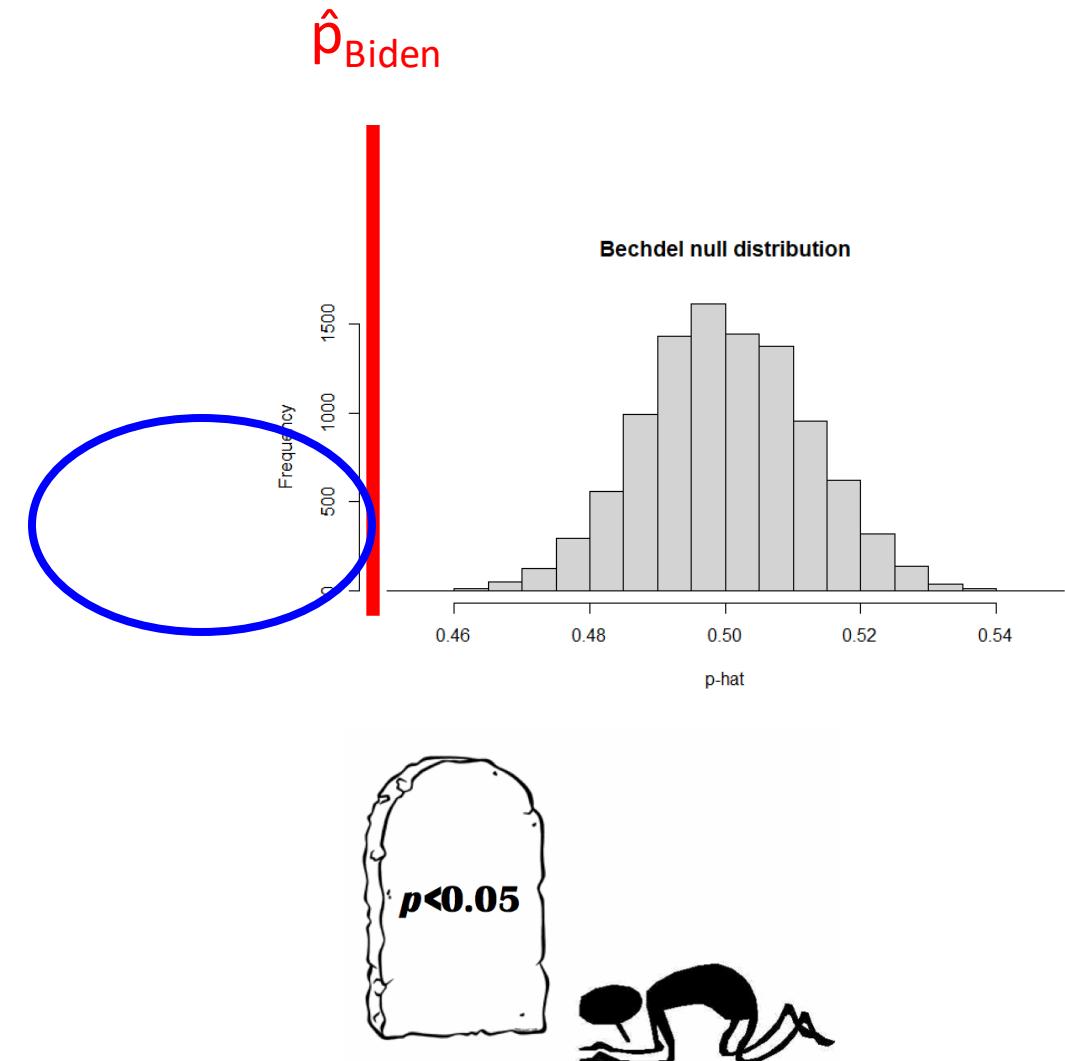
The p-value

The **p-value** is the probability, that we get a statistic as or more extreme than the observed statistic from the null distribution

- $P(\text{Null_Stat} \leq \text{obs_stat} | H_0)$

If the P-value is small, this is evidence against the null hypothesis and the results are often called "statistically significant"

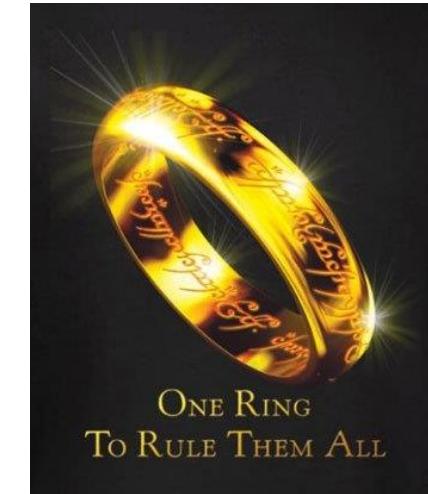
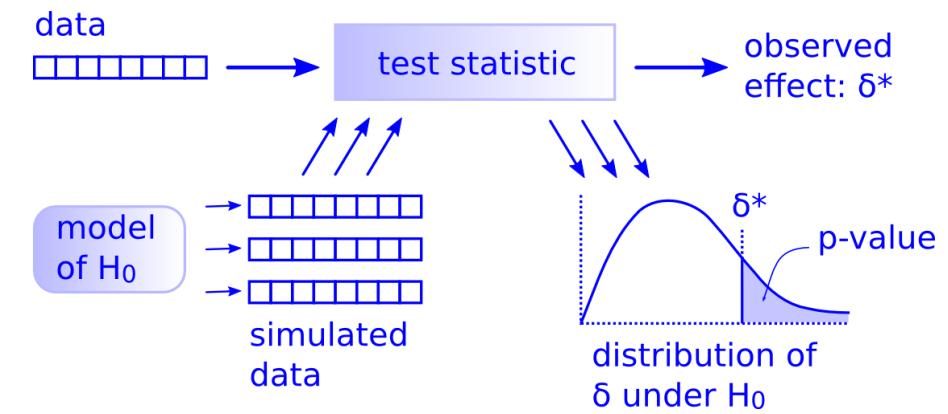
- Convention, p-value < 0.05



Steps needed to run a hypothesis test

To run a hypothesis test, we can use 5 steps:

1. State the null and alternative hypothesis
2. Calculate the observed statistic of interest
3. Create the null distribution
4. Calculate the p-value
5. Make a decision



Bechdel (hypothesis) test

1. State the null hypothesis and the alternative hypothesis

- 50% of the movies pass the Bechdel test: $H_0: \pi = 0.5$
- Less than 50% of movies pass the: $H_A: \pi < 0.5$

2. Calculate the observed statistic

- 803 out of 1794 movies passed the Bechdel test

3. Create a null distribution that is consistent with the null hypothesis

- i.e., the statistics we expect if 50% of the movies passed the Bechdel test

4. Examine how likely the observed statistic is to come from the null distribution

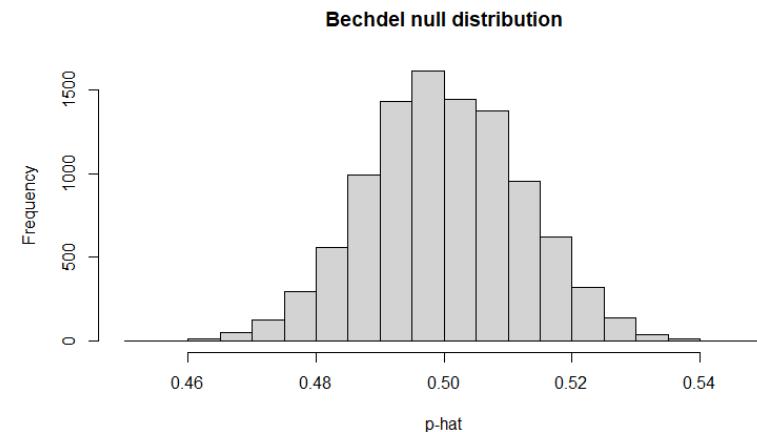
- What is the probability that only 803 of 1794 movies would pass the Bechdel test ($\hat{p} = .448$) if the null hypothesis was true?
- i.e., what is the p-value?

5. Make a judgement

- A small p-value this means that $\pi = .5$ is unlikely, and so it is likely $\pi < .5$
- i.e., we say our results are 'statistically significant'



$$\hat{p} = .448$$



Assessing causal relationships

Causality

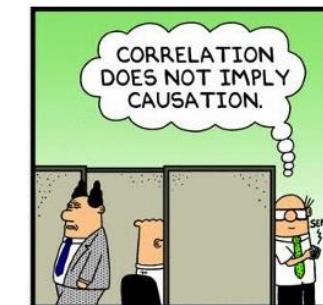
An association is the presence of a reliable relationship between the treatments an outcome

A causal relationship is when changing the value of a treatment variable influences the value outcome variable

A confounding variable (also known as a lurking variable) is a third variable that is associated with both the treatment (explanatory) variable and the outcome (response) variable

- A confounding variable can offer a plausible explanation for an association between the other two variables of interest

The image shows a screenshot of an NPR news article. The header includes the NPR logo and links for NEWS, ARTS & LIFE, MUSIC, PODCASTS & SHOWS, and SEARCH. The main headline is "Chocolate, Chocolate, It's Good For Your Heart, Study Finds". Below the headline is a photo of several dark chocolate bars. A caption states: "There's a growing body of evidence suggesting that compounds found in cocoa beans, called polyphenols, may help protect against heart disease." The author's name, ALLISON AUBREY, is mentioned with a small profile picture.

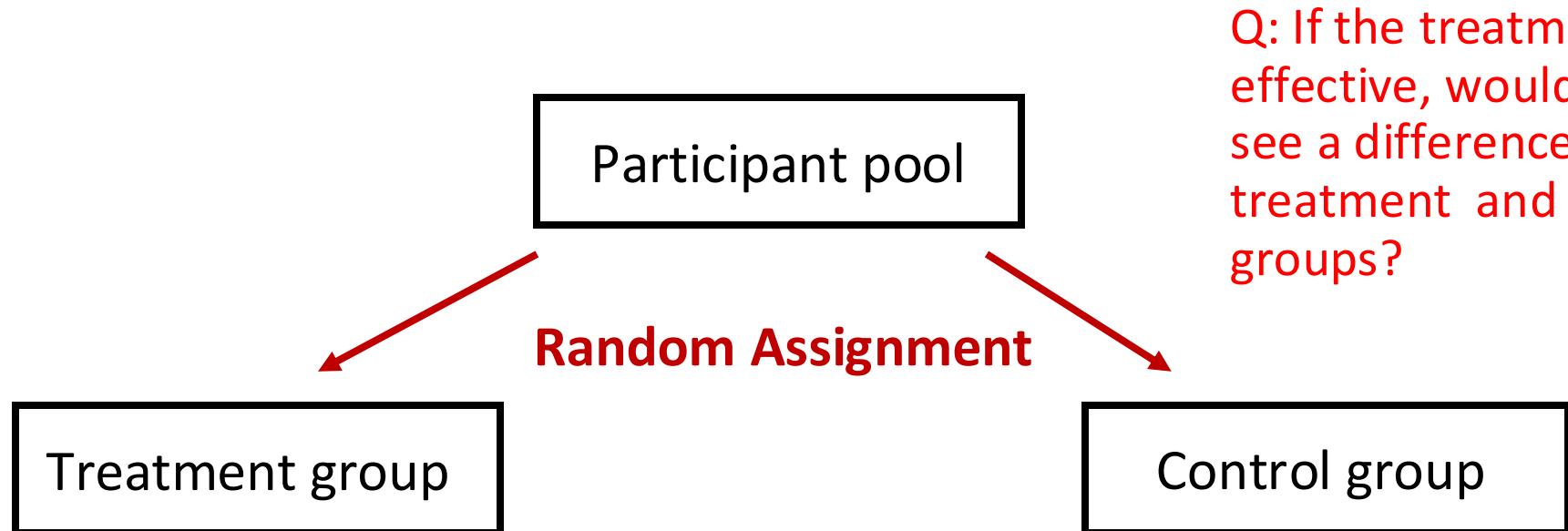


Lurking variable

Randomized Controlled Experiment

Take a group of participant and *randomly assign*:

- Half to a *treatment group* where they get chocolate
- Half in a *control group* where they get a fake chocolate (placebo)
- See if there is more improvement in the treatment group compared to the control group



Case study

RCT to study Botulinum Toxin A (BTA) as a treatment to relieve chronic back pain

- 15 patients in the treatment group (received BTA)
- 16 in the control group (normal saline)

Trials were run double-blind: neither doctors nor patients knew which group they were in.

Results

- 2 patients in the control group had relief from pain (outcome=1)
- 9 patients in the treatment group had relief.

Can this difference be just due to chance?

Neurology®

May 22, 2001; 56 (10) ARTICLES

Botulinum toxin A and chronic low back pain

A randomized, double-blind study

Leslie Foster, Larry Clapp, Marleigh Erickson, Bahman Jabbari

First published May 22, 2001, DOI:
<https://doi.org/10.1212/WNL.56.10.1290>

Step 1: The hypotheses

Null:

- BTA does not lead to an increase in pain relief
 - i.e., if many people were to get BTA and saline, the proportion of people who experienced pain relief would be the same in both groups.
 - $H_0: \pi_{\text{treat}} = \pi_{\text{control}}$

Alternative:

- BTA leads to an increase in pain relief
 - i.e., if many people were to get BTA and saline, the proportion of people who experienced pain relief would be higher for those who received BTA
 - $H_A: \pi_{\text{treat}} > \pi_{\text{control}}$

Neurology®

May 22, 2001; 56 (10) ARTICLES

Botulinum toxin A and chronic low back pain

A randomized, double-blind study

Leslie Foster, Larry Clapp, Marleigh Erickson, Bahman Jabbari

First published May 22, 2001, DOI:
<https://doi.org/10.1212/WNL.56.10.1290>

Step 2: The observed statistic

To calculate an observed statistic we need data:

Let's have our observed statistic mirror our hypotheses

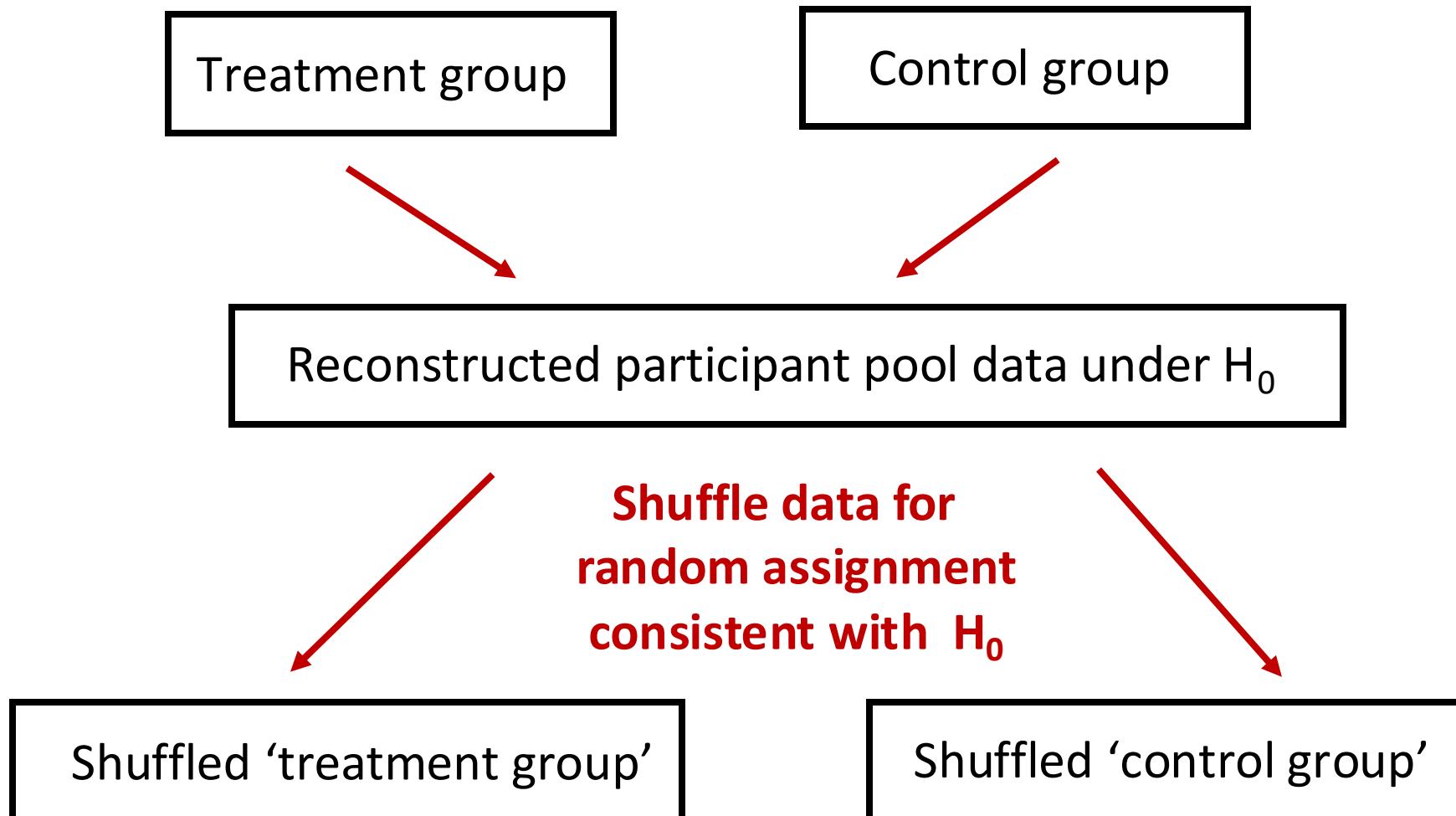
- $H_0: \pi_{\text{treat}} - \pi_{\text{control}} = 0$

Observed statistic is: $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$

$$= 9/15 - 2/16$$
$$= 0.475$$

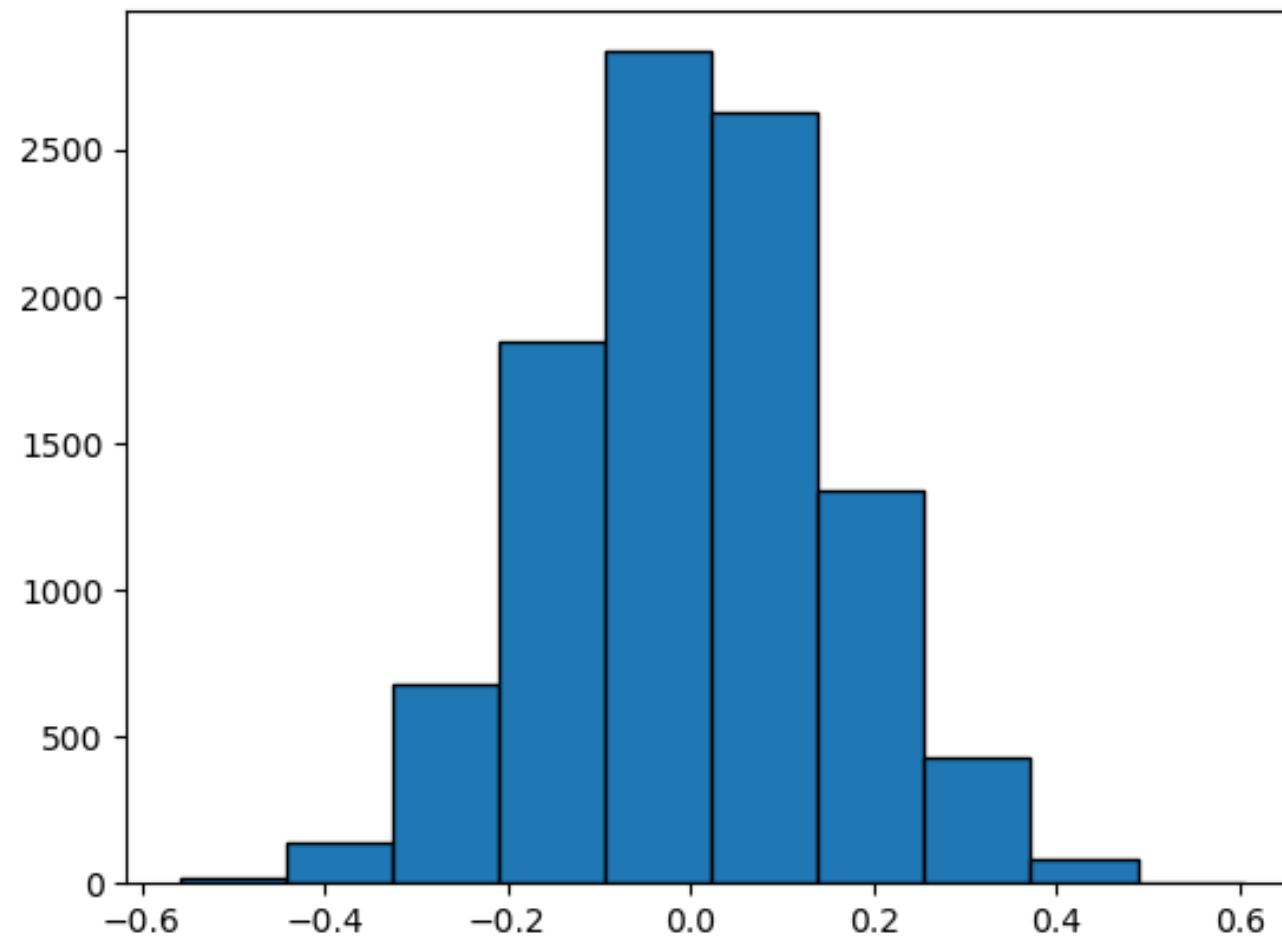
	Group	Result
19	Treatment	1.0
7	Control	0.0
6	Control	0.0
26	Treatment	0.0
17	Treatment	1.0
9	Control	0.0
13	Control	0.0
3	Control	0.0
1	Control	1.0
30	Treatment	0.0
28	Treatment	0.0

3. Create the null distribution!

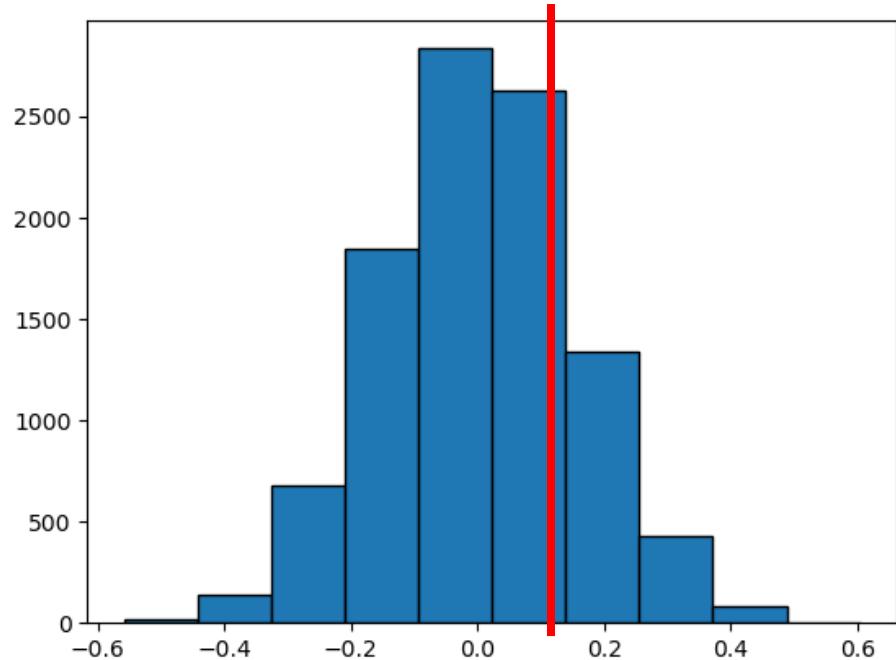


One null distribution statistic: $\hat{p}_{\text{Shuff_Treatment}} - \hat{p}_{\text{Shuff_control}}$

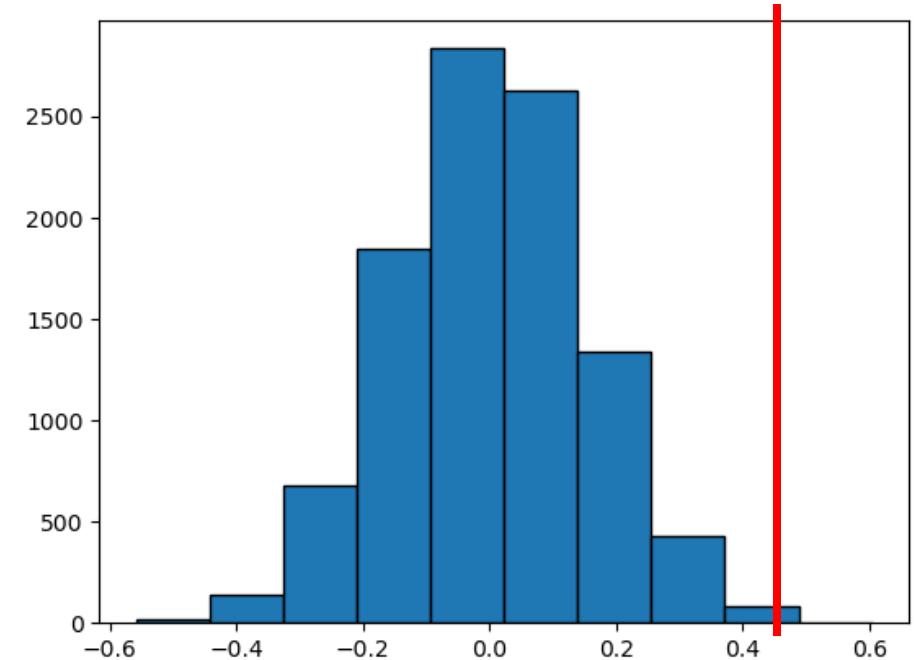
Step 3: Create a null distribution



Step 4: Calculate the p-value

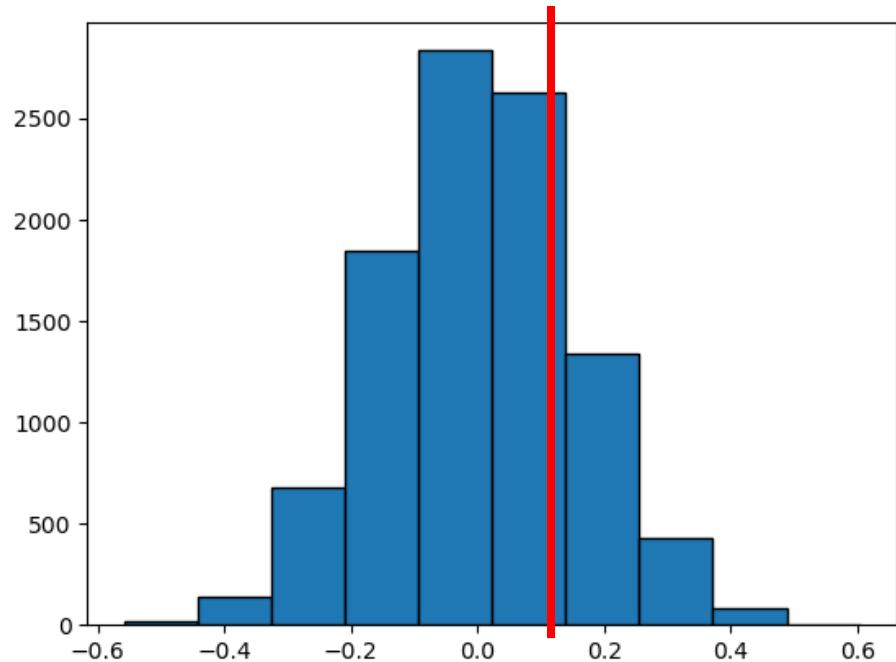


If $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}} = 0.1$ what would the p-value be?

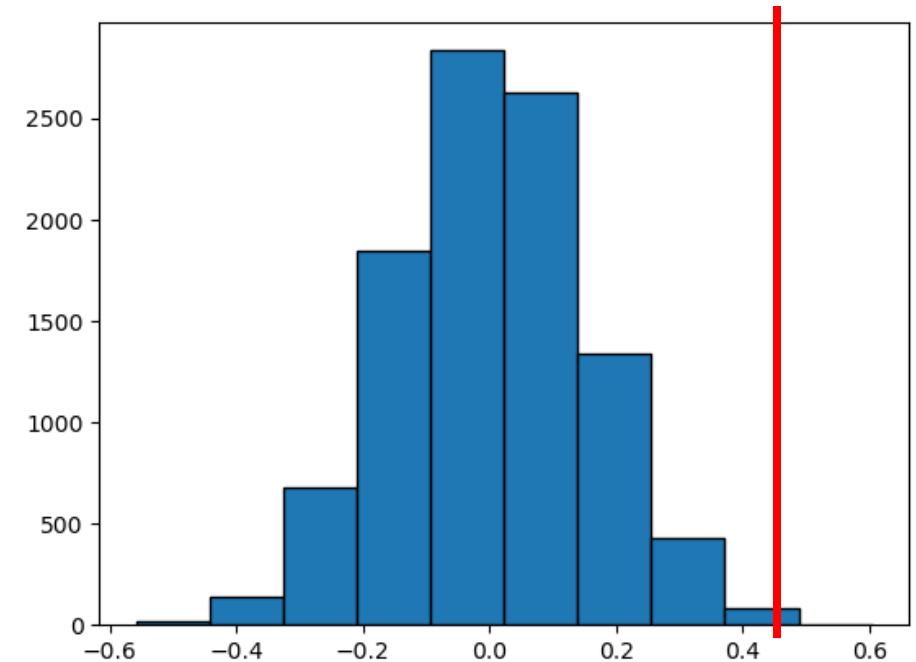


If $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}} = 0.5$ what would the p-value be?

Step 5: Draw a conclusion



If the p-value was 0.19 what would we conclude?



If the p-value was 0.0007 what would we conclude?

Summary: BTA for back pain relief

1. State the null hypothesis and the alternative hypothesis

- BTA does not lead to an increase in pain relief: $H_0: \pi_{\text{treat}} = \pi_{\text{control}}$
- BTA leads to an increase in pain relief: $H_A: \pi_{\text{treat}} > \pi_{\text{control}}$

2. Calculate the observed statistic: $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$

3. Create a null distribution that is consistent with the null hypothesis

- The $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$ statistics we expect if the null hypothesis was true
- i.e., statistics we would expect if there was no difference in pain relief between the two groups

4. Examine how likely the observed statistic is to come from the null distribution

- What is the probability that we would get a $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$ statistic larger than 0.475 if the null hypothesis was true?
- i.e., what is the p-value?

5. Make a judgement

- A small p-value this means that at the proportion of pain relief differed between the two groups
 - i.e., we say our results are 'statistically significant'
- Because our analysis is based on a randomized controlled trial (using random assignment) we can say that BTA causes an increase in pain relief

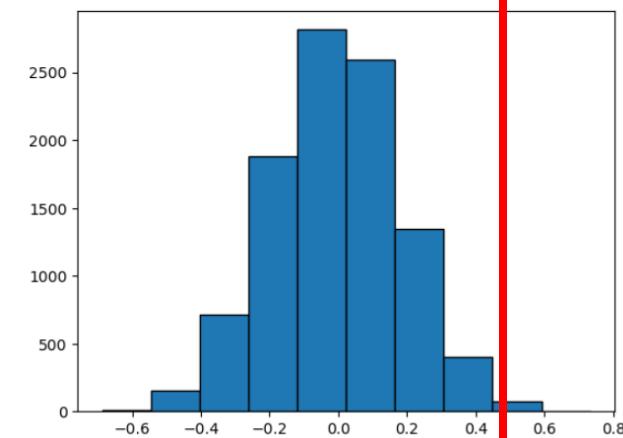
May 22, 2001; 56 (10) ARTICLES

Botulinum toxin A and chronic low back pain
A randomized, double-blind study

Leslie Foster, Larry Clapp, Marleigh Erickson, Bahman Jabbari

First published May 22, 2001, DOI:
<https://doi.org/10.1212/WNL.56.10.1290>

$$\hat{p}_{\text{treat}} - \hat{p}_{\text{control}} = .475$$



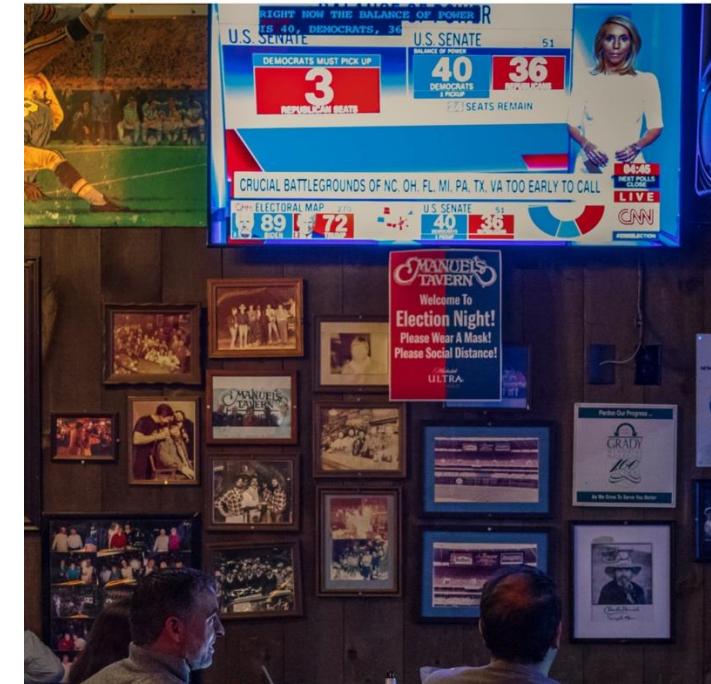


Let's explore this in Jupyter!

The Best Laid Plans to Enjoy (or Ignore) Election Day

Watching the news, attending election parties, singing with friends and getting Botox are among the activities some people have planned.

▶ Listen to this article · 7:03 min [Learn more](#)



Baby birth weights

Question: Is the average weight of babies at birth affected by whether a mother smokes?

To gain insight into this question let's compare:

- A. Birth weights of babies of mothers who smoked during pregnancy
- B. Birth weights of babies of mothers who didn't smoke



Step 1: State the null and alternative hypotheses

Null hypothesis:

- In the population, the distributions of the birth weights of the babies in the two groups are the same.

Alternative hypothesis:

- In the population, the babies of the mothers who didn't smoke were heavier, on average, than the babies of the smokers.

How can we write these hypotheses using symbols we have discussed?

$$H_0: \mu_{\text{non-smoke}} = \mu_{\text{smoke}} \quad \text{or} \quad \mu_{\text{non-smoke}} - \mu_{\text{smoke}} = 0$$

$$H_A: \mu_{\text{non-smoke}} > \mu_{\text{smoke}} \quad \text{or} \quad \mu_{\text{non-smoke}} - \mu_{\text{smoke}} > 0$$

Step 2: Compute the observed statistic

Let's look at a data set from 1236 mother-baby pairs that was collected between 1960 and 1967 among women in the Kaiser Foundation Health Plan in the San Francisco East Bay area

- 742 mothers who did not smoke
- 484 mothers who smoked

Statistic: Difference between average baby weights

- $\bar{x}_{\text{non-smokers}} - \bar{x}_{\text{smoker}}$

Large values of this statistic favor the alternative

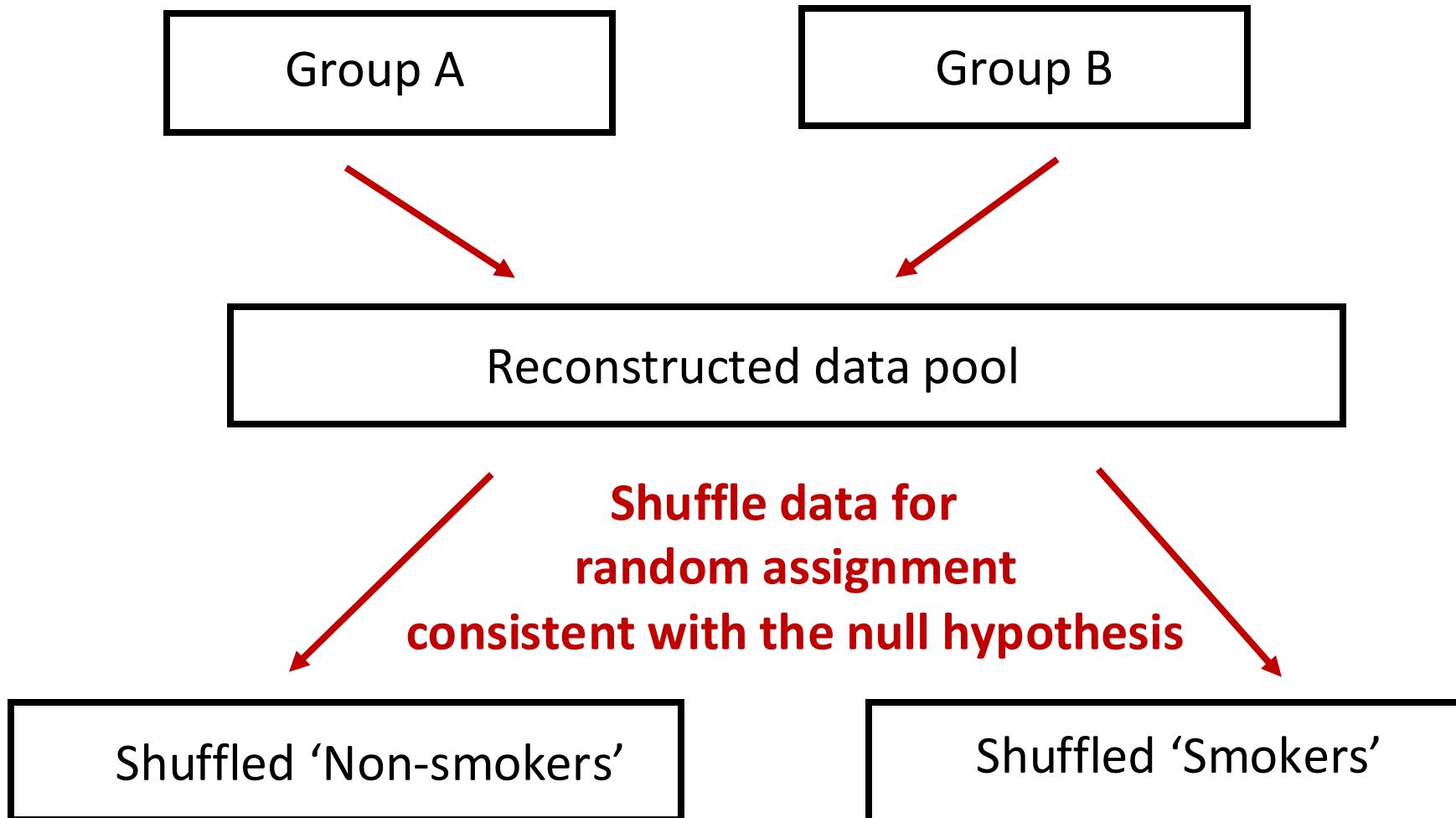
Step 3: Create the null distribution

If the null is true, all rearrangements of the birth weights among the two groups are equally likely

Plan:

- Shuffle all the birth weights
- Assign some to "Group A" and the rest to "Group B", maintaining the two sample sizes
- Find the difference between the averages of the two shuffled groups
- Repeat

Create the null distribution!



