17. Data visualization & LLMs
LING 471

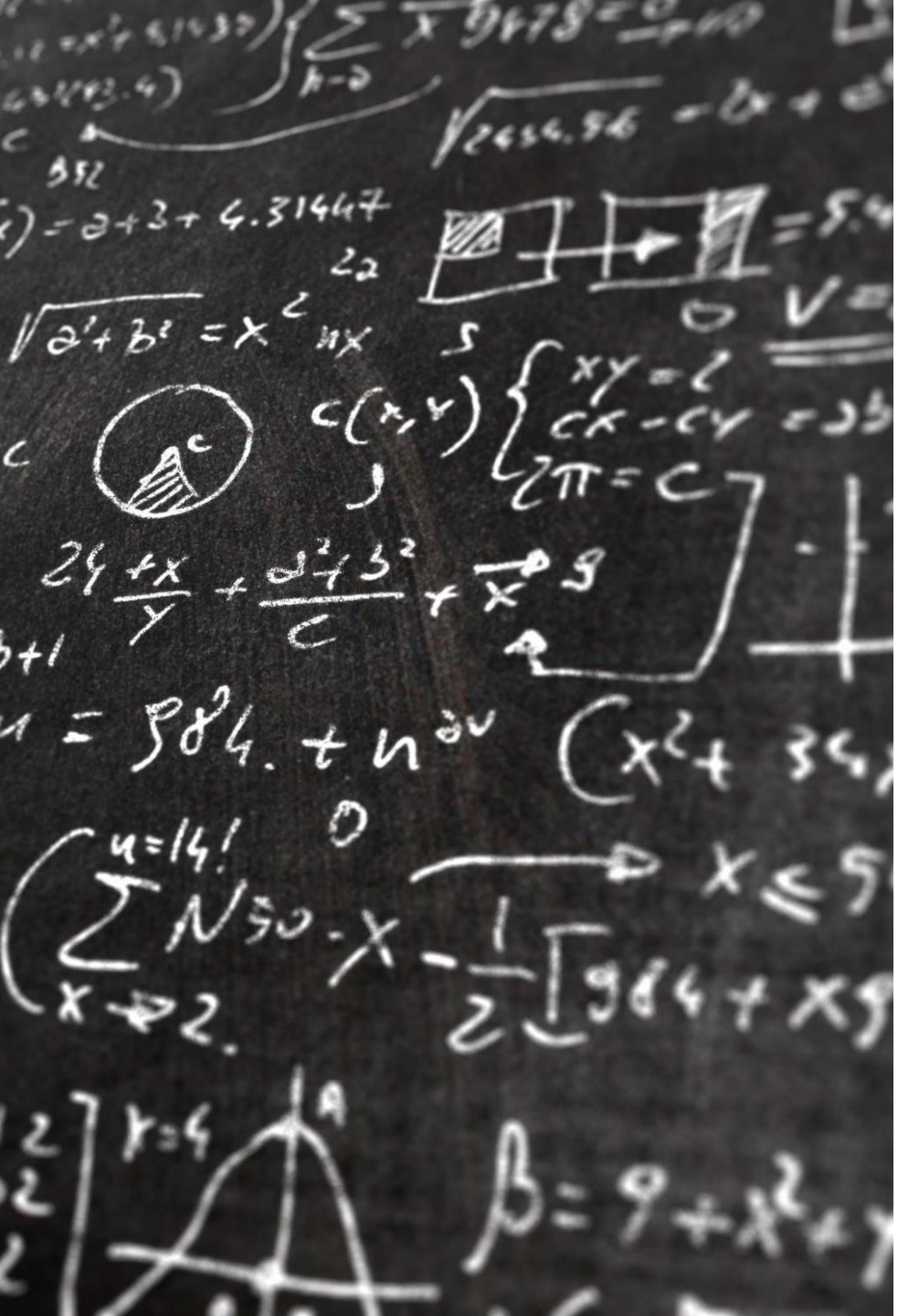
Logistics

- Assignment 4 due tonight
- Sign up for your presentation time slots in the Google Sheet

Learning outcomes

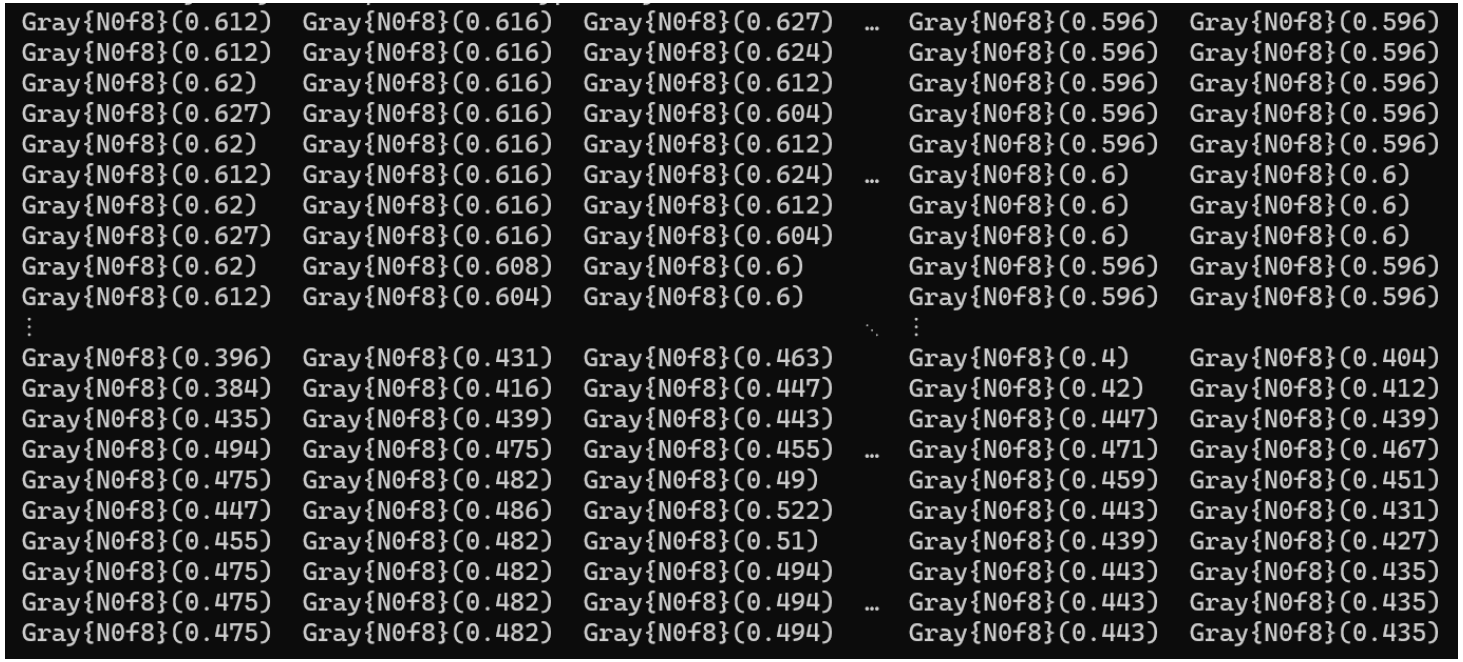
- Describe why **visualizations** are important and give some examples of visualizations
- Describe **line plots**, **scatter plots**, **bar plots**, and **heatmaps**, and create them in Python
- Give a high-level description of **large language models**
- Discuss the issue of "**understanding**" as it relates to large language models
- Use LLMs with API

