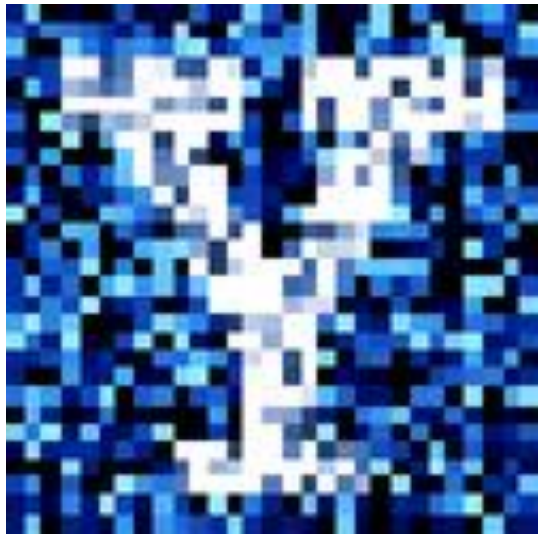
YData: Introduction to Data Science



Class 15: Mapping

Overview

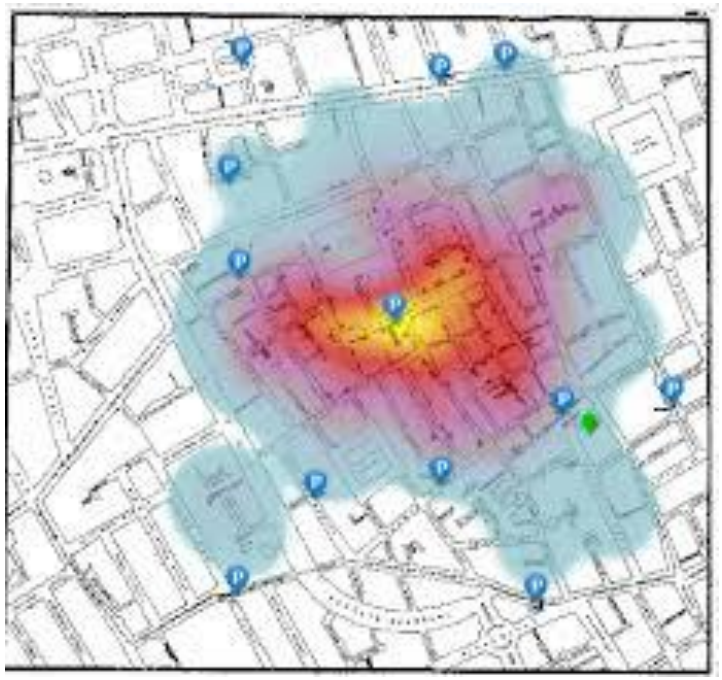
Very quick review of interactive graphics with plotly

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.

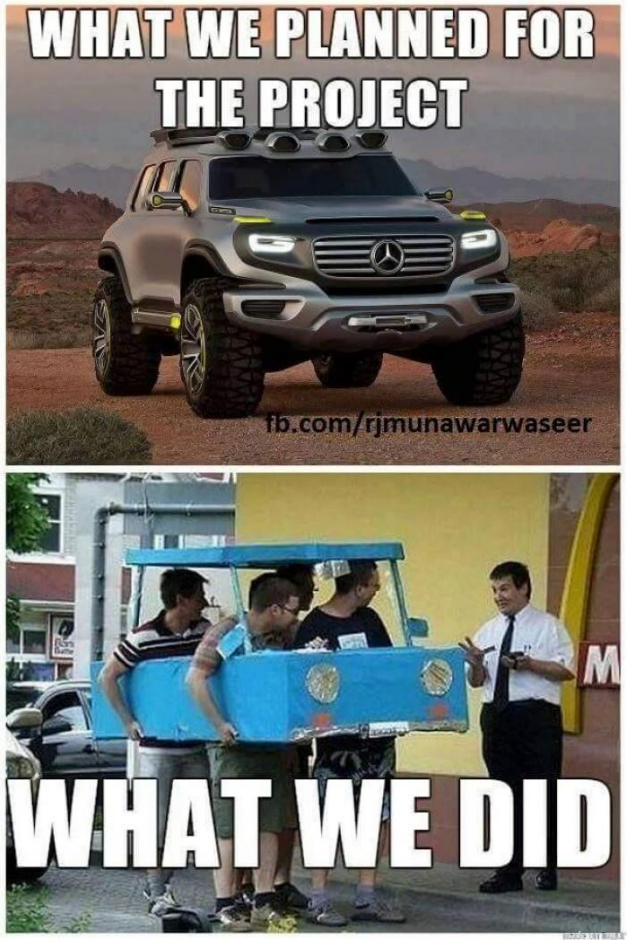
Think about what questions you want to examine, find data, and load it into Python