One null distribution statistic: $\bar{X}_{\text{shffle-non-smokers}} - \bar{X}_{\text{shuffle -smoker}}$



Let's explore this in Jupyter!

Hypothesis tests for correlation

Hypothesis tests for correlation

Is there a positive correlation between the number of pages in a book and the price of the book?



What is the population parameter and the statistic of interest?

Hypothesis testing for correlation

1. Write down the null and alternative in symbols and words

Null hypothesis:

- There is no correlation between book price and the number of pages

Alternative hypothesis:

- There is a positive correlation between book price and the number of pages

In symbols:

$$H_0: \rho = 0$$

$$H_A: \rho > 0$$

Significance tests for correlation

Let's look at the books from Amazon.com

Title	List.Price	NumPages
1,001 Facts that Will Scare the S#!t Out of You	12.95	304
21: Bringing Down the House	15.00	273
100 Best-Loved Poems	1.50	96
1421: The Year China Discovered America	15.99	672

```
amazon = pd.read_csv("amazon.csv")
```

Try this in Python!

Step 2: What is the observed statistic?

- Also say whether you think you will be able to reject the null hypothesis based on a plot of your data

Step 3: Create the null distribution

- To start with: how we can create one point in the null distribution?
 - Hint: think about shuffling the data

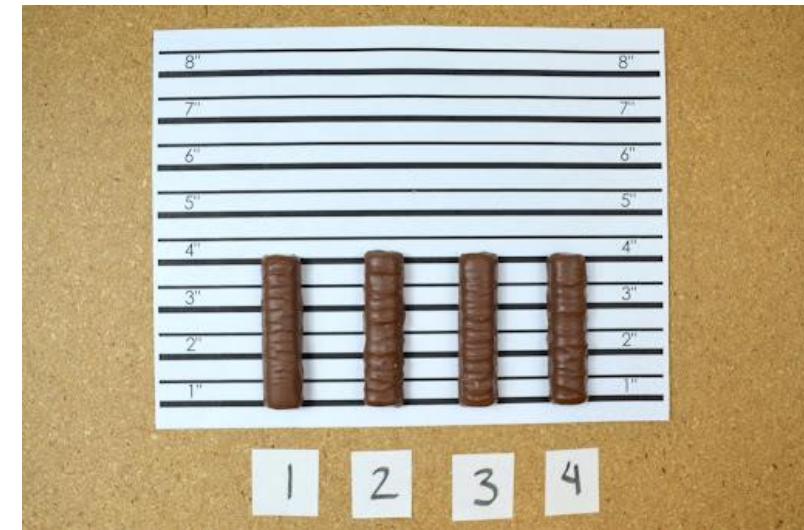
Step 4: What is the p-value that you get?

Step 5: What decision would you make?

Visual hypothesis test

In visual hypothesis tests, we create data visualizations to try to assess whether particular relationships exist in our data.

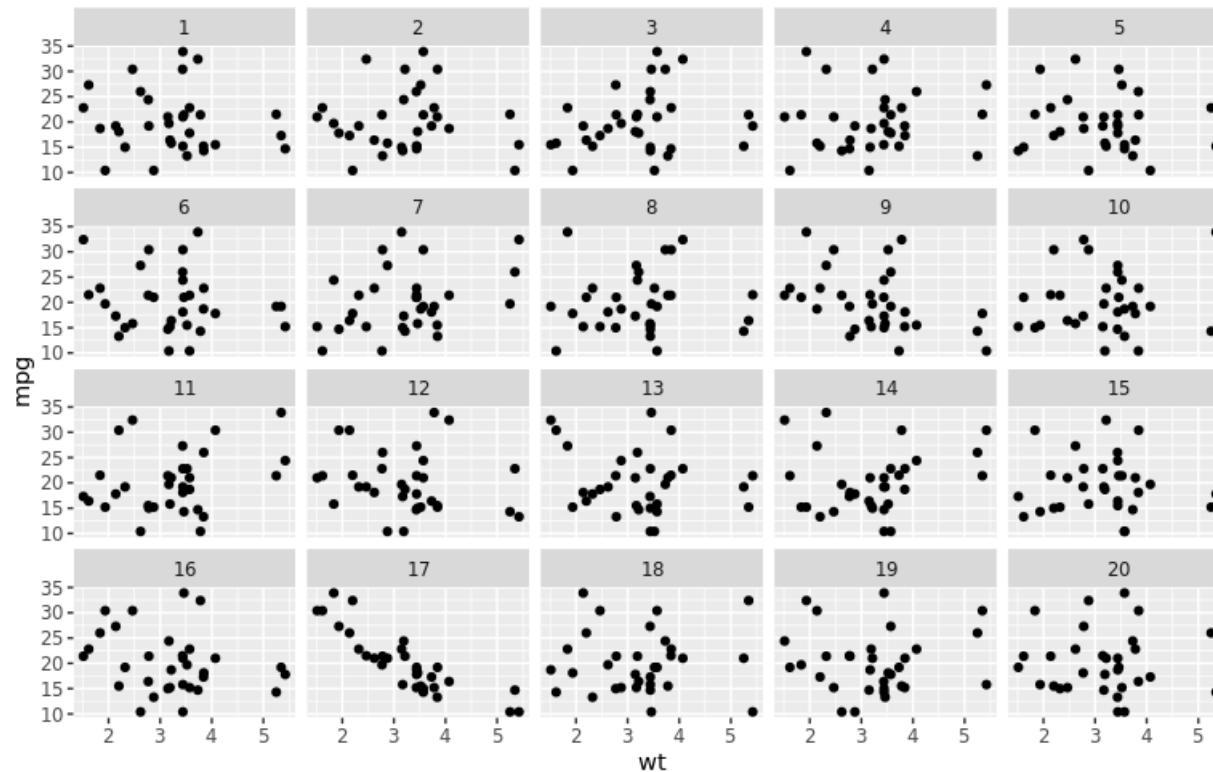
- One way this is done through a visual lineup.



[Which candy bar is 5th Avenue bar?](#)

Visual hypothesis test

Which plot shows the true relationship between a car's weight and the number of miles per gallon a car gets?



Let's try it in Jupyter

Brief mention: two-sided hypothesis tests

Brief mention: two-sided hypothesis tests

So far we have always had a specific prediction for the effect we observed

For example:

- We believed that *less than* 50% of movies passed the Bechdel test
- We believed that babies or mothers who did not smoke would way *more* (on average) than babies of mothers who smoked

This directionality was reflected in our alternative hypotheses

- $H_A: \pi_{\text{Bechdel}} < .5$
- $H_A: \mu_{\text{non-smoke}} > \mu_{\text{smoke}}$

Brief mention: two-sided hypothesis tests

Sometimes we do not know the direction of an effect, we only know that the value specified in the null hypothesis is not correct

For example:

- We just know that 50% of movies do not pass the Bechdel test
 - But it could be than more 50% or less than 50%

We would then write our alternative hypotheses as:

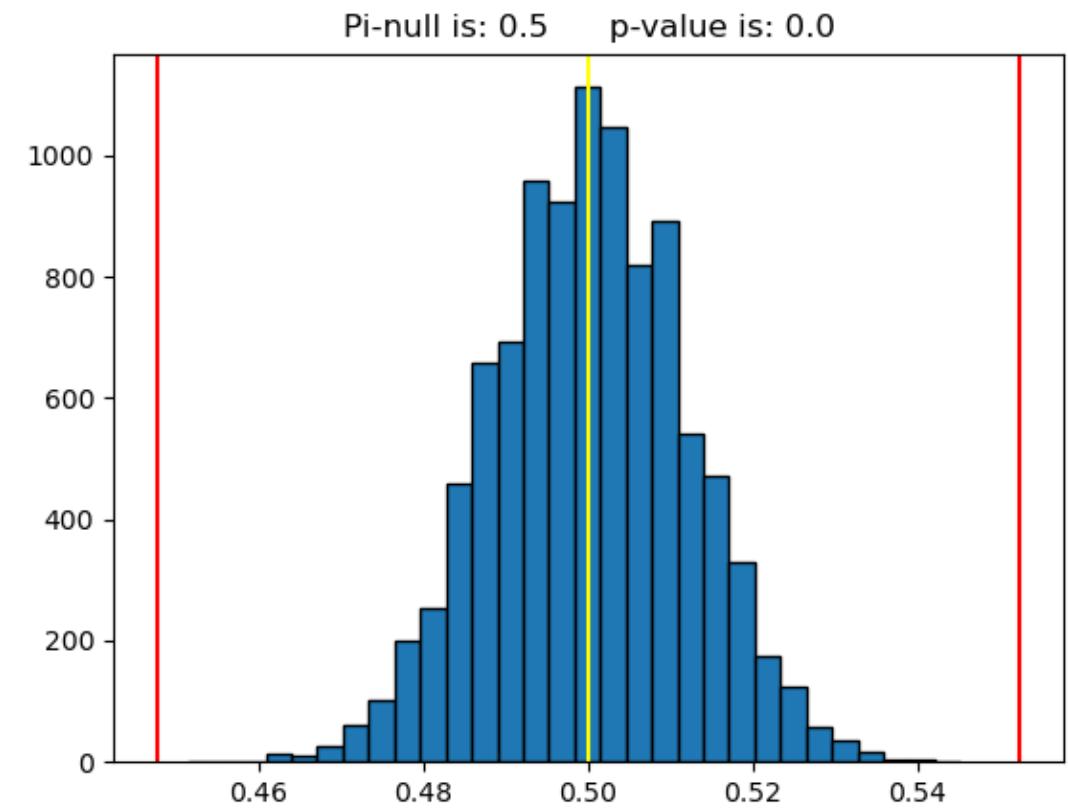
- $H_A: \pi_{\text{Bechdel}} \neq .5$
- $H_A: \mu_{\text{non-smoke}} \neq \mu_{\text{smoke}}$

Brief mention: two-sided hypothesis tests

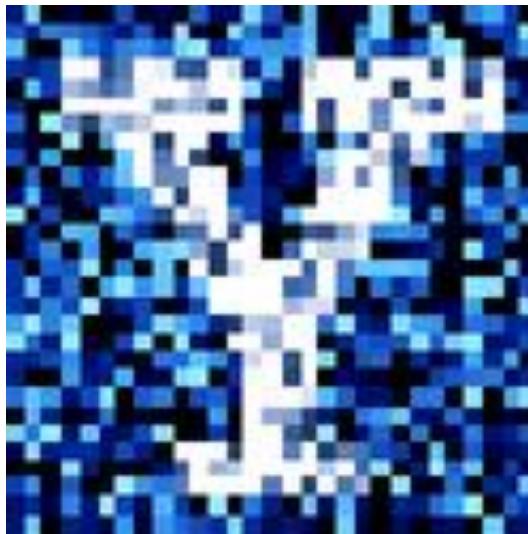
When we have a “two-sided” alternative hypothesis, we need to calculate the the statistics that are “more extreme” than the observed statistic to get the p-value

- i.e., we need to look at both tails of our null distribution to get the p-value

Let's explore this in Jupyter!



YData: Introduction to Data Science



Class 20: Confidence intervals

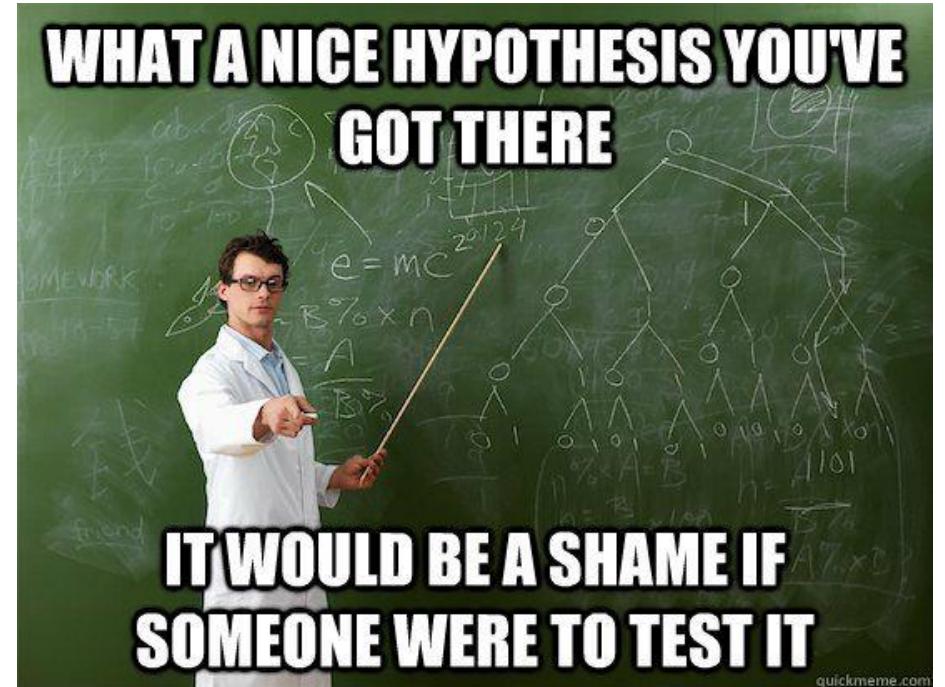
Overview

Review/continuation of hypothesis tests
for correlation

Visual hypothesis tests

Two-sided hypothesis tests

Confidence intervals



Reminder: keep working on your class project

Homework 8 is due on **Sunday November 10th**

A **polished** draft of the project is due on **November 17th**

Review of Statistical Inference

Review: Statistical Inference

Statistical Inference: Making conclusions about a population based on data in a random sample

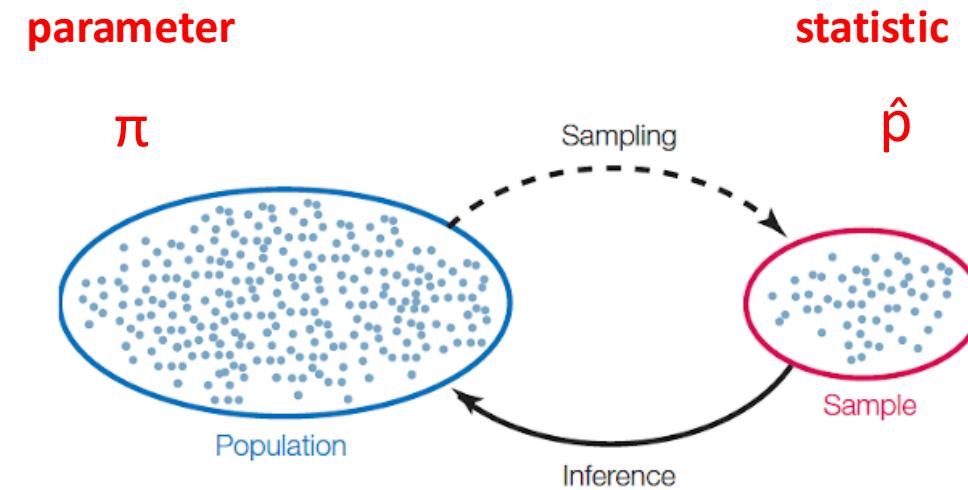
A **parameter** is number associated with the population

- We use Greek symbols to denote parameters

A **statistic** is number calculated from the sample

- We use Latin symbols to denote statistics

A statistic can be used as an estimate of a parameter



	Sample Statistic	Population Parameter
Mean	\bar{x}	μ
Proportion	\hat{p}	π
Correlation	r	ρ

Hypothesis tests

Null and Alternative hypotheses

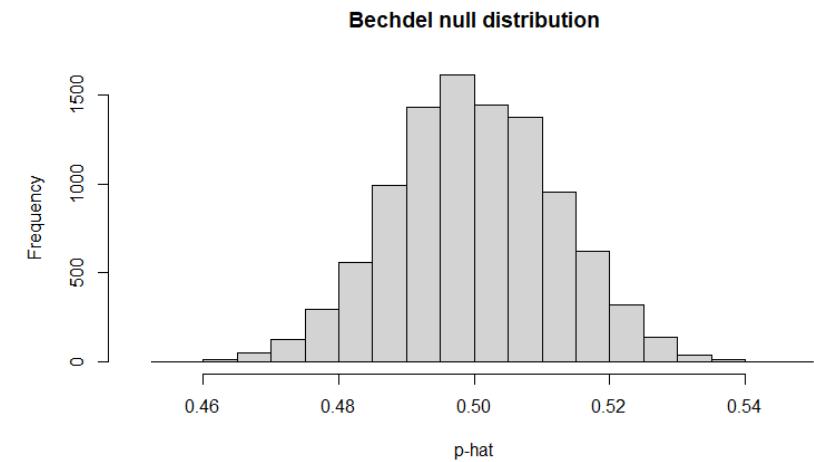
Null hypothesis

- A hypothesis where “nothing interesting” happened
 - E.g., our experiment failed
 - E.g., $H_0: \pi = 0.5$
- We can simulate data under the assumptions of this model to get a "null distribution" of statistics

Alternative hypothesis

- The hypothesis we believe in (would like to see true)
- E.g., $H_A: \pi < 0.5$

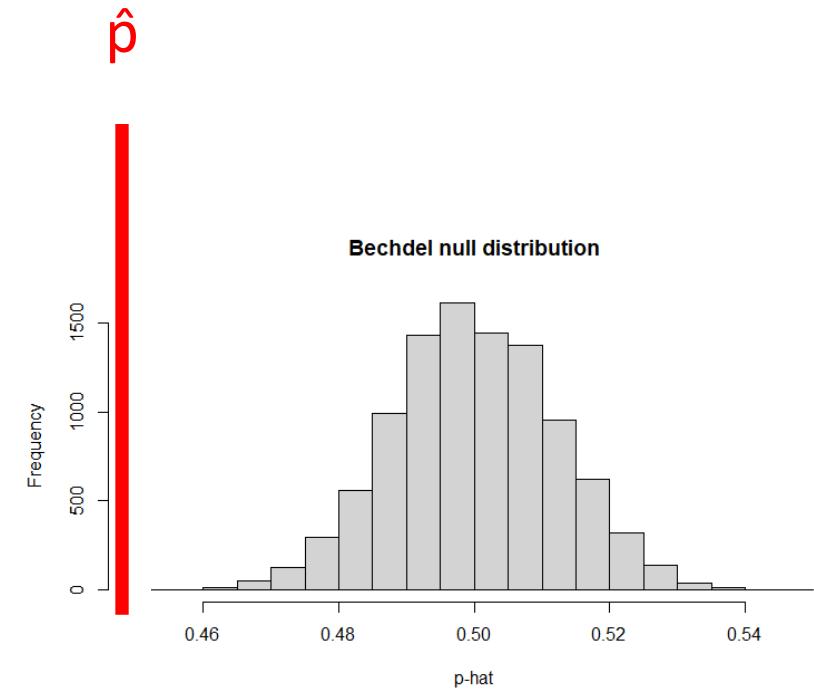
A **test statistic** is the statistic we choose to simulate in order decide between the two hypotheses



Testing the null hypothesis

To resolve choice between null and alternative hypotheses:

- We compare the **observed test statistic** to the statistic values in the null distribution
- If the observed statistic is not consistent with the null distribution, then we can **reject the null hypothesis**
 - E.g., $H_0: \hat{p} \geq 0.5$
 - And we accept the alternative hypothesis
 - E.g., $H_A: \hat{p} < 0.5$



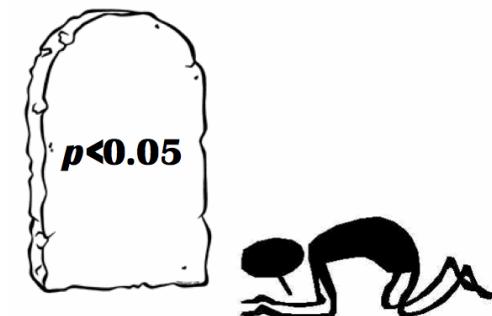
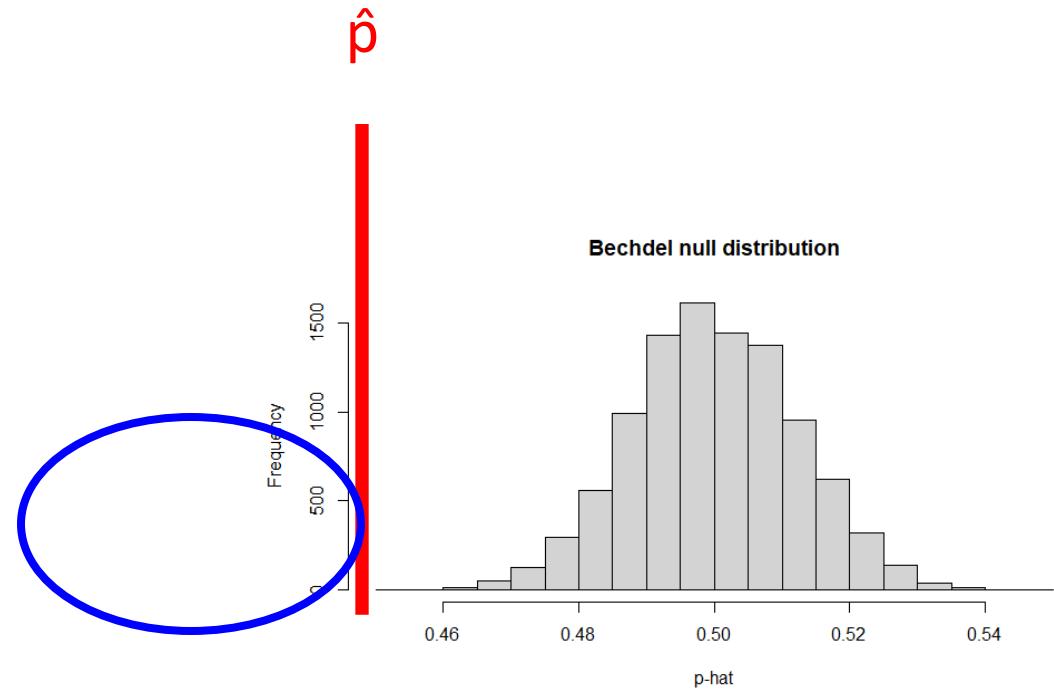
The p-value

The **p-value** is the probability, that we get a statistic as or more extreme than the observed statistic from the null distribution

- $P(\text{Null_Stat} \leq \text{obs_stat} | H_0)$

If the P-value is small, this is evidence against the null hypothesis and the results are often called "statistically significant"

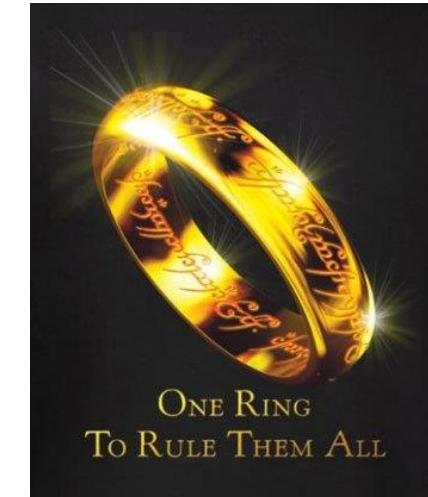
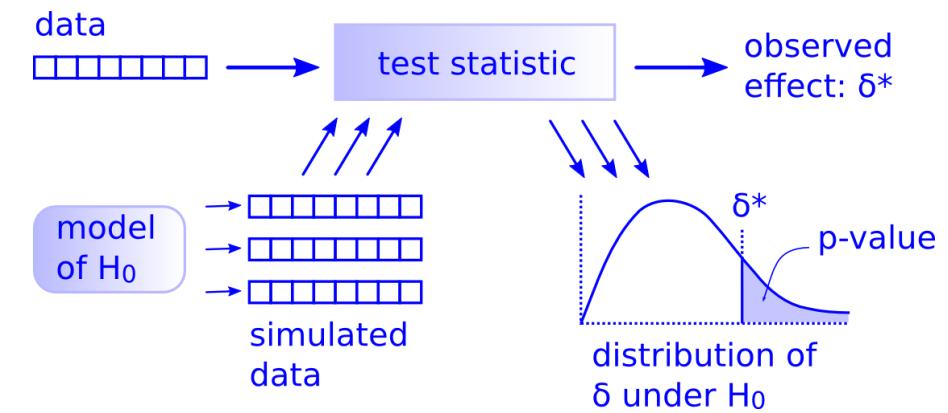
- Convention, p-value < 0.05



Steps needed to run a hypothesis test

To run a hypothesis test, we can use 5 steps:

1. State the null and alternative hypothesis
2. Calculate the observed statistic of interest
3. Create the null distribution
4. Calculate the p-value
5. Make a decision



Summary: BTA for back pain relief

1. State the null hypothesis and the alternative hypothesis

- BTA does not lead to an increase in pain relief: $H_0: \pi_{\text{treat}} = \pi_{\text{control}}$
- BTA leads to an increase in pain relief: $H_A: \pi_{\text{treat}} > \pi_{\text{control}}$

2. Calculate the observed statistic: $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$

3. Create a null distribution that is consistent with the null hypothesis

- The $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$ statistics we expect if the null hypothesis was true
- i.e., statistics we would expect if there was no difference in pain relief between the two groups

4. Examine how likely the observed statistic is to come from the null distribution

- What is the probability that we would get a $\hat{p}_{\text{treat}} - \hat{p}_{\text{control}}$ statistic larger than 0.475 if the null hypothesis was true?
- i.e., what is the p-value?

5. Make a judgement

- A small p-value this means that at the proportion of pain relief differed between the two groups
 - i.e., we say our results are 'statistically significant'
- Because our analysis is based on a randomized controlled trial (using random assignment) we can say that BTA causes an increase in pain relief

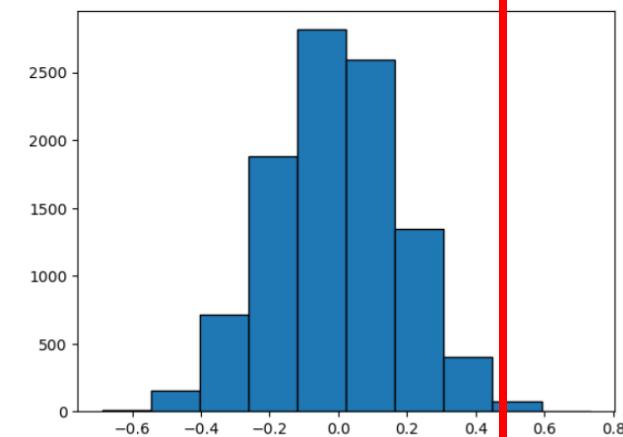
May 22, 2001; 56 (10) ARTICLES

Botulinum toxin A and chronic low back pain
A randomized, double-blind study

Leslie Foster, Larry Clapp, Marleigh Erickson, Bahman Jabbari

First published May 22, 2001, DOI:
<https://doi.org/10.1212/WNL.56.10.1290>

$$\hat{p}_{\text{treat}} - \hat{p}_{\text{control}} = .475$$



Hypothesis tests for correlation

Hypothesis tests for correlation

Is there a positive correlation between the number of pages in a book and the price of the book?



What is the population parameter and the statistic of interest?

Hypothesis testing for correlation

1. Write down the null and alternative in symbols and words

Null hypothesis:

- There is no correlation between book price and the number of pages

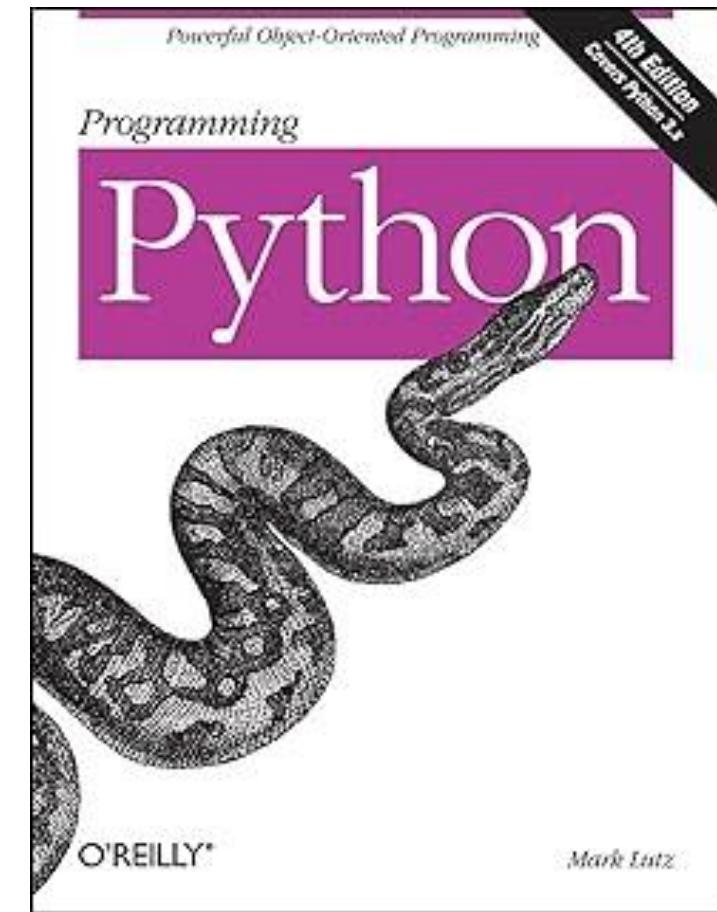
Alternative hypothesis:

- There is a positive correlation between book price and the number of pages

In symbols:

$$H_0: \rho = 0$$

$$H_A: \rho > 0$$



Has 1626 pages

Significance tests for correlation

Let's look at the books from Amazon.com

Title	List.Price	NumPages
1,001 Facts that Will Scare the S#!t Out of You	12.95	304
21: Bringing Down the House	15.00	273
100 Best-Loved Poems	1.50	96
1421: The Year China Discovered America	15.99	672

```
amazon = pd.read_csv("amazon.csv")
```

Try this in Python!

Step 2: What is the observed statistic?

- Also say whether you think you will be able to reject the null hypothesis based on a plot of your data

Step 3: Create the null distribution

- To start with: how we can create one point in the null distribution?
 - Hint: think about shuffling the data

Step 4: What is the p-value that you get?

Has 1012 pages

Step 5: What decision would you make?



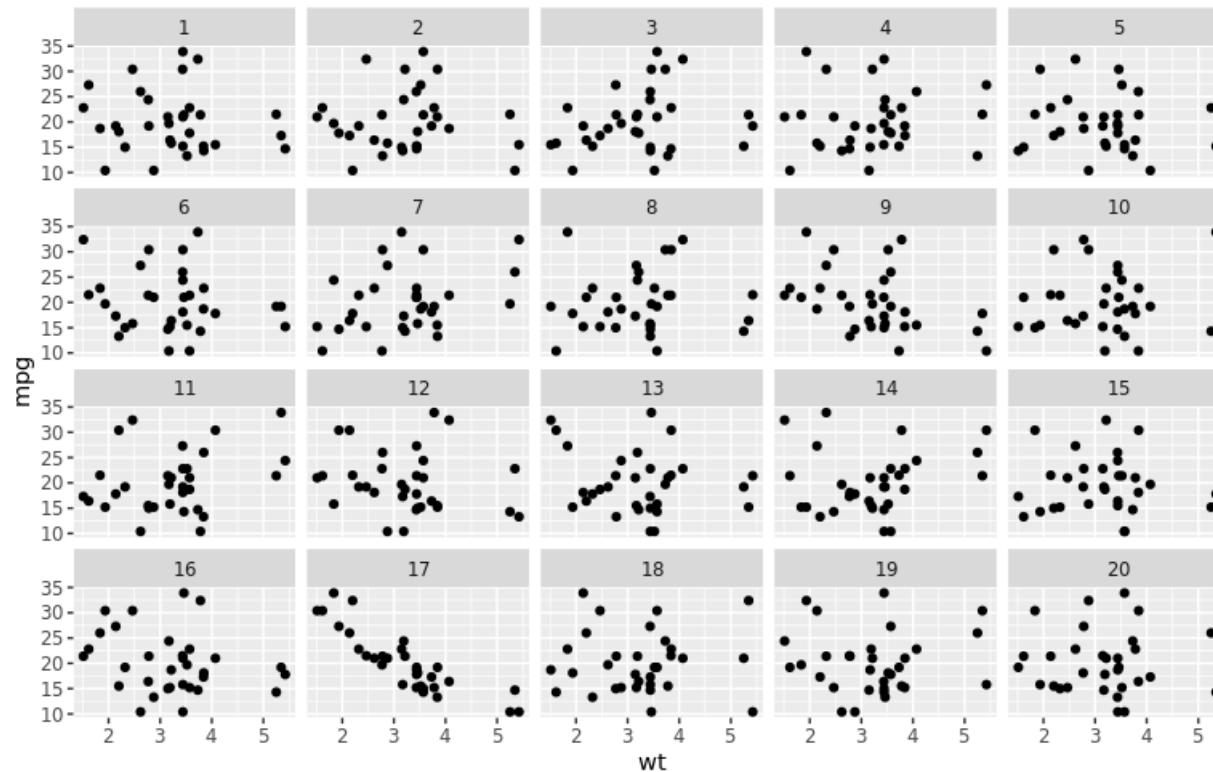
Visual hypothesis test

In visual hypothesis tests, we create data visualizations to try to assess whether particular relationships exist in our data.

One way this is done through a visual lineup

Visual hypothesis test

Which plot shows the true relationship between a car's weight and the number of miles per gallon a car gets?



Let's try it in Jupyter

Brief mention: two-sided hypothesis tests

Brief mention: two-sided hypothesis tests

So far we have always had a specific prediction for the effect we observed

For example:

- We believed that *less than* 50% of movies passed the Bechdel test
- We believed that babies or mothers who did not smoke would weigh *more* (on average) than babies of mothers who smoked

This directionality was reflected in our alternative hypotheses

- $H_A: \pi_{\text{Bechdel}} < .5$
- $H_A: \mu_{\text{non-smoke}} > \mu_{\text{smoke}}$

Brief mention: two-sided hypothesis tests

Sometimes we do not know the direction of an effect, we only know that the value specified in the null hypothesis is not correct

For example:

- We just know that 50% of movies do not pass the Bechdel test
 - But it could be than more 50% or less than 50%

We would then write our alternative hypotheses as:

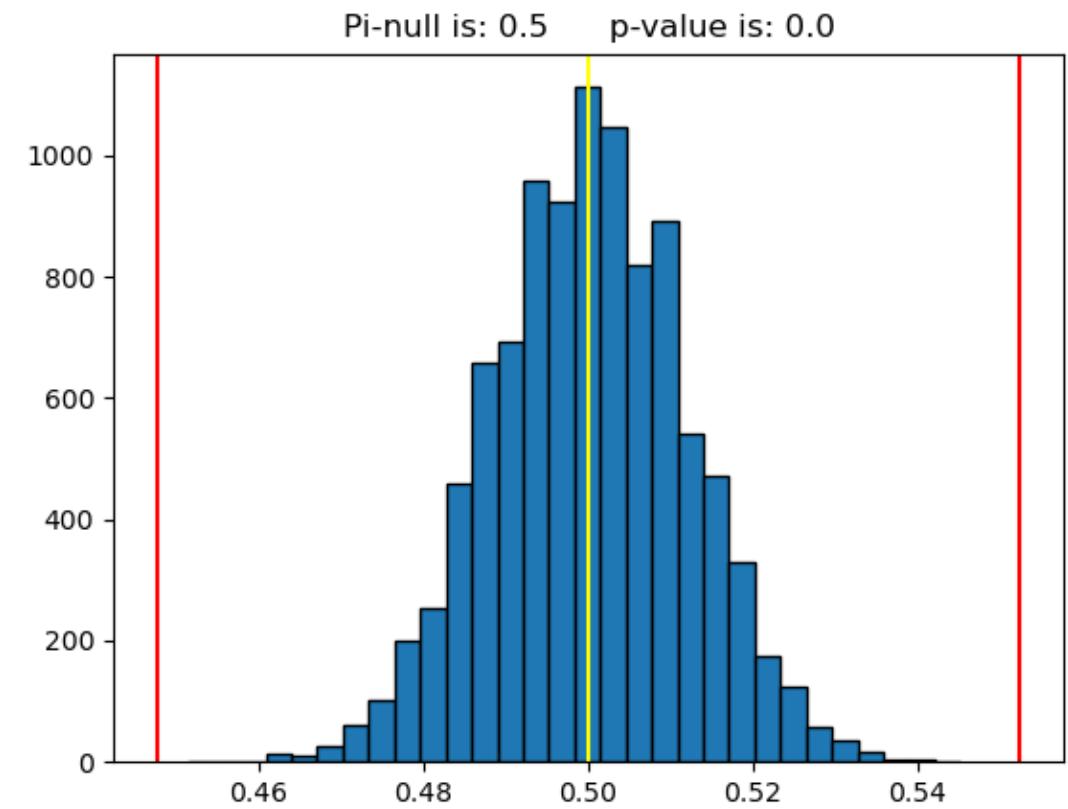
- $H_A: \pi_{\text{Bechdel}} \neq .5$
- $H_A: \mu_{\text{non-smoke}} \neq \mu_{\text{smoke}}$

Brief mention: two-sided hypothesis tests

When we have a “two-sided” alternative hypothesis, we need to calculate the the statistics that are “more extreme” than the observed statistic to get the p-value

- i.e., we need to look at both tails of our null distribution to get the p-value

Let's explore this in Jupyter!



