STATS 507
Data Analysis in Python

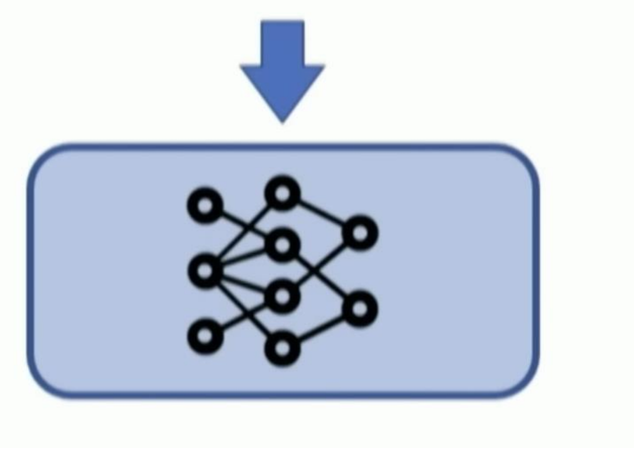
Week1: Syllabus, Installing Python/Jupyter, Intro to Git

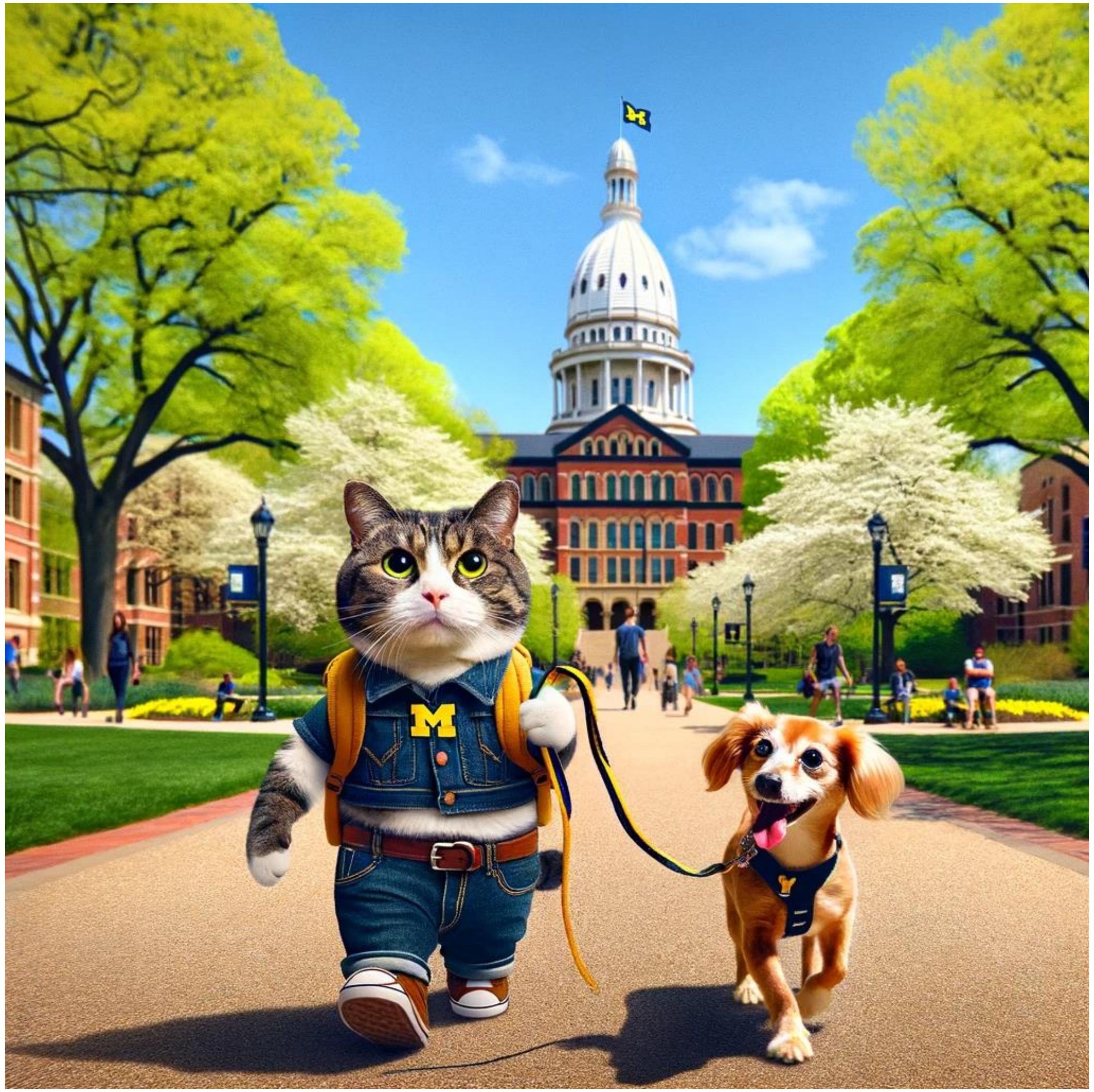
Dr. Xian Zhang

Adapted from slides by Professor Jeffrey Regier

Generating Images from Natural Language

Prompt: “A cat walking a dog at University of Michigan central campus”

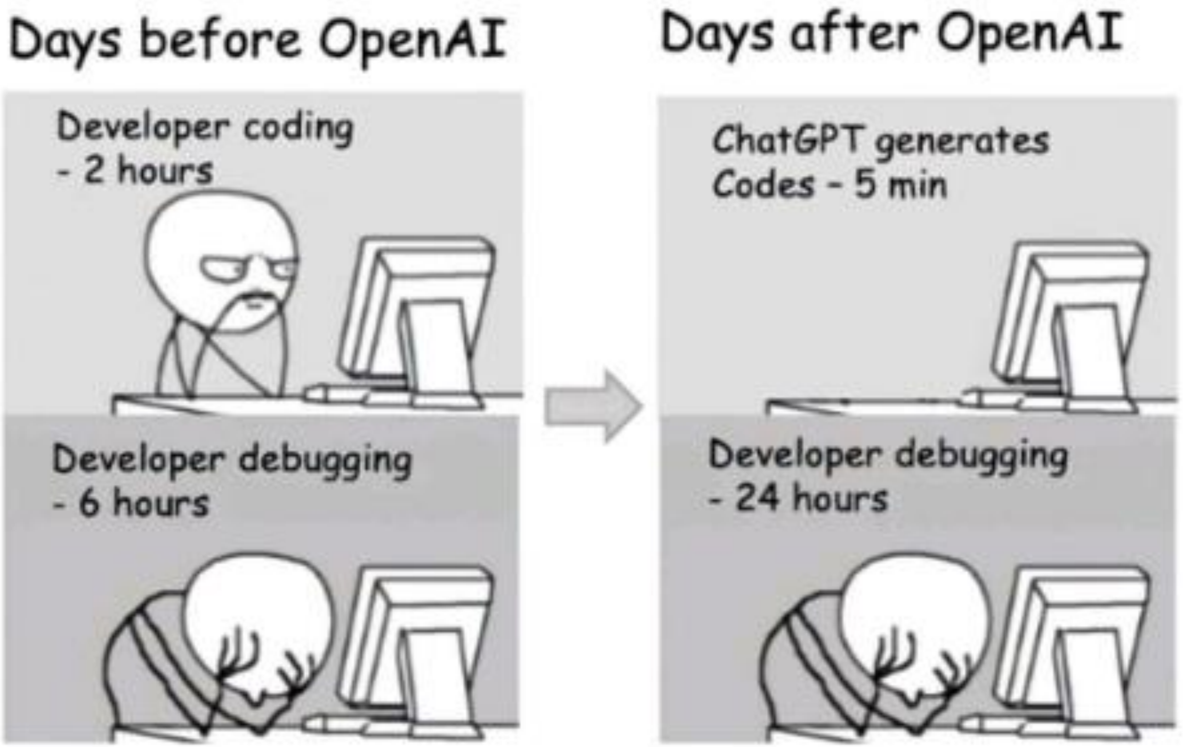




Generated by **DALL·E 3**: able to generate **new data**

Introduction

Big data/ deep learning is revolutionizing so many fields…


Days before OpenAI
Days after OpenAI
Developer coding - 2 hours
Developer debugging - 6 hours
ChatGPT generates Codes - 5 min
Developer debugging - 24 hours

A great tool (used properly) for software engineer/data scientist/education

(The ethics guideline: https://genai.umich.edu/guidance/students)

Isaac Gym from Nvidia Isaac Sim

Robotics

1. Syllabus

2. Intro to Python and Jupyter

3. Intro to Version Control Systems (git)

Course Goals

1. Establish a broad background in **Python** programming
2. Prepare you for the inevitable coding interview
3. Survey popular **tools** in academia/industry for data analysis
4. Learn how to read documentation and quickly get familiar with new tools.