- 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

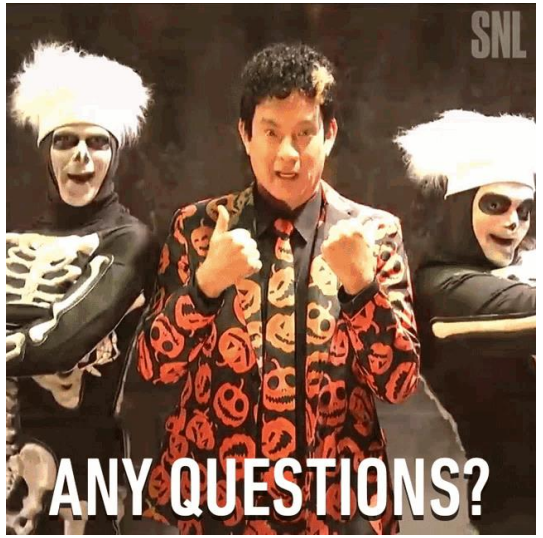
POSITION	C	PF	PG	SF	SG
TEAM					
Atlanta Hawks	18.206896	23.500000	37.096500	9.835881	17.071120
Boston Celtics	26.500000	4.306281	22.600000	30.351780	17.142857
Brooklyn Nets	9.391069	44.119845	38.917057	20.100000	19.500000
Charlotte Bobcats	3.722040	1.563518	8.623920	30.075000	21.486316
Chicago Bulls	3.200000	7.775400	9.030000	27.300000	37.096500

Pivot tables:

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

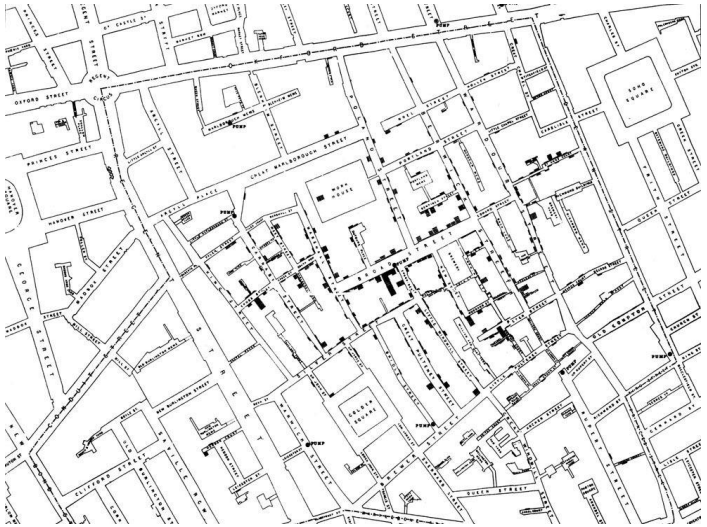
Once we have a 2D table, we can visualize it using:

- px.imshow(df2) # create a heatmap using plotly
- sns.heatmap(df2) # create a heatmap using seaborn