Confidence intervals

Interval estimate based on a margin of error

Null hypothesis tests tell us if a particular parameter value is **implausible**

- E.g., in the Bechdel data we rejected $\pi = .5$

An **interval estimate** give a range of **plausible** values for a population parameter

Example: 42% of American approve of Biden's job performance, plus or minus 3%

How do we interpret this?

Says that the population parameter π lies somewhere between 39% to 45%

- i.e., if they sampled all Americans the true population proportion would be likely be in this range

Confidence Intervals

A **confidence interval** is an interval computed by a method that will contain the *parameter* a specified percent of times

- i.e., if the estimation were repeated many times, the interval will have the parameter x% of the time

The **confidence level** is the percent of all intervals that contain the parameter

Think ring toss...

Parameter exists in the ideal world

We toss intervals at it

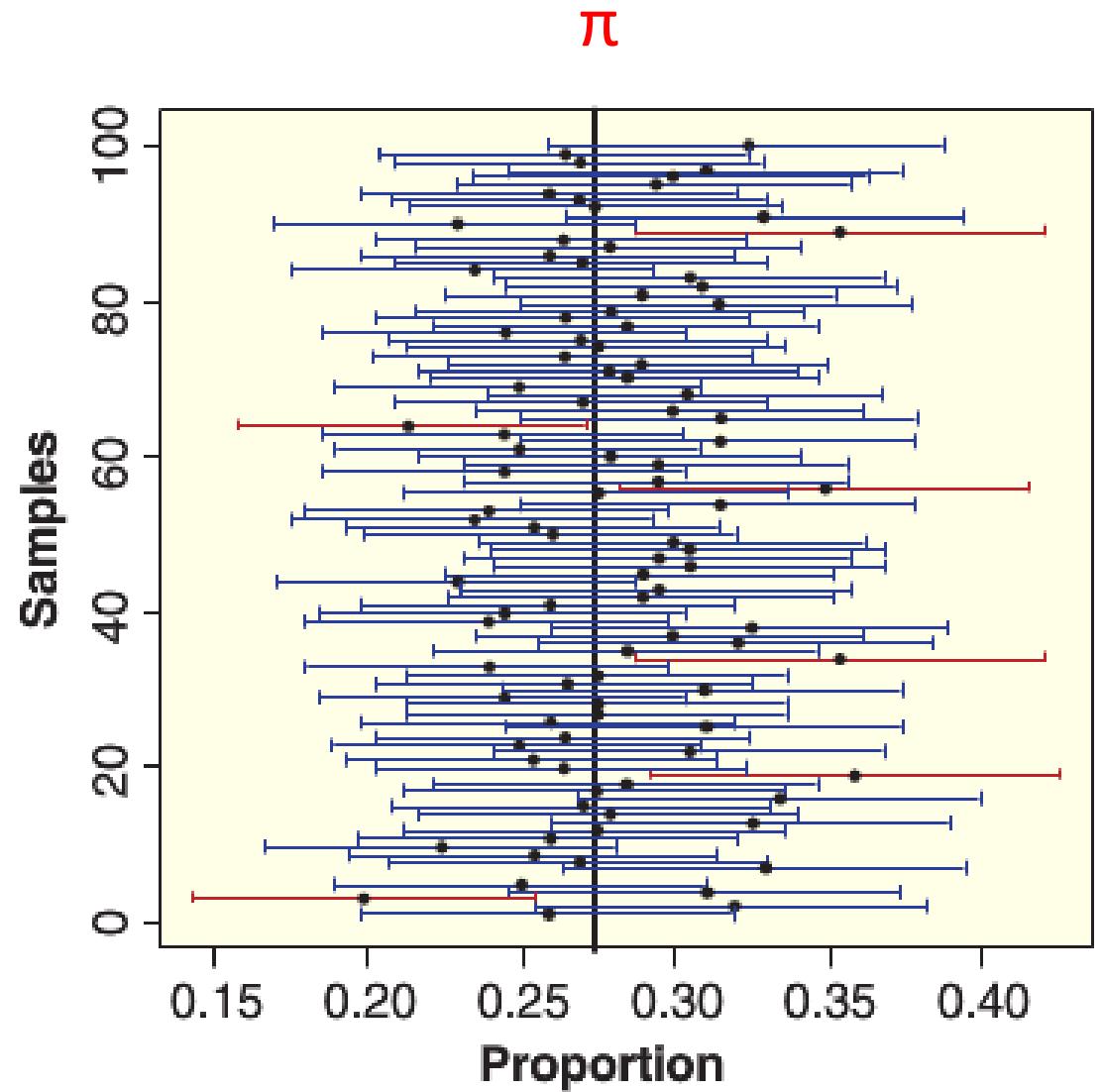
95% of those intervals capture the parameter



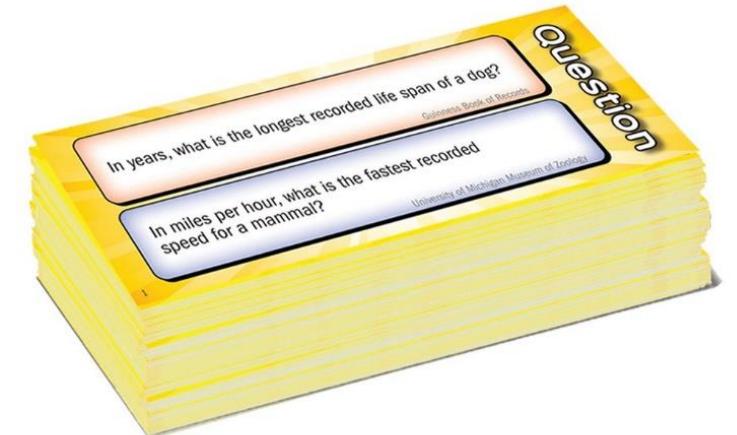
Confidence Intervals

For a **confidence level** of 95%...

95% of the **confidence intervals** will have the parameter in them



Wits and Wagers: 90% confidence interval estimator



I will ask 10 questions that have numeric answers

Please come up with a range of values that contains the true value in it
for 9 out of the 10 questions

- i.e., be a 90% confidence interval estimator

Wits and Wagers...

Question 1: What is the diameter of the moon (in miles)?

Question 2: How many years passed between the first NBA game and the first WNBA game?

Question 3: What percent of U.S. land area does Alaska make up?

Wits and Wagers...

Question 4: On average, how many baseballs are used in a Major League Baseball season?

Question 5: How many rooms are there in the White House?

Question 6: How many votes were cast in the 2012 U.S. presidential election?

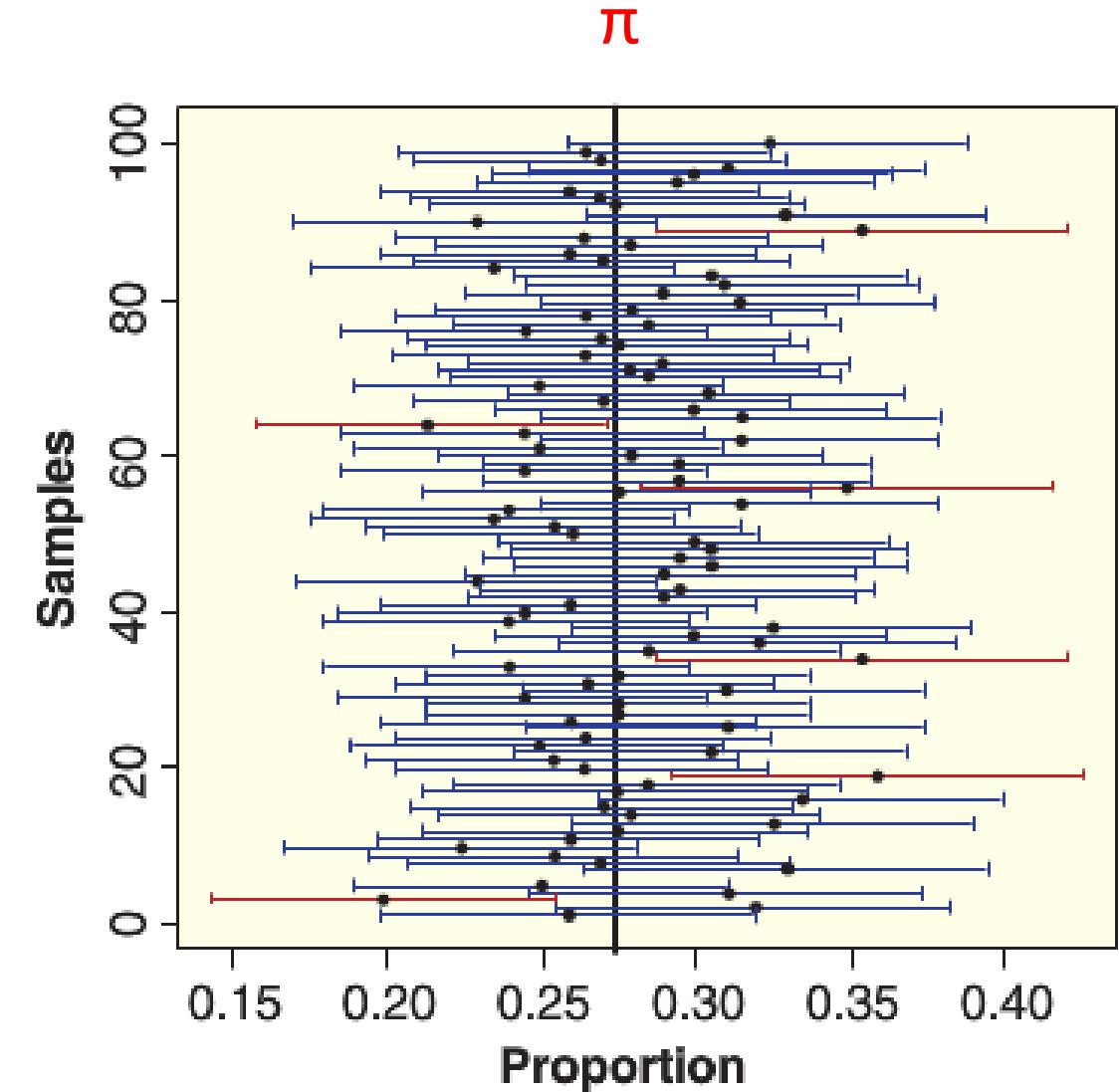
Question 7: Out of the 538 electoral votes, how many did Ronald Reagan receive in the 1984 presidential election ?

Wits and Wagers...

Question 8: How many cases of human spontaneous combustion appeared in medical journals between the years of 1600 and 1900?

Question 9: How many Academy Award nominations did *The Lord of the Rings* movie trilogy receive?

Question 10: In feet, how long was the largest whale ever recorded?



We all have 9 out of 10 correct?!

Note

For any given confidence interval we compute, we don't know whether it has really captured the parameter

But we do know that if we do this 100 times, 90 of these intervals will have the parameter in it

(for a 90% confidence interval)

Tradeoff between interval size and confidence level



There is a tradeoff between the **confidence level** (percent of times we capture the parameter) and the **confidence interval size**

Using hypothesis tests to
construct confidence intervals

Constructing confidence intervals

There are several methods that can be used to construct confidence intervals including

- “Parametric methods” that use probability functions
 - E.g., confidence intervals based on the normal distribution
- A “bootstrap method” where data is resampled from our original sample to approximate a sampling distribution

To learn more about these methods, take Introductory Statistics!

Constructing confidence intervals

We are going to use a less conventional method to get confidence intervals based on the relationship between confidence intervals and hypothesis tests

- The method we will discuss is valid, but can be more computationally expensive than other methods

What we will do is to run a series of hypothesis test with different null hypothesis parameter values

Our confidence interval will be all parameters values where we **fail to reject** the null hypothesis

$$H_0: \pi = \pi_0$$



Failure to reject $\pi = \pi_0$
means π_0 is plausible

Motivation: Bechdel Confidence Interval

From running a hypothesis test on the Bechdel data, we saw that $H_0: \pi = .5$ is unlikely

- i.e., it was not plausible that 50% of movies pass the Bechdel test

But what is a reasonable range of values for the population proportion of movies that pass the Bechdel test?

We can create a confidence interval for π_{Bechdel} to find out!



Very quick review of using hypothesis tests to construct confidence intervals

All parameter values where we fail to reject the null hypothesis make up the confidence interval

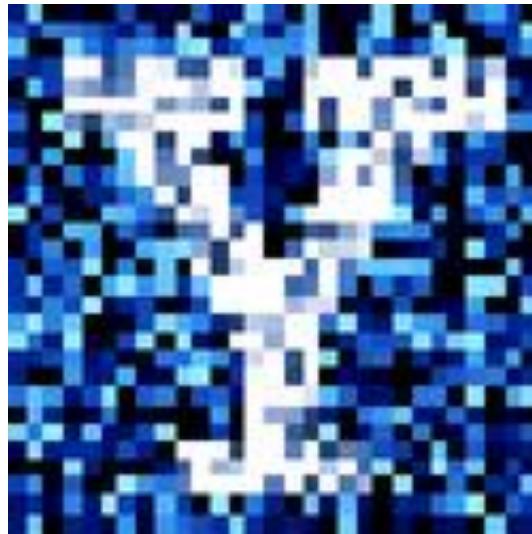
- Using a threshold of p-value < 0.05 yields a 95% CI
- Using a threshold of p-value < 0.01 yields a 99% CI



Let's explore this in Jupyter!

π	p-values
0.4	0
0.41	0.0013
0.42	0.0179
0.43	0.1361
0.44	0.5269
0.45	0.85
0.46	0.296
0.47	0.0614
0.48	0.0067
0.49	0.0004

YData: Introduction to Data Science



Lecture 21: Introduction to machine learning

Overview

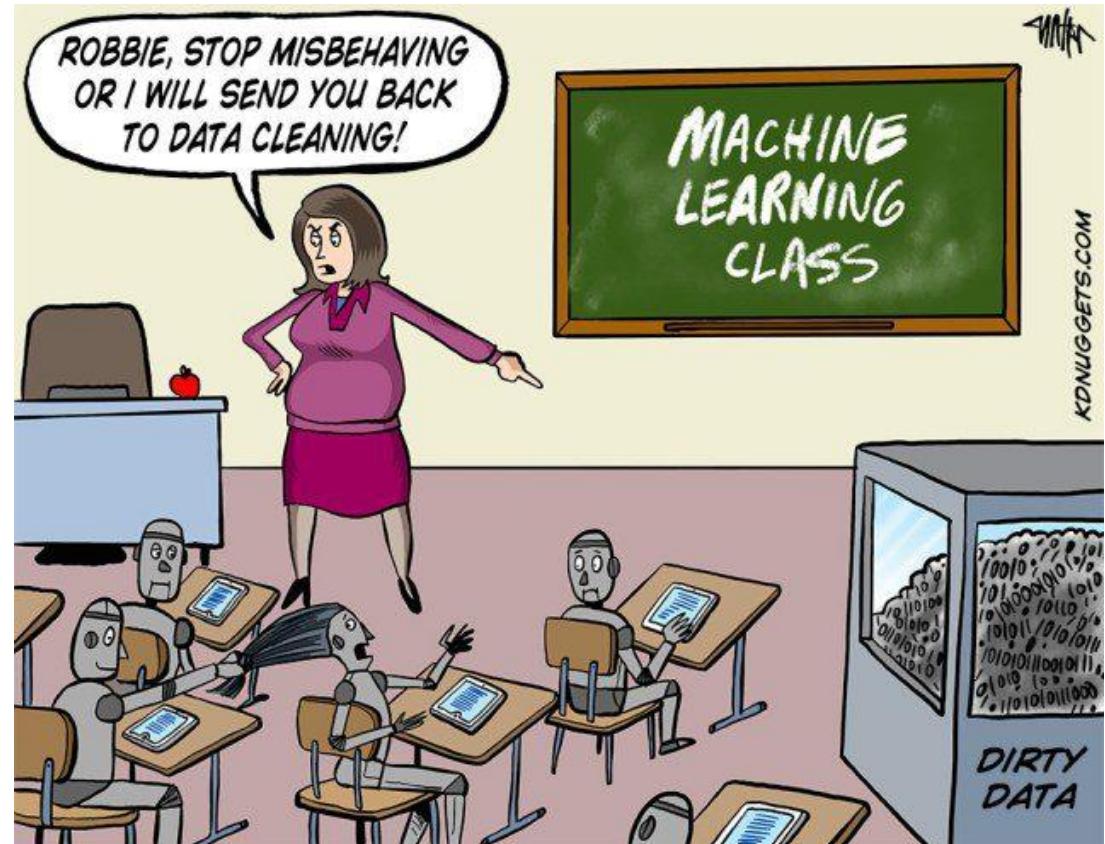
Quick review and continuation of creating confidence intervals using hypothesis tests

Classification

- KNN classifier
- Overfitting

If there is time:

- Other classifiers
- Project time!



Project timeline

Sunday, November 17th

- Projects are due on Gradescope at 11pm
- Email a pdf of your project to your peer reviewers
 - A list of whose paper you will review will be posted to Canvas
 - Fill out the draft reflection on Canvas

Sunday, November 24th

- Jupyter notebook files with your reviews need to be sent to the authors
- A template for doing your review has been posted

Sunday, December 8th

- Project is due on Gradescope
 - Add peer reviews to the Appendix of your project

Homework 9 has been posted

- It is due December 1st

That went as planned,
said no project ever.



My office hours tomorrow (11/13)
will be from 3:30-4:30pm

- (i.e., not from 2-3pm)

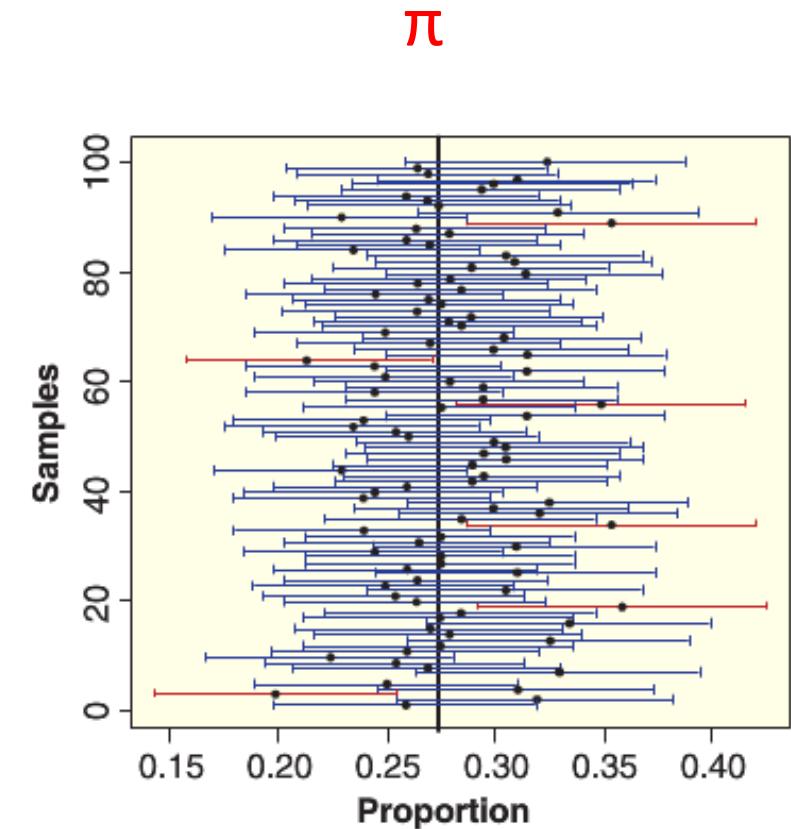
Quick review: confidence Intervals



A **confidence interval** is an interval computed by a method that will contain the *parameter* a specified percent of times

- i.e., if the estimation were repeated many times, the interval will have the parameter x% of the time

The **confidence level** is the percent of all intervals that contain the parameter

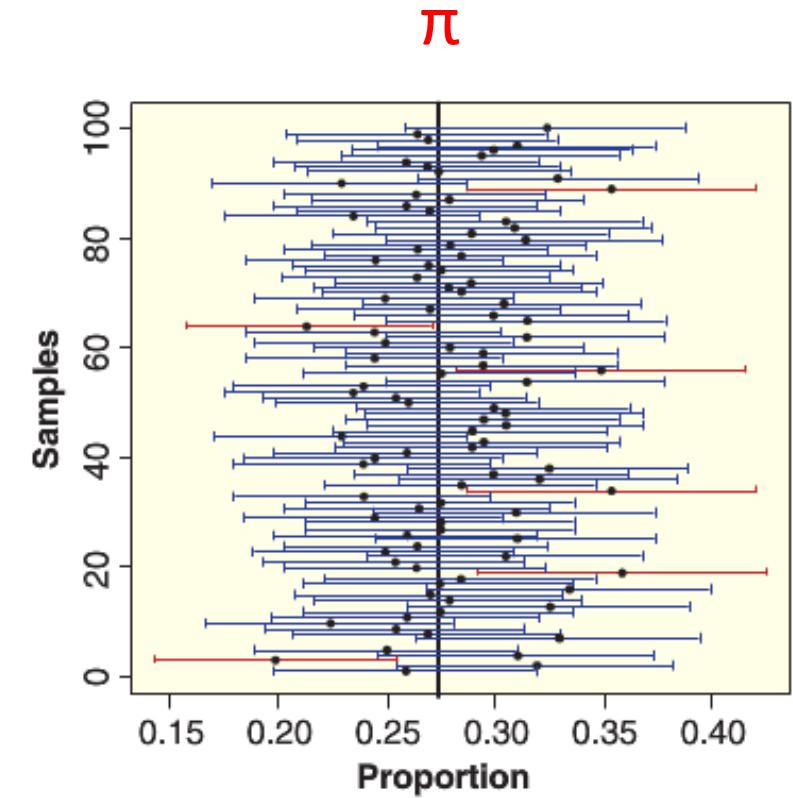
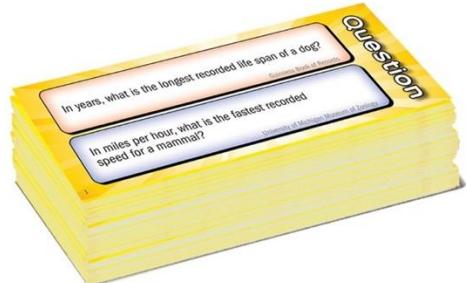


90% Confidence Intervals

For any given confidence interval, we don't know if it has the parameter in it

We just know that it will be in the interval most of the time

- E.g., 9 out of 10 times for a 90% confidence level



Tradeoff between interval size and confidence level



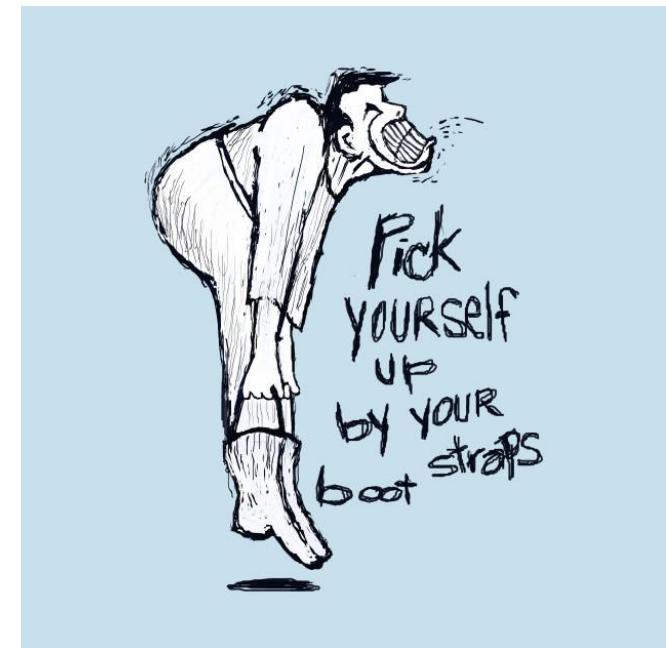
There is a tradeoff between the **confidence level** (percent of times we capture the parameter) and the **confidence interval size**

Review: using hypothesis tests
to construct confidence intervals

Constructing confidence intervals

There are several methods that can be used to construct confidence intervals including

- “Parametric methods” that use probability functions
 - E.g., confidence intervals based on the normal distribution
- A “bootstrap method” where data is resampled from our original sample to approximate a sampling distribution



To learn more about these methods, take Introductory Statistics!

Constructing confidence intervals

We are going to use a less conventional method to get confidence intervals based on the relationship between confidence intervals and hypothesis tests

- The method we will discuss is valid, but can be more computationally expensive than other methods

What we will do is to run a series of hypothesis test with different null hypothesis parameter values

Our confidence interval will be all parameters values where we **fail to reject** the null hypothesis

$$H_0: \pi = \pi_0$$



Failure to reject $\pi = \pi_0$
means π_0 is plausible

Motivation: Bechdel Confidence Interval

From running a hypothesis test on the Bechdel data, we saw that $H_0: \pi = .5$ is unlikely

- i.e., it was not plausible that 50% of movies pass the Bechdel test

But what is a reasonable range of values for the population proportion of movies that pass the Bechdel test?

We can create a confidence interval for π_{Bechdel} to find out!



Very quick review of using hypothesis tests to construct confidence intervals

All parameter values where we fail to reject the null hypothesis make up the confidence interval

- Using a threshold of p-value < 0.05 yields a 95% CI
- Using a threshold of p-value < 0.01 yields a 99% CI



This value is in a 99% confidence interval

π	p-values
0.4	0
0.41	0.0013
0.42	0.0179
0.43	0.1361
0.44	0.5269
0.45	0.85
0.46	0.296
0.47	0.0614
0.48	0.0067
0.49	0.0004
0.5	0

Let's quickly explore this in Jupyter!

Classification

Prediction: regression and classification

We “learn” a function f

- $f(\mathbf{x}) \longrightarrow y$

Input: \mathbf{x} is a data vector of "features"

Output:

- Regression: output is a real number ($y \in \mathbb{R}$)
- Classification: output is a categorical variable y_k

Example: salmon or sea bass?



What are the labels and features in this task?

From Duda, Hart and Stork, 2001

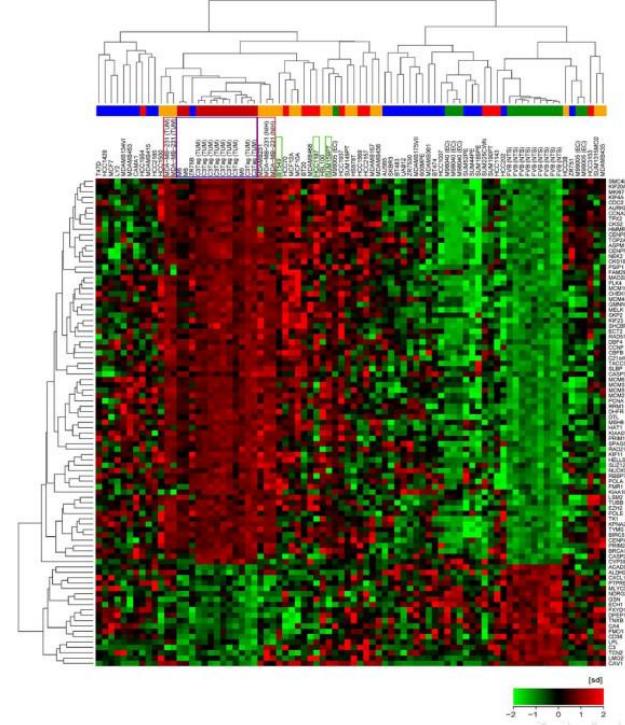
Example: what is in this image?



0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9

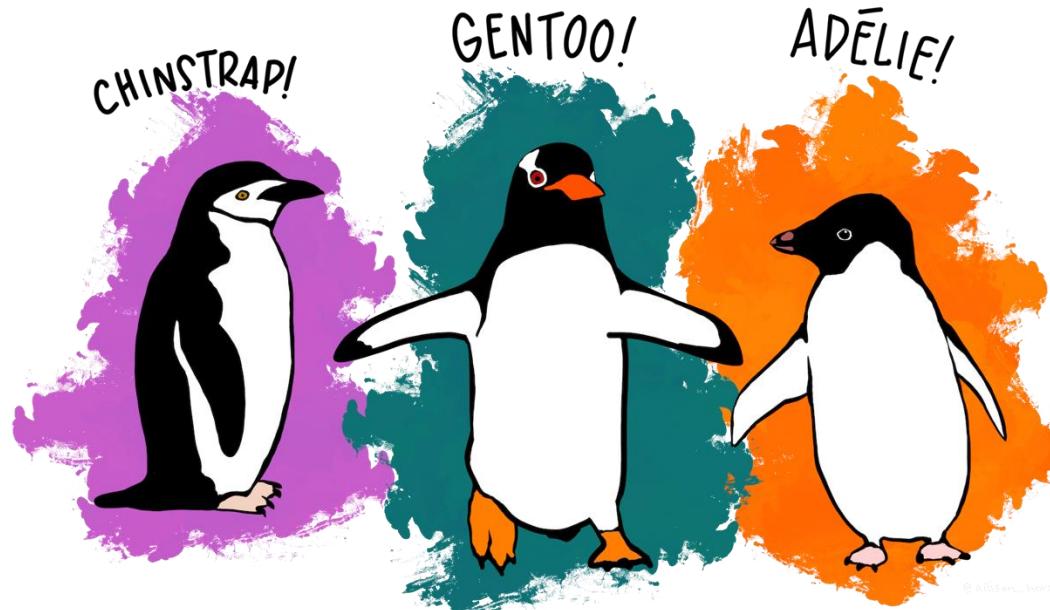
What are the labels and features in this task?

Example: predicting cancer



What are the labels and features in this task?

Example: Penguin species



What are the labels and features in this task?

Example: Chat-GPT predicting/generating text

Question answering:

Are we living in a simulation?

What are the labels and features in this task?

Let's explore features and labels in Jupyter!

k-Nearest Neighbor classifier

Classifiers

Classifiers take a list/array of features X, and return a predicted label y



There are many different types of classifiers

- Decision Trees, Support Vector Machines, Logistic regression, Neural Networks, etc.

The classification process always involves two steps:

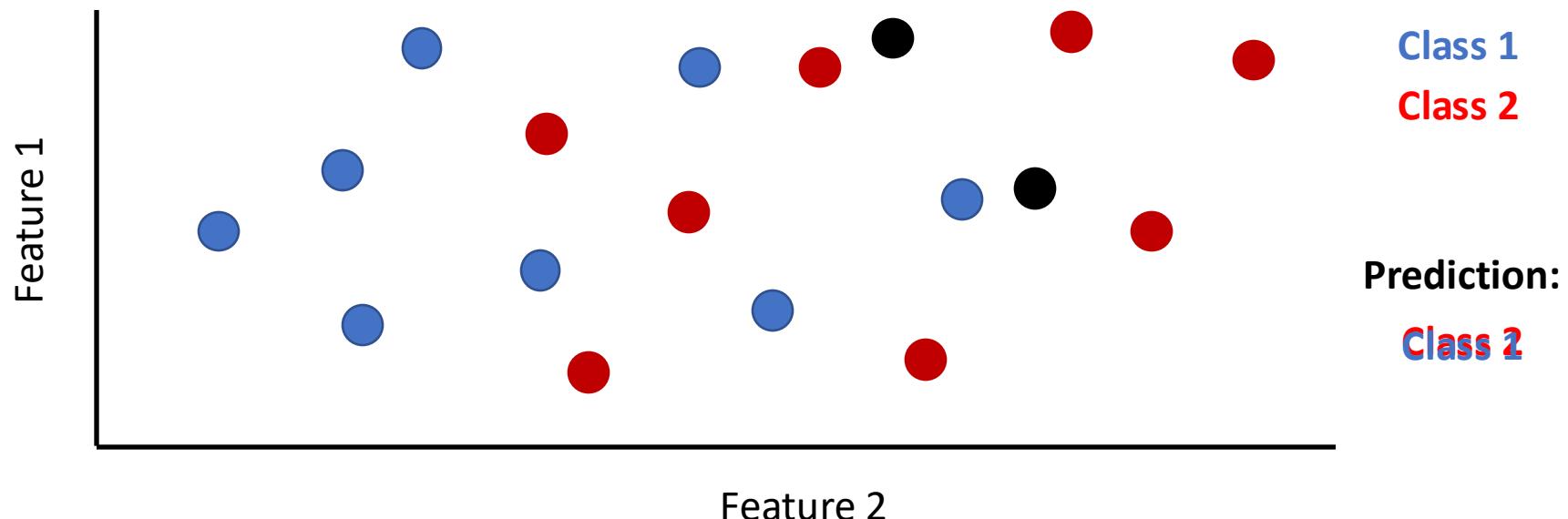
1. **Training** the classifier to learn the relationship between features (X) and labels (y) (from many examples)
 - This is done using the `.fit()` method in scikit-learn
2. **Using** the classifier to predict a label y, from features X for a new data point
 - This is done using the `.predict()` method in scikit-learn

Nearest Neighbor Classifier (k = 1)

Training the classifier: Store all the features with their labels

Making predictions: The label of closest training point is returned

- i.e., we find the distance between a test point and all training points, and the closest point is the prediction



Distance between two points

Suppose we have two data points p_0 and p_1 and we want to compute the distance between these points

If the data points have two features x and y

- Our data is: $p_0 = [x_0, y_0]$ and $p_1 = [x_1, y_1]$
- The distance between the two data points is:

$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$$

If the data points have three features x, y and z

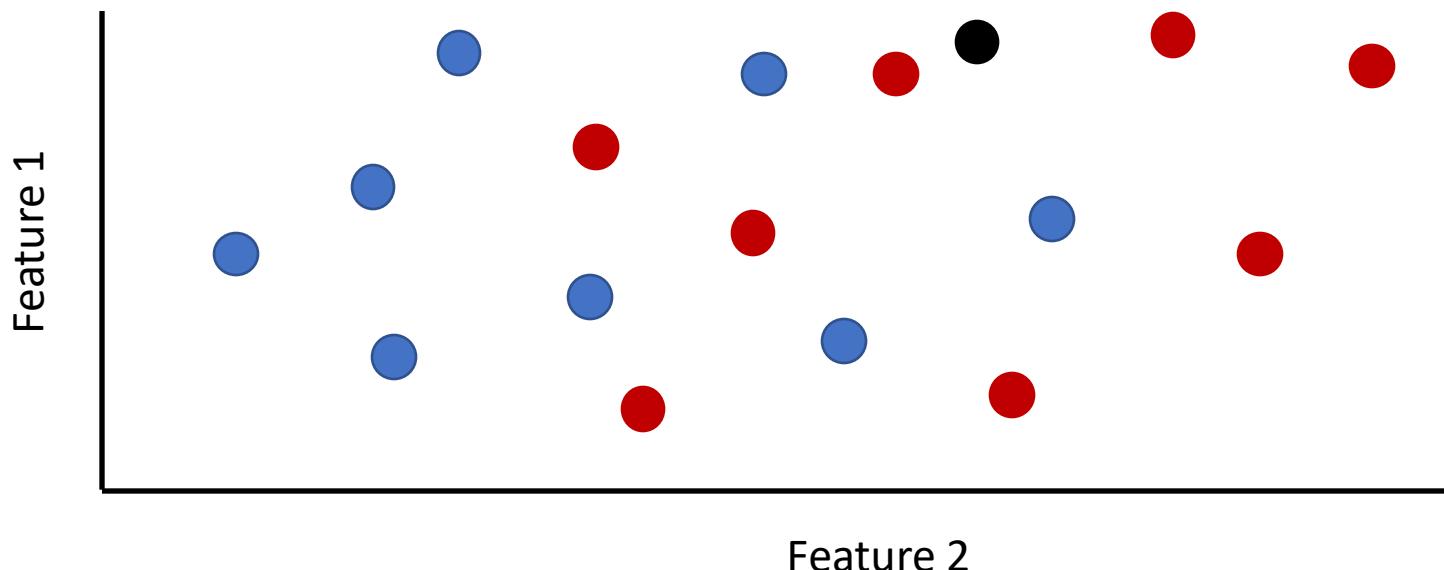
- Our data is: $p_0 = [x_0, y_0, z_0]$ and $p_1 = [x_1, y_1, z_1]$
- The distance between the two data points is:

$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2}$$

Finding the k Nearest Neighbors ($k \geq 1$)

To classify a point:

- Find its k nearest neighbors
- Take a majority vote of the k nearest neighbors to see which of the two classes appears more often
- Assign the point the class that wins the majority vote





KNN classifiers using scikit-learn

We can fit and evaluate the performance of a KNN classifier using:

```
knn = KNeighborsClassifier(n_neighbors = 1)      # construct a classifier
```

```
knn.fit(X_features, y_labels)      # train the classifier
```

```
y_test_predictions = knn.predict(X_test_features) # make predictions
```

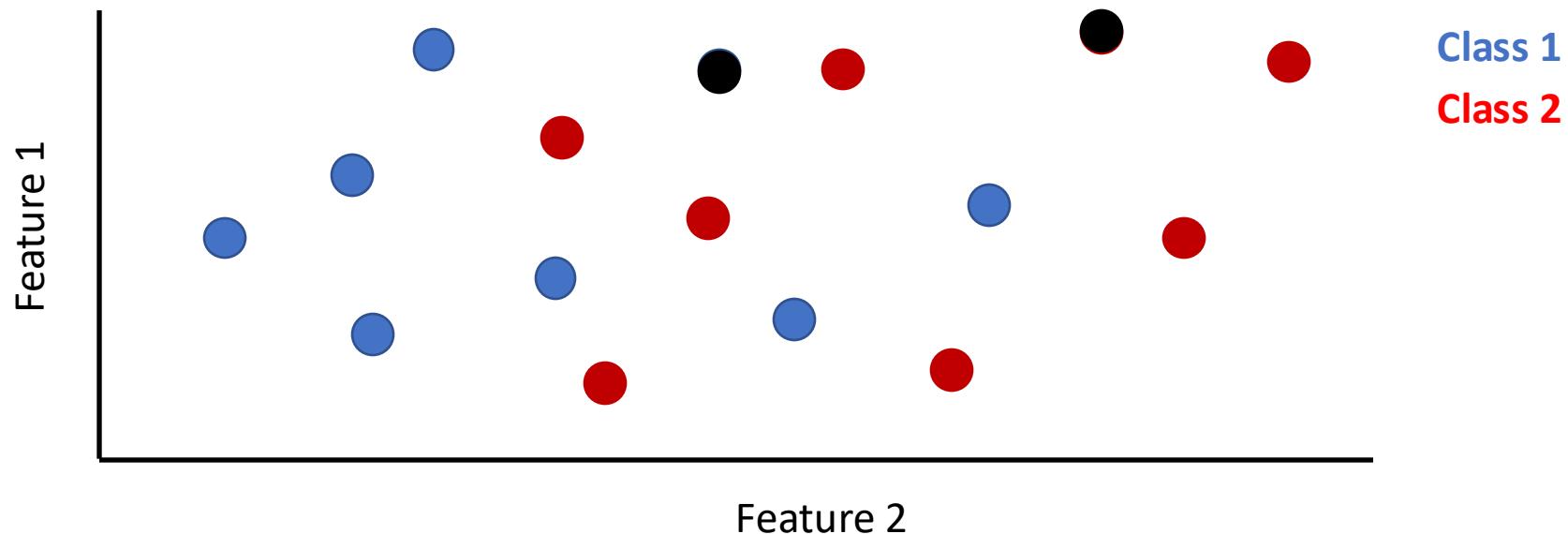
```
np.mean(y_test_predictions == y_test_labels)      # get accuracy
```

Let's explore this in Jupyter!