Data visualization



Why visualize data?

- It tends to be hard to understand trends and patterns in data just by looking at raw numbers
  - Even with equations
- Instead, we turn to visualizations that summarize data or show the relevant concepts
- Raw data, otherwise, is usually too low level



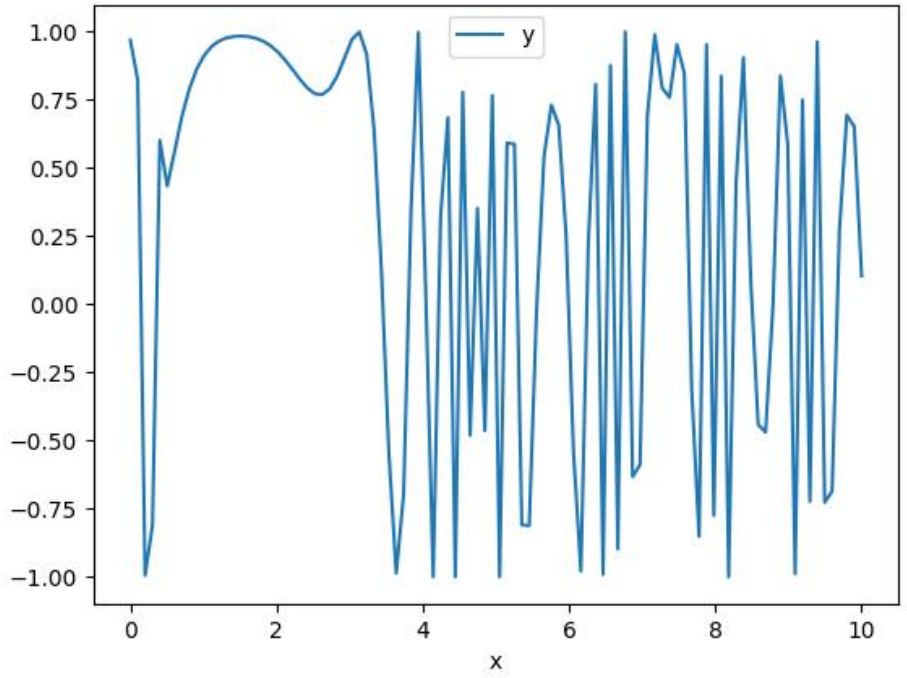
Gray{N0f8}(0.612)	Gray{N0f8}(0.616)	Gray{N0f8}(0.627)	…	Gray{N0f8}(0.596)	Gray{N0f8}(0.596)
Gray{N0f8}(0.612)	Gray{N0f8}(0.616)	Gray{N0f8}(0.624)		Gray{N0f8}(0.596)	Gray{N0f8}(0.596)
Gray{N0f8}(0.62)	Gray{N0f8}(0.616)	Gray{N0f8}(0.612)		Gray{N0f8}(0.596)	Gray{N0f8}(0.596)
Gray{N0f8}(0.627)	Gray{N0f8}(0.616)	Gray{N0f8}(0.604)		Gray{N0f8}(0.596)	Gray{N0f8}(0.596)
Gray{N0f8}(0.62)	Gray{N0f8}(0.616)	Gray{N0f8}(0.612)		Gray{N0f8}(0.596)	Gray{N0f8}(0.596)
Gray{N0f8}(0.612)	Gray{N0f8}(0.616)	Gray{N0f8}(0.624)	…	Gray{N0f8}(0.6)	Gray{N0f8}(0.6)
Gray{N0f8}(0.62)	Gray{N0f8}(0.616)	Gray{N0f8}(0.612)		Gray{N0f8}(0.6)	Gray{N0f8}(0.6)
Gray{N0f8}(0.627)	Gray{N0f8}(0.616)	Gray{N0f8}(0.604)		Gray{N0f8}(0.6)	Gray{N0f8}(0.6)
Gray{N0f8}(0.62)	Gray{N0f8}(0.608)	Gray{N0f8}(0.6)		Gray{N0f8}(0.596)	Gray{N0f8}(0.596)
Gray{N0f8}(0.612)	Gray{N0f8}(0.604)	Gray{N0f8}(0.6)		Gray{N0f8}(0.596)	Gray{N0f8}(0.596)
⋮			⋱	⋮	
Gray{N0f8}(0.396)	Gray{N0f8}(0.431)	Gray{N0f8}(0.463)		Gray{N0f8}(0.4)	Gray{N0f8}(0.404)
Gray{N0f8}(0.384)	Gray{N0f8}(0.416)	Gray{N0f8}(0.447)		Gray{N0f8}(0.42)	Gray{N0f8}(0.412)
Gray{N0f8}(0.435)	Gray{N0f8}(0.439)	Gray{N0f8}(0.443)		Gray{N0f8}(0.447)	Gray{N0f8}(0.439)
Gray{N0f8}(0.494)	Gray{N0f8}(0.475)	Gray{N0f8}(0.455)	…	Gray{N0f8}(0.471)	Gray{N0f8}(0.467)
Gray{N0f8}(0.475)	Gray{N0f8}(0.482)	Gray{N0f8}(0.49)		Gray{N0f8}(0.459)	Gray{N0f8}(0.451)
Gray{N0f8}(0.447)	Gray{N0f8}(0.486)	Gray{N0f8}(0.522)		Gray{N0f8}(0.443)	Gray{N0f8}(0.431)
Gray{N0f8}(0.455)	Gray{N0f8}(0.482)	Gray{N0f8}(0.51)		Gray{N0f8}(0.439)	Gray{N0f8}(0.427)
Gray{N0f8}(0.475)	Gray{N0f8}(0.482)	Gray{N0f8}(0.494)		Gray{N0f8}(0.443)	Gray{N0f8}(0.435)
Gray{N0f8}(0.475)	Gray{N0f8}(0.482)	Gray{N0f8}(0.494)	…	Gray{N0f8}(0.443)	Gray{N0f8}(0.435)
Gray{N0f8}(0.475)	Gray{N0f8}(0.482)	Gray{N0f8}(0.494)		Gray{N0f8}(0.443)	Gray{N0f8}(0.435)

