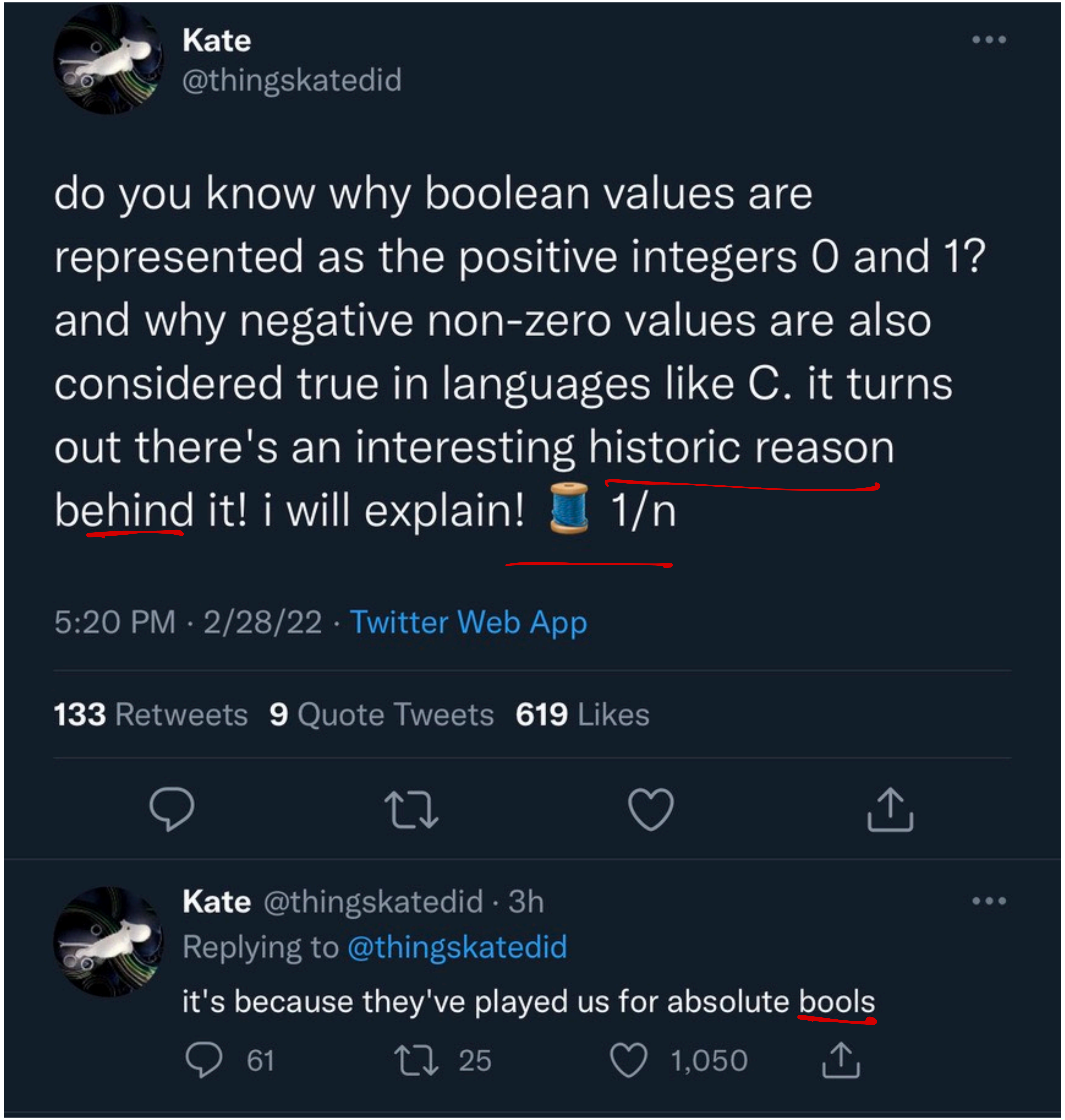
Lecture 9

Data Science, Conclusion

History of Data Science, Spring 2022 @ UC San Diego

Suraj Rampure



Kate
@thingskatedid

do you know why boolean values are represented as the positive integers 0 and 1? and why negative non-zero values are also considered true in languages like C. it turns out there's an interesting historic reason behind it! i will explain! 🧵 1/n

5:20 PM · 2/28/22 · Twitter Web App

133 Retweets 9 Quote Tweets 619 Likes

Kate @thingskatedid · 3h
Replying to @thingskatedid

it's because they've played us for absolute bools

61 25 1,050

Announcements

end of weekend

- Homework 9 is released, and is due **Monday, May 30th at 11:59PM**.
    - Recall, you only need to (satisfactorily) complete 8 homeworks for full credit.
    - Everyone **should** complete Homework 9 (because it's short and fun!) but you only **need** to complete it if:
        - You didn't receive full credit on at least one homework assignment (I messaged you on Slack if this was the case).
        - You missed 2 or more lectures (including today, Lecture 9).
- CAPEs will be released sometime later this week – **please fill them out!**
    - In addition, I will release a separate End-of-Quarter survey to solicit your feedback.

Agenda

- PageRank – one of the first algorithms behind Google's success.
- Python and Jupyter Notebooks.
- Data Science as a field.

PageRank

Page, Brin, and PageRank

googol $= 10^{100}$

- Larry Page (1973-present) and Sergey Brin (1973-present) developed the **Google** search engine (originally known as BackRub) while Ph.D. students at Stanford University.
    - The two dropped out of their Ph.D.s to start Google in 1998, in the garage of Susan Wojcicki (current CEO of YouTube).
- Together, Page and Brin developed the **PageRank** algorithm, which serves as the backbone of the search engine.
    - PageRank assigns each page on the internet a "score" based on its relative importance.
    - This was a novel idea at the time.

The importance of links

- **Key Idea:** The more **incoming links** a page has, the more **important** it is.
- How do we formulate this mathematically?