Evaluation

Training and test accuracy

Q: What would happen if we tested the classifier using the same data with $k = 1$?



Cross-validation

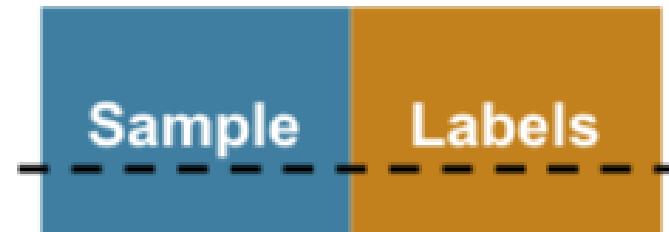
To assess how accurate the predictions of a classifier are, we need to split our data into a **training set** and **test set**

We fit the classifier on the training set

- i.e., we learn the relationship between labels (y) and features (x) on the training set

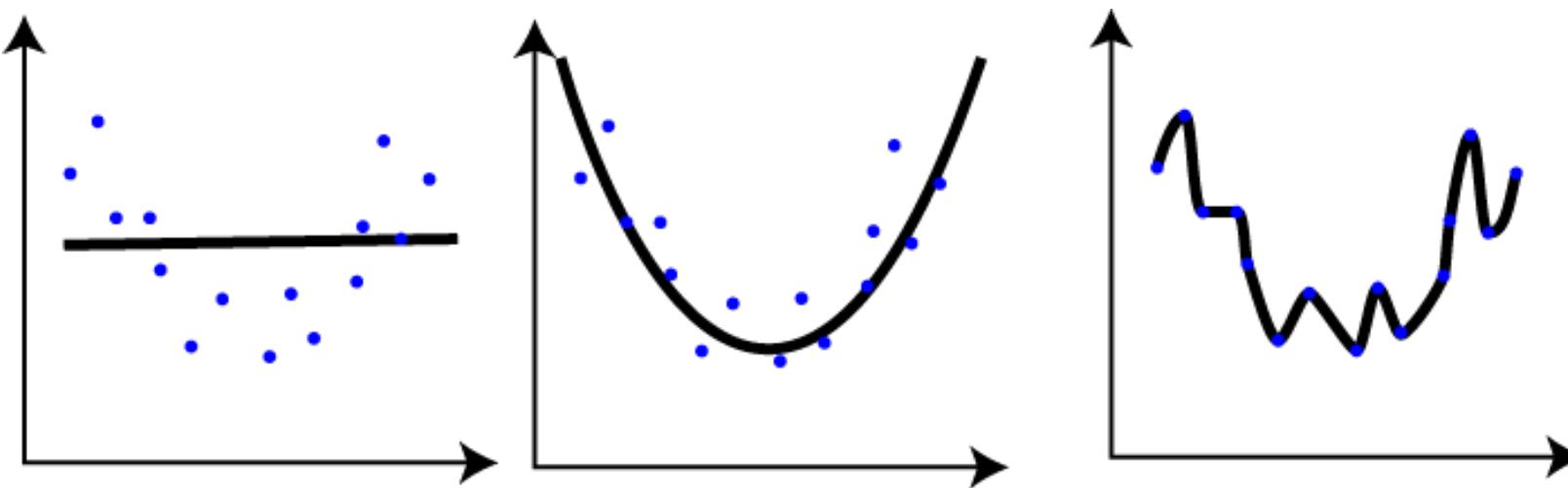
We assess the classifier's performance on the test set

The process of fitting your classifier on a training set and evaluation your classifier on a test set is called **cross-validation**



Cross-validation can help prevent **overfitting**!

Overfitting



Overfitting

If our classifier has over-fit to the training data then:

- a) We might not have a realistic estimate of how accurate its predictions will be on new data
- b) There might be a better classifier that would not over-fit to the data and thus can make better predictions

What we really want to estimate is how well the classifier will make predictions on new data, which is called the **generalization (or test) error**

[Overfitting song...](#)

k-fold cross-validation

Are there any downsides to using **half** the data for training and half for testing?

Since we are only using half the data for training, potentially can build a better model if we used more of our data

- Say 90% of our data for training and 10% of testing

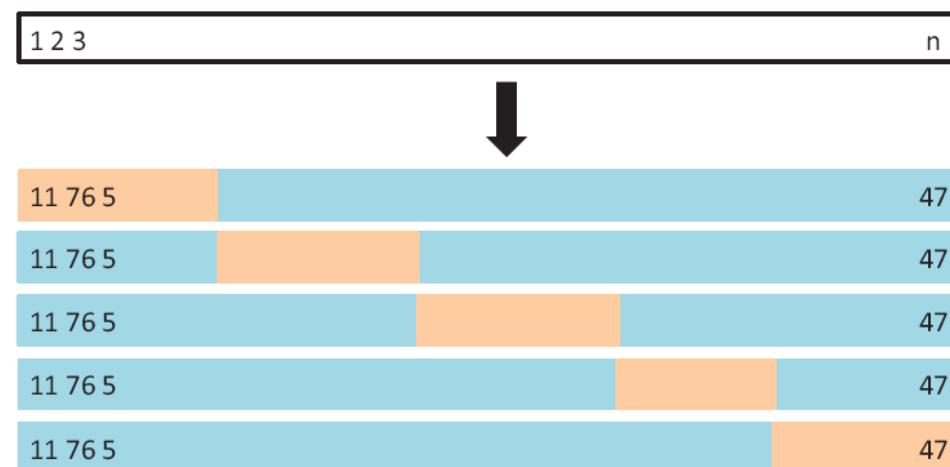
Problem: Then our estimate of the generalization error would be worse

Solutions?

k-fold cross-validation

k-fold cross-validation

- Split the data into k parts
- Train on $k-1$ of these parts and test on the left out part
- Repeat this process for all k parts
- Average the prediction accuracies to get a final estimate of the generalization error



Let's explore
this in Jupyter!

Other classifiers

There are many other classification algorithms such as:

- Support Vector Machines (SVM)
- Decision Trees/Random Forests
- Deep Neural Networks
- etc.

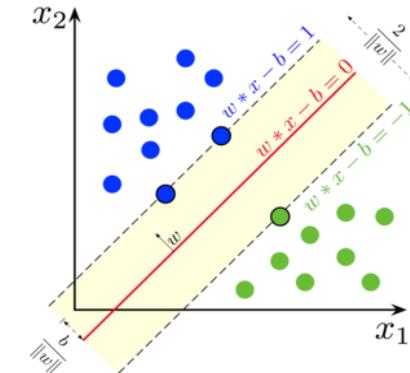
Scikit-learn makes it easy to try out different classifiers get their cross-validation performance

```
svm = LinearSVC()
```

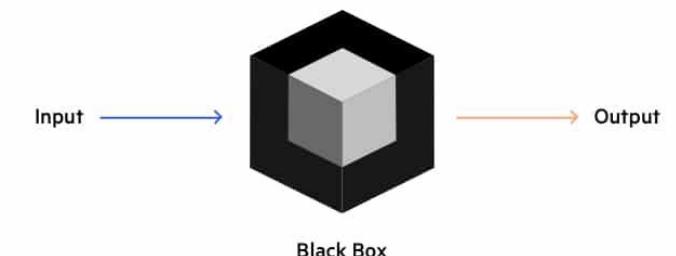
```
scores = cross_val_score(svm, X_features, y_labels, cv = 5)
```

```
scores.mean()
```

Let's quickly try this in Jupyter!



Black Box Testing



Next class, building the KNN classifier ourselves

YData: Introduction to Data Science



Lecture 22: Classification

Overview

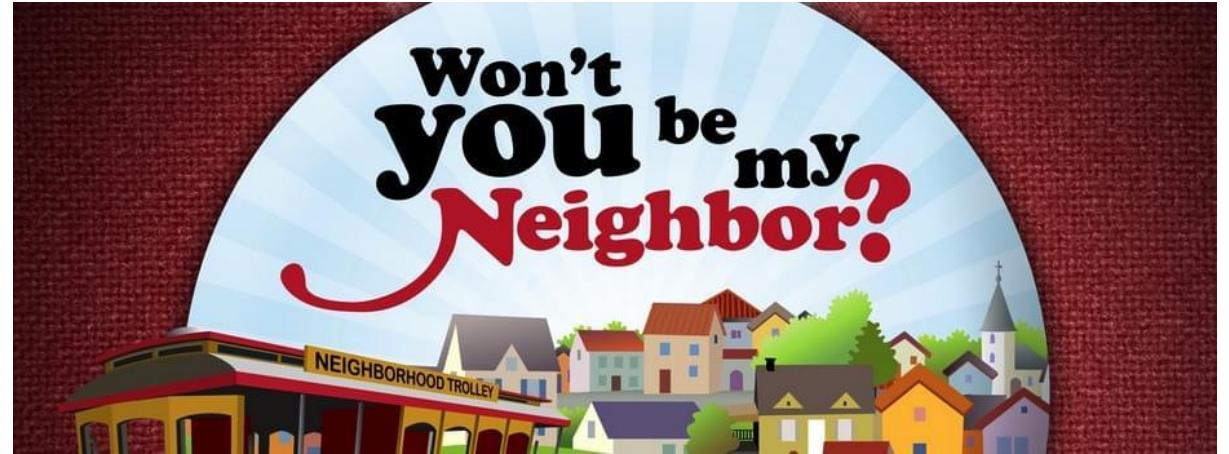
Quick review of KNN classifier

Cross-validation

Other classifiers

Building our own KNN classifier

If there is time: feature normalization



Project timeline

Sunday, November 17th

- Projects are due on Gradescope at 11pm
- Email a pdf of your project to your peer reviewers
 - A list of whose paper you will review is posted on Canvas
 - Fill out the draft reflection on Canvas

Sunday, November 24th

- Jupyter notebook files with your reviews need to be sent to the authors
- A template for doing your review has been posted

Sunday, December 8th

- Project is due on Gradescope
 - Add peer reviews to the Appendix of your project

Homework 9 has been posted

- It is due December 1st

That went as planned,
said no project ever.



your eCards
www.yourecards.com

Extra office hours tomorrow
(11/15) will be from 1-2pm

Review of Classification

Prediction: regression and classification

We “learn” a function f

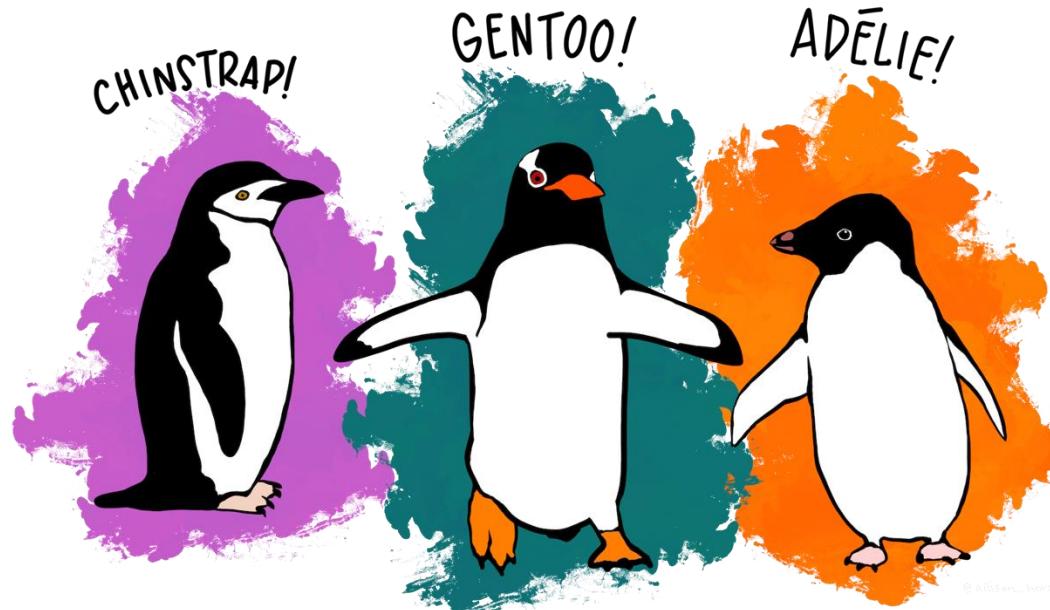
- $f(x) \rightarrow y$

Input: x is a data vector of "features"

Output:

- Regression: output is a real number ($y \in \mathbb{R}$)
- Classification: output is a categorical variable y_k

Example: Penguin species



What are the features and labels in this task?

- Labels (y): Chinstrap, Gentoo, Adelie
- Features (X): Flipper length, bill length, body mass, ...

Classifiers

Classifiers take a list/array of features X, and return a predicted label y



There are many different types of classifiers

- Decision Trees, Support Vector Machines, Logistic regression, Neural Networks, etc.

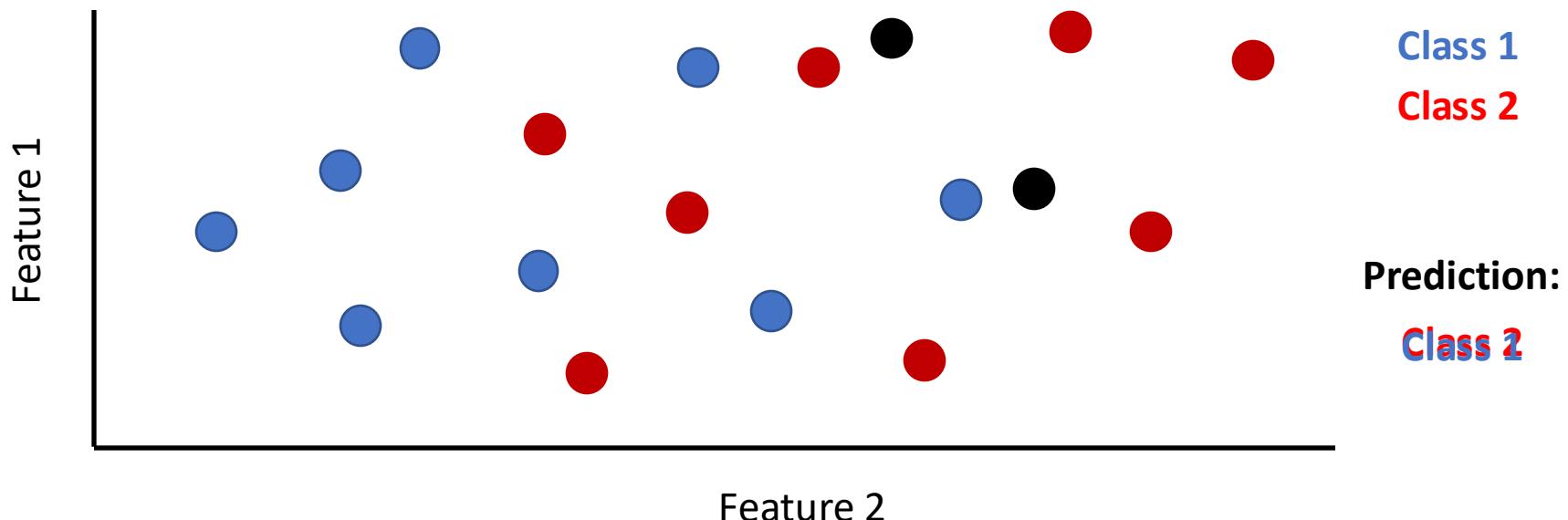
The classification process always involves two steps:

1. **Training** the classifier to learn the relationship between features (X) and labels (y) (from many examples)
 - This is done using the `.fit()` method in scikit-learn
2. **Using** the classifier to predict a label y, from features X for a new data point
 - This is done using the `.predict()` method in scikit-learn

K-Nearest Neighbor Classifier (KNN)

Training the classifier: Store all the features with their labels

Making predictions: The label of closest k training points is returned



KNN classifiers using scikit-learn

We can fit and evaluate the performance of a KNN classifier using:

```
knn = KNeighborsClassifier(n_neighbors = 1)      # construct a classifier
```

```
knn.fit(X_features, y_labels)      # train the classifier
```

```
penguin_predictions = knn.predict(X_penguin_features) # make predictions
```

```
np.mean(penguin_predictions == y_penguin_labels)      # get accuracy
```

Evaluation

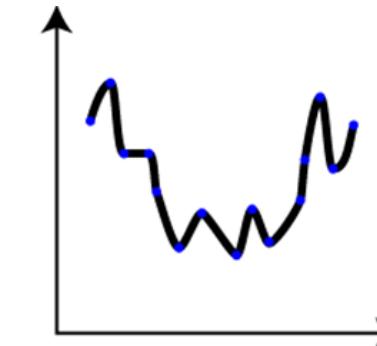
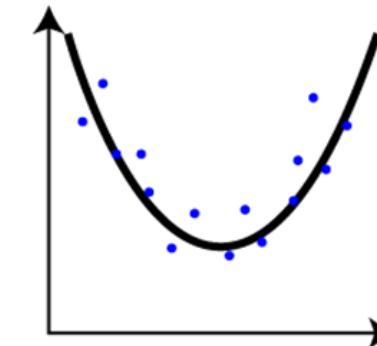
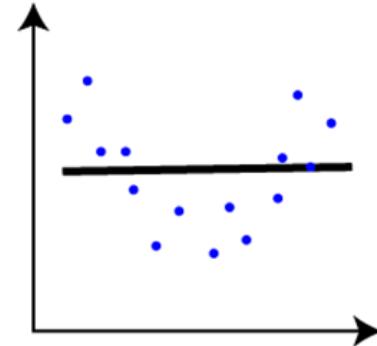
Review: overfitting

Overfitting occurs when our classifier matches too close to the training data and doesn't capture the true underlying patterns

If our classifier has overfit to the training data then:

- a. We might not have a realistic estimate of how accurate its predictions will be on new data
- b. There might be a better classifier that would not over-fit to the data and thus can make better predictions

What we really want to estimate is how well the classifier will make predictions on new data, which is called the **generalization (or test) accuracy**



[Overfitting song...](#)

Review: Cross-validation

To assess how accurate the predictions of a classifier are, we need to split our data into a **training set** and **test set**

We fit the classifier on the training set

- i.e., we learn the relationship between labels (y) and features (x) on the training set

We assess the classifier's performance on the test set

The process of fitting your classifier on a training set and evaluation your classifier on a test set is called **cross-validation**

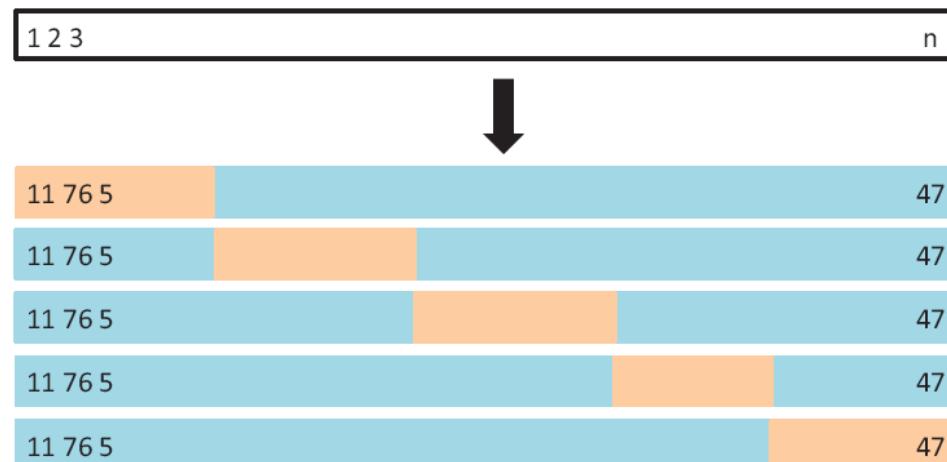
Cross-validation can help prevent **overfitting**!



k-fold cross-validation

k-fold cross-validation

- Split the data into k parts
- Train on $k-1$ of these parts and test on the left out part
- Repeat this process for all k parts
- Average the prediction accuracies to get a final estimate of the generalization error

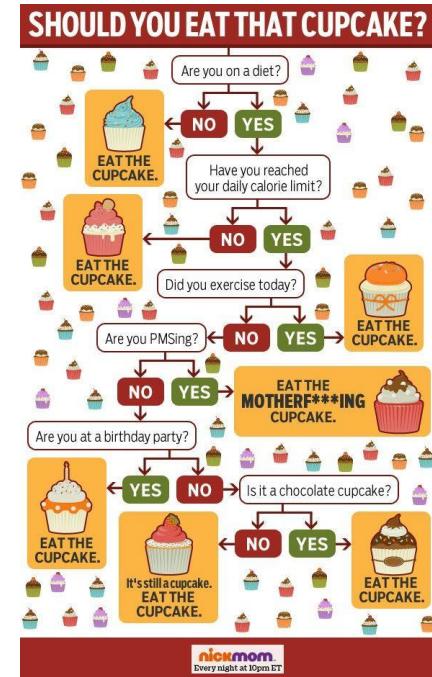
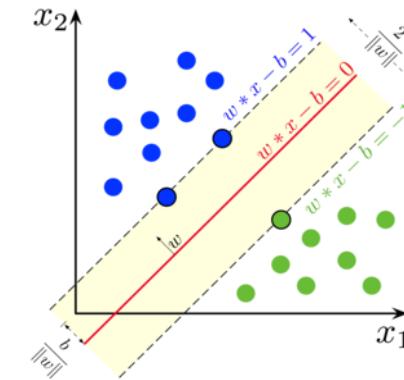


Let's try this in
Jupyter!

Other classifiers

There are many other classification algorithms such as:

- Support Vector Machines (SVM)
- Decision Trees/Random Forests
- Deep Neural Networks
- etc.



Scikit-learn makes it easy to try out different classifiers
get their cross-validation performance

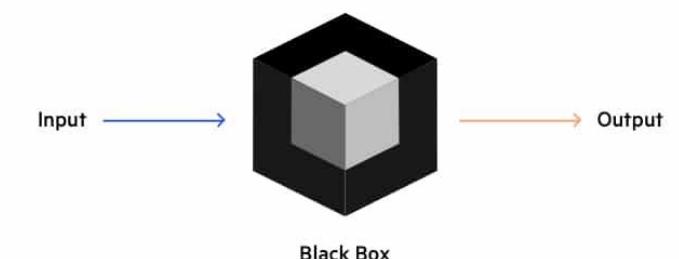
```
svm = LinearSVC()
```

```
scores = cross_val_score(svm, X_features, y_labels, cv = 5)
```

```
scores.mean()
```

Let's quickly try this in Jupyter!

Black Box Testing



Building a KNN classifier

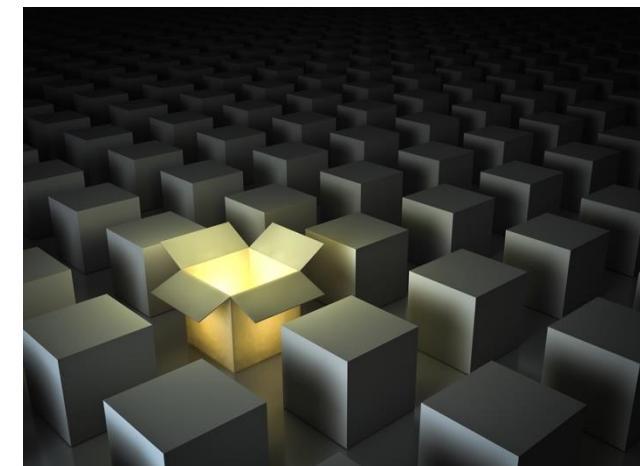
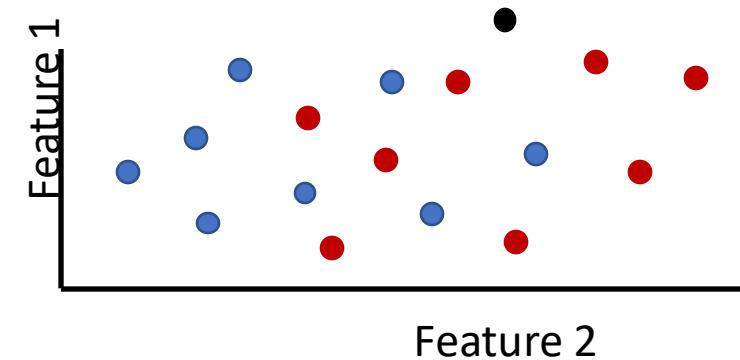


Building the KNN classifier

So far we have used a KNN classifier

- and we have some idea of how it works

Let's now see if we can write to to
implement the classifier ourselves...



Steps to build a KNN classifier

We build our KNN classifier by creating a series of functions...

1. `euclid_dist(x1, x2)`

$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2}$$

- Calculates the Euclidean distance between two points x_1 and x_2

2. `get_labels_and_distances(test_point, X_train_features, y_train_labels)`

- Finds the distance between a test point and all the training points

3. `classify_point(test_point, k, X_train_features, y_train_labels)`

- Classifies one test point by returning the majority label of the k closest points

4. `classify_all_test_data(X_test_data, k, X_train_features, y_train_labels)`

- Classifiers all test points

Let's continue exploring this in Jupyter!

Feature normalization

Review: Distance between two points

Two features x and y: $D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$

Three features x, y, and z: $D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2}$

- And so on for more features...

It's important the features are standardized

- If not, features that typically have larger values will dominate the distance measurement

Feature normalization

In order to deal with features that are measured on very different scales, we can normalize the features

With a z-score normalization, we normalize each feature to have:

- A mean of 0
- A standard deviation of 1

We can do this in Python using:

```
scalar = StandardScaler()  
scalar.fit(X_train)  
X_train_transformed = scalar.transform(X_train)  
X_test_transformed = scalar.transform(X_test)
```

bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
46.1	15.1	215.0	5100.0
37.3	17.8	191.0	3350.0
51.3	18.2	197.0	3750.0
39.5	16.7	178.0	3250.0
48.7	15.1	222.0	5350.0

$$\frac{x - \bar{x}}{s}$$

$$z_i = \frac{x_i - \bar{x}}{s}$$

Feature normalization

To avoid overfitting ("data leakage") we can:

- Calculate the mean and standard deviation on the training
- Apply these means and standard deviations to normalize the training and test sets

To do this in a cross-validation loop, we can use a pipeline:

```
scalar = StandardScaler()
```

```
knn = KNeighborsClassifier(n_neighbors = 1)
```

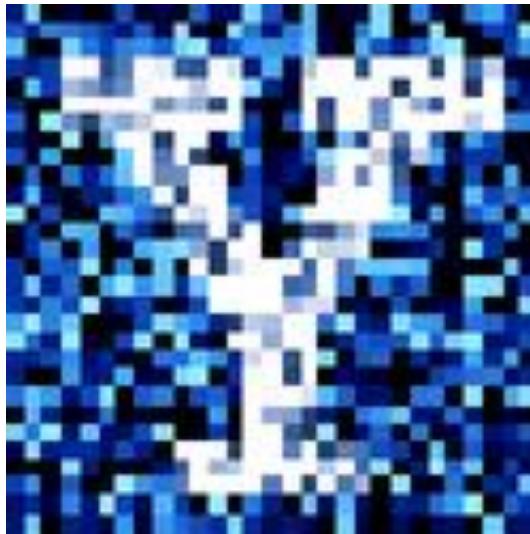
```
cv = KFold(n_splits=5)
```

```
pipeline = Pipeline([('transformer', scalar), ('estimator', knn)])
```

```
scores = cross_val_score(pipeline, X_penguin_features, y_penguin_labels, cv = cv)
```

Let's explore this in Jupyter!

YData: Introduction to Data Science



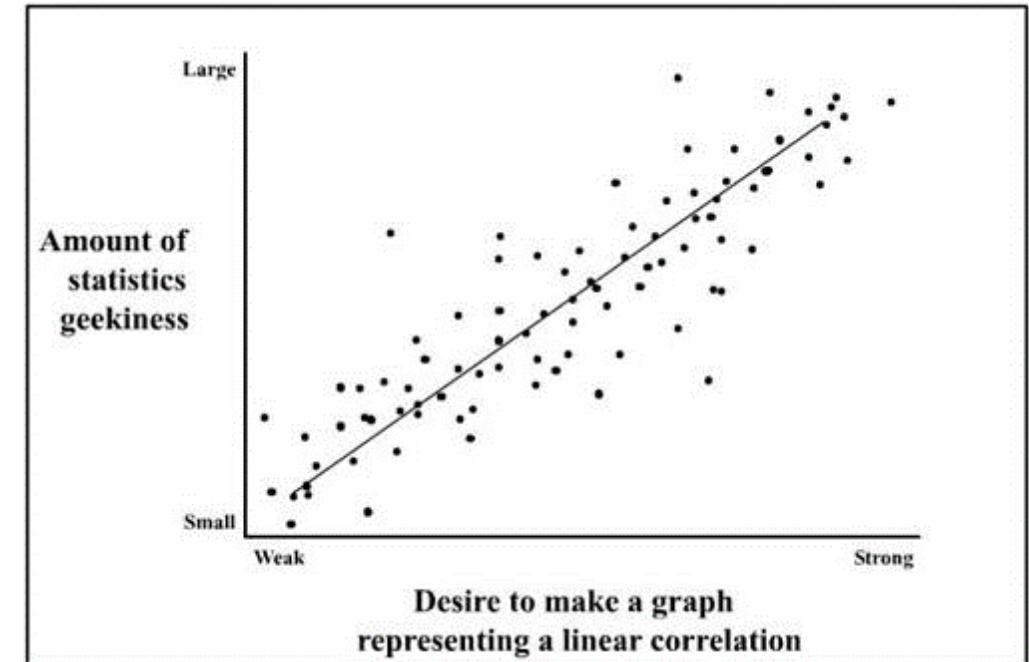
Lecture 23: Linear regression and unsupervised learning

Overview

Linear regression

Unsupervised learning/clustering

- K-means cluster
- If there is time: hierarchical clustering



Project timeline

~~Sunday, November 17th~~

- Projects are due on Gradescope at 11pm
- Email a pdf of your project to your peer reviewers
 - A list of whose paper you will review is posted on Canvas
 - Fill out the draft reflection on Canvas

~~Sunday, November 24th~~

- Jupyter notebook files with your reviews need to be sent to the authors
- A template for doing your review has been posted

~~Sunday, December 8th~~

- Project is due on Gradescope
 - Add peer reviews to the Appendix of your project

~~Homework 9 has been posted~~

- It is due December 1st



Prediction: regression and classification

We “learn” a function f

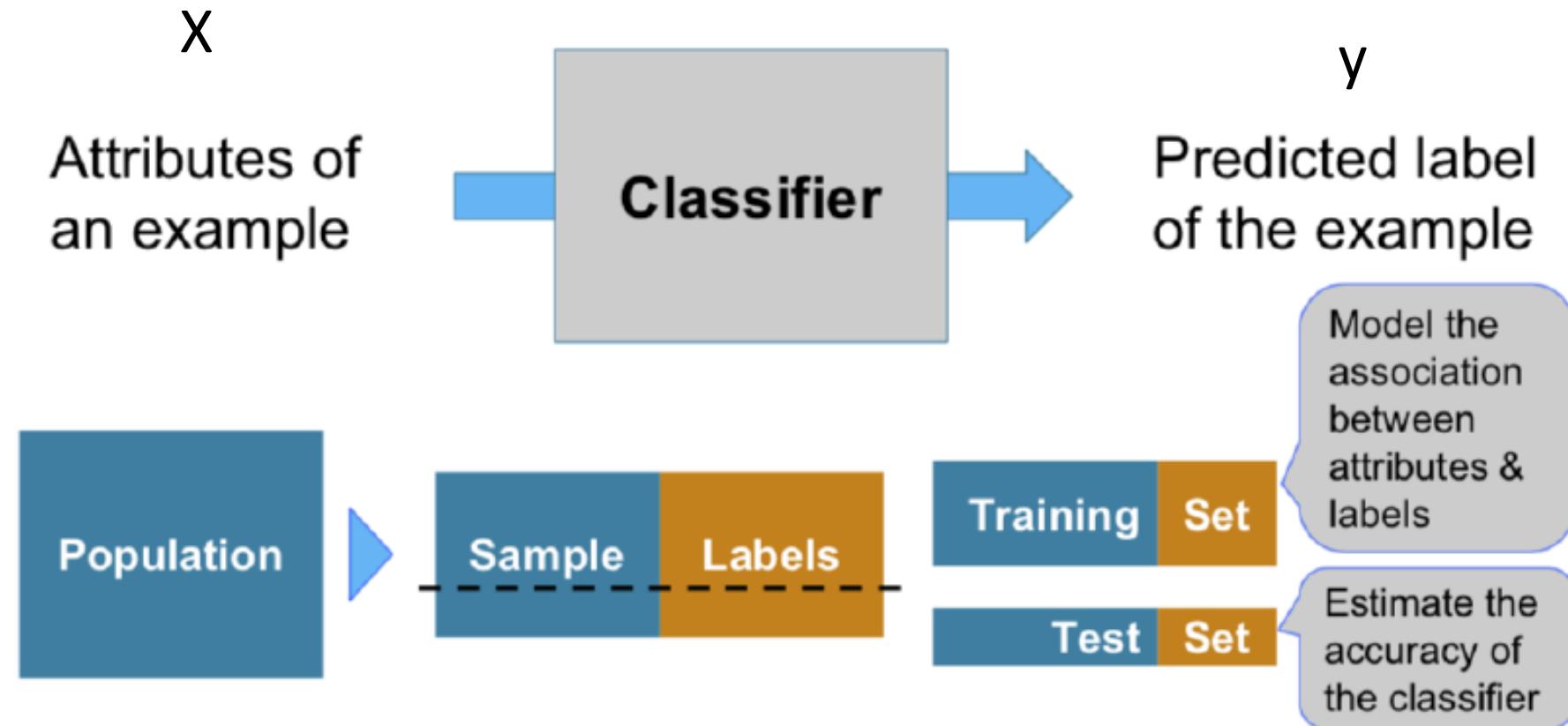
- $f(\mathbf{x}) \longrightarrow y$

Input: \mathbf{x} is a data vector of "features"