Types of visualizations: Raw numbers

- One way to visualize data is to print out raw numbers
- Maybe as a string
- Maybe as a table
- Sometimes helpful if numbers are summaries (like mean, standard deviation)
- Otherwise, not very informative

Types of visualizations: Equations

- Often made up of simple mathematical functions, e.g., $f(x) = x^4$
- Sometimes require additional visualizations (e.g., what, exactly, does $f(x) = \sin(x^2 - 3x + 4 - \ln(x)^8)$ look like?)
- Often compact, but not everything is easy to convert to a function



Types of visualizations: Plots

- Some trends in data can be summarized as specific types of images called plots
- Different plots can show different things
- One example is a heatmap
  - The image on the right is a visualization of the table of raw numbers

Choosing a type of plot

- Many plots have a specific type of relationship they are trying to show
- A key to good communication is often choosing the right kind of plot to visualize your data
- Often more art than science to choose the right plot
  - And many artistic choices need to be made regardless
- Often better to think about your “story” first and then choose an appropriate plot

Plot types and creation in Python/Pandas

Creating plots in Python: matplotlib

- There are many packages that you can use in Python to create plots
- We will be using one called matplotlib
  - There is an interface to it through Pandas
  - Often serves as a backend for other packages
- Other options might include seaborn, GR, and plotnine

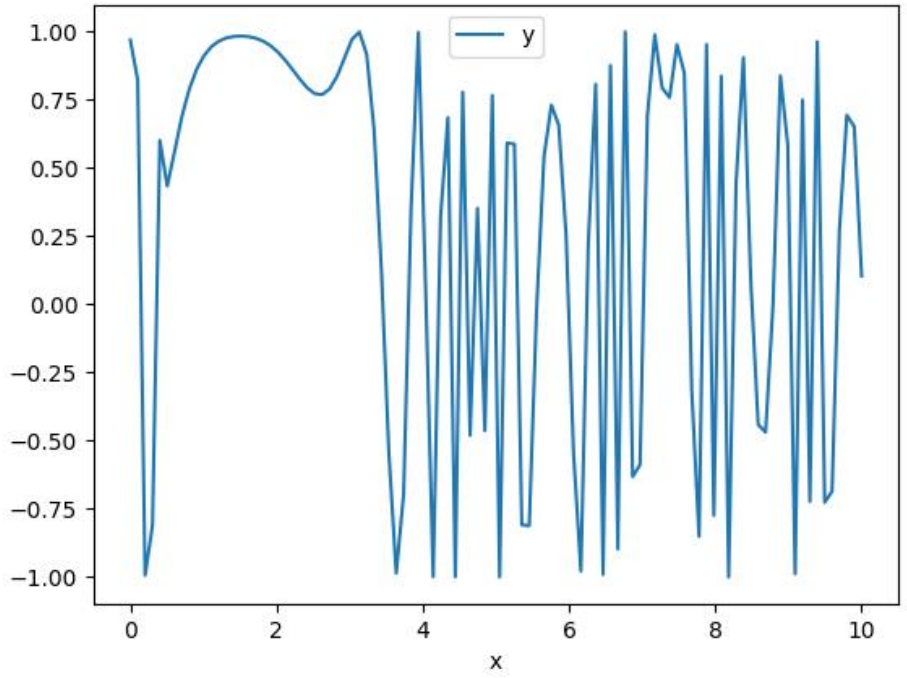
Set up code

- We need to import both matplotlib and pandas to use the pandas interface
- Can use matplotlib without pandas for more options