Tool/programming languages might be obsolete one day…

…But not your ability to learn new frameworks and solve problems!

Scope of STATS 507

Part 1: Introduction to Python

Data types, functions, classes, objects, algorithm thinking, functional programming

Part 2: Numerical Computing and Data Visualization

numpy, scipy, scikit-learn, matplotlib, Seaborn

Part 3: Dealing with structured data

pandas, regular expressions, retrieving web data, SQL, real datasets

Part4: Intro to Deep Learning

PyTorch, Perceptron, Multi-layer perceptron, SGD, regularization, CNN…

Prerequisites

Some background on **programming** and **statistics**

Come speak to me if:

- This is your first programming course
- You have never taken a probability or statistics course

This course is probably not for you if:

- You have no programming background

MORE ON THIS LATER

Instructors

Xian Zhang, *xianz@umich.edu*

Ph.D., Software Engineer

https://sites.google.com/view/xian-zhang/home

Graduate Student Instructors

Roman Kouznetsov, *roko@umich.edu*

Statistics PhD student & 507 alum

Julian Bernado, *bernado@umich.edu*

Statistics PhD student & 507 alum

My office hours:

Friday 3:30 PM – 5:00 PM

Zoom: https://umich.zoom.us/j/8786375189

GSI Office hours:

Monday 10:30 AM – 12:30 PM

Zoom: https://umich.zoom.us/j/9285005633

Wednesday 10:00 AM – 12:00 PM
Angell Hall 219

Course Website

Canvas

- Read the **syllabus**
- Look ahead at lecture slides and in-class practice
- Download the homework **assignments**
- **Ask questions** about homework assignments
- Find **office hours** times and **zoom** links
- **Submit** homework solutions
- Check your **grades**
- and more!

Getting help

	Canvas discussion board	GSI office hours	my office hours	email GSIs	email me
questions about homework	☺ *	☺		✗	✗
questions about lecture / slides / course topics		☺	😃	✗	✗
questions / concerns about grading	✗	☺	☺ **	☺	
personal matters; concerns about course; extended illness	✗		✓	✗	✓

* For questions about homework that cannot be asked without revealing a solution, please ask during GSI office hours rather than through Canvas.

** Please ask the GSIs first about homework grading; come to me if your concern is not resolved.

Course Format

This is a **lecture** format class and **not recorded**. Slides will be uploaded the night before class.

We meet Monday & Wednesday, from 8:30 - 10:00 AM in AUD C AH. We will end by 9:50 am.

I do not take attendance.

Textbooks

We recommend the following books, although we will not follow them closely:

The first part of the course is based on *Python for Everybody* by Charles Severance:

https://www.py4e.com/book.php

The second part of the course is based on *Understanding Deep Learning* by Simon J.D. Prince:

https://udlbook.github.io/udlbook/

Online resources

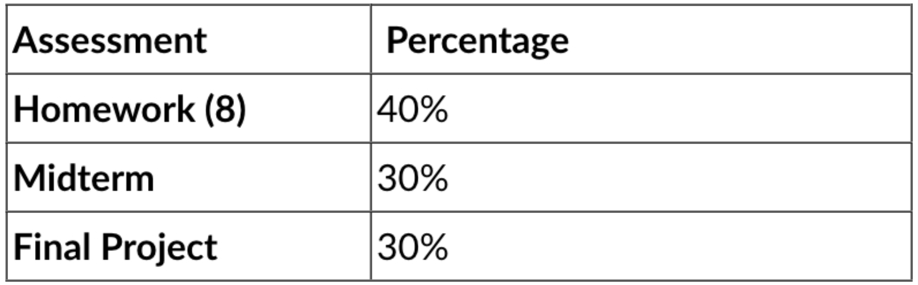
Another goal of this course is to **get you comfortable reading docs**!