Output:

- Regression: output is a real number ($y \in \mathbb{R}$)
- Classification: output is a categorical variable y_k

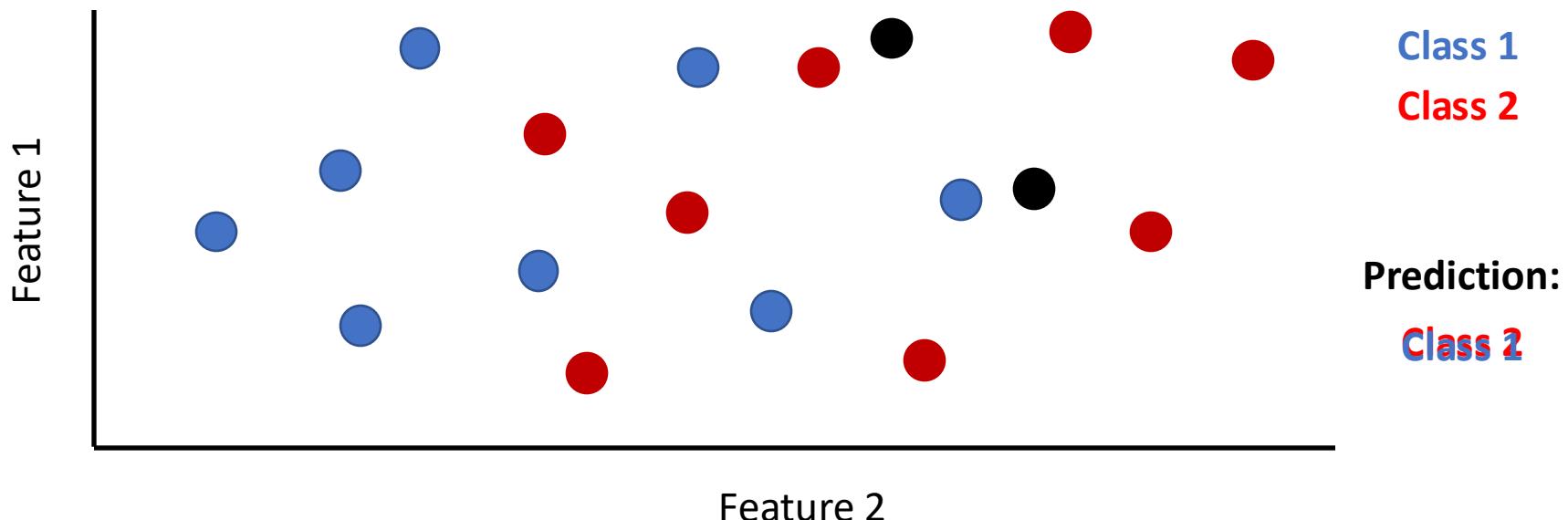
Training a classifier



K-Nearest Neighbor Classifier (KNN)

Training the classifier: Store all the features with their labels

Making predictions: The label of closest k training points is returned



KNN classifiers using scikit-learn

We can fit and evaluate the performance of a KNN classifier using:

```
knn = KNeighborsClassifier(n_neighbors = 1)      # construct a classifier
```

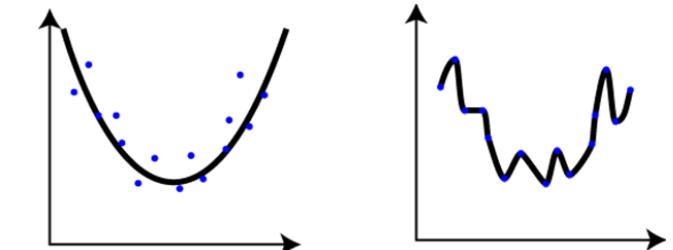
```
knn.fit(X_features, y_labels)      # train the classifier
```

```
penguin_predictions = knn.predict(X_penguin_features) # make predictions
```

```
np.mean(penguin_predictions == y_penguin_labels)      # get accuracy
```

Review: overfitting

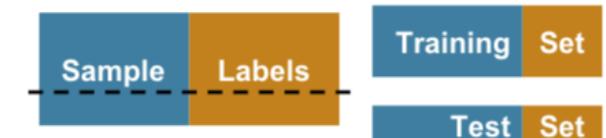
Overfitting occurs when our classifier matches too close to the training data and doesn't capture the true underlying patterns



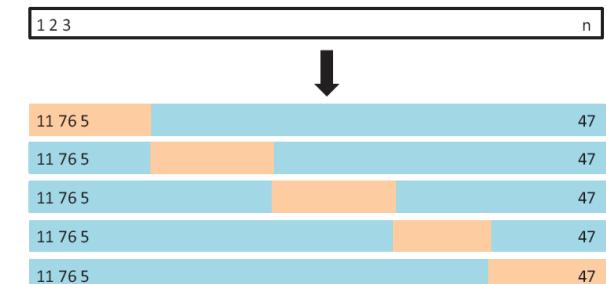
[Overfitting song...](#)

To avoid overfitting we split our data into a training and test set

- Classifier learns relationship on training data
- Classifier's performance is evaluated on the test data



We can use k-fold cross-validation to get a better estimate of the test accuracy



Other classifiers

We also built our own KNN classifier

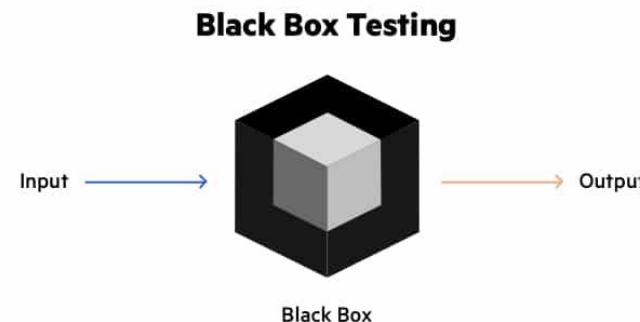
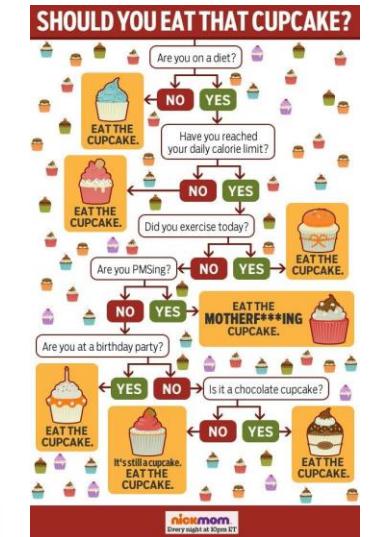
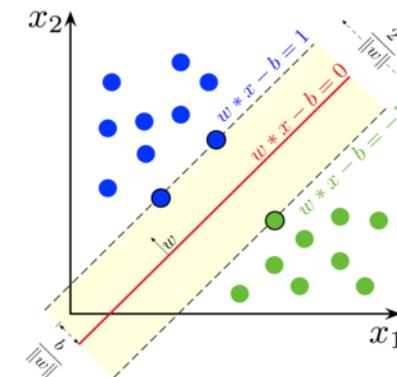
Scikit-learn makes it easy to try out different classifiers get their cross-validation performance

- E.g., SVM, random forests, neural networks, etc.

`svm = LinearSVC()`

```
scores = cross_val_score(svm,  
X_features, y_labels, cv = 5)
```

```
scores.mean()
```



Linear regression

Prediction: regression and classification

We “learn” a function f

- $f(x) \rightarrow y$

Input: x is a data vector of "features"



Output:

- Regression: output is a real number ($y \in \mathbb{R}$)
- Classification: output is a categorical variable y_k

Regression

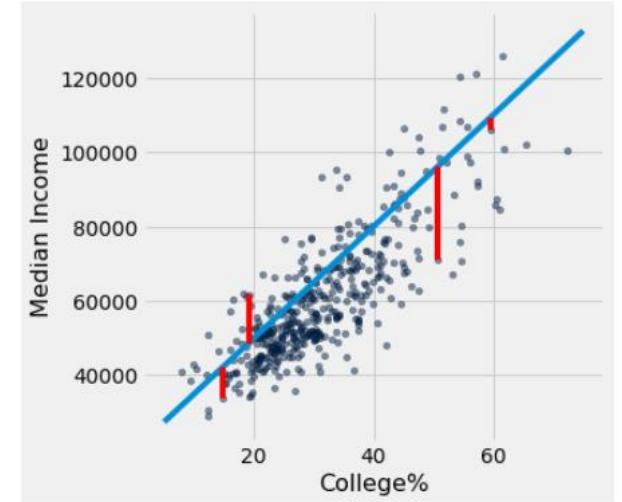
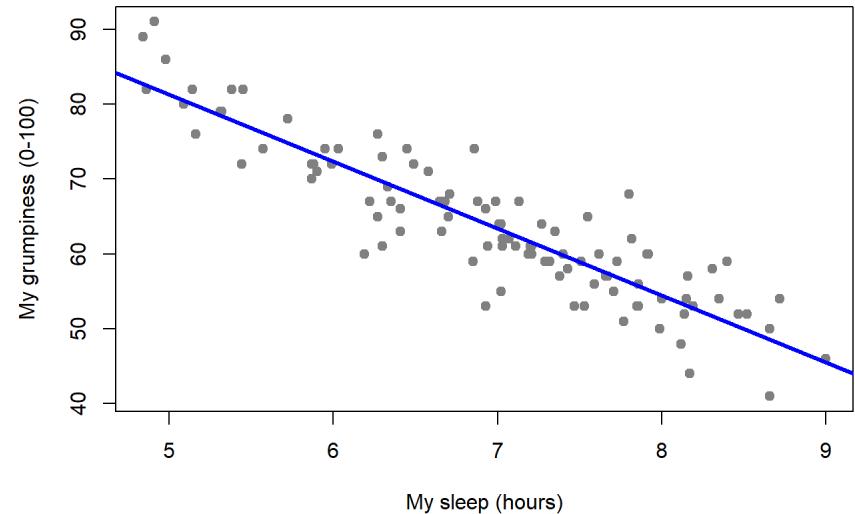
Regression is method of using one variable x to predict the value of a second variable y

- i.e., $\hat{y} = f(x)$
- Linear regression: $\hat{y} = \text{intercept} + \text{slope} \cdot x$

$$\hat{y} = b_0 + b_1 \cdot x$$

The coefficients for these regression models are found by minimizing root mean square error (RMSE)

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$



[Regression line app](#)

Multiple regression

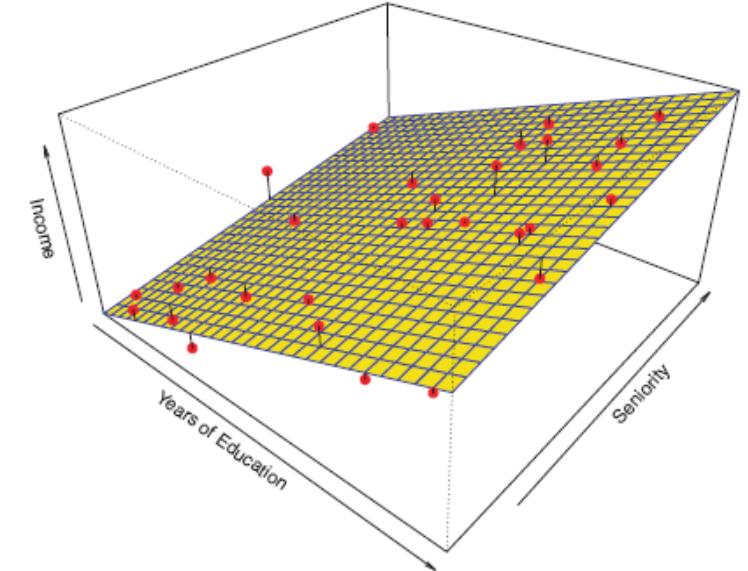
In multiple regression we try to predict a quantitative response variable y using several features x_1, x_2, \dots, x_k

We estimate coefficients using a data set to make predictions \hat{y}

$$\hat{y} = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + \dots + b_k \cdot x_k$$



Learn the b_i 's on the training set. Assess prediction accuracy on test set.

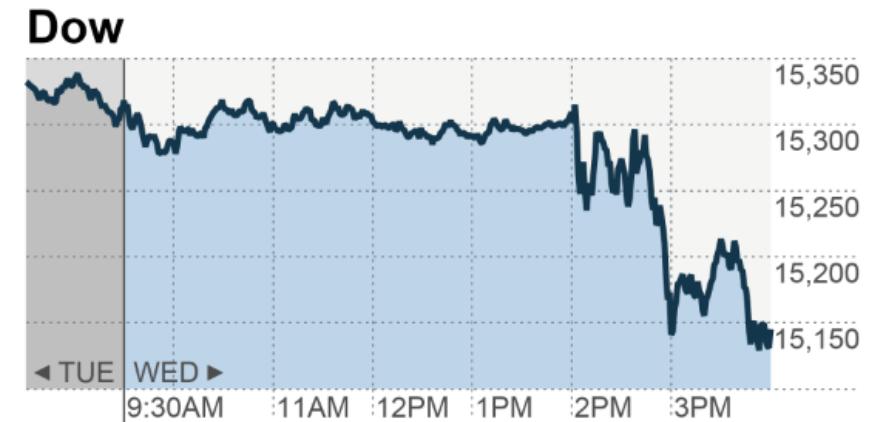


Multiple regression

$$\hat{y} = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + \dots + b_k \cdot x_k$$

There are many uses for multiple regression models including:

- To make predictions as accurately as possible
- To understand which predictors (x) are related to the response variable (y)



Linear regression models in scikit-learn

We can use scikit-learn to create linear regression models

- You can also use the stats models package to do this

```
linear_model = LinearRegression() # construct a linear regression model
```

```
linear_model.fit(X_train_features, y_train) # train the classifier
```

"Learns" b_0, b_1, \dots by minimizing
the RMSE on the training data

train the classifier

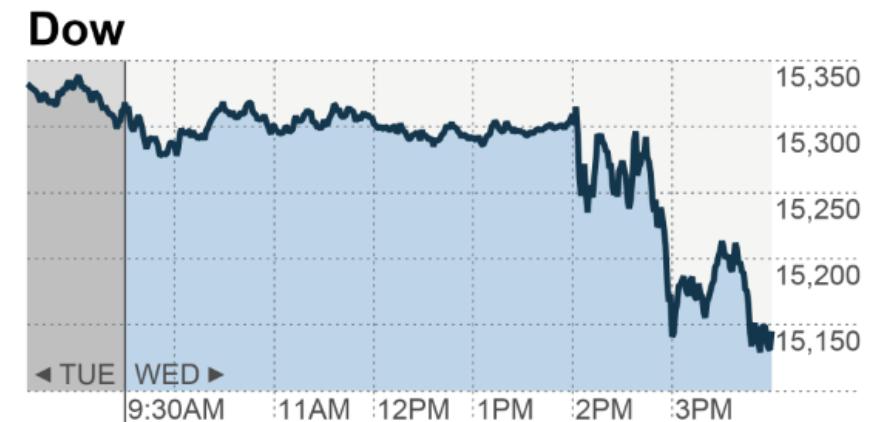
Numeric values
to predict

```
y_predictions = linear_model.predict(X_test_features) # make predictions
```

```
RMSE = np.sqrt(np.mean((y_test - y_predictions)**2)) # get the RMSE
```

Real world example

Rather than predicting stock prices...



Let's predict the mass of penguins!

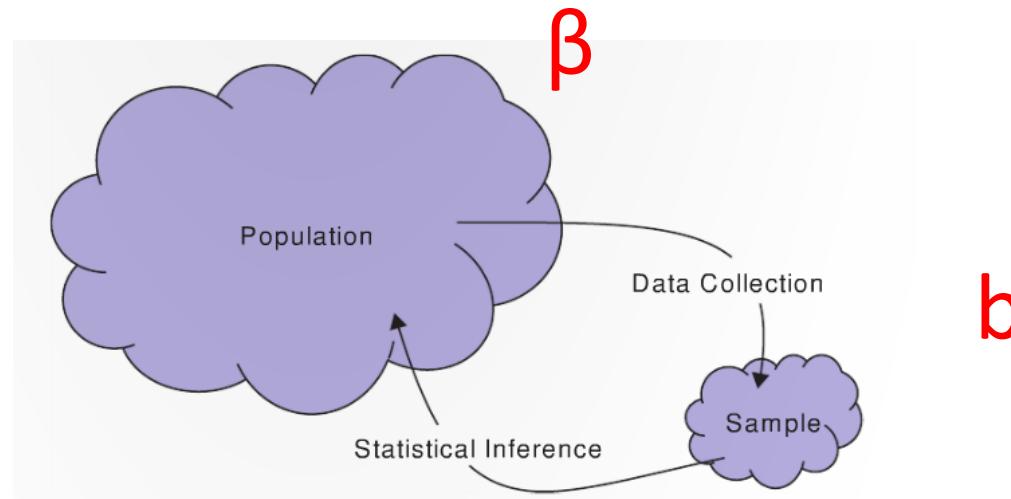
Let's try this in Jupyter!

Inference for simple linear regression

Inference for simple linear regression

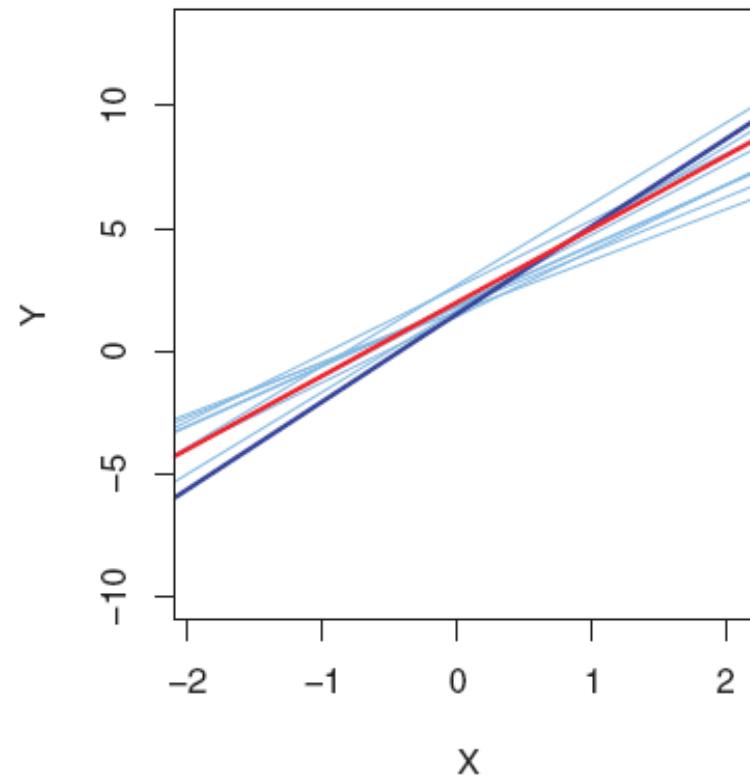
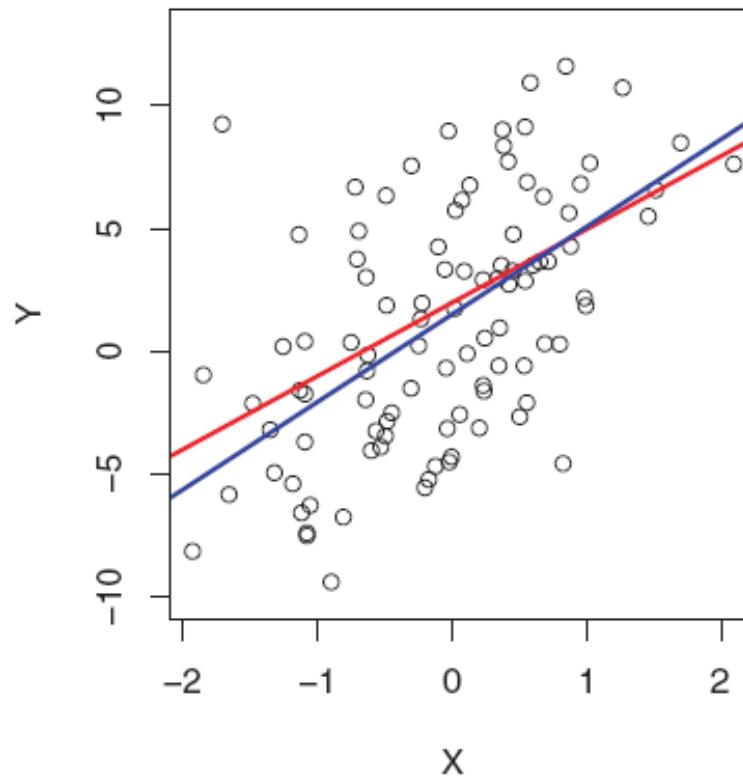
The Greek letter β is used to denote the slope *of the population*

The letter b is used to denote the slope *of the sample*



Population: β_0 and β_1

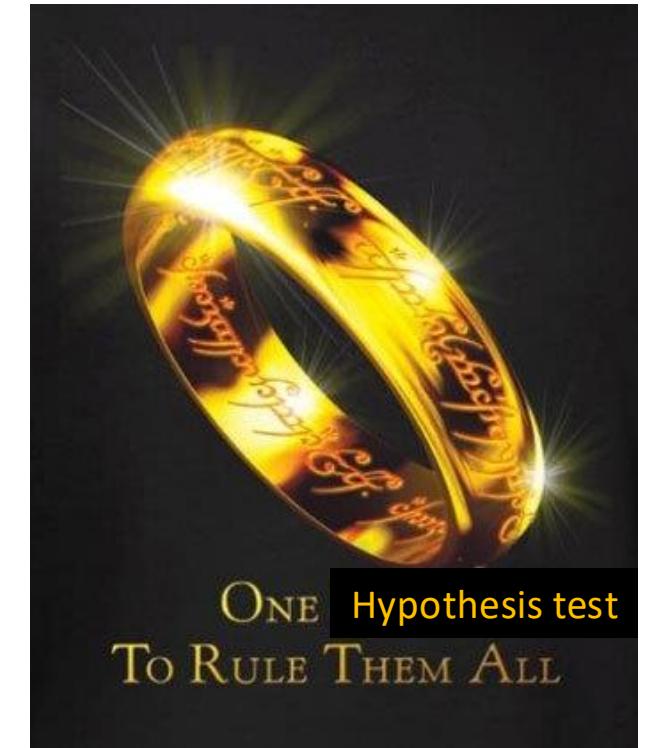
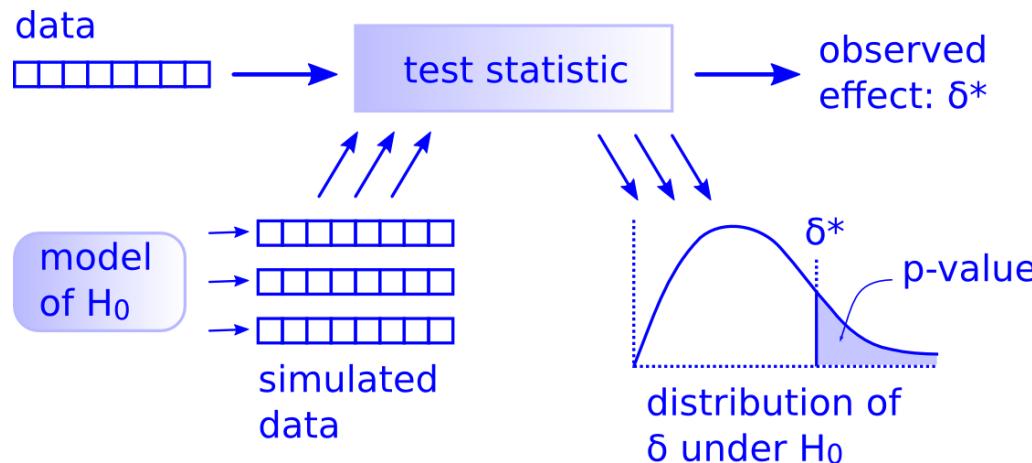
Sample estimates: b_0 and b_1



Hypothesis test for regression coefficients

We can run a hypothesis test to see if there is a linear relationship between a feature x , and a response variable y

There is only one [hypothesis test!](#)



Hypothesis test for regression coefficients

We can run a hypothesis test to see if there is a linear relationship between a feature x , and a response variable y

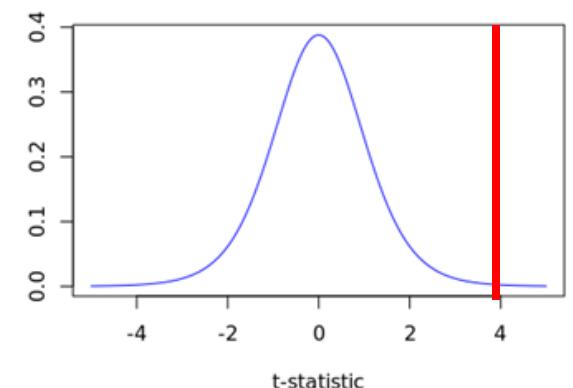
- $H_0: \beta_1 = 0$ (slope is 0, so no linear relationship between x and y)
- $H_A: \beta_1 \neq 0$

One type of hypothesis test we can run is based on a t-statistic:

$$t = \frac{b_1 - 0}{\hat{SE}_{b_1}}$$

- The t-statistic comes from a t-distribution with $n - k$ degrees of freedom
 - (If a few conditions are met)

We could also shuffle our data, fit models, and extract b_1 coefficients to create a null distribution



Confidence intervals for regression coefficients

The confidence interval for the slope coefficients:

$$\beta_1$$

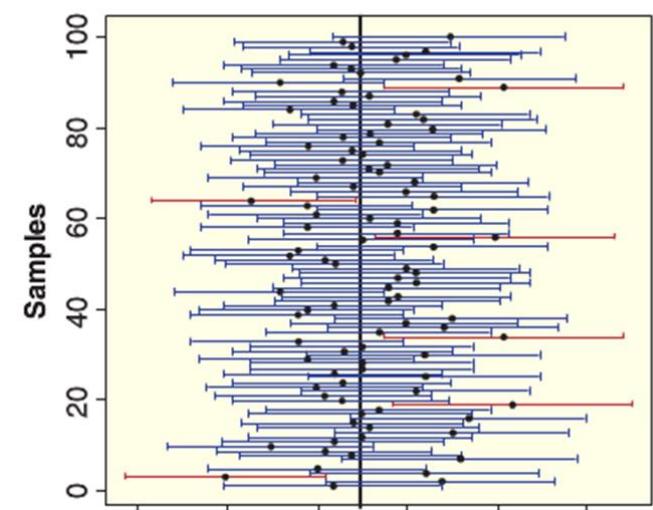
We can use the statsmodels package to run hypothesis tests and create confidence intervals using “parametric” methods based on t-distributions

```
import statsmodels.api as sm

sm_linear_model = sm.OLS(y_train,
                          X_train_with_constant).fit()

sm_linear_model.summary()
```

Let's try this in Jupyter!



Unsupervised learning

Supervised learning and unsupervised learning

In **supervised learning** we have a set of features X, along with a label y

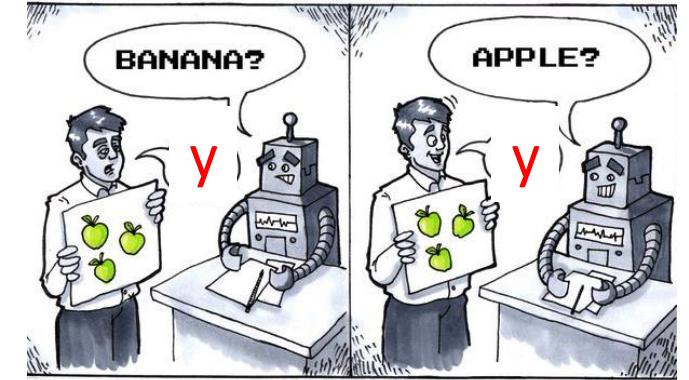
- We use the features X to predict y on new data

In **unsupervised learning**, we have features X, but **no** response variable y

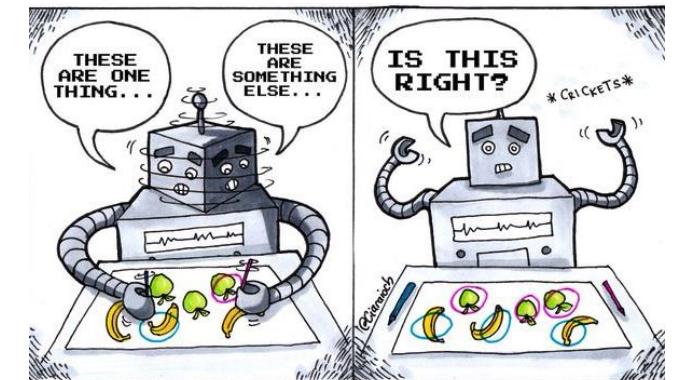
Unsupervised learning can be useful in order to find structure in the data and to visualize patterns

A key challenge in unsupervised learning is that there is no real ground truth response variable y

- So we don't have measures like the mean prediction accuracy



Supervised Learning



Unsupervised Learning

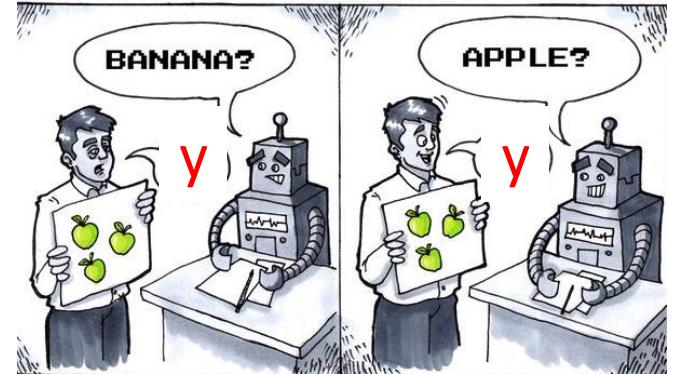
Unsupervised learning

Given we are almost at the end of the semester, we will focus on clustering, which is one type of unsupervised learning:

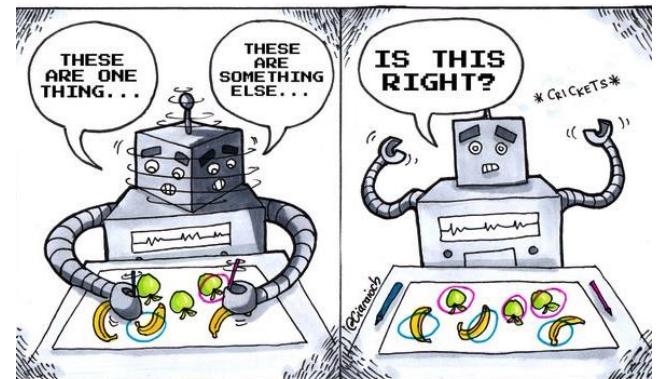
In **clustering** we try to group similar data points together

Another type of unsupervised learning:

- **Dimensionality reduction** where we try to find a smaller set of features that captures most of the variability original larger feature set
 - E.g., principal component analysis (PCA)



Supervised Learning



Unsupervised Learning

Clustering



So tell me how
many clusters do
you see?



Clustering

Clustering divides n data points x_i 's into subgroups

- Data points in the same group are similar/homogeneous
- Data points in different groups are different from each other

A diagram illustrating a data matrix. A red bracket on the left indicates the number of rows is n . A red bracket at the top indicates the number of columns is p . The matrix itself is a grid of cells, with the first few rows labeled as $x_{11}, x_{12}, \dots, x_{1p}$, $x_{21}, x_{22}, \dots, x_{2p}$, and so on down to $x_{n1}, x_{n2}, \dots, x_{np}$. The cells are outlined in black, with the first two rows highlighted by a blue border.

x_{11}	x_{12}	\cdots	x_{1p}
x_{21}	x_{22}	\cdots	x_{2p}
\vdots	\vdots	\ddots	\vdots
x_{n1}	x_{n2}	\cdots	x_{np}

Examples:

- Examining gene expression levels to group cancer types together
- Examining consumer purchasing behavior to perform market segmentation

Clustering can be:

- **Flat:** no structure beyond dividing points into groups
- **Hierarchical:** Population is divided into smaller and smaller groups (tree like structure)

K-means clustering

K-means clustering partitions the data into K distinct, non-overlapping clusters

- i.e., each data point x_i belongs to exactly one cluster C_k

The number of clusters, K , needs to be specified prior to running the algorithm

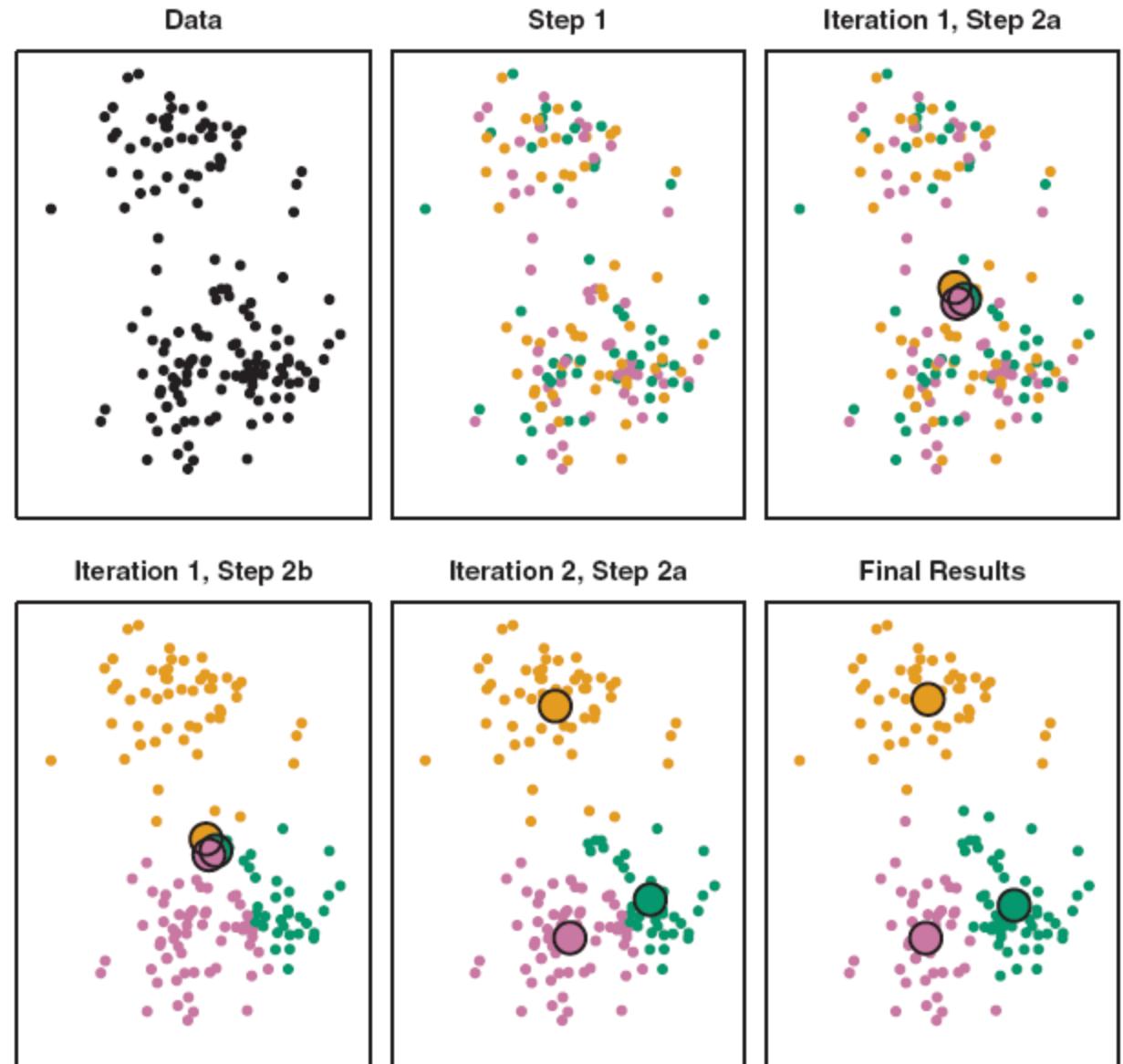
The goal is to minimize the within-cluster variation

- e.g., to make the Euclidean distance for all points within a cluster as small as possible

Finding the exact optimal solution is computationally intractable (there are k^n possible partitions), but a simple algorithm exists to find a local optimum which is often works well in practice.