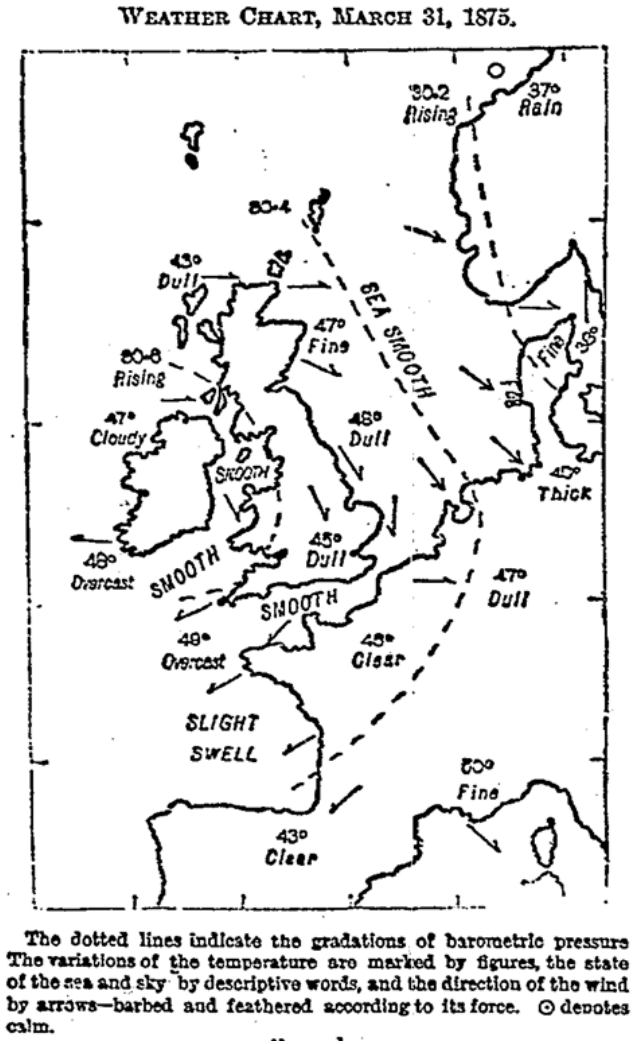


Maps

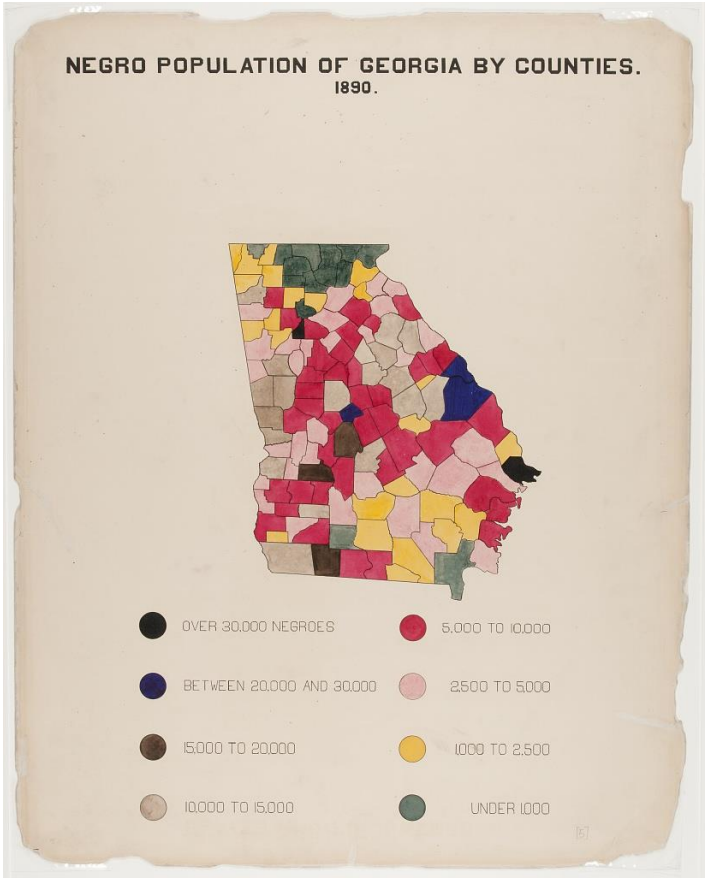
Review Maps



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 contrains “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

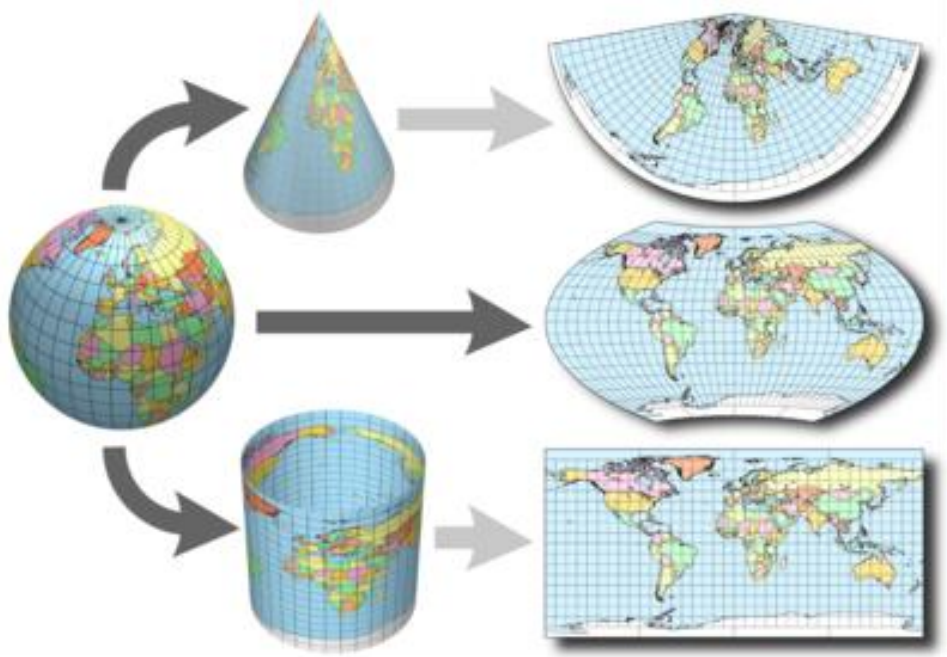
A coordinate reference system (CRS) is a framework used to precisely measure locations on the surface of the Earth as coordinates

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.