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

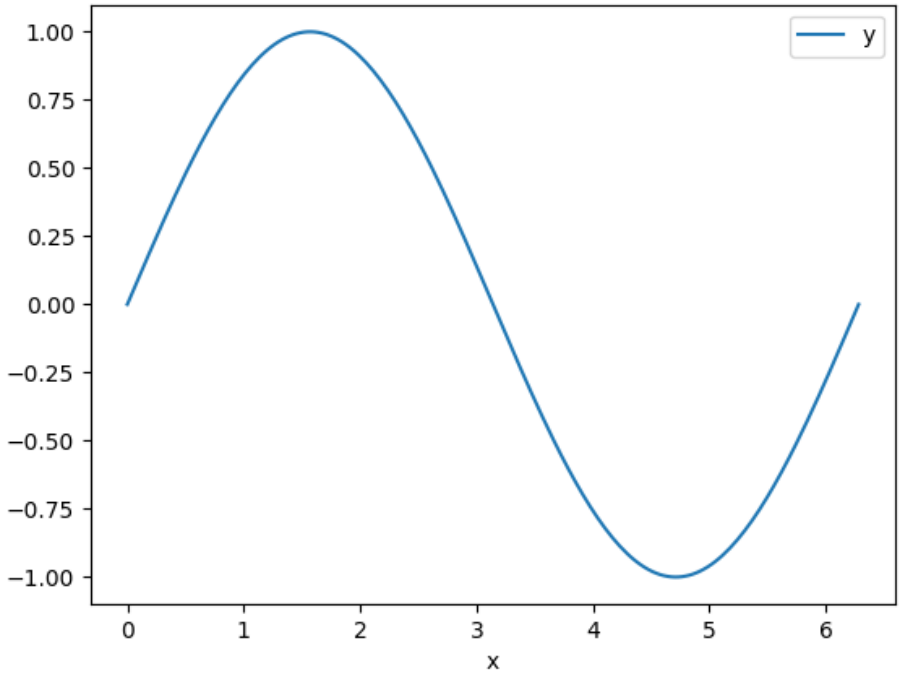
Line plots

- A line plot basically connects all data points with lines
- Order matters here
  - Wrong connecting lines might be drawn otherwise
- Useful for continuous relationships



Creating a line plot

```
x = np.linspace(0, 2*np.pi, num=200)
y = np.sin(x)
df = pd.DataFrame({'x': x, 'y': y})
df.plot('x', 'y')
plt.show()
```

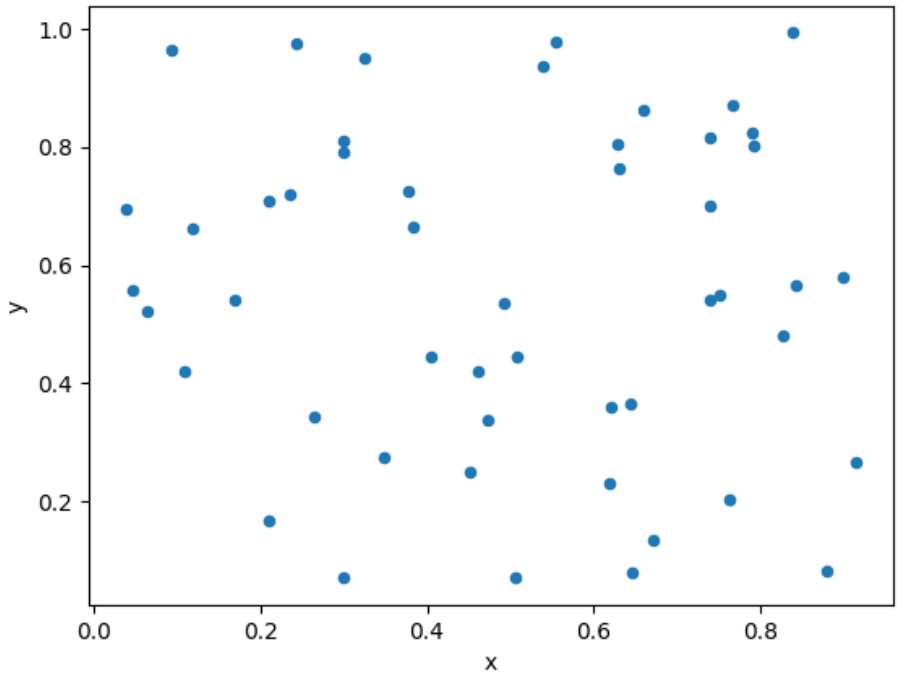


Scatterplot

- A scatter plot shows each data point as a small object in the plotting space
  - Often a dot or square
- Order matters a lot less here
- Useful for data that may not show a continuous relationship

Making a scatterplot

```
x = np.random.rand(50)
y = np.random.rand(50)
df = pd.DataFrame({'x': x, 'y': y})
df.plot.scatter('x', 'y')
plt.show()
```



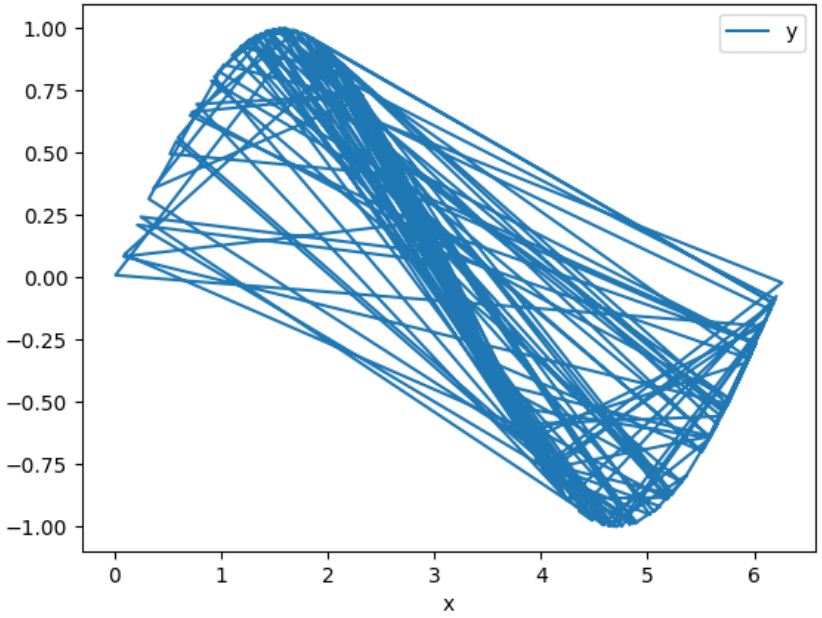
Ordering for scatter vs. line

```
x = np.random.rand(200)*2*np.pi
y = np.sin(x)
df = pd.DataFrame({'x': x, 'y': y})
df.plot('x', 'y')
plt.show()
```