Idea #1

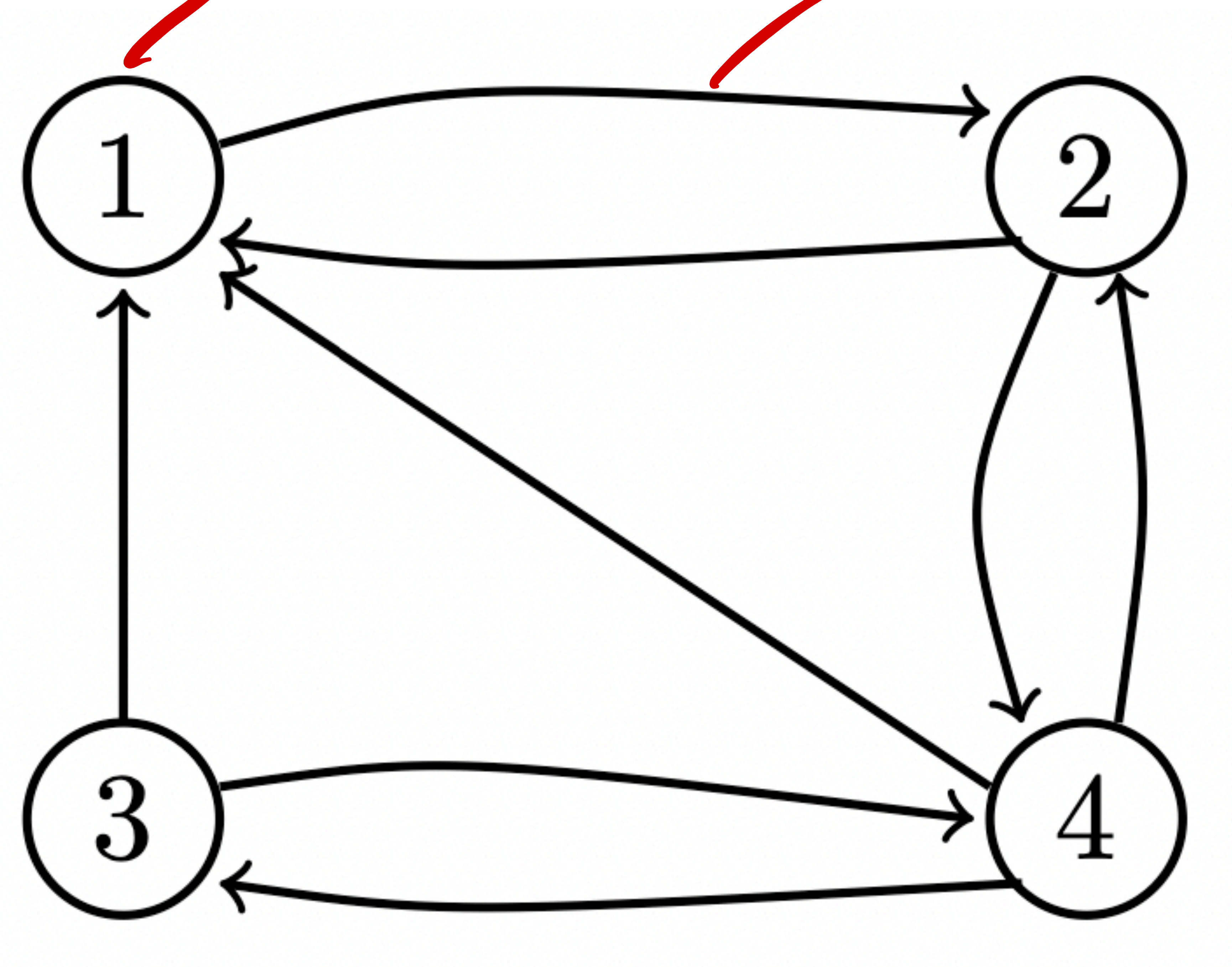
x_i = # incoming links to site i

$x_1 = 3$, $x_2 = 2$, $x_3 = 1$

$x_4 = 2$

webpages

links



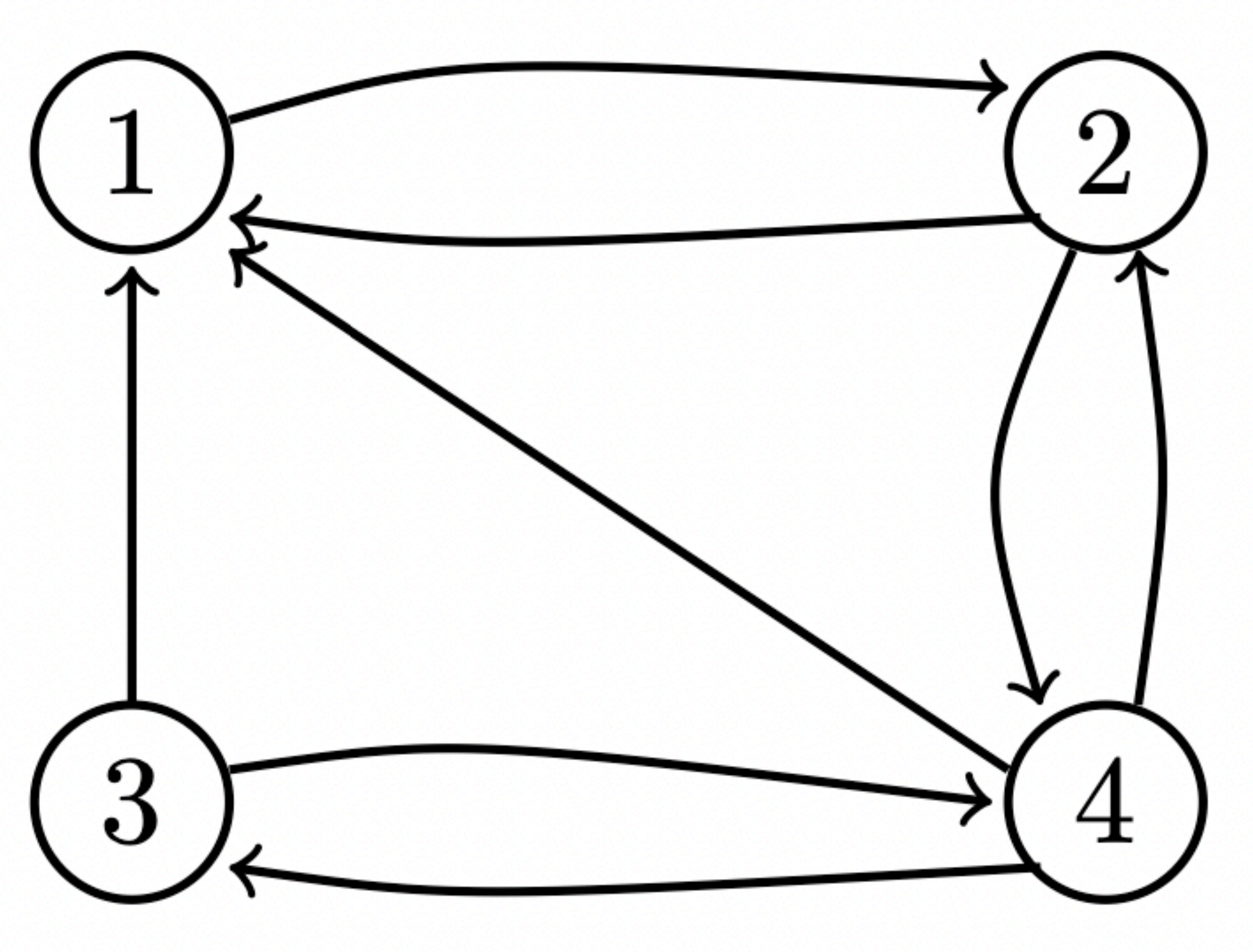
(source)

Idea # 2

$$x_i = \sum_{j \to i} x_j$$

i.e. score of a site is equal to the sum of the scores of the incoming sites

$$\begin{aligned} x_1 &= x_2 + x_3 + x_4 \\ x_2 &= x_1 + x_4 \\ x_3 &= x_4 \\ x_4 &= x_2 + x_3 \end{aligned}$$



only solution is $x_1 = x_2 = x_3 = x_4 = 0$

Ideas

- Idea 1: The score of a page is equal to the **number of pages that link to it**.
    - Issue: this doesn’t account for the **importance** of the incoming links.
        - If my personal website links to the UCSD homepage, that doesn’t mean much.
        - If the UCSD homepage links to my personal website, that means a lot.
- Idea 2: The score of a page is equal to the **sum of the scores of the pages that link to it**.
    - Issue: no non-zero solution!

PageRank

$$x_i = \sum_{j \to i} \frac{x_j}{n_j}$$

x_i = score of website i

n_i = # of websites that website i links to

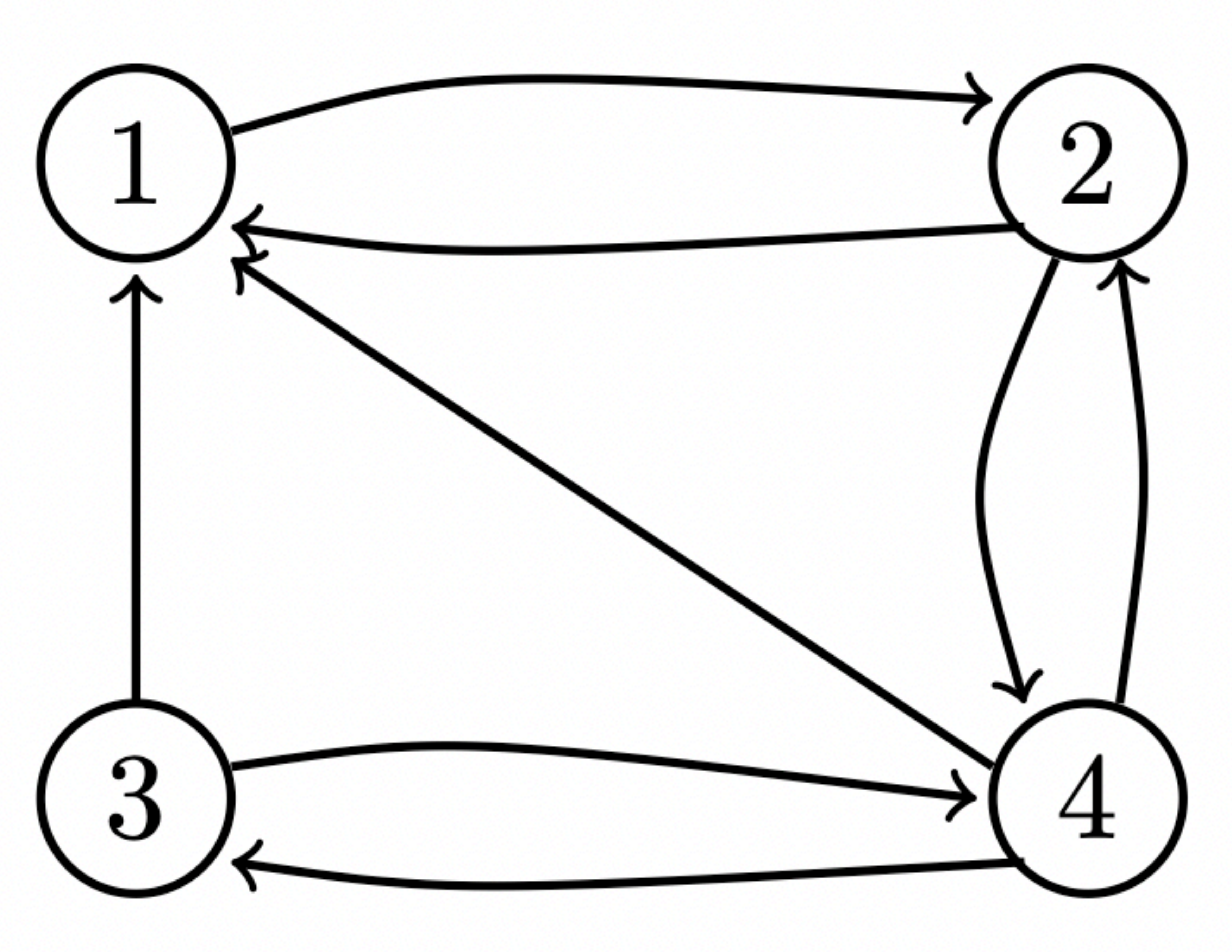
PageRank: The score of a page is equal to a **weighted sum of the scores of the pages that link to it, where each page's score is weighted by the number of outgoing links that page has.**