DYMAXION

YOU LIKE ISAAC ASIMOV, XML, AND SHOES WITH TOES. YOU THINK THE SEGWAY GOT A BAD RAP. YOU OWN 3D GOGGLES, WHICH YOU USE TO VIEW ROTATING MODELS OF BETTER 3D GOGGLES. YOU TYPE IN DVORAK.

WINKEL-TRIPEL

NATIONAL GEOGRAPHIC ADOPTED THE WINKEL-TRIPEL IN 1998, BUT YOU'VE BEEN A WT FAN SINCE *LONG* BEFORE 'NATGEO' SHOWED UP. YOU'RE WORRIED IT'S GETTING PLAYED OUT, AND ARE THINKING OF SWITCHING TO THE KAVRAYSKIY. YOU ONCE LEFT A PARTY IN DISGUST WHEN A GUEST SHOWED UP WEARING SHOES WITH TOES. YOUR FAVORITE MUSICAL GENRE IS "POST-".

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'D JUST ELECT *NORMAL* 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 CAR'S 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 CONFLICT-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

(EQUIRECTANGULAR)

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 CAHILL MAP IT'S BASED—... YOU HAVE A FRAMED REPRODUCTION AT HOME?! WHOA. ...LISTEN, FORGET THESE QUESTIONS. ARE YOU DOING ANYTHING TONIGHT?

GALL-PETERS

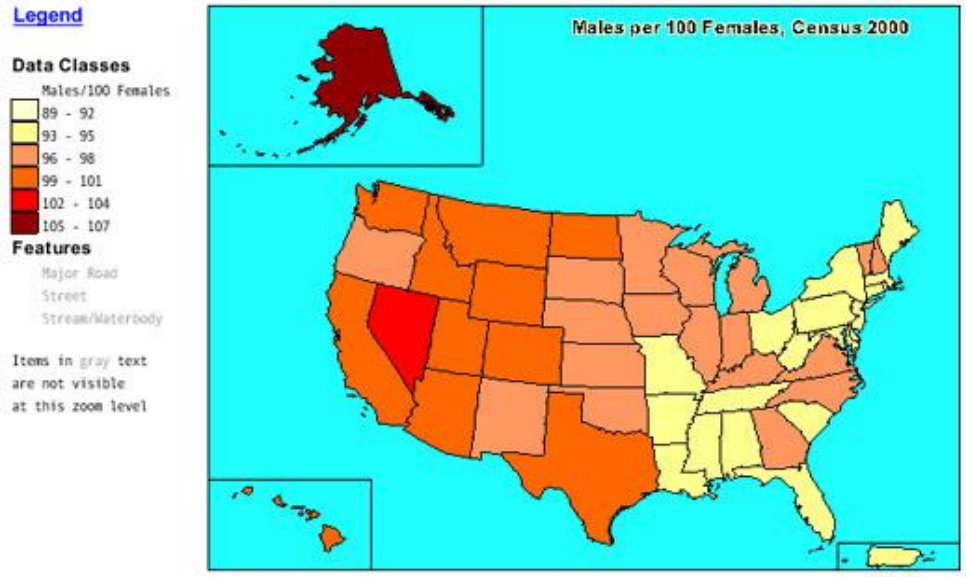
I *HATE* YOU.