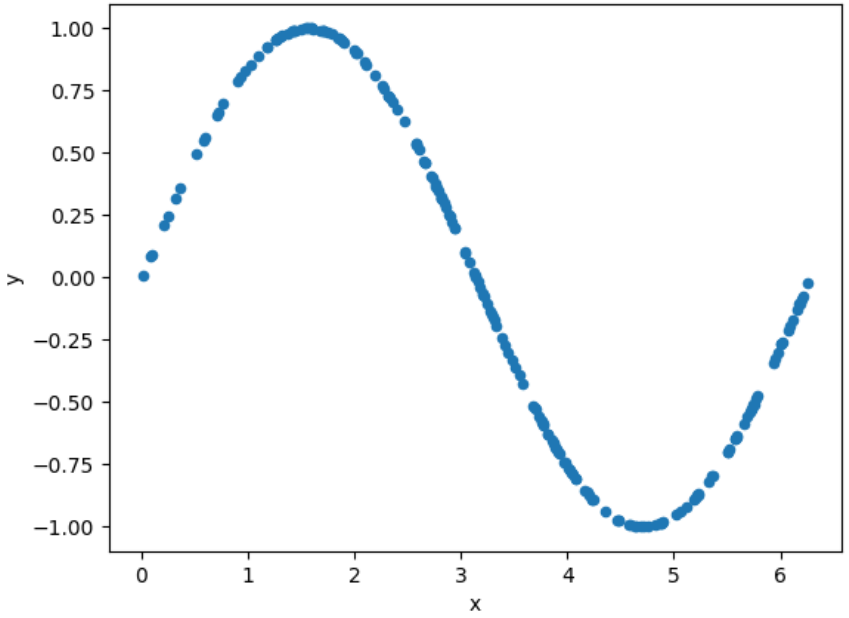
```
x = np.random.rand(200)*2*np.pi
y = np.sin(x)
df = pd.DataFrame({'x': x, 'y': y})
df.plot.scatter('x', 'y')
plt.show()
```

Ordering for scatter vs. line

LINE PLOT

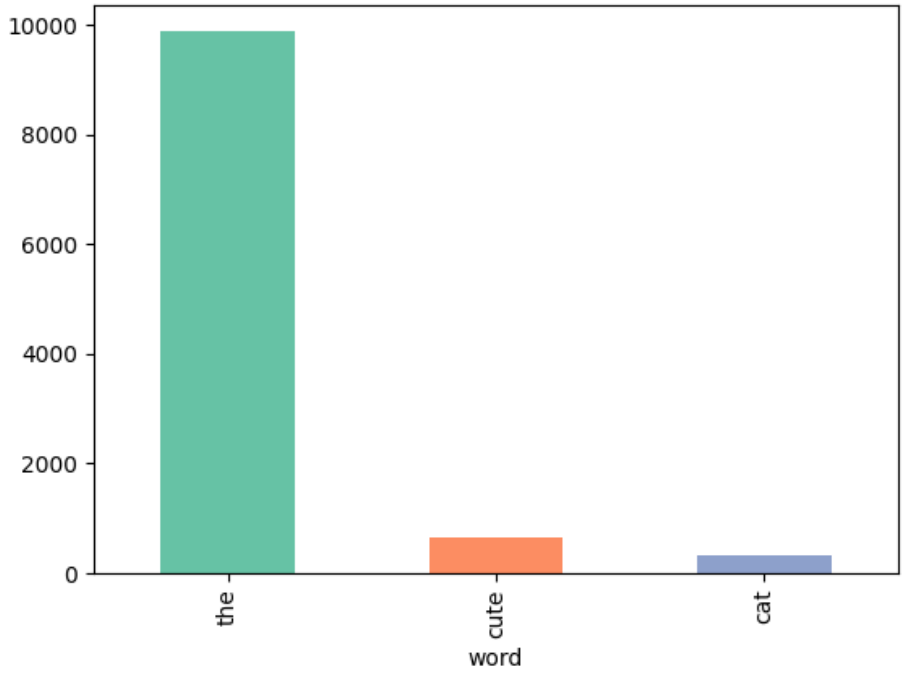


SCATTER PLOT



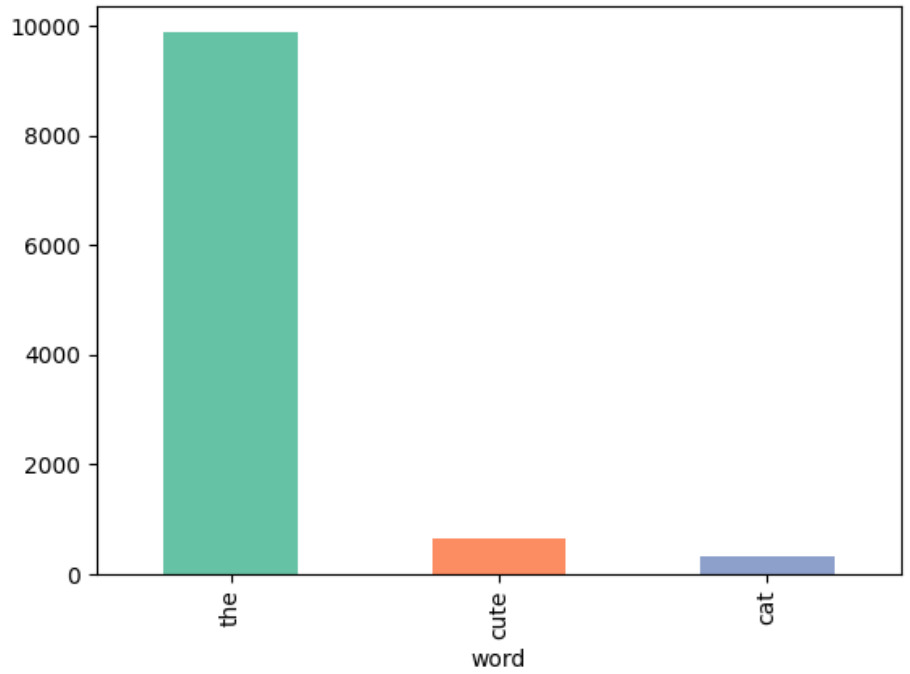
Barplot

- Usually used to represent categorical data
- That is, data where you can't imagine there being intermediate values
  - E.g., counts of words, income in each month, number of each color marble in a bag