$$x_1 = \frac{1}{2}x_2 + \frac{1}{2}x_3 + \frac{1}{3}x_4$$

$$x_2 = 1x_1 + \frac{1}{3}x_4$$

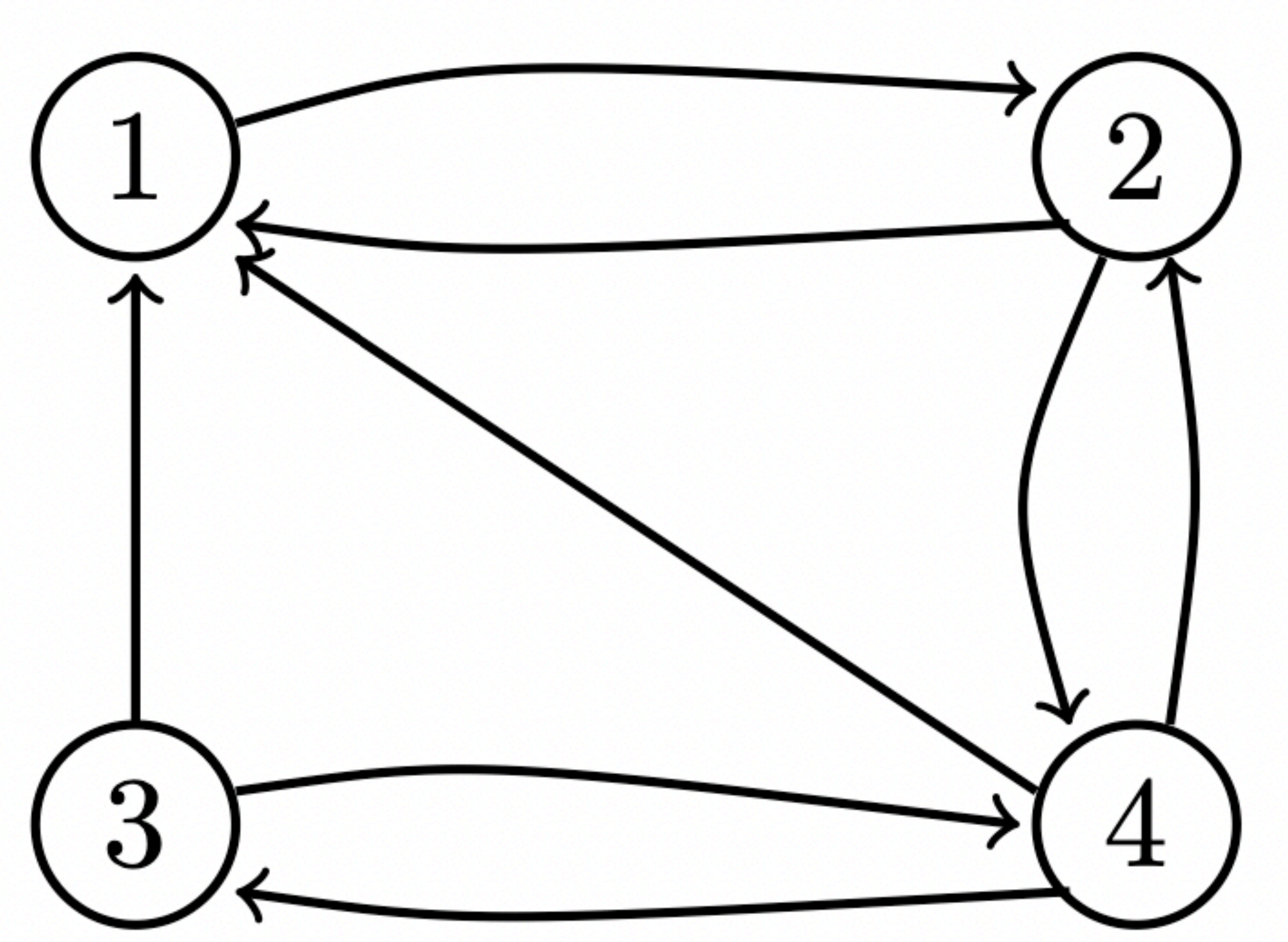
$$x_3 = \frac{1}{3}x_4$$

$$x_4 = \frac{1}{2}x_2 + \frac{1}{2}x_3$$



$$\begin{aligned}
x_1 &= \tfrac{1}{2}x_2 + \tfrac{1}{2}x_3 + \tfrac{1}{3}x_4 \\
x_2 &= 1x_1 + \tfrac{1}{3}x_4 \\
x_3 &= \tfrac{1}{3}x_4 \\
x_4 &= \tfrac{1}{2}x_2 + \tfrac{1}{2}x_3
\end{aligned}$$

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad A = \begin{bmatrix} 0 & 1/2 & 1/2 & 1/3 \\ 1 & 0 & 0 & 1/3 \\ 0 & 0 & 0 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \end{bmatrix}$$



$$\boxed{\vec{x} = A\vec{x}}$$

infinite # of solutions

⇒ add constraint that $\boxed{\sum x_i = 1}$

thus, scores are probabilities

Solving the PageRank problem

columns sum to 1!

- The set of equations devised on the previous slide can be written as a matrix-vector equation of the form $\boxed{\vec{x} = A\vec{x},}$ where

looking for eigenvector w/ eigenval of 1

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, \text{TO}\; A = \begin{array}{c} \text{FROM} \\ \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{3} \\ 1 & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \\ \begin{array}{cccc} 1\to & 2\to & 3\to & 4\to \end{array} \end{array}$$

stochastic matrix

- All we now need to do is solve for $\vec{x}$.
- $\vec{x}$ is known as an **eigenvector**, corresponding to the **eigenvalue** of 1.

Aside

A square matrix

λ constant

$\vec{v}$ vector

$$A\vec{v} = \lambda\vec{v}$$

eigenvector

eigenvalue

An iterative system

- Let's place the restriction on $\overrightarrow{x}$ that all of its elements must sum to 1 – in this way, we can interpret $\overrightarrow{x}$ as a probability distribution over all webpages.

- At time $t = 0$, let's assume users are equally likely to be on any of the 4 webpages, so $\overrightarrow{x_0} = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}^T$.

- At time $t = 1$, users move between websites according to the matrix A – so $\overrightarrow{x_1} = A\overrightarrow{x_0}$.

- Then, $\overrightarrow{x_2} = A\overrightarrow{x_1} = A^2\overrightarrow{x_0}$.

- More generally, $\overrightarrow{x_n} = A^n\overrightarrow{x_0}$.

- **Key Idea:** $\overrightarrow{x_n}$ will **converge** to a steady-state matrix, which is equal to the solution $\overrightarrow{x}$ on the previous slide.

$\vec{X}_t$ = distribution of websites at time t

$$A = \begin{bmatrix} 0 & 1/2 & 1/2 & 1/3 \\ 1 & 0 & 0 & 1/3 \\ 0 & 0 & 0 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \end{bmatrix}$$

$$\vec{X}_0 = \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}$$

$$\vec{x}_1 = A\,\vec{x}_0 = \begin{bmatrix} \frac{1}{2}\cdot\frac{1}{4} + \frac{1}{2}\cdot\frac{1}{4} + \frac{1}{3}\cdot\frac{1}{4} \\ 1\cdot\frac{1}{4} + 0 + \frac{1}{3}\cdot\frac{1}{4} \\ \vdots \\ \cdots \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ \vdots \\ \\ \\ \end{bmatrix}$$

Generally:

$$\boxed{\vec{x}_n = A^n \vec{x}_0}$$

$$\vec{x}_2 = A\,\vec{x}_1 = A(A\,\vec{x}_0)$$

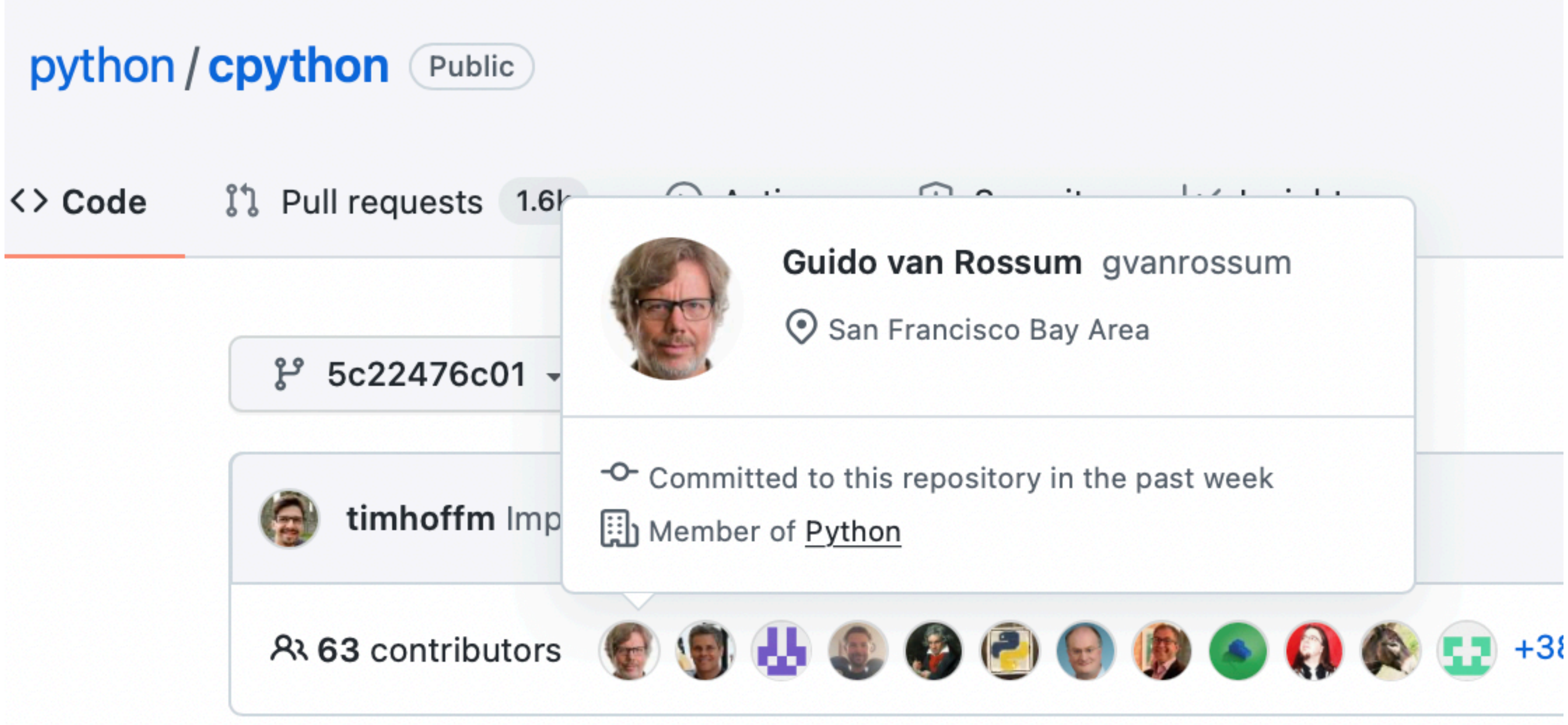
$$\vec{x}_5 = A\,\vec{x}_4 = A^2\vec{x}_3 = A^3\vec{x}_2 = A^4\vec{x}_1 = A^5\vec{x}_0$$