K-means clustering

1. Randomly assign points to clusters C_k
2. Calculate cluster centers as means of points in each cluster
3. Assign points to the closest cluster center
4. Recalculate cluster center as the mean of points in each cluster
5. Repeat steps 3 and 4 until convergence



K-means clustering

Because only a local minimum is found, different random initializations will lead to different solutions

- One should run the algorithm multiple times to get better solutions

Let's explore this in Jupyter!



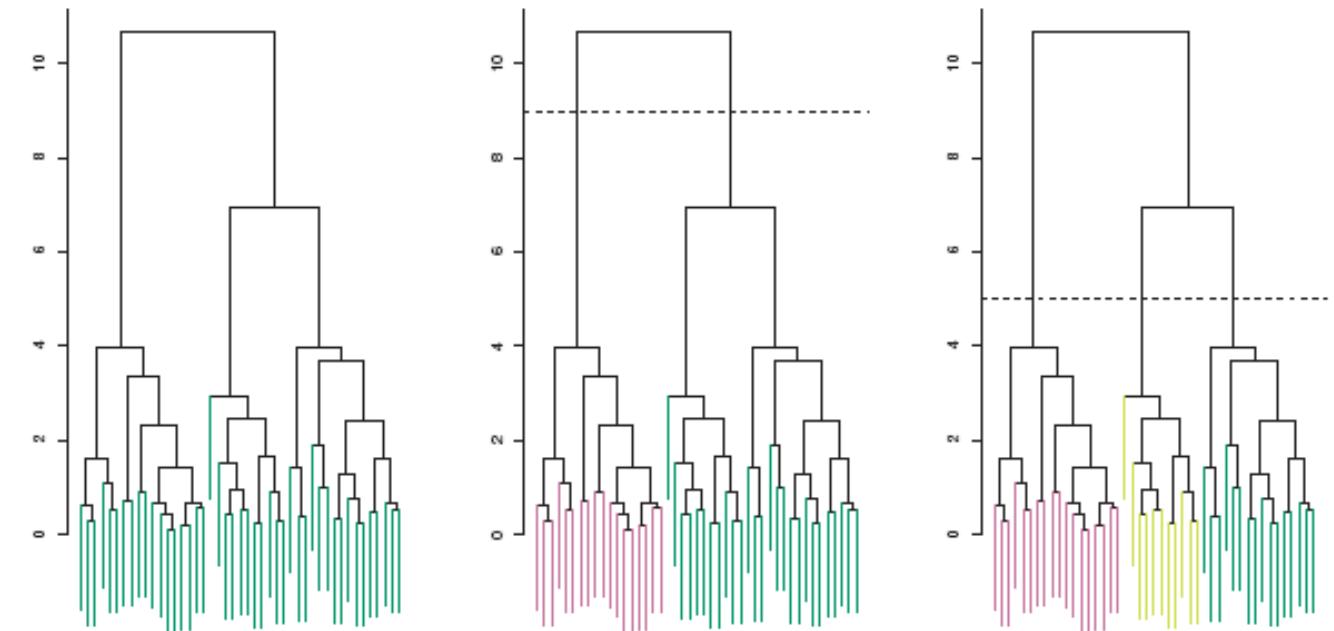
Hierarchical clustering

Hierarchical clustering

In **hierarchical clustering** we create a dendrogram which is a tree-based representation of successively larger clusters.

We can cut the dendrogram at any point to create as many clusters as desired

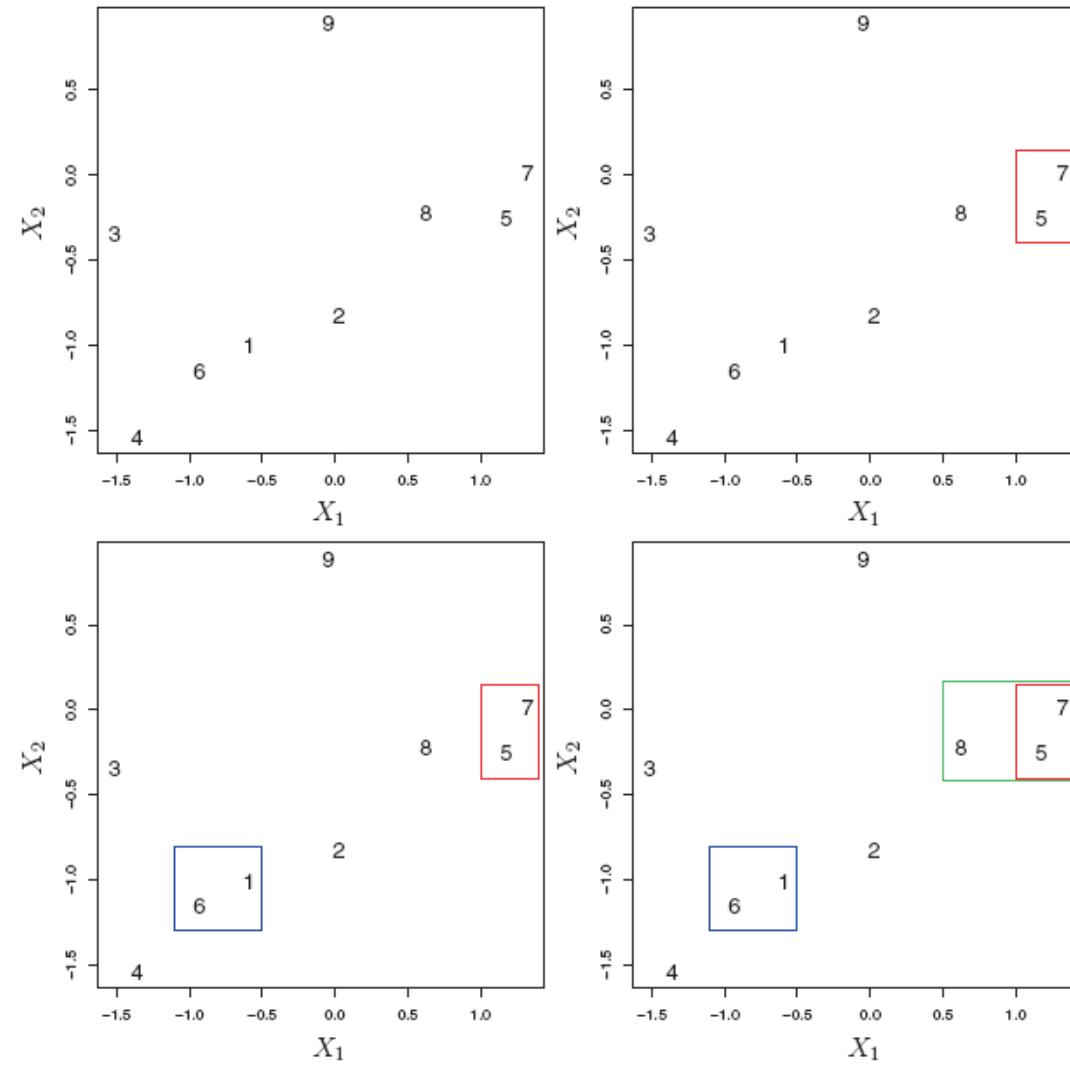
- i.e., don't need to specify the number of clusters, K , beforehand



Hierarchical clustering

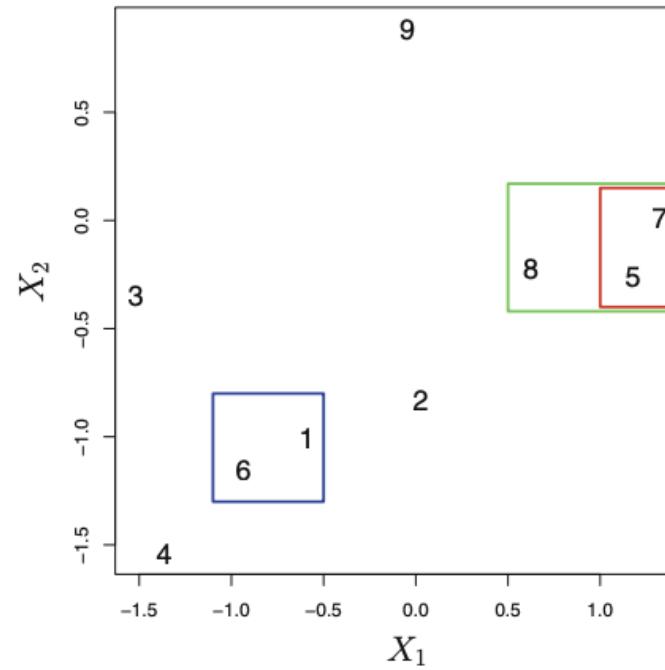
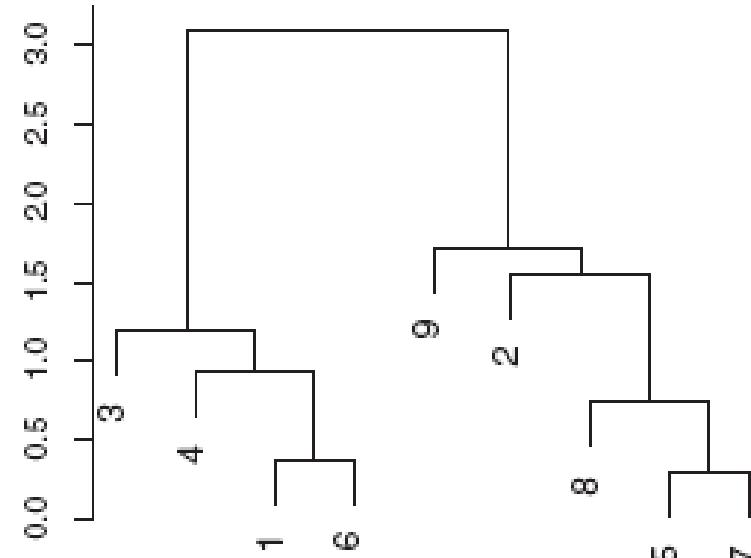
We can create a hierarchical clustering of the data using simple bottom-up agglomerative algorithm:

1. Choosing a (dis)similarity measure
 - E.g., The Euclidean distance
2. Initializing the clustering by treating each point as its own cluster
3. Successively merging the pair of clusters that are most similar
 - i.e., calculate the similarity between all pairs of clusters and merging the pair that is most similar
4. Stopping when all points have been merged into a single cluster



Hierarchical clustering

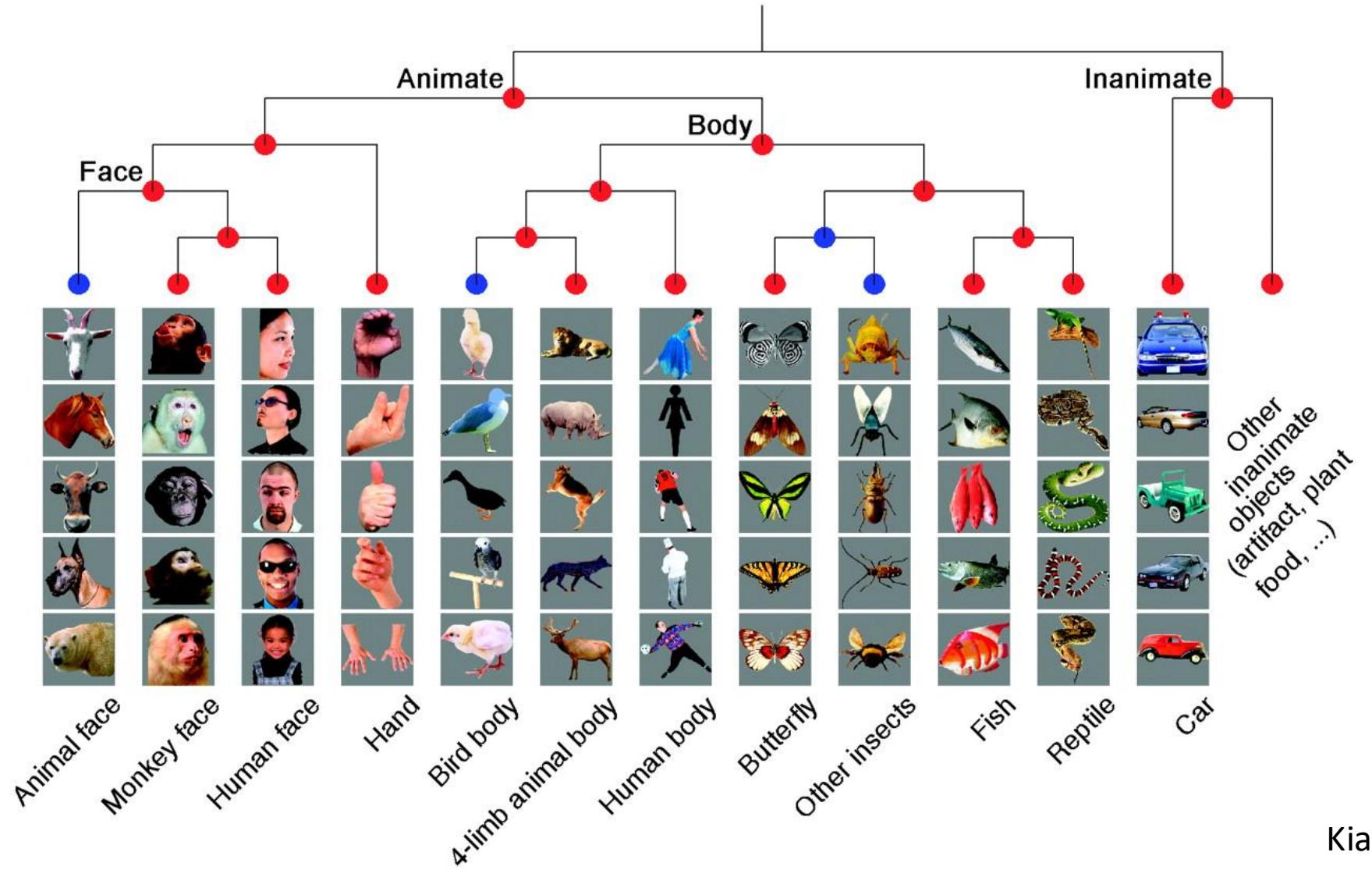
The vertical height that two clusters/points merge show how similar the two *clusters* are



Note: horizontal distance between *individual points* is not important:

- point 9 is considered as similar to point 2 as it is to point 7

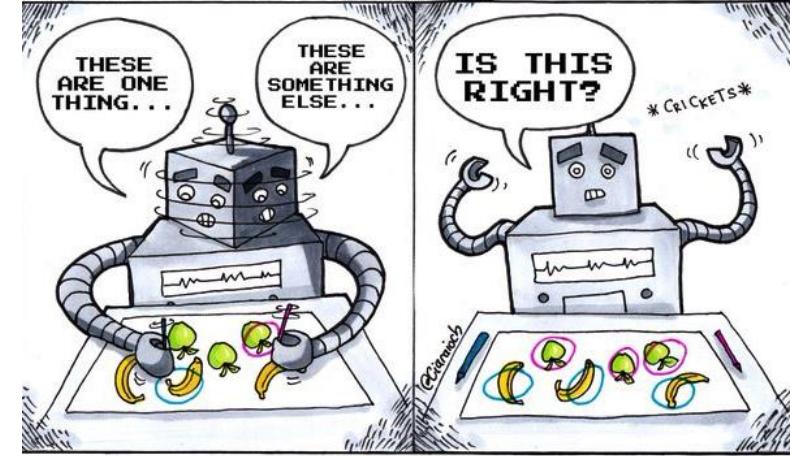
Hierarchical clustering example



Issues with clustering

Choices made can effect the results:

- Feature normalization and/or dissimilarity measure
- K-means: choice of K
- For hierarchical cluster: linkage and cut height



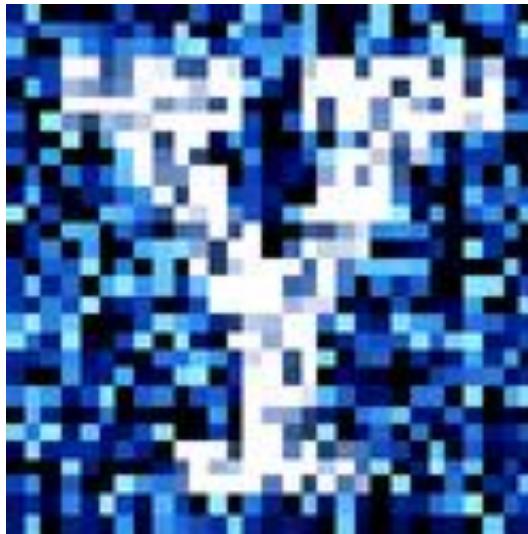
Unsupervised Learning

Potential approaches to deal with these issues:

- Try a few methods and see if one gives interesting/useful results
- Validate that you get similar results on a second set of data

Let's explore this in Jupyter!

YData: Introduction to Data Science



Lecture 24: Unsupervised learning

Overview

Unsupervised learning/clustering

- K-means cluster
- Hierarchical clustering

If there is time: objected oriented programming

OUR ANALYSIS SHOWS THAT THERE ARE THREE KINDS OF PEOPLE IN THE WORLD: THOSE WHO USE K-MEANS CLUSTERING WITH K=3, AND TWO OTHER TYPES WHOSE QUALITATIVE INTERPRETATION IS UNCLEAR.



Project timeline

~~Sunday, November 17th~~

- Projects are due on Gradescope at 11pm
- Email a pdf of your project to your peer reviewers
 - A list of whose paper you will review is posted on Canvas
 - Fill out the draft reflection on Canvas

~~Sunday, November 24th~~

- Jupyter notebook files with your reviews need to be sent to the authors
- A template for doing your review has been posted

~~Sunday, December 8th~~

- Project is due on Gradescope
 - Add peer reviews to the Appendix of your project

~~Homework 9 has been posted~~

- It is due December 1st



Prediction: regression and classification

We “learn” a function f

- $f(x) \rightarrow y$

Input: x is a data vector of "features"



Output:

- Regression: output is a real number ($y \in \mathbb{R}$)
- Classification: output is a categorical variable y_k

Review: Regression

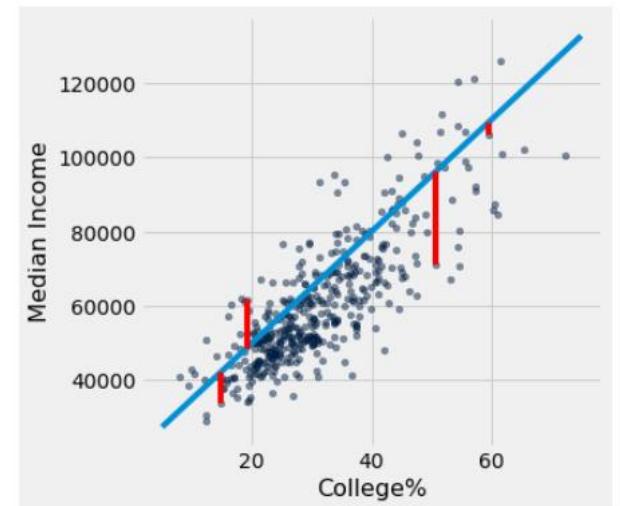
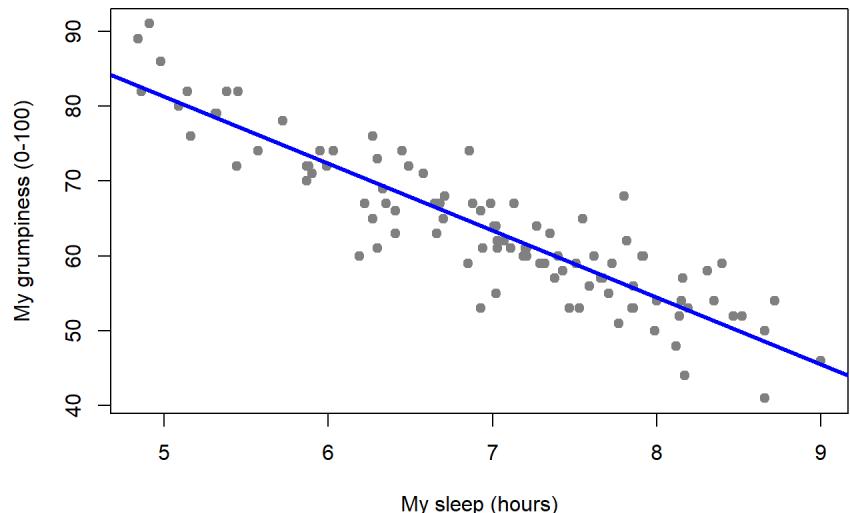
Regression is method of using one variable x to predict the value of a second variable y

- i.e., $\hat{y} = f(x)$
- Linear regression: $\hat{y} = \text{intercept} + \text{slope} \cdot x$

$$\hat{y} = b_0 + b_1 \cdot x$$

The coefficients for these regression models are found by minimizing root mean square error (RMSE)

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$



[Regression line app](#)

Review: Multiple regression

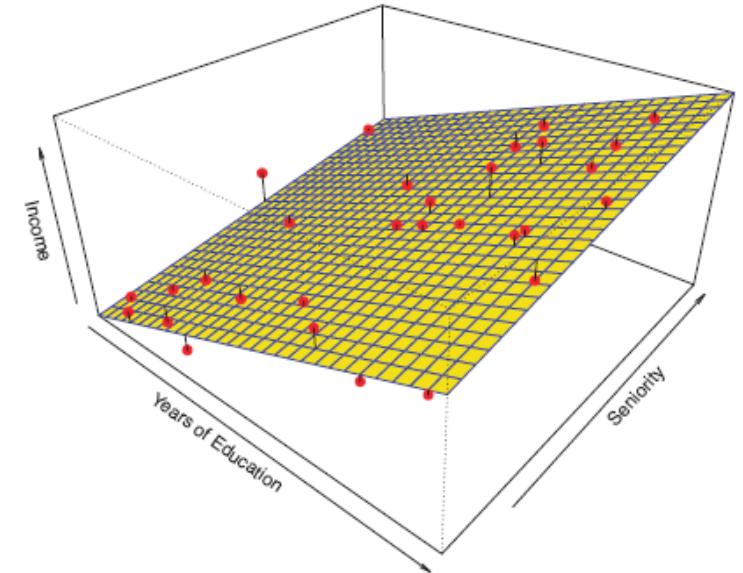
In multiple regression we try to predict a quantitative response variable y using several features x_1, x_2, \dots, x_k

We estimate coefficients using a data set to make predictions \hat{y}

$$\hat{y} = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + \dots + b_k \cdot x_k$$



Learn the b_i 's on the training set. Assess prediction accuracy on test set.



Review: Linear regression models in scikit-learn

We can use scikit-learn to create linear regression models

- You can also use the stats models package to do this

```
linear_model = LinearRegression() # construct a linear regression model
```

```
linear_model.fit(X_train_features, y_train) # train the classifier
```

"Learns" b_0, b_1, \dots by minimizing
the RMSE on the training data

train the classifier

Numeric values
to predict

```
y_predictions = linear_model.predict(X_test_features) # make predictions
```

```
RMSE = np.sqrt(np.mean((y_test - y_predictions)**2)) # get the RMSE
```

Review: Inference for regression

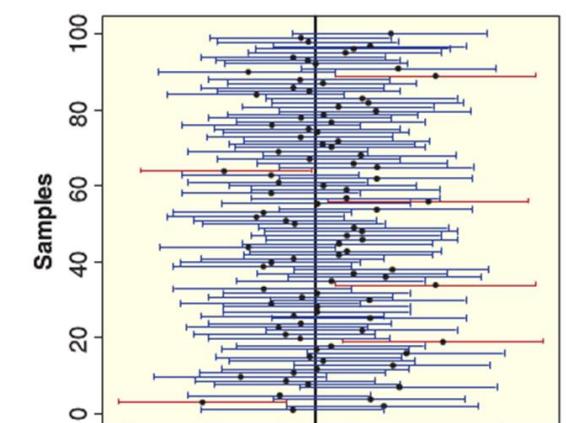
We can run a hypothesis test to see if there is a linear relationship between a feature x , and a response variable y

- $H_0: \beta_1 = 0$ (slope is 0, so no linear relationship between x and y)
- $H_A: \beta_1 \neq 0$

The confidence interval for the slope coefficients:

```
import statsmodels.api as sm  
  
sm_linear_model = sm.OLS(y_train, X_train_with_constant).fit()  
  
sm_linear_model.summary()
```

$$\beta_1$$



Unsupervised learning

Supervised learning and unsupervised learning

In **supervised learning** we have a set of features X, along with a label y

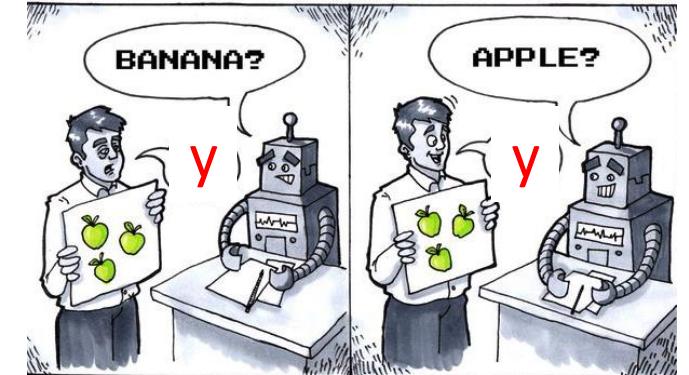
- We use the features X to predict y on new data

In **unsupervised learning**, we have features X, but **no** response variable y

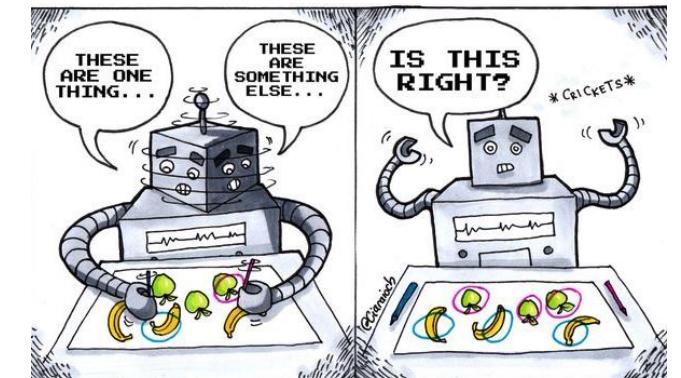
Unsupervised learning can be useful in order to find structure in the data and to visualize patterns

A key challenge in unsupervised learning is that there is no real ground truth response variable y

- So we don't have measures like the mean prediction accuracy



Supervised Learning



Unsupervised Learning

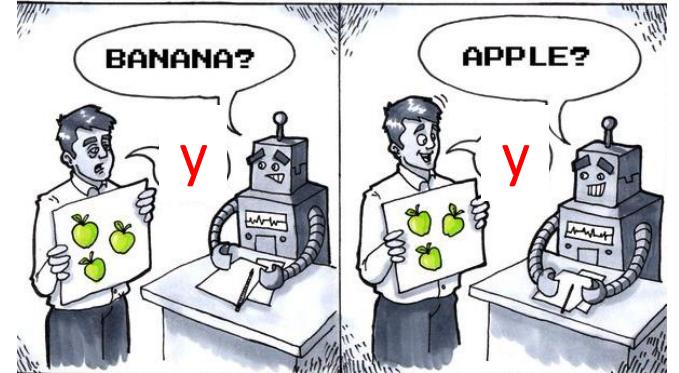
Unsupervised learning

Given we are almost at the end of the semester, we will focus on clustering, which is one type of unsupervised learning:

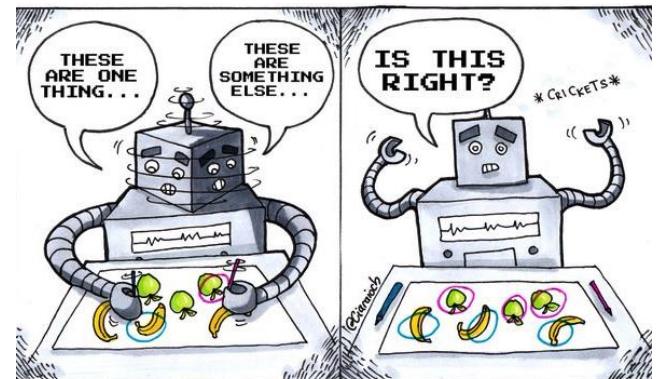
In **clustering** we try to group similar data points together

Another type of unsupervised learning:

- **Dimensionality reduction** where we try to find a smaller set of features that captures most of the variability original larger feature set
 - E.g., principal component analysis (PCA)



Supervised Learning



Unsupervised Learning

Clustering



So tell me how
many clusters do
you see?



Clustering

Clustering divides n data points x_i 's into subgroups

- Data points in the same group are similar/homogeneous
- Data points in different groups are different from each other

A diagram illustrating a data matrix. A red bracket on the left indicates the number of rows is n . A red bracket at the top indicates the number of columns is p . The matrix itself is a grid of cells, with the first few rows labeled as $x_{11}, x_{12}, \dots, x_{1p}$, $x_{21}, x_{22}, \dots, x_{2p}$, and so on down to $x_{n1}, x_{n2}, \dots, x_{np}$. The cells are outlined in black, with the first two rows highlighted by a blue border.

x_{11}	x_{12}	\cdots	x_{1p}
x_{21}	x_{22}	\cdots	x_{2p}
\vdots	\vdots	\ddots	\vdots
x_{n1}	x_{n2}	\cdots	x_{np}

Examples:

- Examining gene expression levels to group cancer types together
- Examining consumer purchasing behavior to perform market segmentation

Clustering can be:

- **Flat:** no structure beyond dividing points into groups
- **Hierarchical:** Population is divided into smaller and smaller groups (tree like structure)

K-means clustering

K-means clustering partitions the data into K distinct, non-overlapping clusters

- i.e., each data point x_i belongs to exactly one cluster C_k

The number of clusters, K , needs to be specified prior to running the algorithm

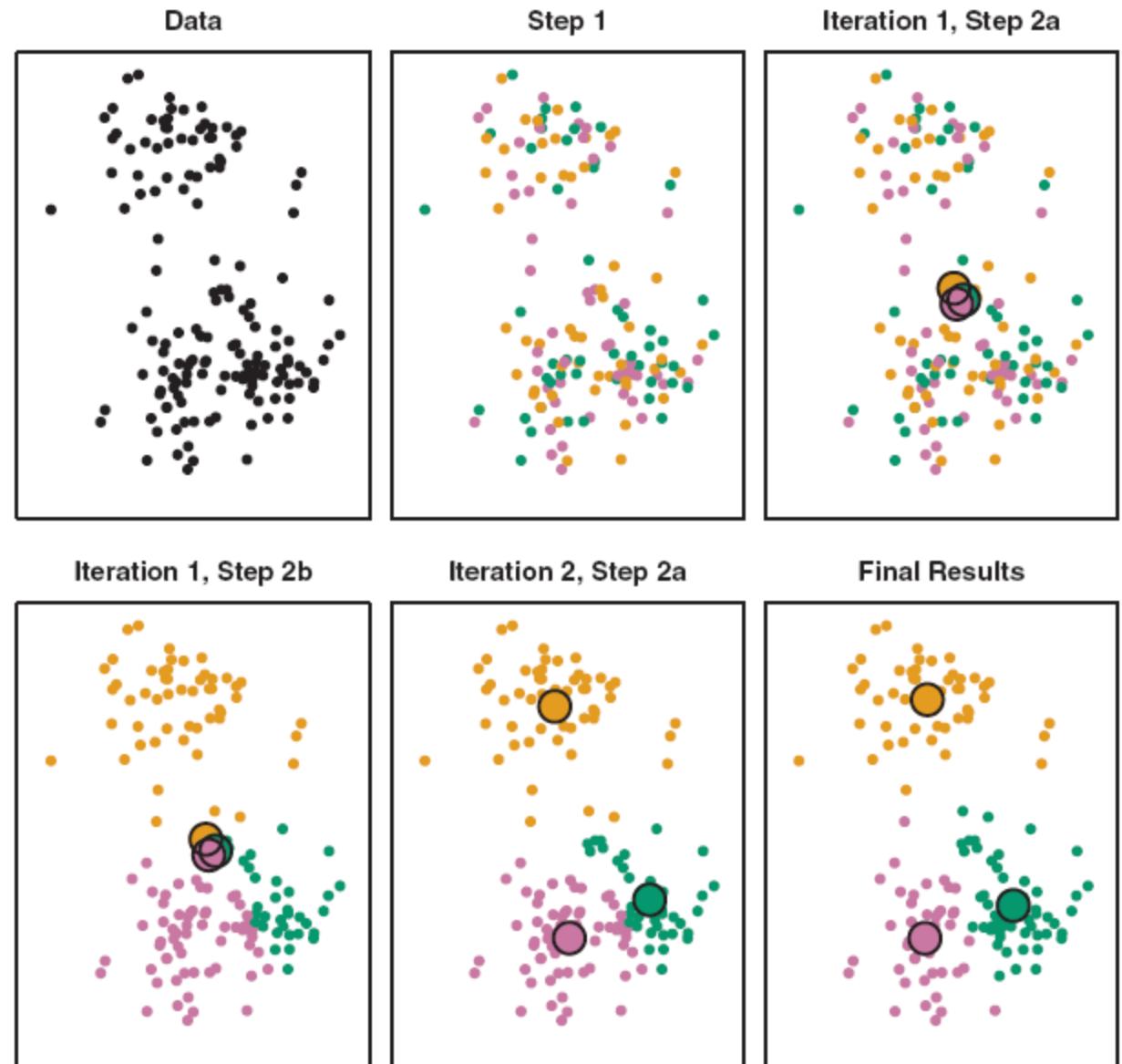
The goal is to minimize the within-cluster variation

- e.g., to make the Euclidean distance for all points within a cluster as small as possible

Finding the exact optimal solution is computationally intractable (there are k^n possible partitions), but a simple algorithm exists to find a local optimum which is often works well in practice.

K-means clustering

1. Randomly assign points to clusters C_k
2. Calculate cluster centers as means of points in each cluster
3. Assign points to the closest cluster center
4. Recalculate cluster center as the mean of points in each cluster
5. Repeat steps 3 and 4 until convergence



K-means clustering

Because only a local minimum is found, different random initializations will lead to different solutions

- One should run the algorithm multiple times to get better solutions

Let's explore this in Jupyter!



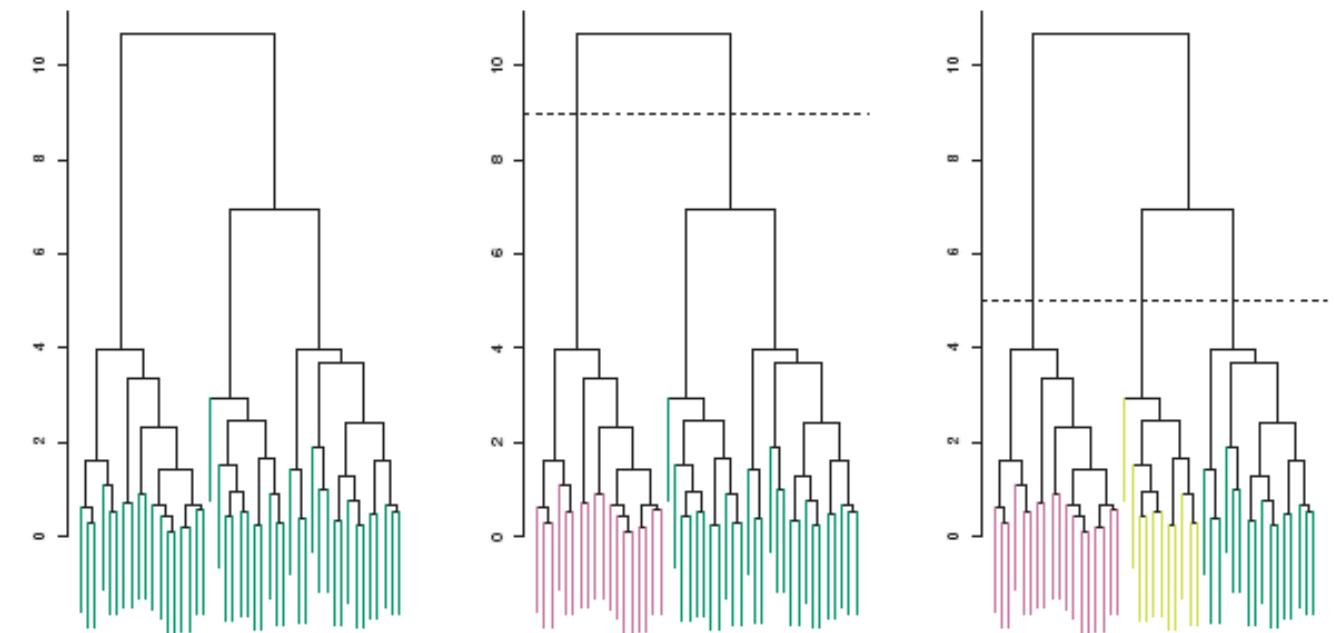
Hierarchical clustering

Hierarchical clustering

In **hierarchical clustering** we create a dendrogram which is a tree-based representation of successively larger clusters.