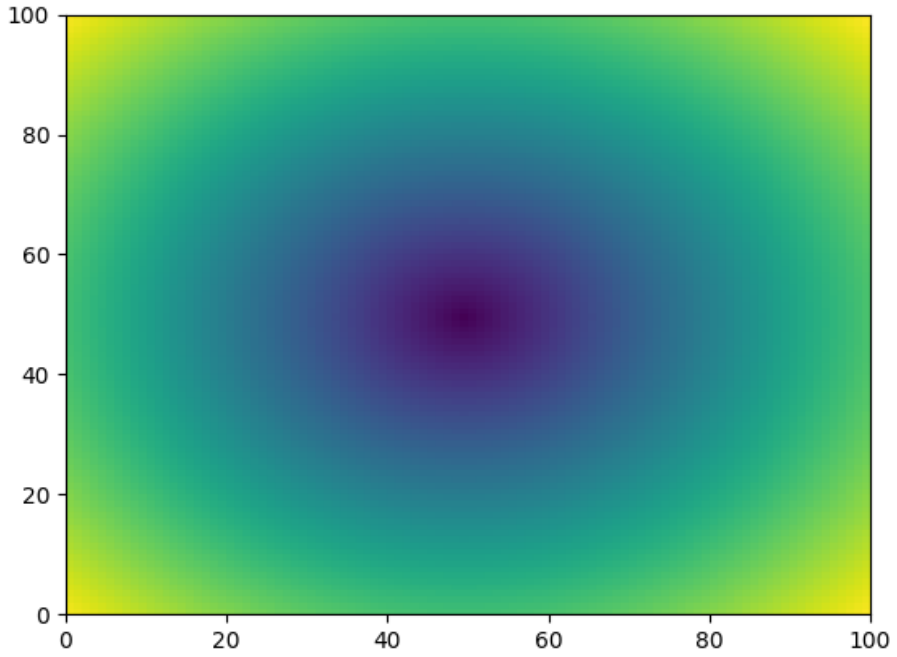
Making a barplot

```
df = pd.DataFrame({'word': ['the', 'cute',
'cat'], 'count': [9876, 654, 320]})

n_col = len(set(df.word))

colors = [plt.cm.Set2(x) for x in
range(n_col)]

df.plot.bar('word', 'count', color=colors)

plt.legend('', frameon=False)

plt.subplots_adjust(bottom=0.15)

plt.show()
```



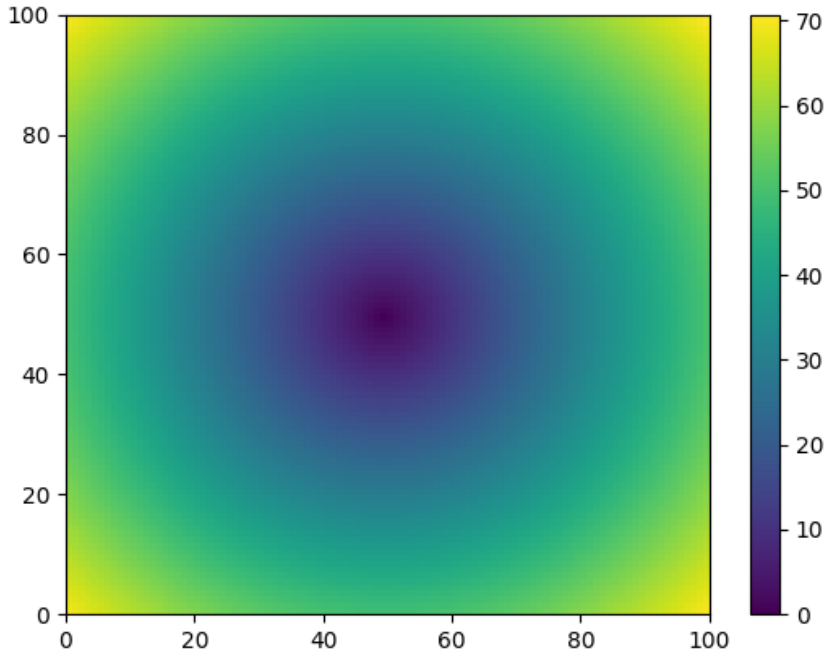
Heatmap

- Usually used to show how two variables affect a third variable
- Use a color map to assign a color to the range of values
  - Many color maps available
  - Often not using linear spacing
- Requires a matrix, rather than two vectors
  - Two vectors can be given to express units on x- and y-axes
- Can’t make right from Pandas



Making a heatmap

```
a = np.zeros((100, 100))
for i in range(a.shape[0]):
    for j in range(a.shape[1]):
        a[i, j] = np.linalg.norm([i-49, j-49])


plt.pcolormesh(a)
plt.colorbar()
plt.show()
```