Python and Jupyter Notebooks

van Rossum and Python

- Guido van Rossum (1956-present) is a Dutch-American software engineer and current Microsoft employee. He is the inventor of the **Python** programming language.
- He first released the language in 1991, while working at CWI (a research institute in the Netherlands).
    - It was designed to be **easier to read** than C, but just as capable.
    - Python is named after the British comedy group **Monty Python**.
    - Key to Python is the fact that it is **open source**, meaning that anyone can contribute to its development. (GitHub)
- Van Rossum held the title of "benevolent dictator for life" until 2018, when he stepped down.

Per his personal website, Van Rossum owns a "PYTHON" license plate.



Guido van Rossum gvanrossum

San Francisco Bay Area

Committed to this repository in the past week

Member of Python

timhoffm Imp

63 contributors

Guido van Rossum still actively contributes to Python. Screenshot taken from the Python GitHub repo.

“Computer Programming for All”

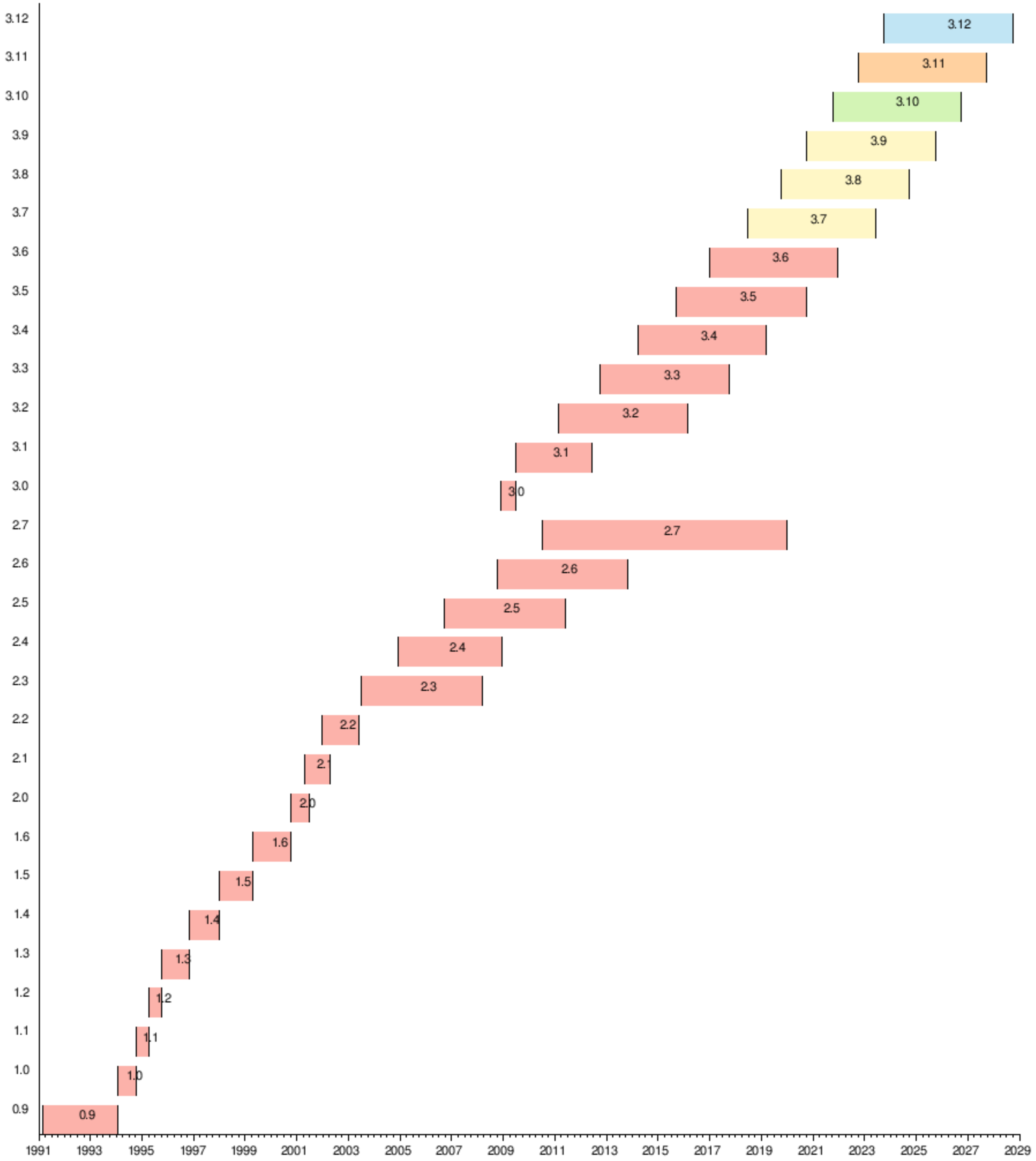
- In 1999, while at Van Rossum submitted a proposal to DARPA (Defense Advanced Research Projects Agency) named “Computer Programming for All.”
- In it, he declared that computer programming should be taught to everyone in elementary school.
    - Reason: Programming develops logical reasoning skills and helps develop familiarity with how a computer works.
    - He cited Python as being a good language to teach as it is beginner-friendly, widely used in industry, and easy to customize by writing modules.
        - Initial plan: teach using a **subset** of Python (similar to the relationship between **babypandas** and **pandas**).
        - Java, C, and C++ are too complicated for beginners.

The evolution of Python

(source)

- Python 0.9: Released in **1991**.
- Python 1: First version released in **1994**. Not in use today at all.
  - Added lambdas.
- Python 2: First version released in **2000**, only officially deprecated on January 1st, 2020 (last version 2.7.18).
  - Added list comprehension.
- Python 3: First released in **2006**, latest version 3.10.2 (January 14th, 2022).
  - Python 3 is not **backwards compatible** with Python 2, meaning that code written for Python 3 won't necessarily run with Python 2 (and vice versa).
  - Backwards compatibility was broken in order to improve various features of the language. The lack of it is why many developers continued to use Python 2 more than a decade after Python 3 was released.