We can cut the dendrogram at any point to create as many clusters as desired

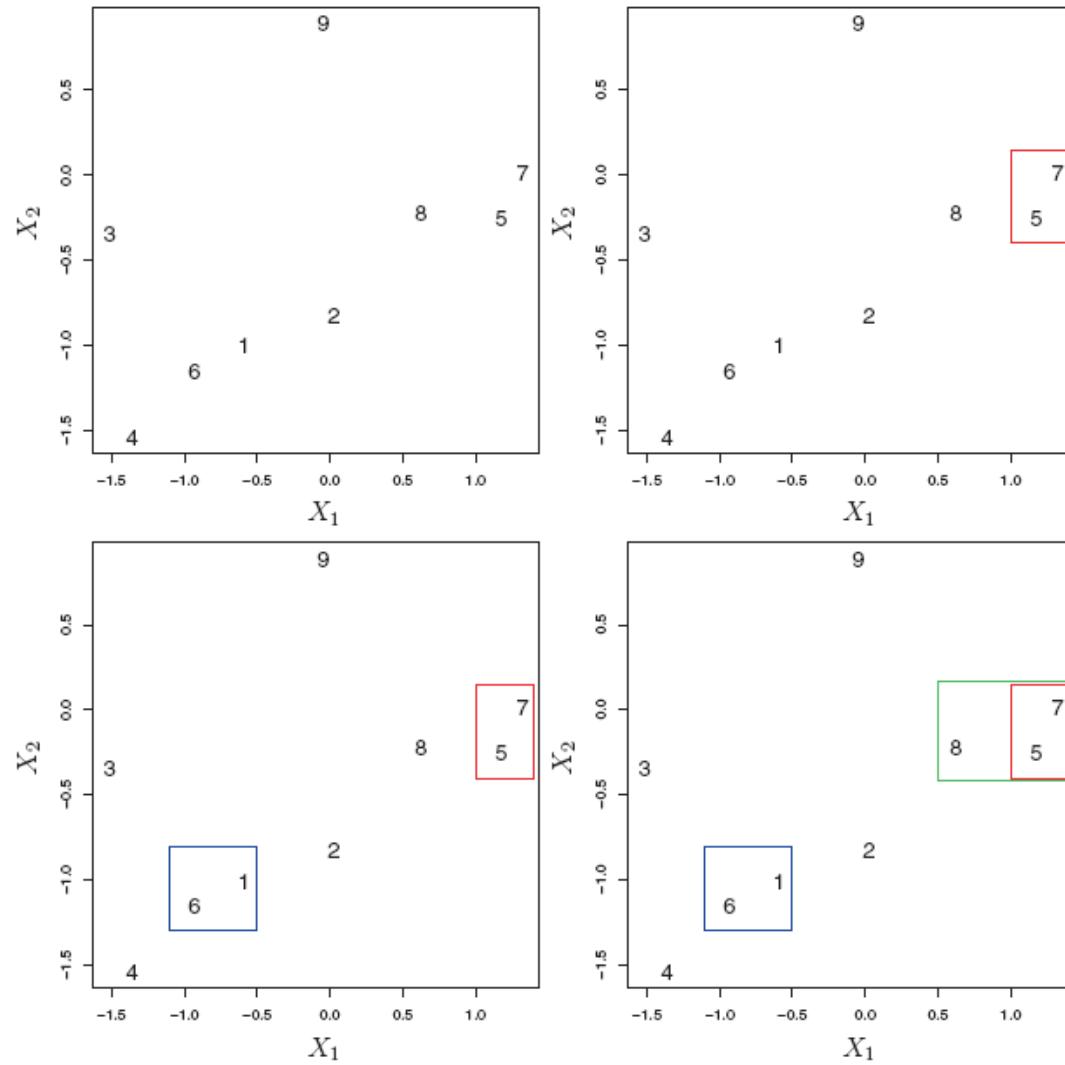
- i.e., don't need to specify the number of clusters, K , beforehand



Hierarchical clustering

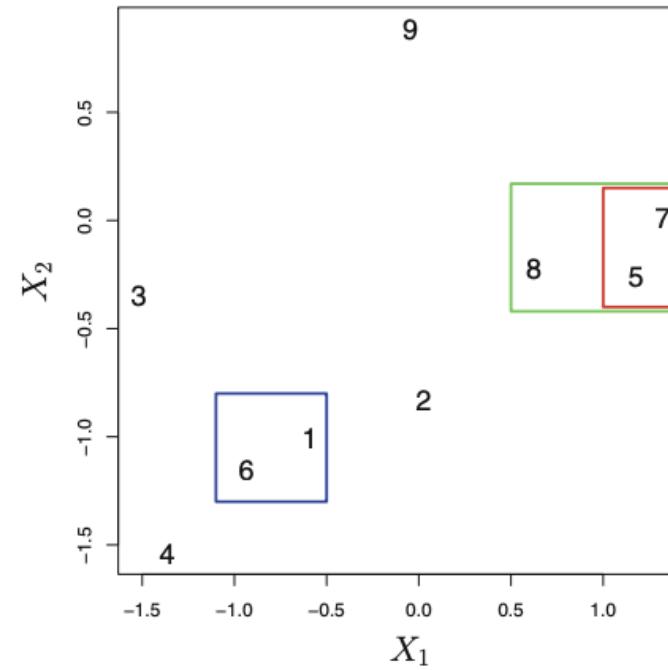
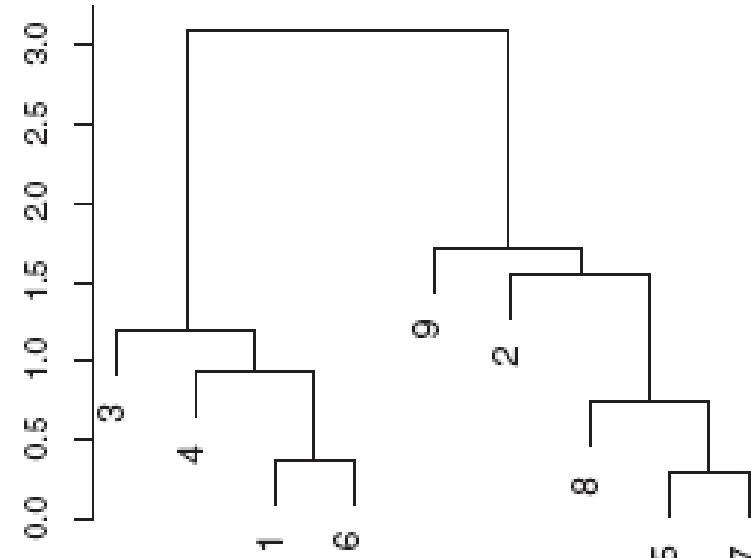
We can create a hierarchical clustering of the data using simple bottom-up agglomerative algorithm:

1. Choosing a (dis)similarity measure
 - E.g., The Euclidean distance
2. Initializing the clustering by treating each point as its own cluster
3. Successively merging the pair of clusters that are most similar
 - i.e., calculate the similarity between all pairs of clusters and merging the pair that is most similar
4. Stopping when all points have been merged into a single cluster



Hierarchical clustering

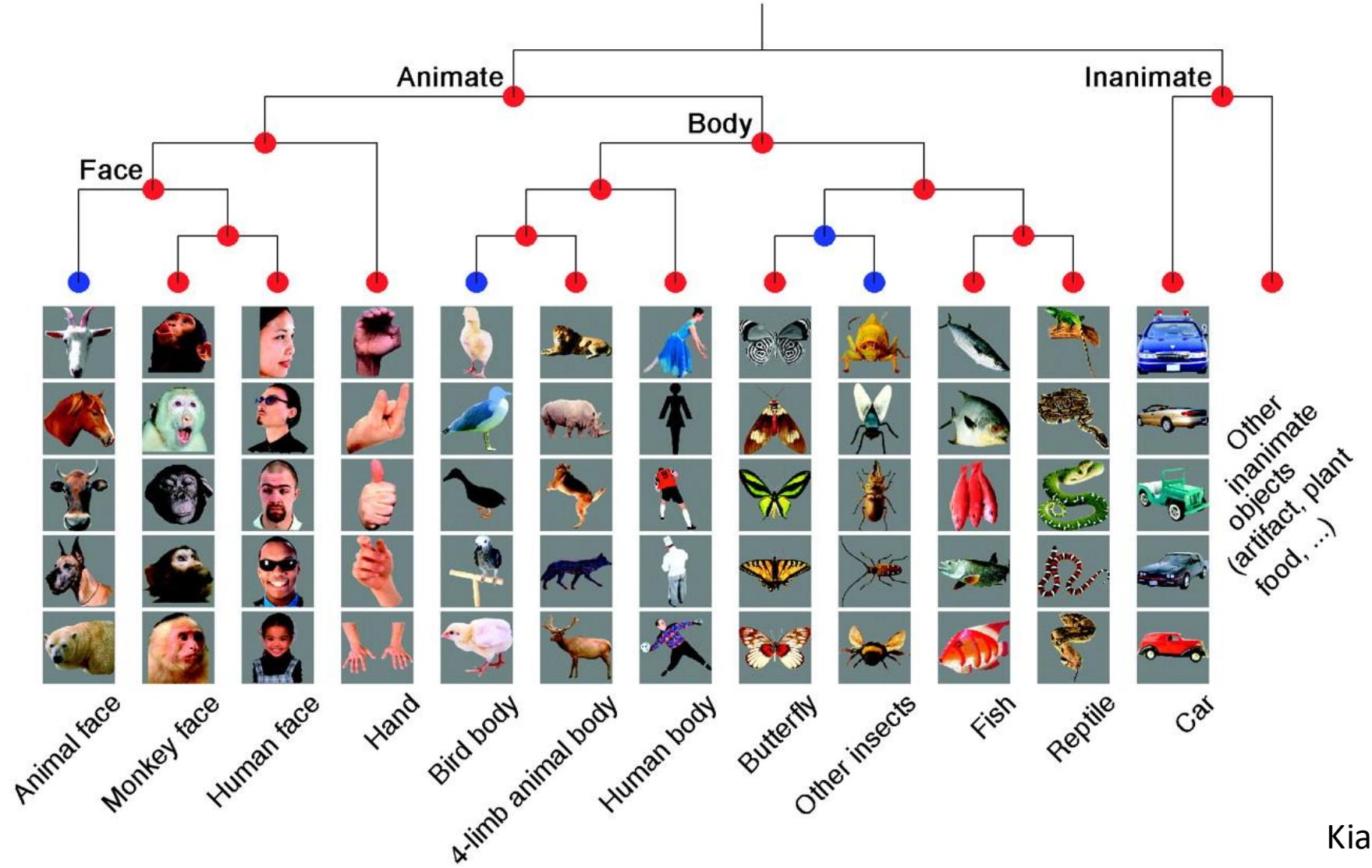
The vertical height that two clusters/points merge show how similar the two *clusters* are



Note: horizontal distance between *individual points* is not important:

- point 9 is considered as similar to point 2 as it is to point 7

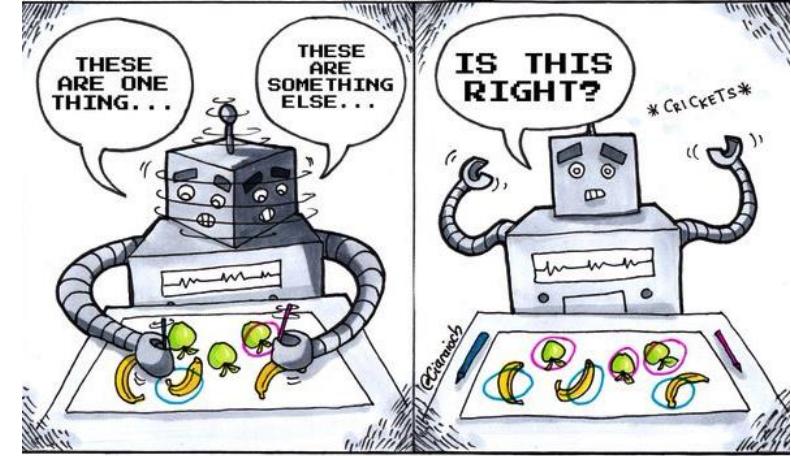
Hierarchical clustering example



Issues with clustering

Choices made can effect the results:

- Feature normalization and/or dissimilarity measure
- K-means: choice of K
- For hierarchical cluster: linkage and cut height



Unsupervised Learning

Potential approaches to deal with these issues:

- Try a few methods and see if one gives interesting/useful results
- Validate that you get similar results on a second set of data

Let's explore this in Jupyter!

Object oriented programming

Object-oriented programming

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain:

1. **data**: in the form of fields
 - often known as attributes or properties
2. **methods**: functions that operate on the data

Have we seen any objects in this class yet?

```
knn = KNeighborsClassifier(n_neighbors = 1)      # constructor  
knn.fit(X_train, y_train)                      # method
```

Using object-oriented programming makes it easier to design larger software systems

- E.g., useful for writing packages, such as the ones we have used in this class

Object-oriented programming

A **class** defines what the object is

- i.e., classes have code that lists the types of data stored and the methods

An **object** is a realization of classes (often called an instance of a class)

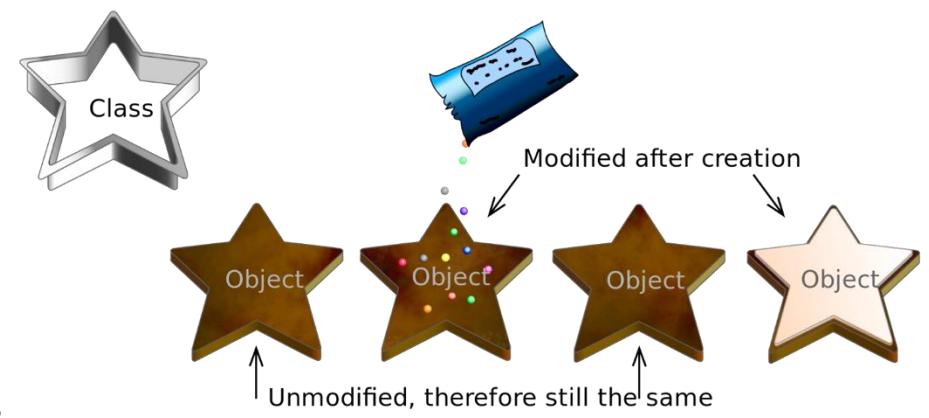
- Stores actual data that can be manipulated with methods

```
knn1 = KNeighborsClassifier(n_neighbors = 1)
```

```
knn5 = KNeighborsClassifier(n_neighbors = 5)
```

Useful because it allows us to have many object instances

- E.g., A KNN classifier trained on different data, different values of k, etc.



Object-oriented programming

A **constructor** defines what should happen when an object is created

- In Python the constructor is defined with the `__init__(self)` method

```
class KNN:
```

```
# Constructor
def __init__(self, n_neighbor):
    self.k = n_neighbor
```

```
# create an instance of an object
my_knn = KNN(n_neighbor = 5)

# get the stored value of k
my_knn.k
```

Object-oriented programming

Methods are functions that are applied to the data stored in the object

For example, the `.fit()` method in an KNN object would save training and test data

```
class KNN:
```

```
... # constructor, etc. above
```

```
def fit(self, X_train, y_train):
```

```
    self.X_train = X_train
```

```
    self.y_train = y_train
```

```
# call the .fit() method  
my_knn.fit(X_train, y_train)
```

```
# retrieve the training data  
my_knn.X_train
```

Object-oriented programming

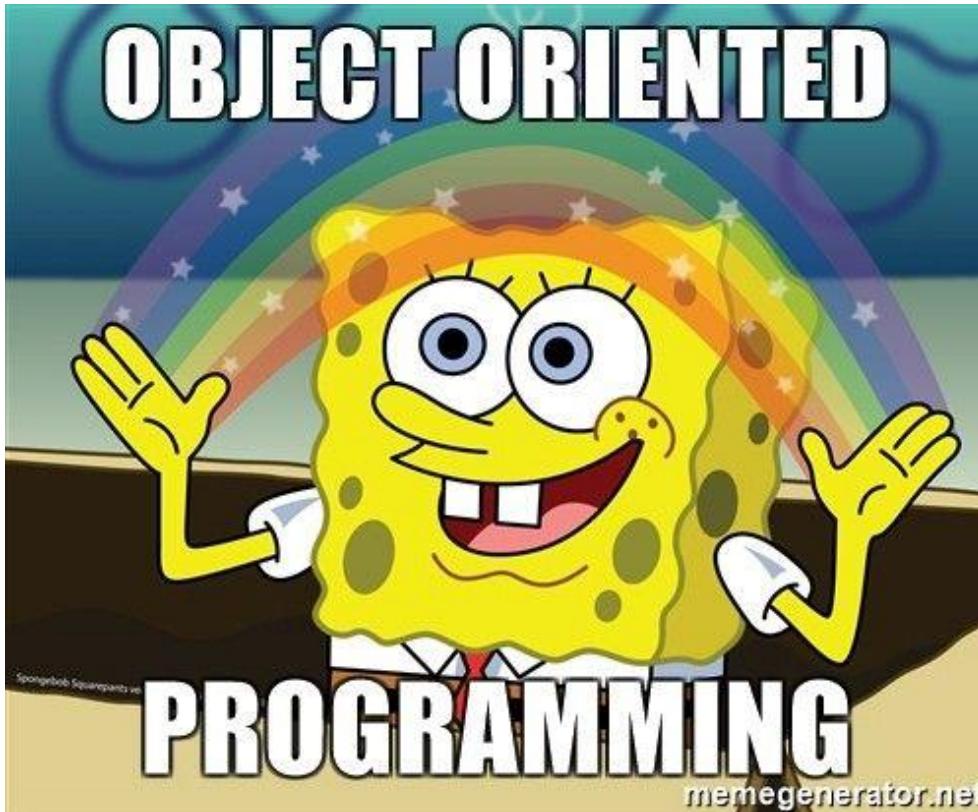
The `__str__` method defines what should happen when the `print()` function is called on the object

```
# str method
def __str__(self):
    return 'I am an object'
```

```
# create an instance of an object
my_knn = KNN(n_neighbor = 5)

# print the object
print(my_knn)
```

Questions?



Let's explore this in Jupyter!

YData: Introduction to Data Science



Lecture 25: Odds and ends

Overview

Review of:

- Clustering
- Objected oriented programming

Miscellaneous topics

- Widgets for interactive analyses
- Creating a webpage on GitHub
- Installing Jupyter on your own computer

If there is time

- Ethics in Data Science

ODDS

AND

ENDS

Project timeline

~~Sunday, November 17th~~

- Projects are due on Gradescope at 11pm
- Email a pdf of your project to your peer reviewers
 - A list of whose paper you will review is posted on Canvas
 - Fill out the draft reflection on Canvas

~~Sunday, November 24th~~

- Jupyter notebook files with your reviews need to be sent to the authors
- A template for doing your review has been posted

Sunday, December 8th

- Project is due on Gradescope
 - Add peer reviews to the Appendix of your project

~~Homework 9 has been posted~~

- It is due December 1st

That went as planned,
said no project ever.



your eCards
homelifecards.com

~~Final exam review session~~

- Monday December 9th at 3pm
- In this room

Review of Clustering



So tell me how
many clusters do
you see?



How many clusters?



Six Clusters



Two Clusters



Four Clusters

Supervised learning and unsupervised learning

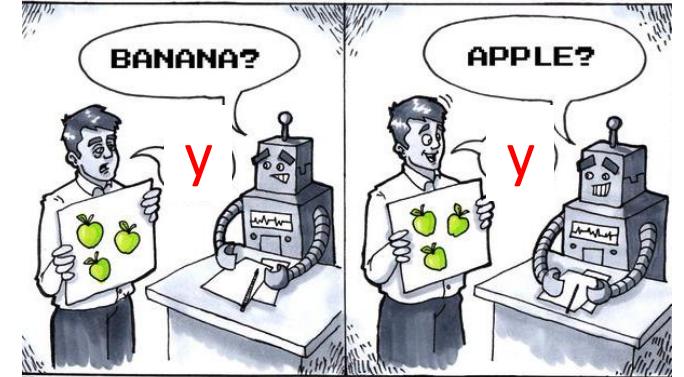
In **unsupervised learning**, we have features X, but **no response variable y**

- Unsupervised learning can be useful in order to find structure in the data and to visualize patterns,
- but there is no real ground truth response variable y

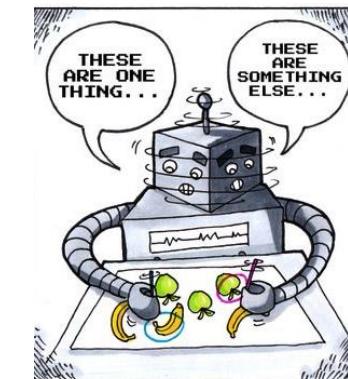
1. **Clustering** where we try to group similar data points together

2. **Dimensionality reduction** where we try to find a smaller set of features that captures most of the variability in the data

- Principal component analysis (PCA)



Supervised Learning



Unsupervised Learning

Clustering

Clustering divides n data points x_i 's into subgroups

- Data points in the same group are similar/homogeneous
- Data points in different groups are different from each other

A diagram showing a data matrix with n rows and p columns. The matrix is represented by a grid of boxes. A red bracket on the left indicates the number of rows n . A red bracket at the top indicates the number of columns p . The first row is highlighted in blue, and the second row is also highlighted in blue. The last row is highlighted in red.

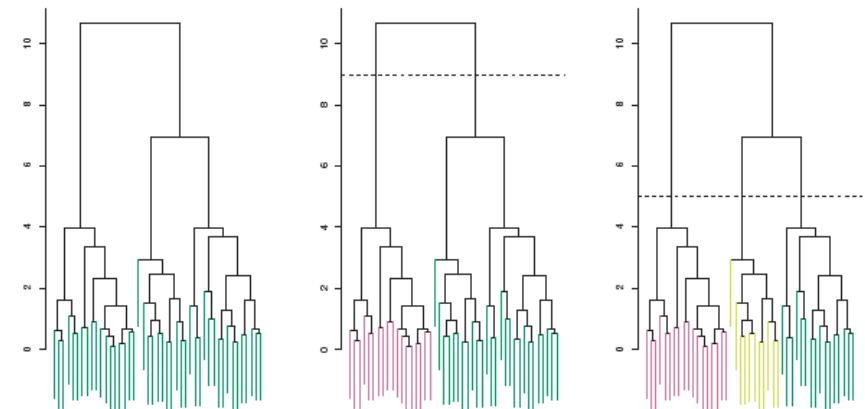
x_{11}	x_{12}	\cdots	x_{1p}
x_{21}	x_{22}	\cdots	x_{2p}
\vdots	\vdots	\ddots	\vdots
x_{n1}	x_{n2}	\cdots	x_{np}

Flat clustering: k-means

- Specify k clusters in advance and each point is assigned to one cluster

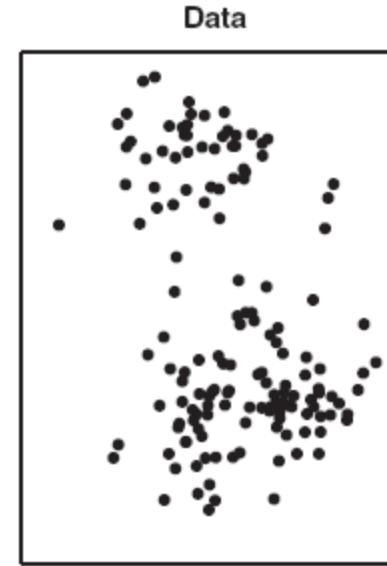
Hierarchical clustering

- Tree of nested clusters is created and we “cut” the tree to get a particular number of clusters



K-means clustering

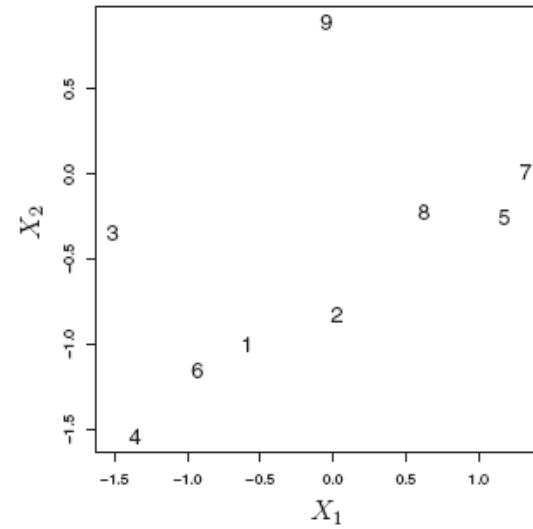
1. Randomly assign points to clusters C_k
2. Calculate cluster centers as means of points in each cluster
3. Assign points to the closest cluster center
4. Recalculate cluster center as the mean of points in each cluster
5. Repeat steps 3 and 4 until convergence



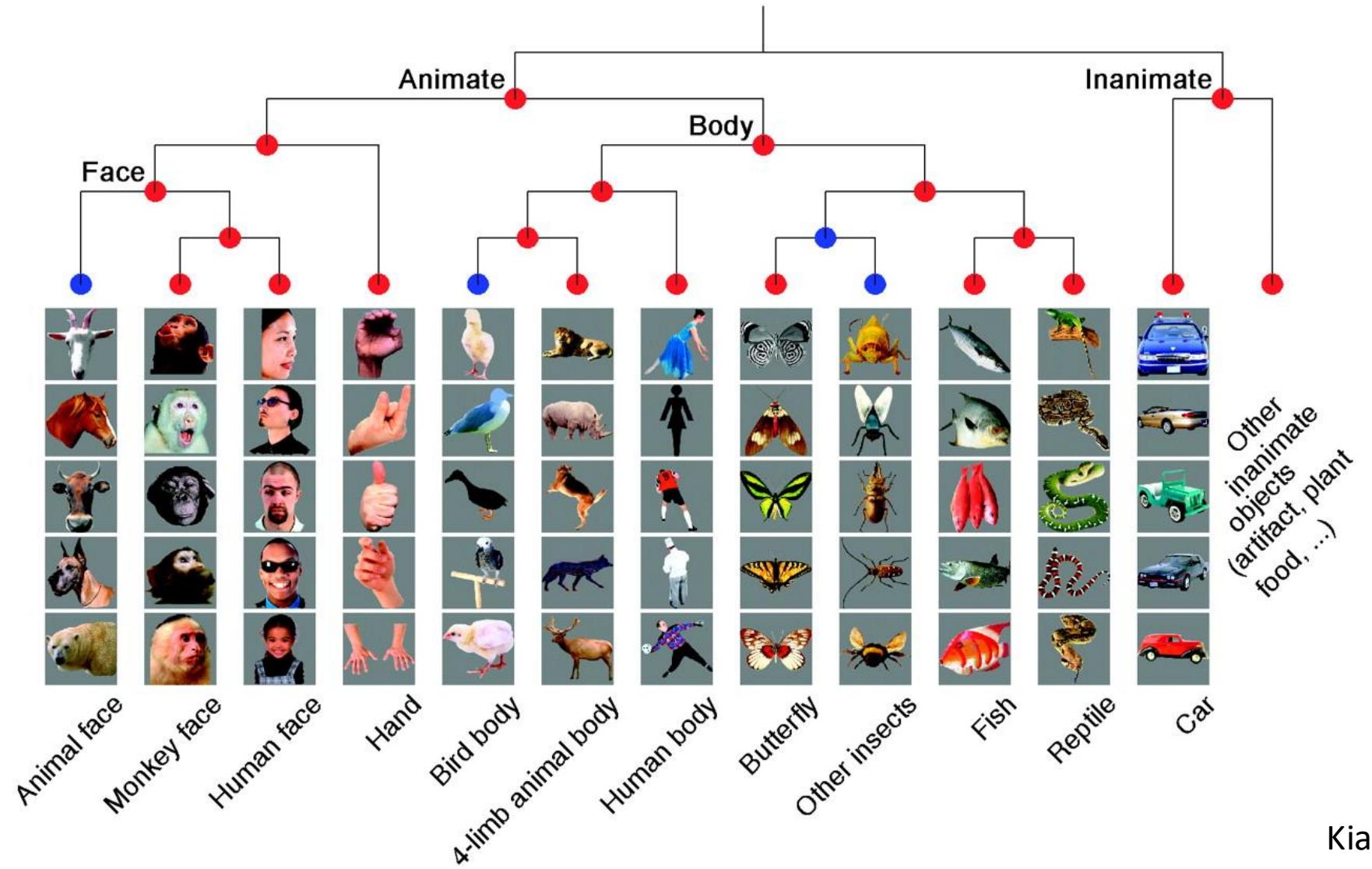
Hierarchical clustering

We can create a hierarchical clustering of the data using simple bottom-up agglomerative algorithm:

1. Choosing a (dis)similarity measure
 - E.g., The Euclidean distance
2. Initializing the clustering by treating each point as its own cluster
3. Successively merging the pair of clusters that are most similar
 - i.e., calculate the similarity between all pairs of clusters and merging the pair that is most similar
4. Stopping when all points have been merged into a single cluster



Hierarchical clustering example



Review: Object oriented programming

Object-oriented programming

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain:

1. **data**: in the form of fields
 - often known as attributes or properties
2. **methods**: functions that operate on the data

Using object-oriented programming makes it easier to design larger software systems

- E.g., useful for writing packages, such as the ones we have used in this class

Object-oriented programming

A **class** defines what the object is

- i.e., classes have code that lists the types of data stored and the methods

An **object** is a realization of classes (often called an instance of a class)

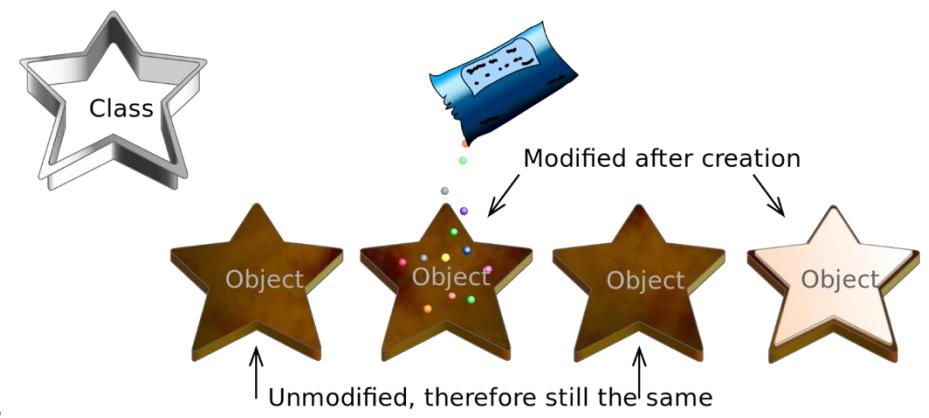
- Stores actual data that can be manipulated with methods

```
knn1 = KNeighborsClassifier(n_neighbors = 1)
```

```
knn5 = KNeighborsClassifier(n_neighbors = 5)
```

Useful because it allows us to have many object instances

- E.g., A KNN classifier trained on different data, different values of k, etc.



Object-oriented programming

A **constructor** defines what should happen when an object is created

- In Python the constructor is defined with the `__init__(self)` method

```
class StorageObj:
```

```
# Constructor
def __init__(self, a_num):

    self.n = a_num
    self.b = False
    self.s = "Hi"
```

```
# create an instance of an object
my_obj = StorageObj(3)
```

```
# get the value stored in n
my_obj.n
my_obj.s
```

Object-oriented programming

Methods are functions that are applied to the data stored in the object

For example, the `.fit()` method in an KNN object would save training and test data

```
class StorageObj:
```

```
... # constructor, etc. above
```

```
def mult_num(self, val = 7):
```

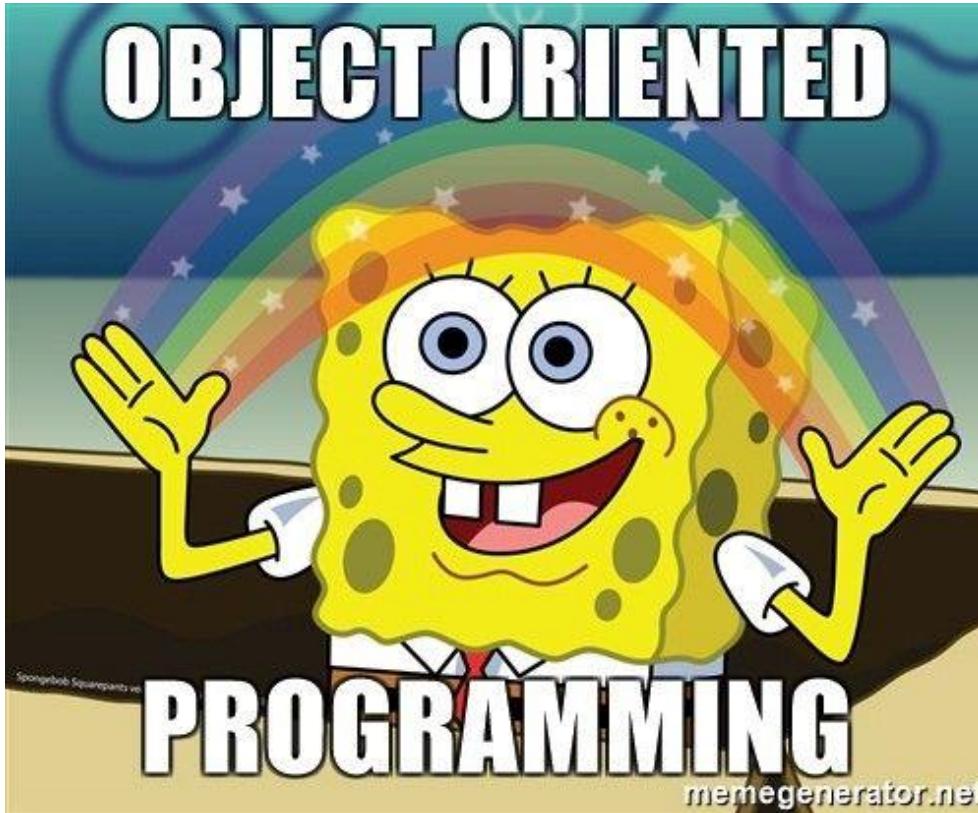
```
    return self.n * val
```

```
# create an instance of an object  
my_obj = StorageObj(3)
```

```
# call the .mult_num() method  
my_obj.mult_num()
```

```
my_obj.n = 10  
my_obj.mult_num(5)
```

Questions?



Let's quickly review this in Jupyter!

Jupyter widgets

Jupyter widgets

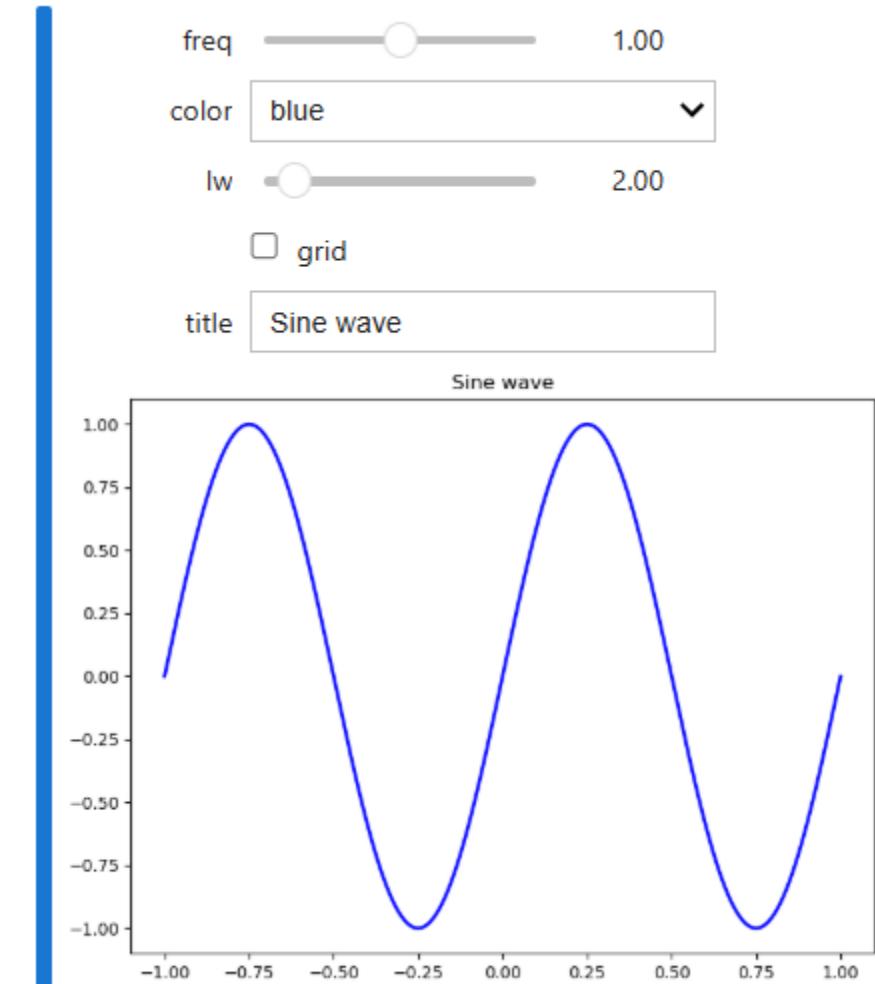
Jupyter widgets allow you to add buttons, sliders, etc. to a Jupyter notebook so that you can create interactive visualizations

We can add widgets by importing:

```
import ipywidgets as widgets
```

We can then create a widget using:

```
bu = widgets.Button(description="Click")
```



Jupyter widgets

There are several ways to connect widgets to figures

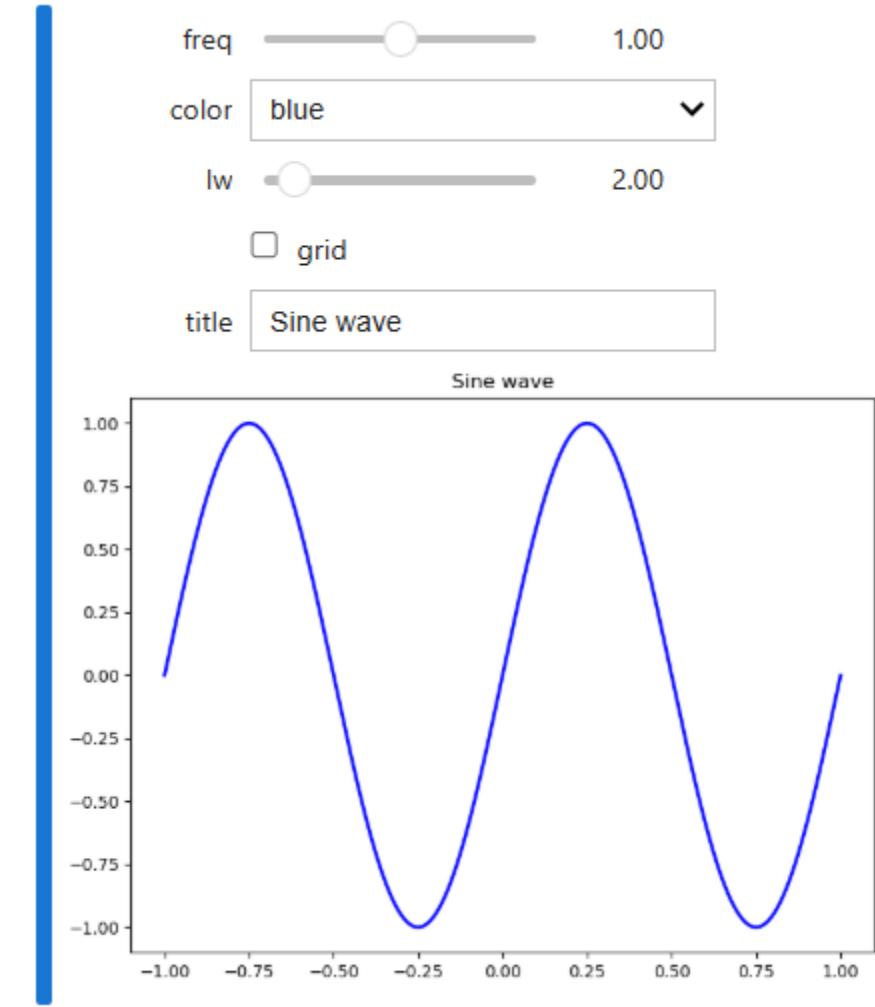
- One way is through `widgets.interact()`

```
def bandwidth_widget(bw = 1):  
    sns.kdeplot(cars.horsepower,  
                lw=3,  
                bw_adjust=bw)
```

```
widgets.interact(bandwidth_widget,  
                 bw = (.1, 3));
```



Adds a slider widget because this is a range of continuous values



Let's try it in Jupyter!