Read and understand what you can, google (chatgpt) terms you don’t understand.

- *NumPy quickstart*, https://numpy.org/devdocs/user/quickstart.html
- *SciPy tutorial*, https://docs.scipy.org/doc/scipy/tutorial/index.html
- *Pyplot tutorial*, https://matplotlib.org/stable/tutorials/pyplot.html
- *Pandas tutorial*, https://pandas.pydata.org/pandas-docs/stable/user_guide/
- *Seaborn tutorial*, https://seaborn.pydata.org/tutorial.html
- *Scikit-learn tutorial*, https://scikit-learn.org/stable/tutorial/
- *PyTorch* tutorial, https://pytorch.org/tutorials/

Grading

Final grades will be based on



Assessment	Percentage
Homework (8)	40%
Midterm	30%
Final Project	30%

The distribution of final grades is expected to be similar to what it has been for previous offering of STATS 507:

45% A/A+, 70% A-/A/A+, 98% B- or above.

Homework (40%)

Students enter 507 with a wide variety of programming backgrounds.

The course workload is heavy for some students and light for others.

1. Students with the strongest programming backgrounds may complete each weekly homework assignment in as little as **5 hours**.
2. Many students complete the homework in around **10 hours**.
3. Students with little prior programming experience will likely learn the most; however, they may need to devote as much as **25 hours weekly** to completing the homework assignments!

Homework & late days (40%)

Homework due dates are strict, However,

- You have **seven** “late days” to use over the course of the semester.
- For each late day you spend, you extend the deadline of a homework by 24 hours.
- You may spend multiple late days per homework.
- Once you have turned in your homework you may not spend more late days to turn in your homework again.

The purpose of this late day policy is to enable you to deal with unexpected circumstances (e.g., illness, family emergencies, job interviews) without having to come to me. Of course if dire circumstances arise (e.g., long-term illness that causes you to miss multiple weeks of lecture), please speak with me as promptly as possible. Due to the university grading schedule, you may not use late days to extend the deadline of the last homework assignment.

Don’t plagiarize!

- You may discuss homeworks with your fellow students, but the work you submit must be your own.
- You may not share your homework solutions notebook with classmates, and you may not access other students’ solutions.
- You must disclose in your homework whom (if anyone) you discussed the homework with.
- Submissions will be checked with the MOSS Plagiarism Detector.
- A zero grade will be given for submissions that contain any plagiarized code.
- All incidents of academic dishonesty will be reported to Rackham, which
- typically administers additional penalties upon conviction.

Midterm (30%)

An **in-person** midterm is scheduled on :

Wednesday, 10/09/24 in AUD C AH

Evaluation of internalization of core ideas covered in lecture.

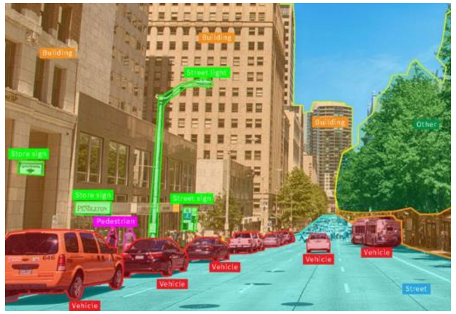
Final Projects (30%)

Assuming knowing Python and related tools, the final project for this course is to build ANY end-to-end project with hugging face where you can use any open-sourced dataset and model (pretrained or fine-tuned).

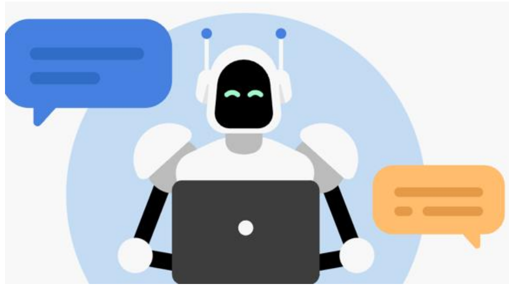
More details on submission guidelines and evaluation criteria will be provided with the project announcement.

Music generation

Computer Vision



Large Language Models



Questions?