Timeline of Python versions



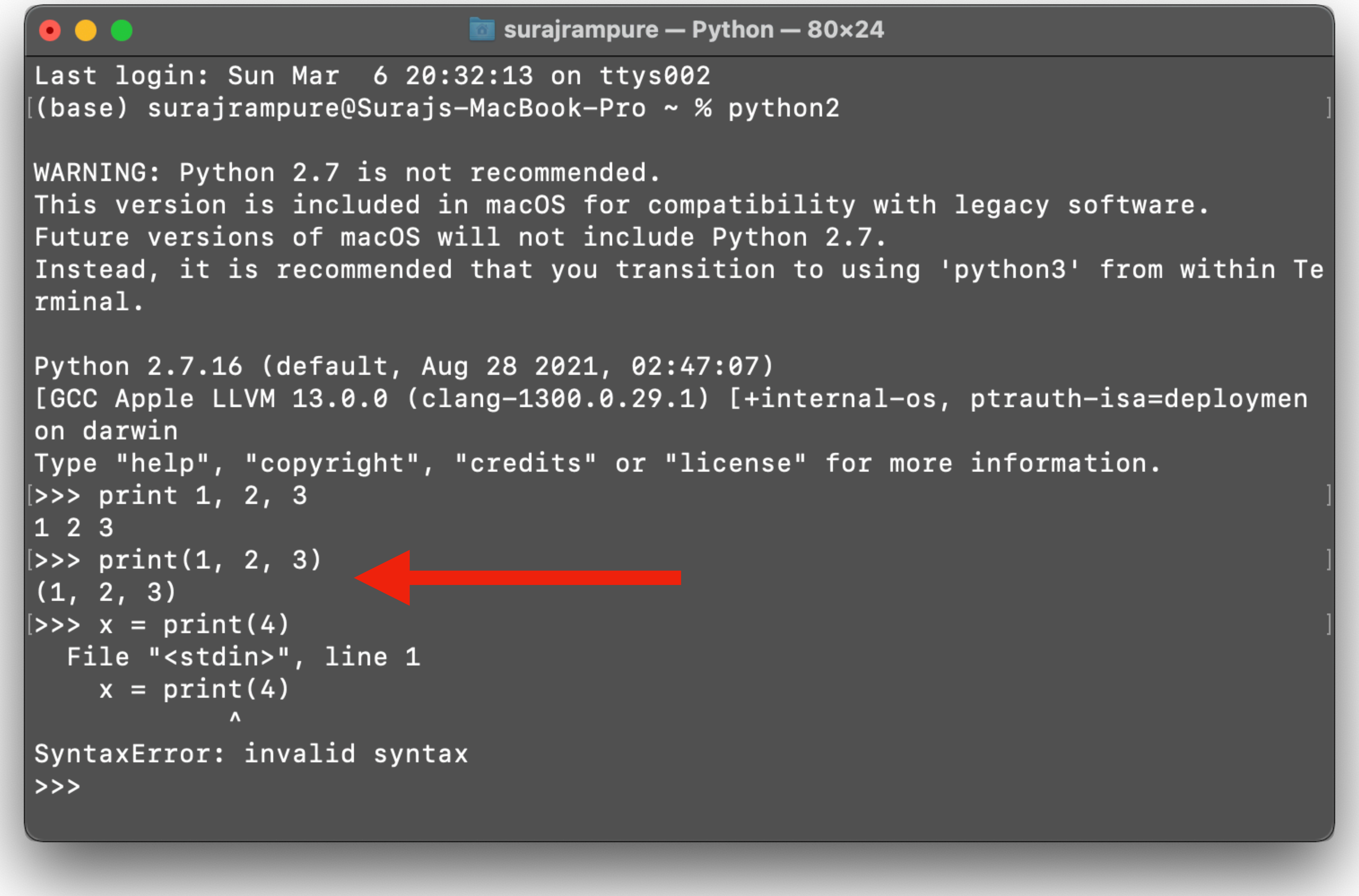
(source)

Differences between v2 and v3

- This document details all of the changes between Python 2 and 3; some are listed below.
- **Printing**: In Python 2, `print` is a keyword, like `for` or `if`. In Python 3, `print` is a function.
- **Division**: In Python 2, dividing two integers always returns an integer (i.e. integer division). In Python 3, dividing two integers always returns a float (i.e. the true result).
- **Some variable names:** In Python 2, you were able to use `True`, `False`, and `None` as variable names. In Python 3, you cannot.
- **List comprehension**: In Python 2, the loop variable used in a list comprehension can overwrite the global definition of that variable if it exists globally. In Python 3, this bug is fixed.

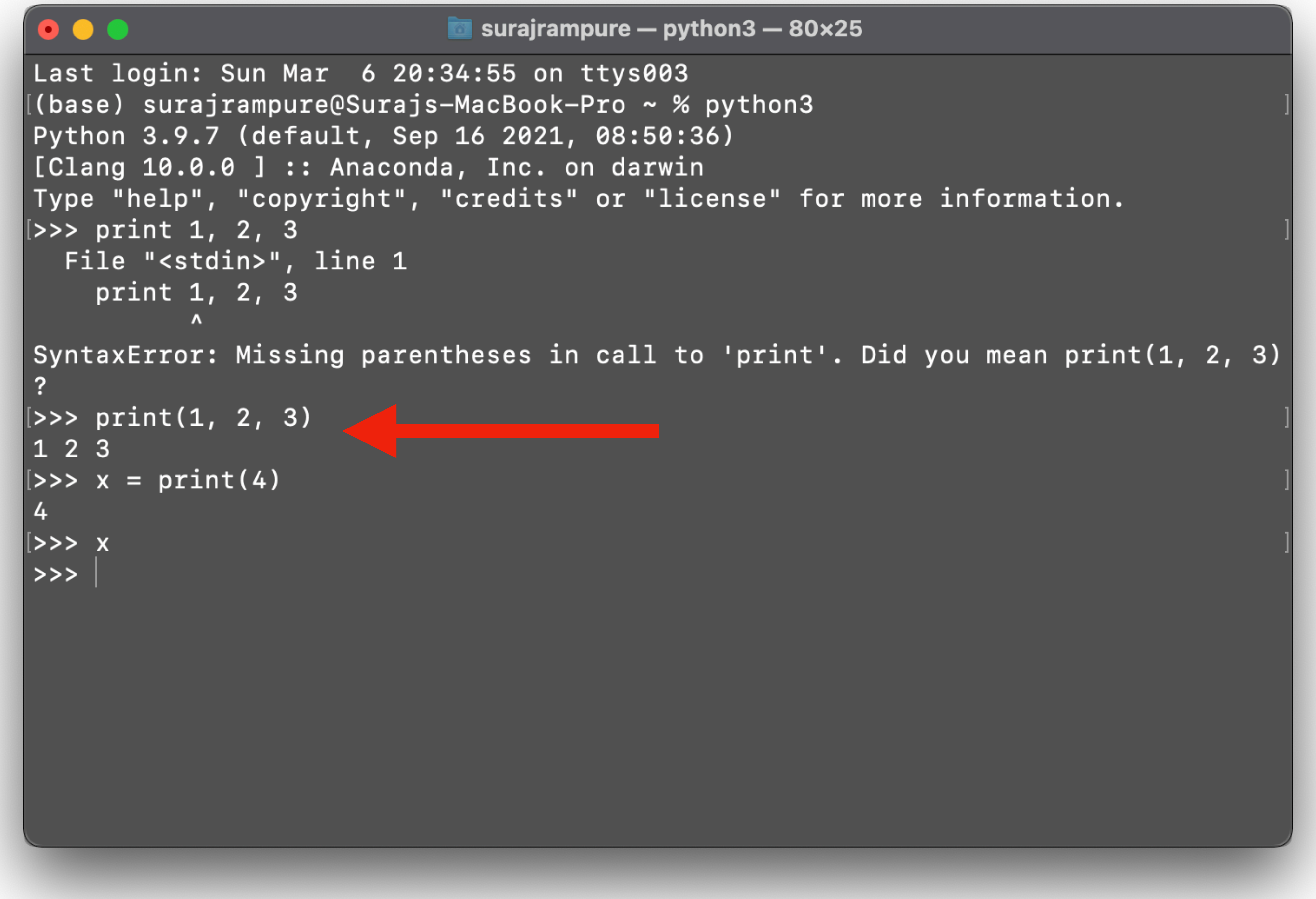
(demo)

Python 2: **`print`** is a **keyword**, like `for` or `if`.



```
Last login: Sun Mar  6 20:32:13 on ttys002
(base) surajrampure@Surajs-MacBook-Pro ~ % python2

WARNING: Python 2.7 is not recommended.
This version is included in macOS for compatibility with legacy software.
Future versions of macOS will not include Python 2.7.
Instead, it is recommended that you transition to using 'python3' from within Te
rminal.

Python 2.7.16 (default, Aug 28 2021, 02:47:07)
[GCC Apple LLVM 13.0.0 (clang-1300.0.29.1) [+internal-os, ptrauth-isa=deploymen
on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print 1, 2, 3
1 2 3
>>> print(1, 2, 3)
(1, 2, 3)
>>> x = print(4)
  File "<stdin>", line 1
    x = print(4)
            ^
SyntaxError: invalid syntax
>>>
```

Python 3: **`print`** is a **function**.



```
Last login: Sun Mar  6 20:34:55 on ttys003
(base) surajrampure@Surajs-MacBook-Pro ~ % python3
Python 3.9.7 (default, Sep 16 2021, 08:50:36)
[Clang 10.0.0 ] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print 1, 2, 3
  File "<stdin>", line 1
    print 1, 2, 3
          ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print(1, 2, 3)
?
>>> print(1, 2, 3)
1 2 3
>>> x = print(4)
4
>>> x
>>>
```

Pérez and IPython

- Fernando Pérez (1972-present) is a Colombian-American physicist and software engineer, and is currently a professor of Statistics at UC Berkeley. He developed **IPython**, which later evolved into **Project Jupyter**.