Maps

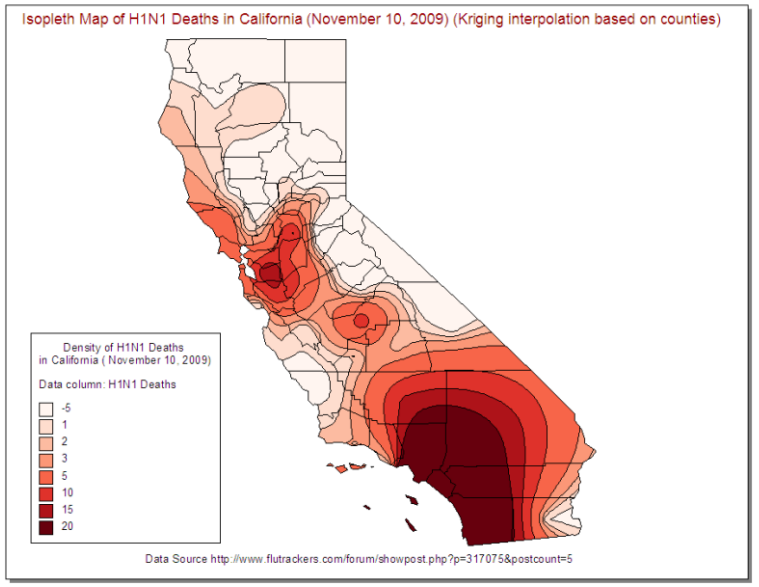
Choropleth maps: shades/colors in predefined areas based on properties of a variable

Isopleth maps: creates regions based on constant values

Choropleth map



Isopleth map



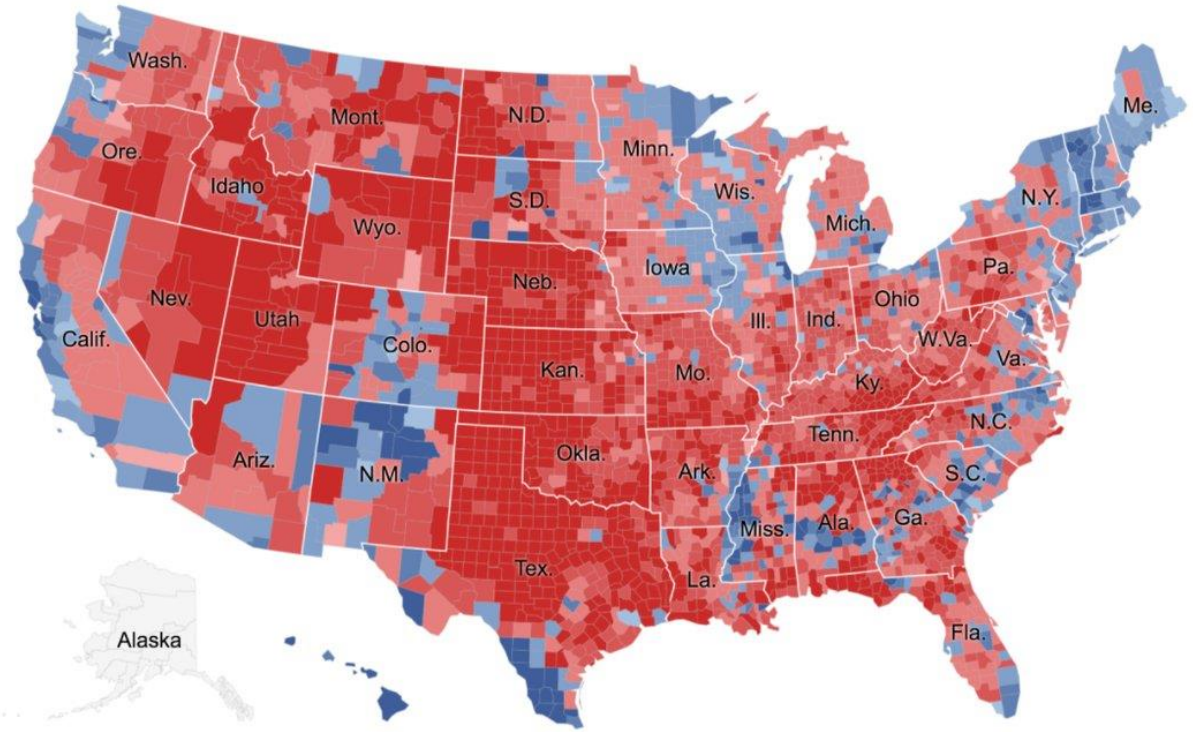
Choropleth maps

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)

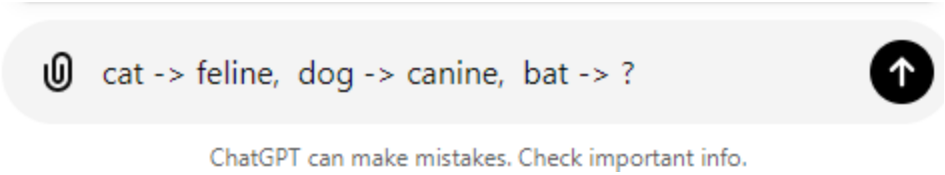
The range always includes start but excludes end

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)
```



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!