Hosting a webpage on GitHub

Hosting a webpage on GitHub

You've visiting many webpages in your life

- E.g., <https://canvas.yale.edu/>

Let's now discuss how we can create our own webpage and host it for free using GitHub.com

Webpages

Web pages are written in Hypertext Markup Language (HTML)

- Typically webpages end with the extension .html
 - Have you seen any files like this related to this class?

Webpage consists of **text** and **tags**

- Text is the content displayed on the webpage
- Tags allow one to insert links, images, and other meta-data

Tags have the form <></>

- For example, we can make text bold using

The word **bold** is bold

Example of a basic web page

```
<html>  
<title> My cool page </title>  
  
<body>  
    This is my cool webpage  
      
<br><br>  
    Here is <a href="https://canvas.yale.edu/">Canvas </a>  
  
</body>  
  
</html>
```

This is my **cool** webpage

This is a link to Canvas

Let's create this simple webpage

Hosting webpages on GitHub pages

Webpages we've created

Your homework and your class project were saved as .html documents before you printed them to pdfs

GitHub.com is a service that allows people to share code

- Also allows one to host webpages for free!
- So if you save your project as a .html document, you can share it on the web!

Let's go through the steps now to host a webpage on GitHub



Installing Python/Jupyter on your own computer

Installing Jupyter on your own computer

Throughout this semester we have been using Jupyter on the YCRC cluster

Yale Center for Research Computing

Once the semester is over, you will lose access to this cluster

- So please be sure to backup any work you want to save!

colab

If you want to keep using Jupyter, you will need to either:

- Use Google Colab
- Install Jupyter on your own computer

ANACONDA®

Environments and Anaconda



In order use packages on your own computer (e.g. pandas, matplotlib, etc) you first need to install

Environments allow one to switch between different versions of packages

- This is useful because packages are often updated

One of the most popular environment managers **conda**

- i.e., it allows you to install packages and switch between environments that have different packages installed

To download anaconda please go to:

<https://www.anaconda.com/download/>



Creating a conda environment



Once anaconda is installed, we can create an environment that has the data science packages we need.

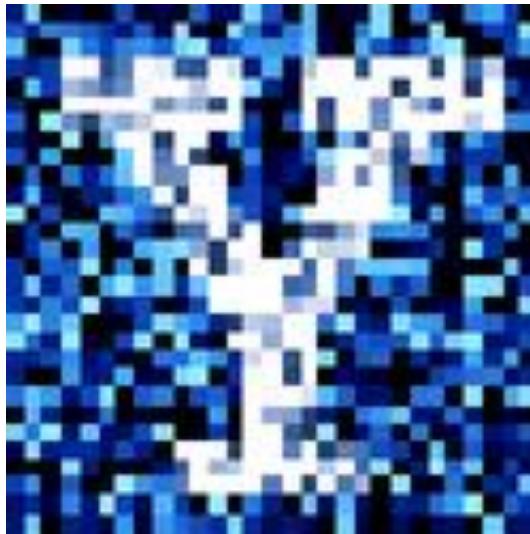
One of the easiest ways to do this is to use a .yml file that has a list of the packages we would like to install

- E.g., ydata123.yml

Let's try this now...



YData: Introduction to Data Science



Lecture 26: Web scraping, LLMs Ethics, and wrap-up

Overview

Quick review of web pages

Web scraping

Brief demo of running a LLM/chatbot
in Python

Ethics

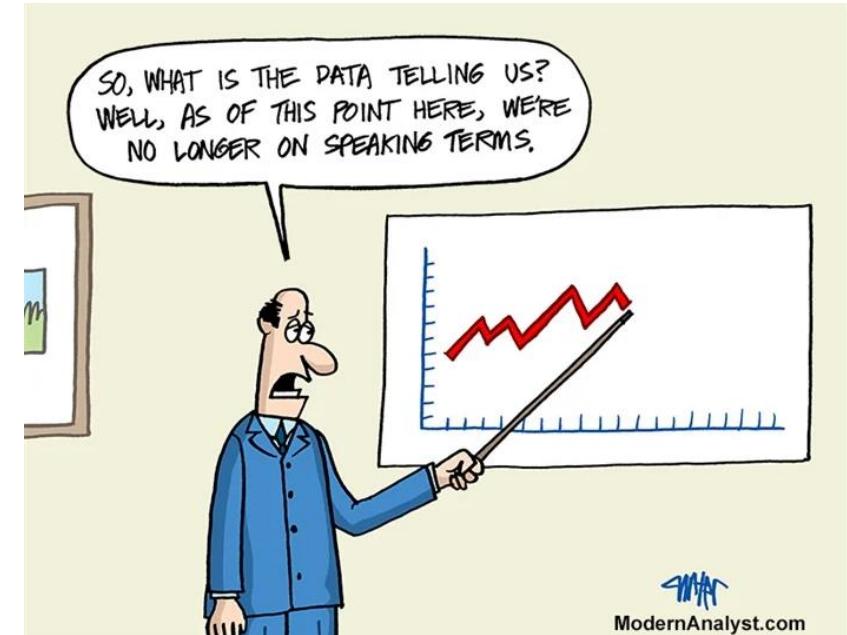
Wrap-up



Project timeline

Sunday, December 8th

- Project is due on Gradescope
 - Add peer reviews to an Appendix of your project



Please also fill out the final project reflection on Canvas!

- Will be very valuable to have your feedback on how the project and class overall went

Announcement

Exam review session: Monday December 9th from 3-4pm in this room

Final exam: Monday December 16th at 7pm **in this classroom!**



Quick review of webpages

Review: Webpages

Webpages are written in Hypertext Markup Language (HTML)

- Typically webpages end with the extension .html

Webpage consists of **text** and **tags**

- Text is the content displayed on the webpage
- Tags allow one to insert links, images, and other meta-data

Tags have the form <> </>

- For example, we can make text bold using

The word **bold** is bold

Review: Example of a basic webpage

```
<html>  
<title> My cool page </title>  
<body>  
    This is my cool webpage  
      
<br><br>  
    Here is <a href="https://canvas.yale.edu/">Canvas </a>  
</body>  
</html>
```

This is my **cool** webpage

This is a link to Canvas

Additional webpage tags

Headers: <h2> Second biggest header </h2>

Inserting images:

Lists:

```
<ul>
    <li> Item 1 </li>
    <li> Item 2 </li>
</ul>
```

Additional webpage tags - tables

```
<table>
  <tr>
    <td> 1 </td>
    <td> 2 </td>
  </tr>           1 2
  <tr>           4 5
    <td> 4 </td>
    <td> 5 </td>
  </tr>
</table>
```

Let's take a quick look at some webpages

- [A page I created hosted on GitHub](#)
- [Department of Statistics and Data Science](#)
- [Wikipedia page](#)

Let's look at the source code and inspect elements of these pages using the Chrome web browser

[W3School](#) is a good site to learn more about creating webpages

Web scraping

Web scraping

Web scraping is the processing of extracting webpage content through code

In Python, the most widely used package for web scraping is [Beautiful Soap](#) which can extract relevant information from webpages

```
import requests  
from bs4 import BeautifulSoup
```

BeautifulSoup

Web scraping

We first need to download the webpage using the `requests` module

```
# the web address
url = "http://blah.com/page.html"

# request the webpage
response = requests.get(url)

# see if the request was successful
print(response)
```

Common responses

- **200:** success
- **400:** URL was badly formed
 - i.e., bad web address
- **403:** "Forbidden" Server understood the request but refused it
 - Often because it detects you're trying to scrape the site

Web scraping

We can parse a successful response with Beautiful Soap using:

```
soup = BeautifulSoup(response.text, 'html.parser') # create the object  
  
tables = soup.find_all('table') # get all tables on a webpage  
headers = soup.find_all(['h1', 'h2']) # get all tables on a webpage  
# etc.
```

The `.find_all()` method returns tag elements which can be used to extract the relevant information related to particular tags

Let's try this in Jupyter!

Brief discussion of Large Language Models

Brief discussion of Large Language Models

Large language models (LLMs) are taking over the world

- They can write code and even [analyze data](#)

One can download free, open source, LLMs through the [Hugging Face platform](#)

Let's very briefly look at running a LLM locally on our own computers...

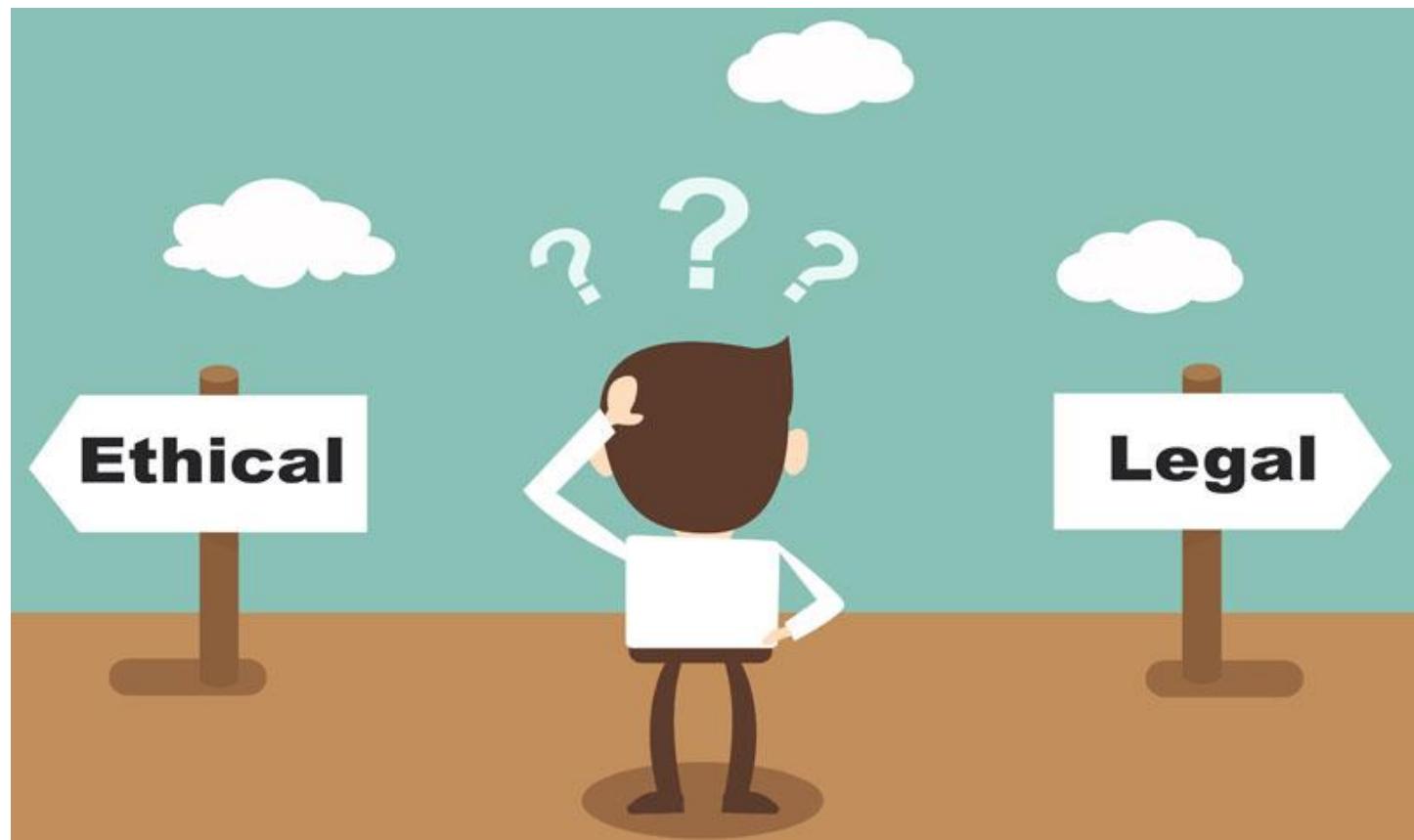


ChatGPT

Gemini



Ethics



Ethics in Data Science

Ethics of:

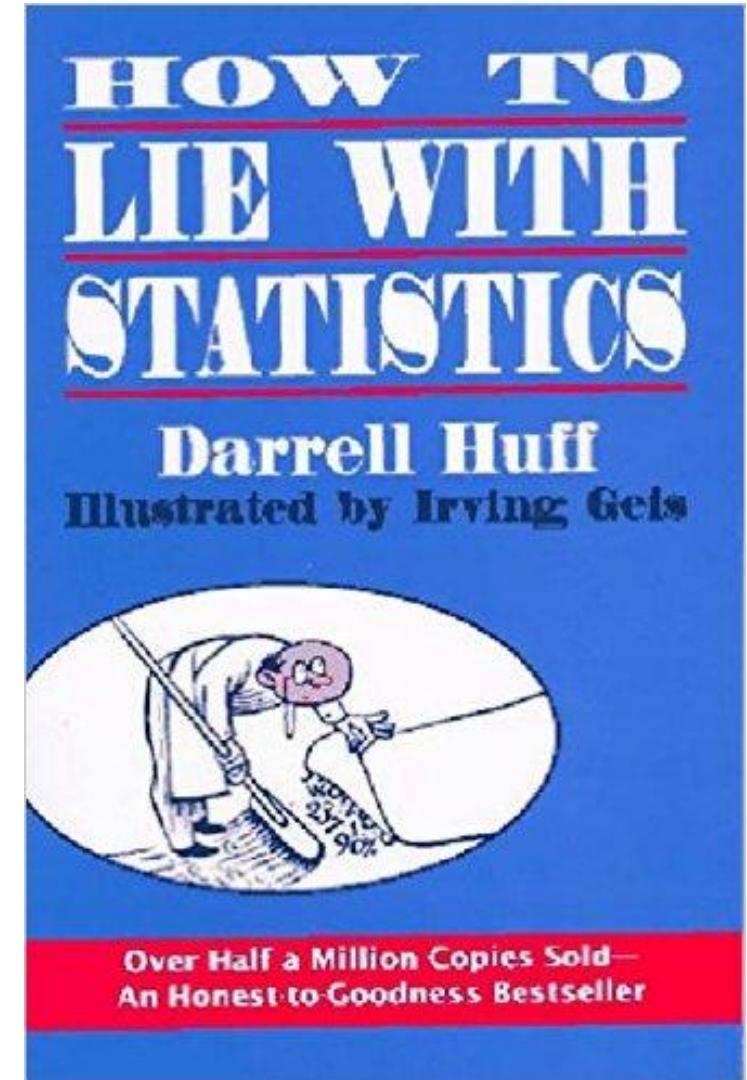
1. Data presentation
2. Using valid data
3. Data scraping TOS and privacy
4. Reproducibility
5. Citations/peer review
6. Disclosure
7. Ethics in Statistical analyses
8. Ethics of creating powerful tools

1. Ethics of data presentation

Data should be displayed in an honest way that gives an accurate picture of trends

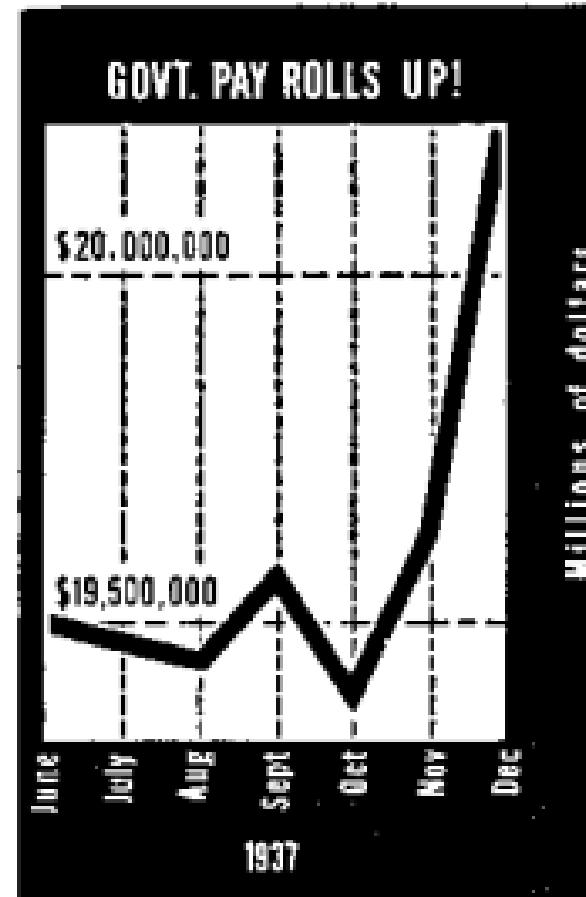
Darrell Huff wrote a classic book in the 1950's pointing out ways that people lie with statistics

The book was banned as training material at the VA



Ethics of data presentation

What is potentially misleading with this figure?



From a 1938 article in Dun's Review titled 'GOVERNMENT PAY ROLLS UP!'

Alumni Survey Results

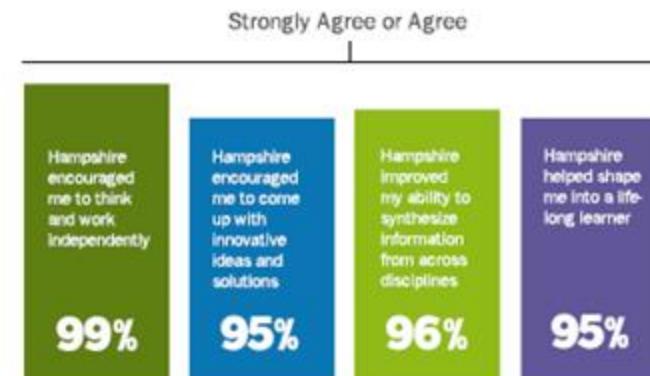
2. Using valid data

Is almost everyone satisfied with Hampshire College?

As part of a strategic-planning process, in spring 2013 Hampshire College launched a survey of alums. Via email, the College invited 8,160 alums to fill out an online questionnaire administered by the campus's Alumni and Family Relations and Institutional Research offices. A total of 1,920 surveys were completed, yielding a response rate of 24%.

Note: The percentages in the data (below) are based on the number of responses received for each question.

To what extent do you agree with the following statements?



Please rate your student experience at Hampshire.



3. Data scraping, terms of service and privacy

Scraping publicly available data is fine (e.g., Wikipedia) but what about scraping data if:

- It violates a website's Terms of Service?
- User privacy?

Kirkegaard and Bjerrekaer scraped okcupid and data on 68,371 users publicly available including usernames, dating preferences, etc.

Submitted: 8th of May 2016
Published: 3rd of November 2016

The OKCupid dataset: A very large public dataset of dating site users

Emil O. W. Kirkegaard*

Julius D. Bjerrekær†



Open Differential
Psychology

- Is this ok?

4. Reproducibility

Do scientists have an ethical obligation to make sure their research is reproducible?

nature methods

Access provided by Massachusetts Institute of Technology

Altmetric: 5 Citations: 5 [More detail >>](#)

Commentary

Ethical reproducibility: towards transparent reporting in biomedical research

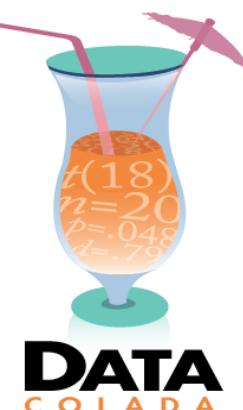
Reproducibility

Do scientists have an obligation to share data/code?

- What if it could hurt your career?
 - Others could prove you wrong, make new findings on your own data, etc.

What should you do if you find one of your papers is wrong?

- You need to retract the paper!



[NEWSLETTERS](#) [SIGN IN](#) [NPR SHOP](#)

[NEWS](#) [CULTURE](#) [MUSIC](#) [PODCASTS & SHOWS](#) [SEARCH](#)

EDUCATION

Harvard professor who studies dishonesty is accused of falsifying data

JUNE 26, 2023 · 1:15 PM ET

 Juliana Kim



Francesca Gino has been teaching at Harvard Business School for 13 years.
Maddie Meyer/Getty Images

5. Citations

If you got an idea from someone else you should always cite their work!

- What is the term for failing to do this?

You should also cite other background work that is relevant

- What if they didn't cite you?

What about citing someone because they will be a reviewer of your paper?

- How do you deal with someone else's questionable behavior?

6. Disclosure of conflicts of interest

If you have a conflict of interest you should always disclose it

- Even if you think it doesn't affect your judgement it might



7. Ethics in Statistics

P-hacking (data dredging):

Keep trying different hypothesis tests on a data set until you reach ‘statistical significance’ ($p < 0.05$)

File drawer effect:

- Try a million studies until one is significant

8. Ethics of creating powerful tools

Some prominent people are concerned about job loss due to machine learning, or even computers posing an existential threat to humans

- Is this something we should be concerned with as Data Scientists?

Ethics in machine learning

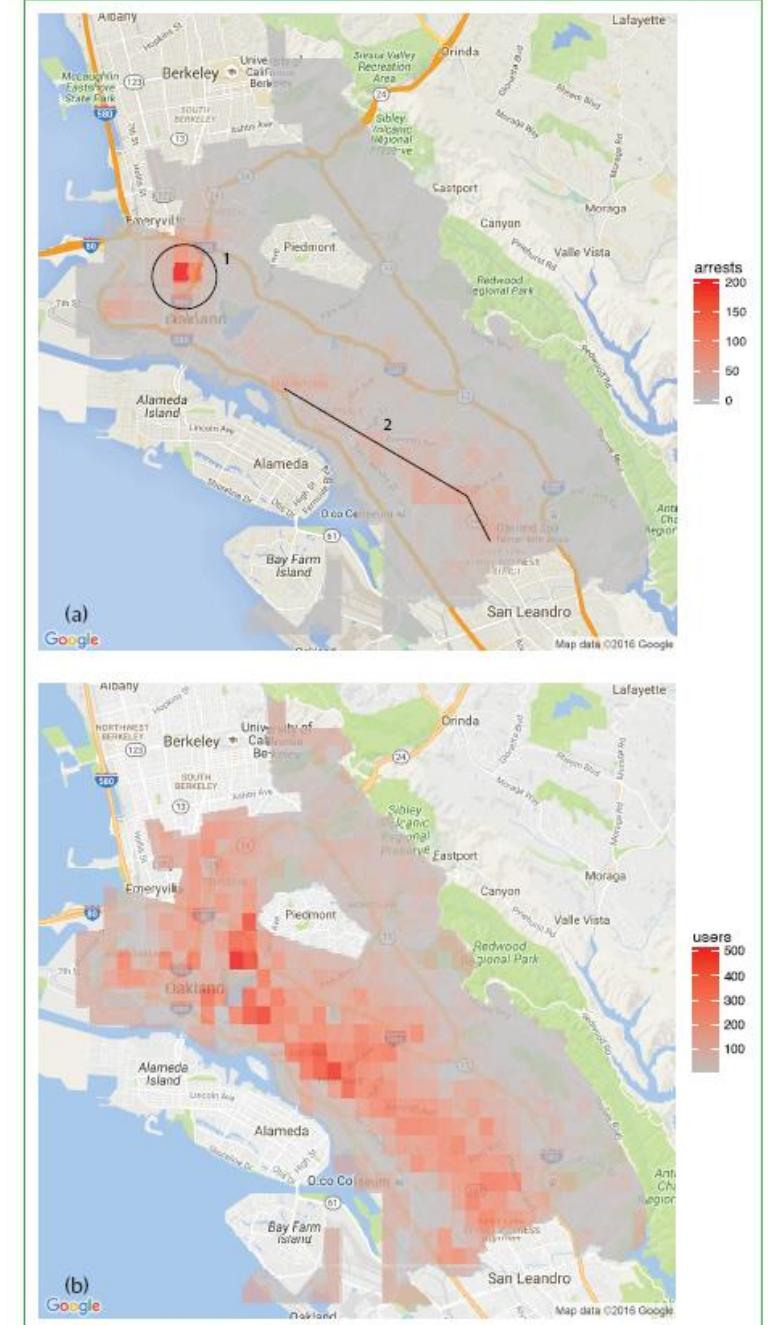
Idea: use ML to police areas with most crimes

- E.g. more police where most drug arrests were made

Possible results

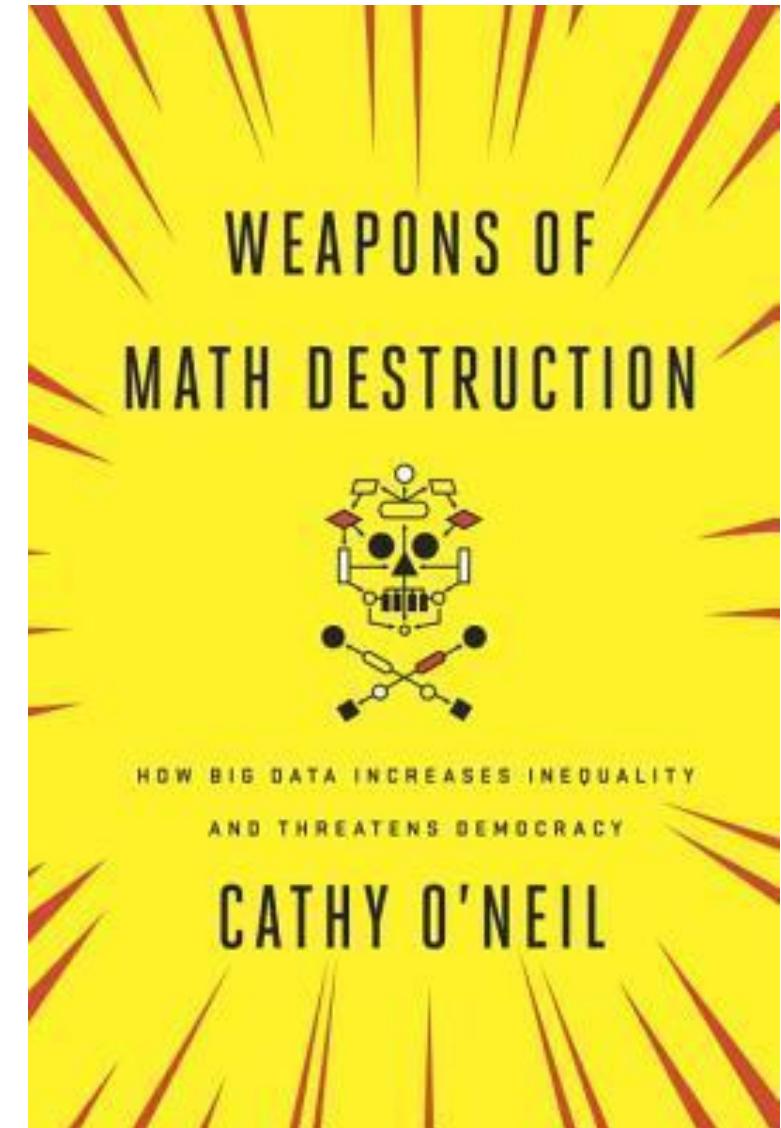
- Higher arrest rates for drugs found in these areas seemingly showing the ML algorithm is working

Any potential problems with this?



Additional reading

https://www.ted.com/talks/cathy_o_neil_the_era_of_blind_faith_in_big_data_must_end



Wrap up and conclusions

Topics covered

What is Data Science?

Python basics

Descriptive statistics

Array computations

Manipulating data tables

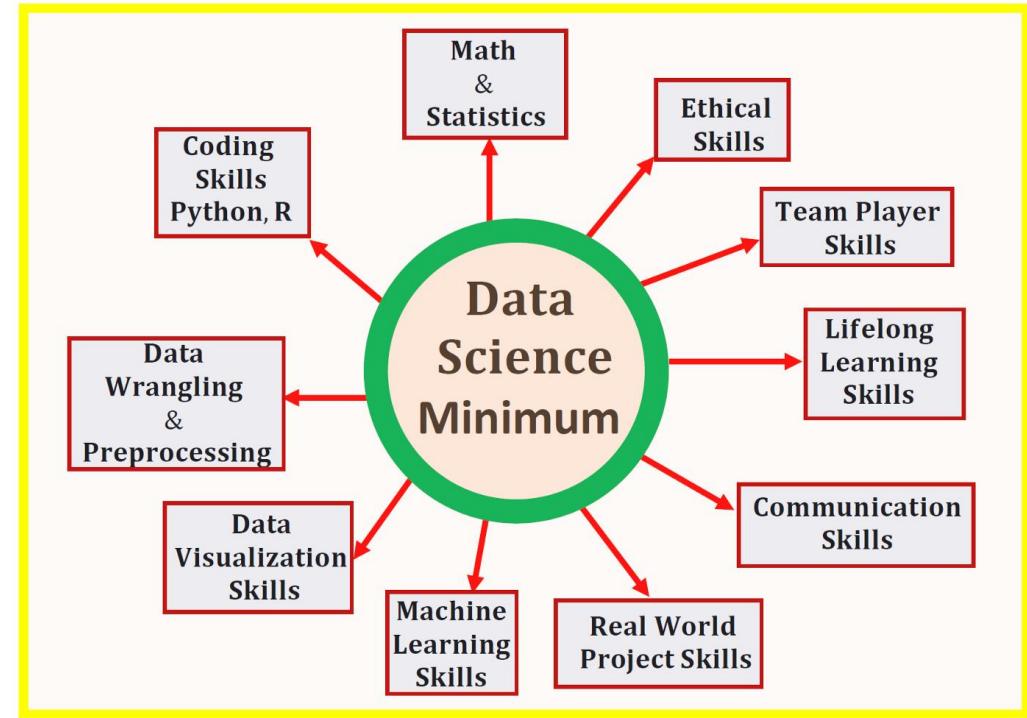
Data visualization

Mapping

Text manipulation and data cleaning

Statistical perspective: hypothesis tests and confidence intervals

Machine learning perspective: supervised and unsupervised learning



Learning goals

1. Understand concepts in data science

- Learn basic computational skills for analyzing data
- Understand concepts in statistics and machine learning

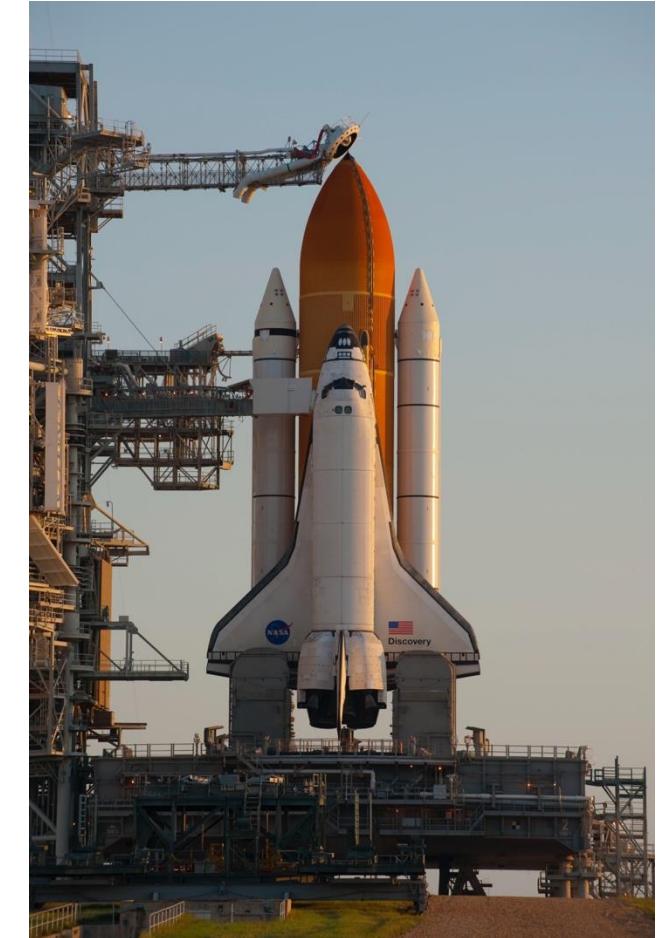
2. Gain practical data science skills applicable to any domain

3. See how data science analyses can be applied to real-world data from a variety of domains

- There will be ~weekly readings on data science related topics

There are no prerequisites for this class

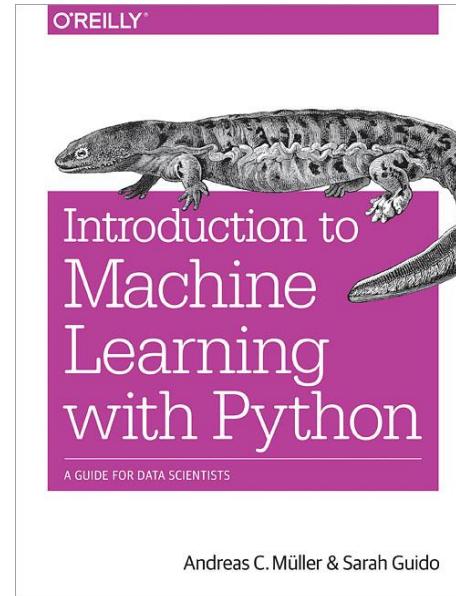
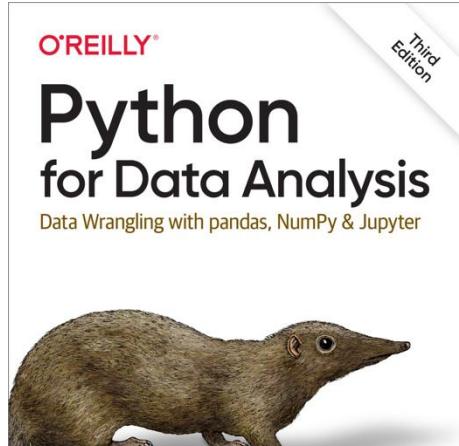
- E.g., no prior knowledge of statistics or programming is required



Next steps

1. Take more advanced Statistics and Data Science classes offered at Yale!
 - S&DS 100, S&DS 240, YData connector classes, ...
2. There are many good books and online resources to learn more Python

3. Profit!



THE WALL STREET JOURNAL.
[Home](#) [World](#) [U.S.](#) [Politics](#) [Economy](#) [Business](#) [Tech](#) [Markets](#) [Opinion](#) [Books & Arts](#) [Real Estate](#)

LIFE & WORK | JOURNAL REPORTS: COLLEGE RANKINGS

Top Colleges for High-Paying Jobs in Data Science

Median salary over first 10 years is \$100,323

Or make scientific breakthrough, change policy/the world, etc.!

Good luck with the end of the semester!

Good luck finishing your final projects!

Review session: Monday December 9th from 3-4pm in this room