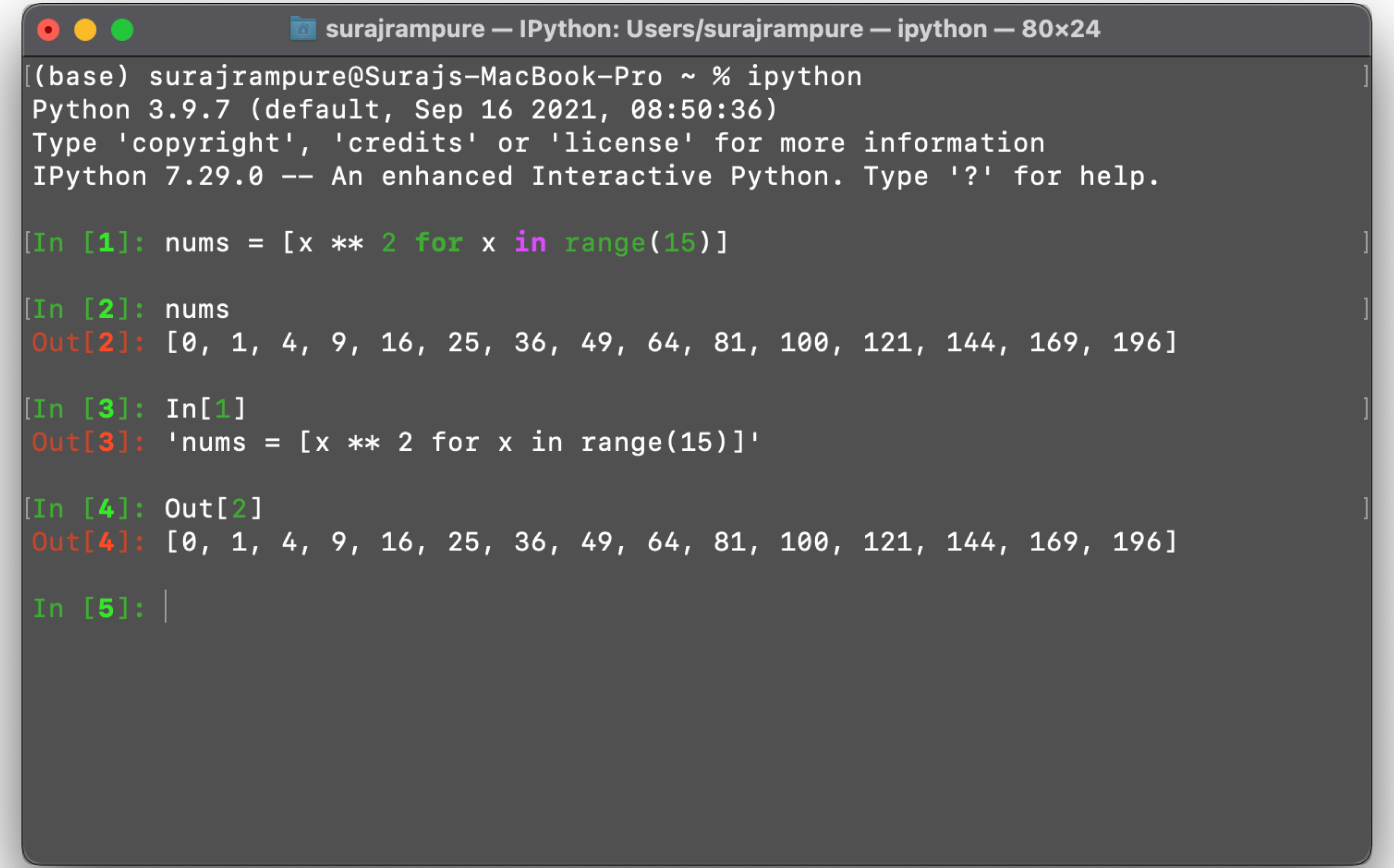
- Pérez developed IPython – which stands for interactive Python – as a physics Ph.D. student at the University of Colorado, Boulder in 2001.

    - IPython began its life in the Terminal, where one could manipulate data and create visualizations in an exploratory, interactive manor, **in a way that outputs of previous lines of code were programmatically accessible.**

        - Version 0.0.1 of IPython is available on GitHub.

    - **matplotlib**, the de-facto standard visualization library in Python, was developed by **John Hunter** as a way of bringing MATLAB-style visualization to IPython.

    - In 2012, **Brian Granger** and **Min Ragan-Kelley** worked with Pérez to add a notebook interface to IPython (ipynb stands for "interactive python notebook").



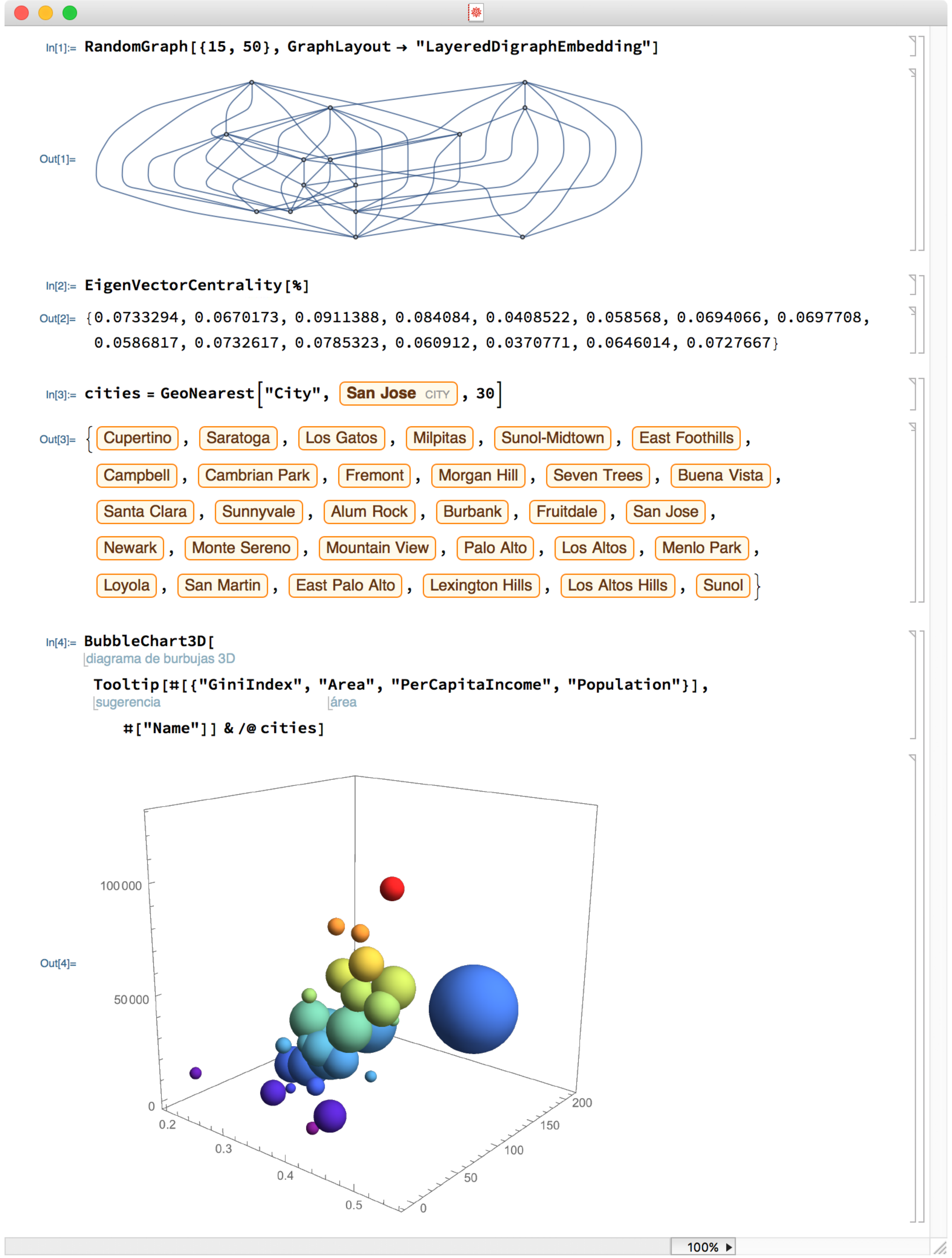
surajrampure — IPython: Users/surajrampure — ipython — 80×24

```
(base) surajrampure@Surajs-MacBook-Pro ~ % ipython
Python 3.9.7 (default, Sep 16 2021, 08:50:36)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.29.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: nums = [x ** 2 for x in range(15)]

In [2]: nums
Out[2]: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196]

In [3]: In[1]
Out[3]: 'nums = [x ** 2 for x in range(15)]'

In [4]: Out[2]
Out[4]: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196]

In [5]:
```

Screenshot of ipython in the Terminal (which still works today). Note that the inputs and outputs of earlier lines of code are programmatically accessible.

“I was a graduate student in physics and I had started using Python to analyze the data for my Ph.D. thesis. I realized that it was possible to use Python in that interactive, exploratory manner, but it was limited. I thought maybe I could build a small tool that would make that process of running a bit of code, maybe plotting, visualizing some data, continuing to write code based on what I'm looking at in the figure, to open a data file -- that exploratory process -- easier.”

- Fernando Pérez (source)



```
In[1]:= RandomGraph[{15, 50}, GraphLayout → "LayeredDigraphEmbedding"]

Out[1]=

In[2]:= EigenVectorCentrality[%]

Out[2]= {0.0733294, 0.0670173, 0.0911388, 0.084084, 0.0408522, 0.058568, 0.0694066, 0.0697708,
         0.0586817, 0.0732617, 0.0785323, 0.060912, 0.0370771, 0.0646014, 0.0727667}

In[3]:= cities = GeoNearest["City", San Jose CITY, 30]

Out[3]= {Cupertino, Saratoga, Los Gatos, Milpitas, Sunol-Midtown, East Foothills,
         Campbell, Cambrian Park, Fremont, Morgan Hill, Seven Trees, Buena Vista,
         Santa Clara, Sunnyvale, Alum Rock, Burbank, Fruitdale, San Jose,
         Newark, Monte Sereno, Mountain View, Palo Alto, Los Altos, Menlo Park,
         Loyola, San Martin, East Palo Alto, Lexington Hills, Los Altos Hills, Sunol}

In[4]:= BubbleChart3D[
        diagrama de burbujas 3D
         Tooltip[#[{"GiniIndex", "Area", "PerCapitaIncome", "Population"}],
         sugerencia                       área
            #["Name"]] & /@ cities]

Out[4]=
```