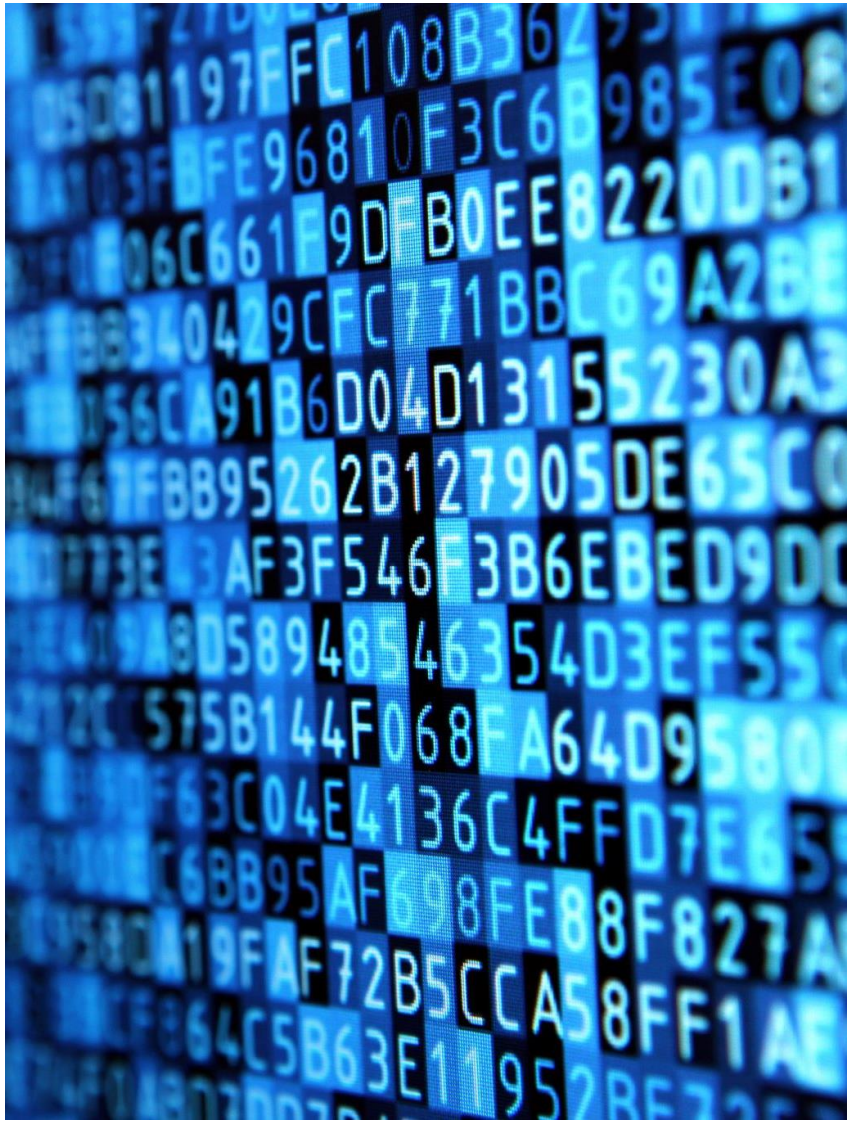
Keyword arguments and extra functions

- Many plots require some keyword arguments to make fully interpretable
  - E.g., choosing colormap in the heatmap with the cmap argument
- Some also require extra functions to modify
  - plt.xlabel (i.e., matplotlib.pyplot.xlabel) – change x-axis label
  - plt.ylabel (i.e., matplotlib.pyplot.ylabel) – change y-axis label
- Generally, need to check matplotlib docs to see what you need

Programming activity

- Access the code in the class repo
- Work through the plot examples on the previous slides
- Try modifying the data, adding axis labels, etc.

Large language models



What is a large language model (LLM)?

- Consider the skipgram we trained last week on a very small amount of data
- Imagine we made the network ***MUCH, MUCH*** larger
  - So large it makes you worry about global warming
- Also, imagine we trained it on ***WAY, WAY*** more data
  - Let's say… The entire internet
- This is, roughly, a large language model like GPT

What are LLMs trained to do?

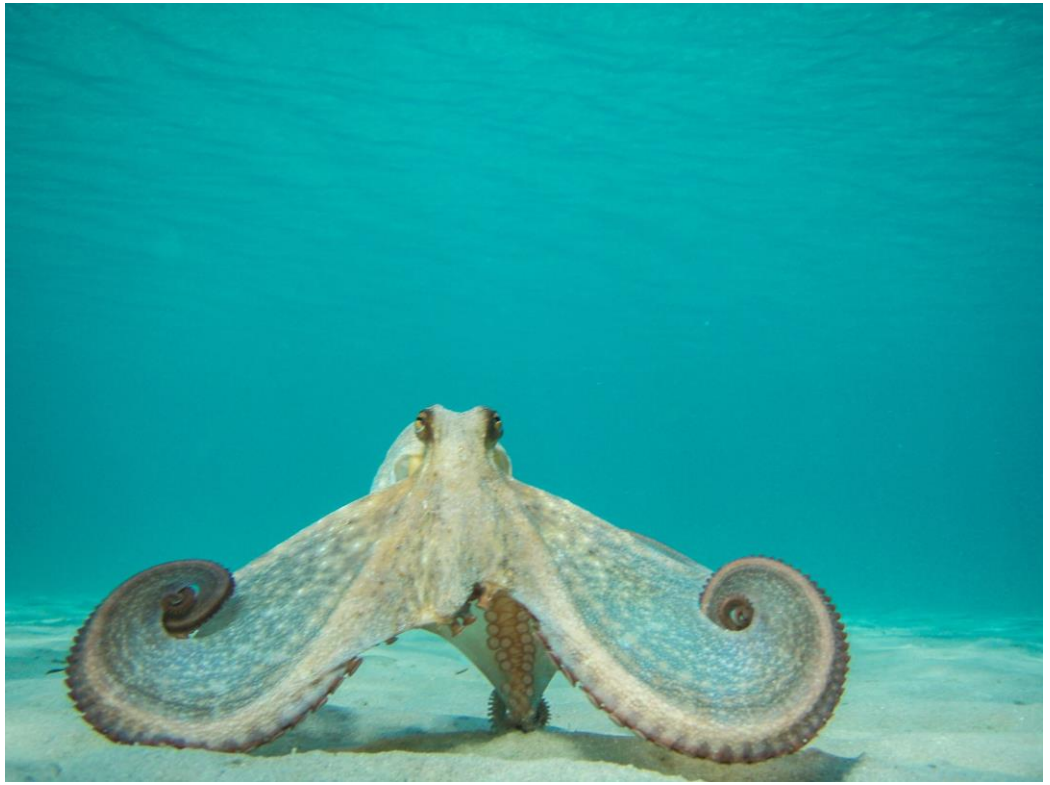
- Similar to the skipgram model, they are trained to predict subsequent words given a set of previous words
  - The architecture and task are different, but conceptually similar to the skipgram
- They are usually later fine-tuned for specific tasks
  - Sometimes, just having humans say “yes, I like this output” or not

The octopus scenario: Can LLMs understand language?