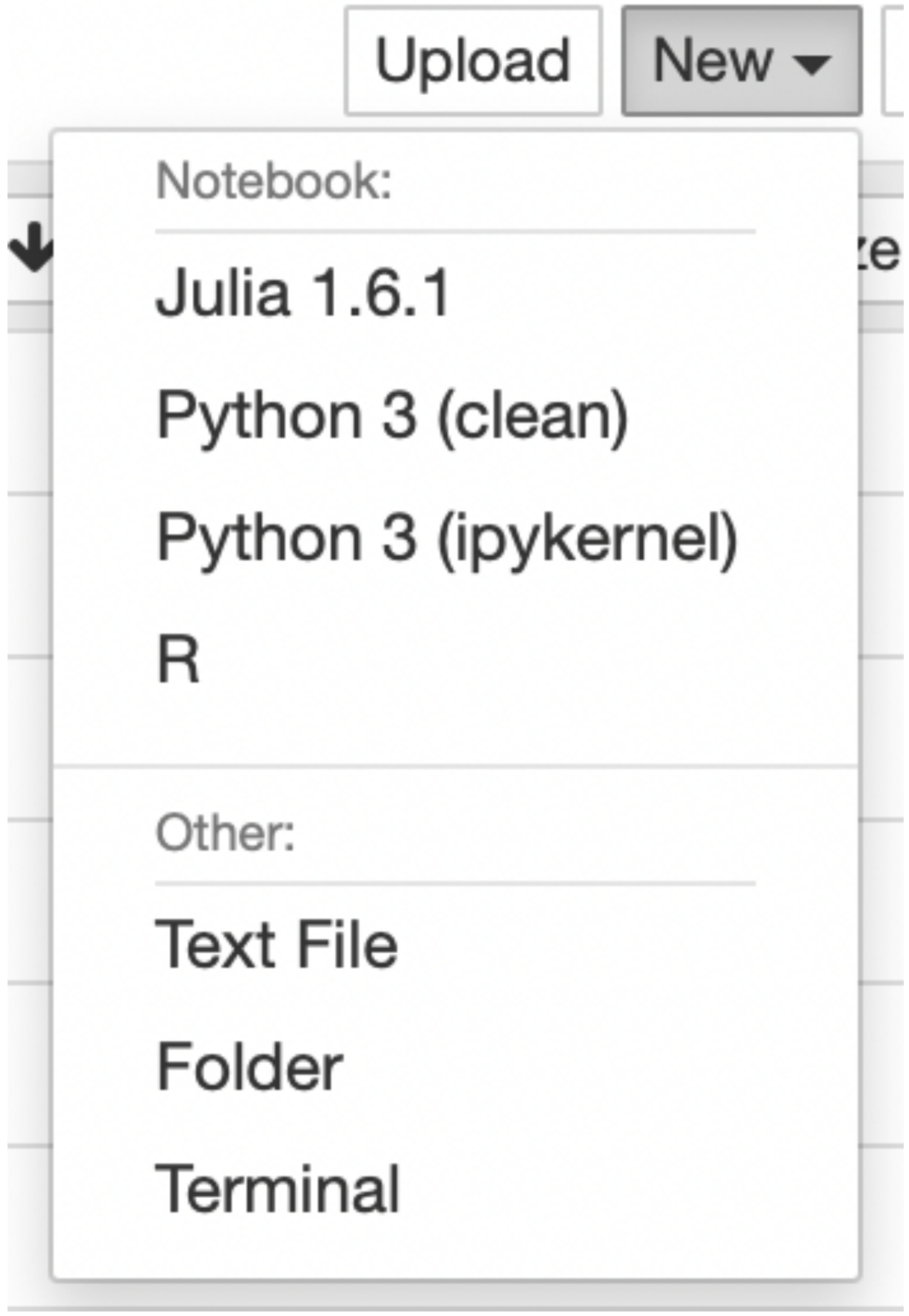
Part of the inspiration of both IPython and the IPython/Jupyter Notebook interface was Mathematica, a notebook-based computation environment developed by WolframAlpha.

Mathematica supported notebooks as early as 1988.

Jupyter Notebooks

- In 2014, the IPython Project evolved into **Project Jupyter**, which supports the development of Jupyter Notebooks and other related tools across a variety of languages.
  - Jupyter stands for **Ju**lia, **Pyt**hon, and **R** – the three core languages that Project Jupyter supports.
  - Project Jupyter is non-profit and open-source.
- Different **kernels** enable the use of Jupyter notebooks with different languages.
  - The default kernel is IPython, though there exist kernels for over 100 languages (including Java)!
- In 2021, Nature recognized Jupyter Notebooks as being one of 10 pieces of software that "**transformed science**" (source).

The DataHub that you have access to for this class has Julia and R kernels built-in.



Data science as a field

From Lecture 1: origins of the term "data science"

- John Tukey, the originator of many ideas in modern data science, wrote "The Future of Data Analysis" in 1962[1,2], in which he said:

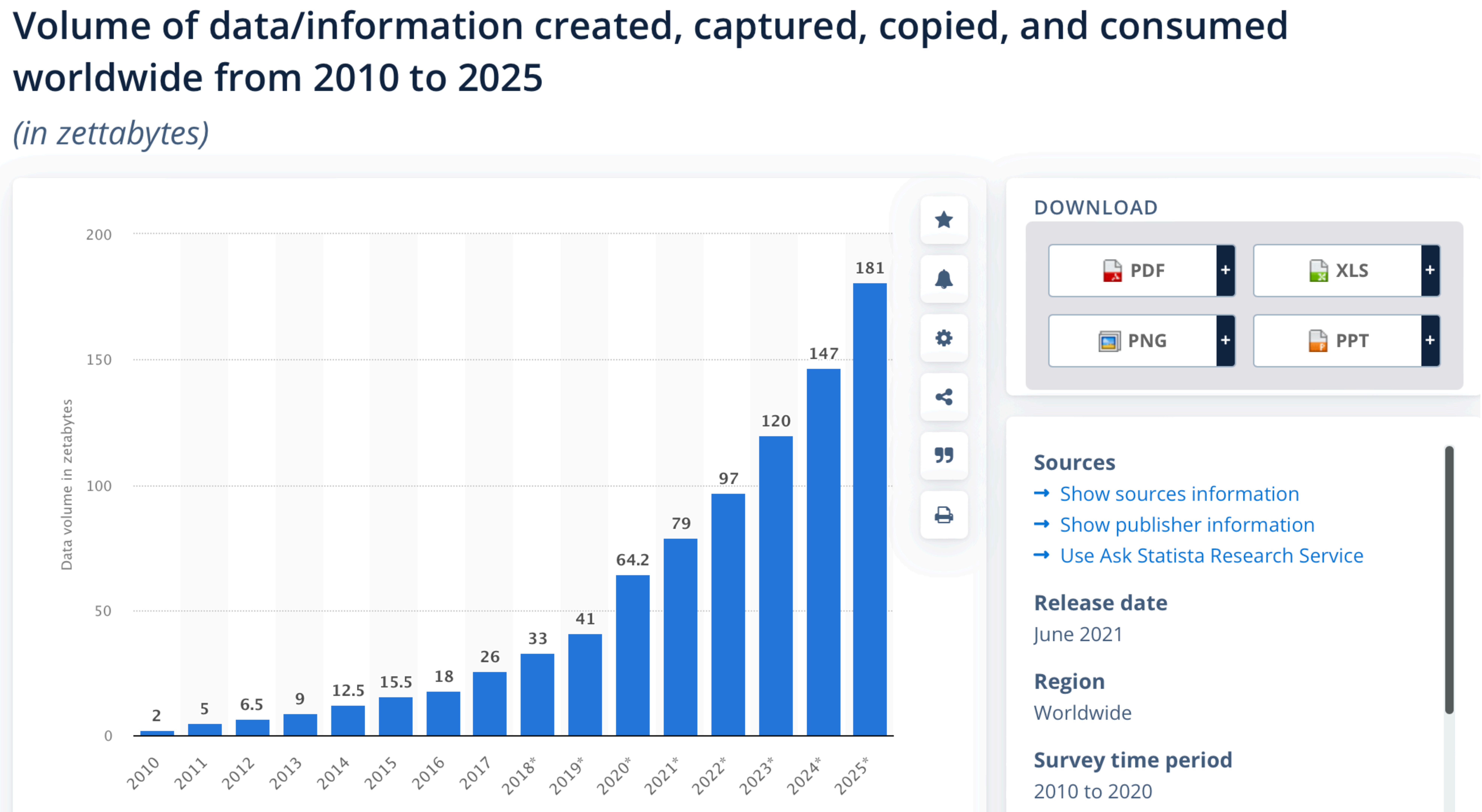
  > *"For a long time I have thought I was a statistician, interested in inferences from the particular to the general. But as I have watched mathematical statistics evolve, I have had cause to wonder and to doubt… All in all, I have come to feel that my central interest is in data analysis, which I take to include,* ***among other things: procedures for analyzing data, techniques for interpreting the results of such procedures, ways of planning the gathering of data to make its analysis easier, more precise or more accurate, and all the machinery and results of (mathematical) statistics which apply to analyzing data****"*

- In 1974[3], Peter Naur defined "data science" as being:

  > *"The science of dealing with data, once they have been established, while the relation of the data to what they represent is delegated to other fields and sciences."*

[1] Tukey, "The Future of Data Analysis"
[2] Donoho, "50 years of Data Science"
[3] Naur, "Concise Survey of Computer Methods"

Why?



Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025

(in zettabytes)

Sources

→ Show sources information

→ Show publisher information

→ Use Ask Statista Research Service

Release date

June 2021

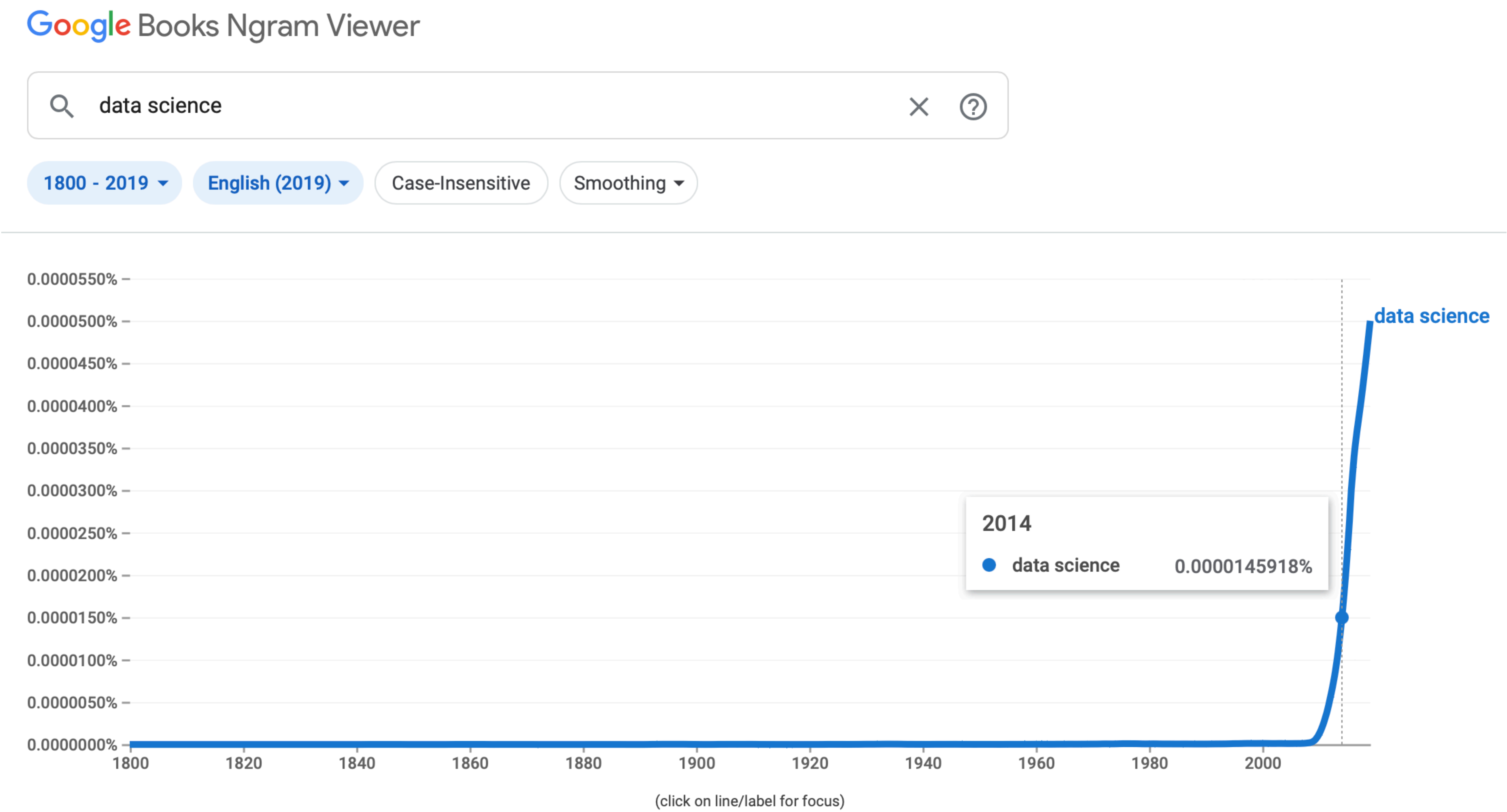
Region

Worldwide

Survey time period

2010 to 2020

1 zettabyte = 10^{21} bytes = 1 trillion terabytes (visualization source)



Google Books Ngram Viewer

data science

1800 - 2019

English (2019)

Case-Insensitive

Smoothing

2014

● data science 0.0000145918%

(click on line/label for focus)

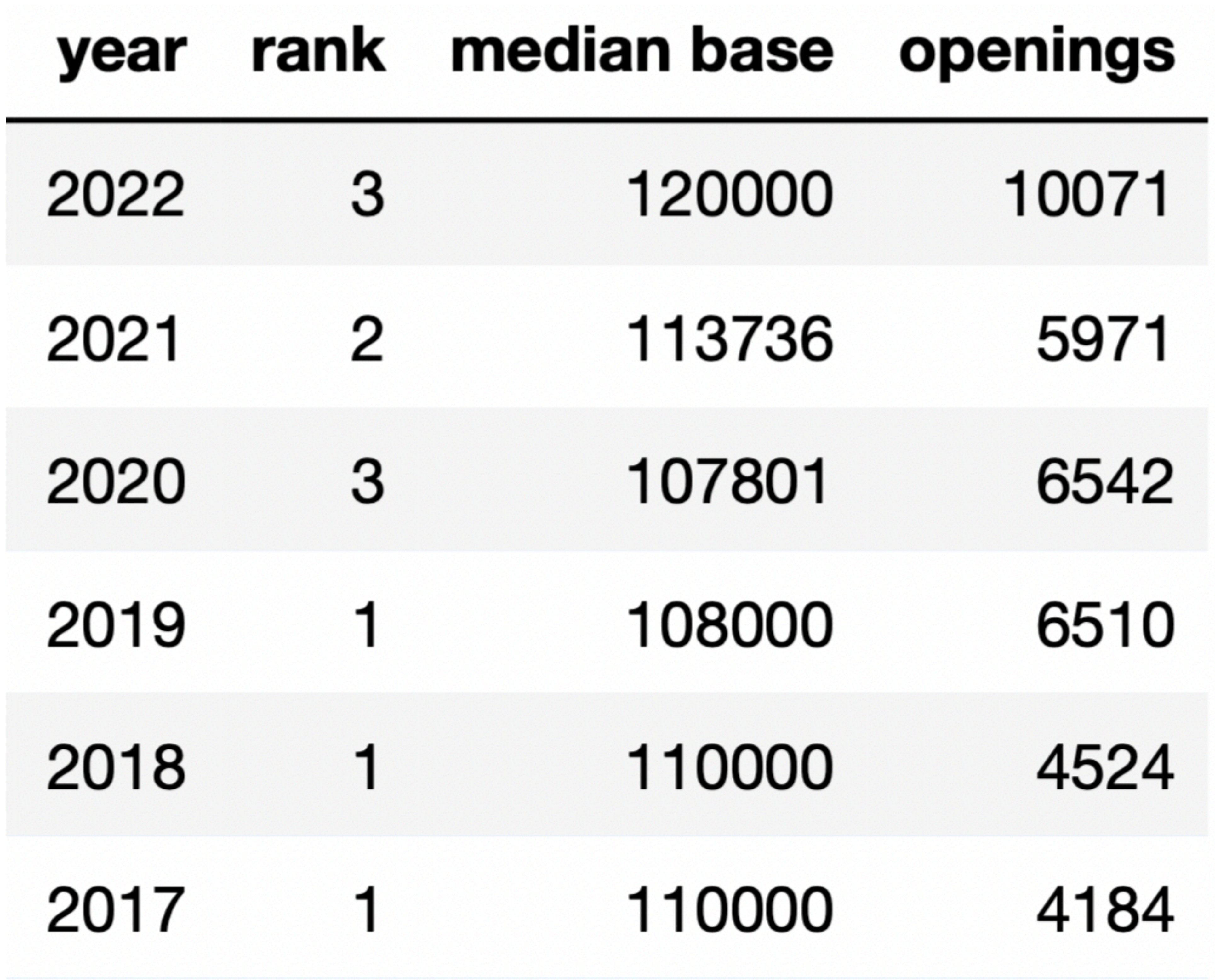
The recent rise of data science

- **1997:** C. F. Jeff Wu, then a professor at the University of Michigan, proclaims that statistics should be renamed data science and that statisticians should be renamed data scientists.
- **2001**: William Cleveland, a professor of Statistics and Computer Science at Purdue University, writes *Data Science: An Action Plan for Expanding the Technical Areas of the Field of Statistics*.
- **2002:** The *Data Science Journal*, organized by CODATA (the Committee on Data for Science and Technology) is first published.
- **2012:** Thomas Davenport and D.J. Patil write *Data Scientist: The Sexist Job of the 21st Century* in the Harvard Business Review.
- **2015:** The US appoints its D.J. Patil as its first Chief Data Scientist.

“…my role as the U.S. CDS will be to responsibly source, process, and leverage data in a timely fashion to enable transparency, provide security, and foster innovation for the benefit of the American public, in order to maximize the nation’s return on its investment in data.”

- *D.J. Patil (UCSD ’96) in 2015, when appointed the first* ***Chief Data Scientist of the US*** (source)

Job growth



year	rank	median base	openings
2022	3	120000	10071
2021	2	113736	5971
2020	3	107801	6542
2019	1	108000	6510
2018	1	110000	4524
2017	1	110000	4184

Collected from Glassdoor's "50 Best Jobs in America".

Conclusion

What could we have spent more time covering?