- Summarizing from “Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data” (and our reading for today)
- Consider two people stranded on separate islands: Alex and Blake
- There is a telephone line connecting Alex and Blake, and they talk to each other a lot
- At some point, an ultra-smart octopus intercepts the phone conversations and starts learning how Blake responds to Alex
- Eventually, the octopus reconfigures the telephone system to respond to Alex (and stranding Blake)
- One day, Alex frantically calls Blake and says, “Help! There is a bear on a rampage here, but I have some sticks! What do I do? How do I defend myself?”
- How does the octopus respond?

The octopus

- The octopus is representing an LLM in this scenario
- It can learn many different linguistic patterns and perhaps fool Alex, but it has embodied experience with very little of it
- Is the octopus's "knowledge" of language actually an understanding?



Slippery concepts: “Understanding,” “Meaning,” “Semantics”

- The octopus example is, in part, supposed to highlight the imprecision of common usages of terms like “understanding”
  - There are some times where it might make sense to say a computer device “understands” you
  - E.g., “I asked Siri to set an alarm, and she understood”
- But, other scenarios are hard to say
  - Does the octopus “understand” language?
  - How about GPT?

Conflicting definitions of semantics and understanding

EMBODIED

- Understanding a word requires being able to relate it back to experiences you have had before
  - A computer cannot do this

DISTRIBUTIONAL

- Understanding a word requires being able to relate it to other words
  - A computer can do this

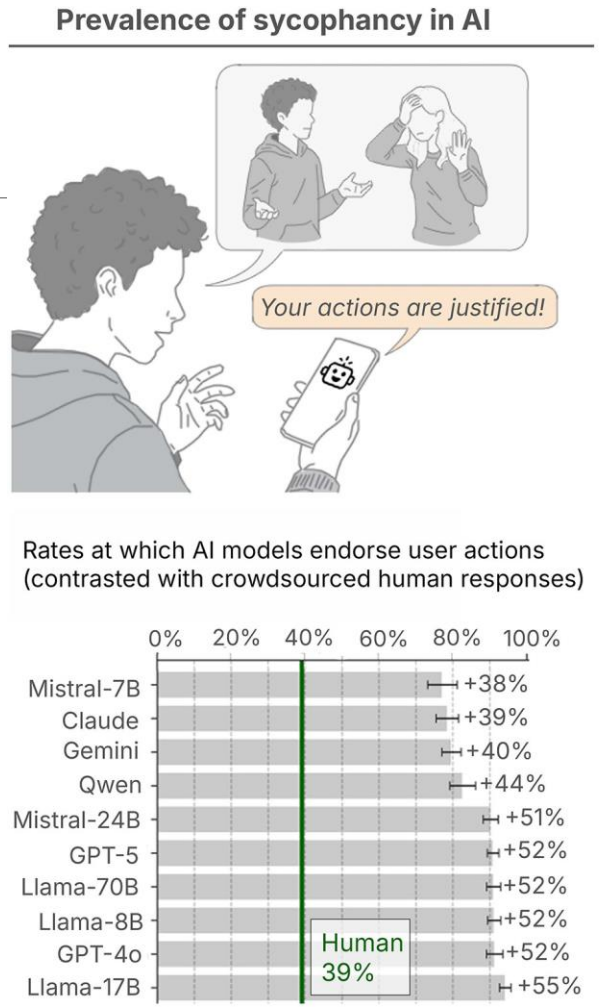
Which definition is correct? More satisfying? More natural?

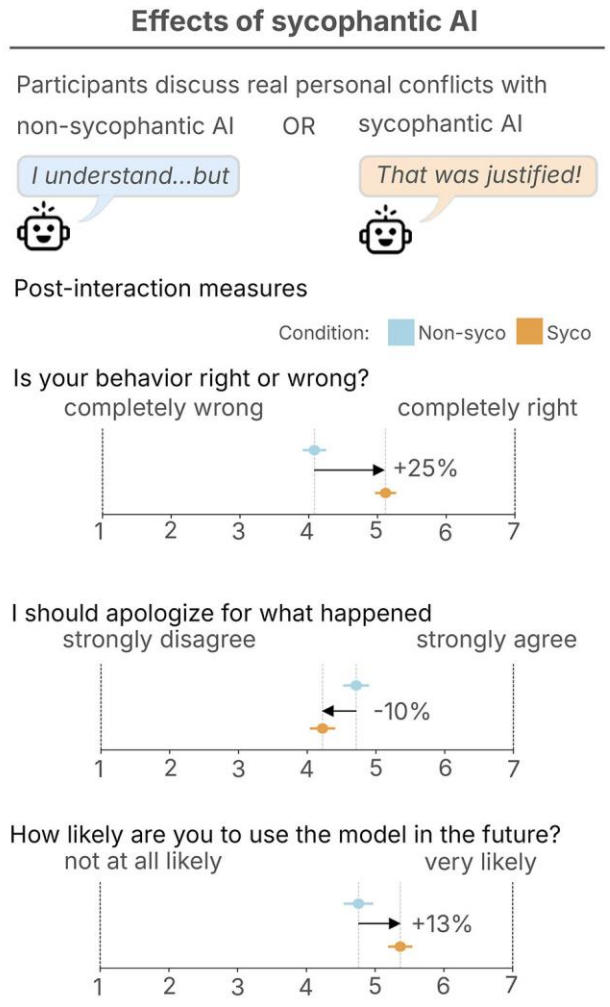
LLMs and harm

- Large language models pose substantial risk (and potential opportunity), and can commit harm
  - Risk: giving dangerous information, sounding convincing about things it's incorrect about
  - Opportunity: speed up some drudgery work (like templates)
  - Harm: reiterating societal biases, reinforcing linguistic standards and linguistic marginalization, etc.

LLMs and harm

- Sycophancy in LLM (Cheng et al. 2026)
  - AI affirmed users' actions more often than humans, even when queries involved deception, illegality, or other harms.
  - Interaction with sycophantic AI reduced participants' willingness to take responsibility and repair interpersonal conflicts, while increasing their conviction that they were right.
  - Sycophantic models were trusted and preferred despite distorting judgment.





LLM exploration activity

- Access the demo code from the class repo
- Get a free API from Google AI Studio
- Try different prompts in the Python code