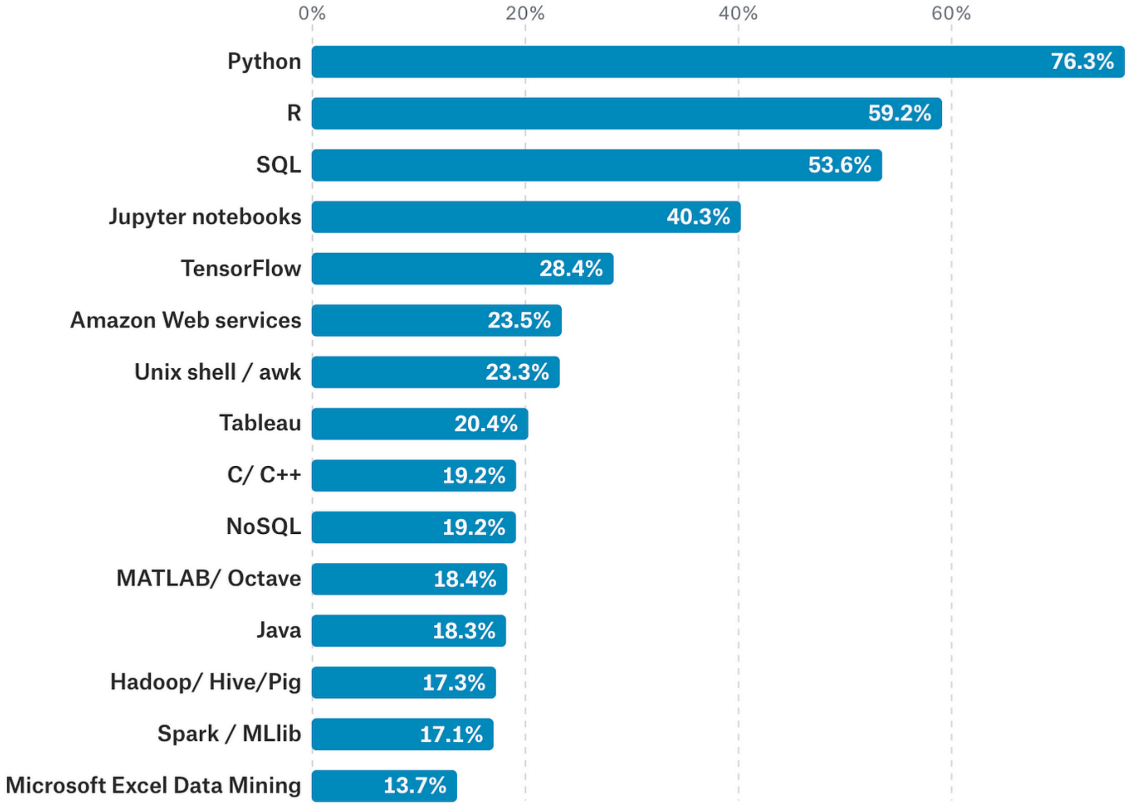
1. Syllabus

2. Intro to Python and Jupyter

3. Intro to version Control

Why Python now?

Increasing, Python is the language of data science and general programming.



- Popular
- Easy to learn
- Open-sourced (no license needed)

Data science community Kaggle’s annual survey “The State of Data Science& Machine Learning” asks the question “What tools are used at work?” (reference)

What is Python?

Python is a **dynamically typed**, **interpreted** programming language

- Created by Guido van Rossum and first released in 1991

Design philosophy: simple, readable

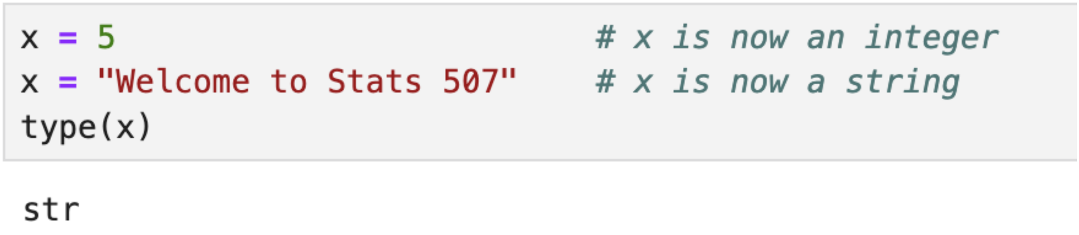
Python is dynamically typed.

Design philosophy: simple, readable

Dynamically typed

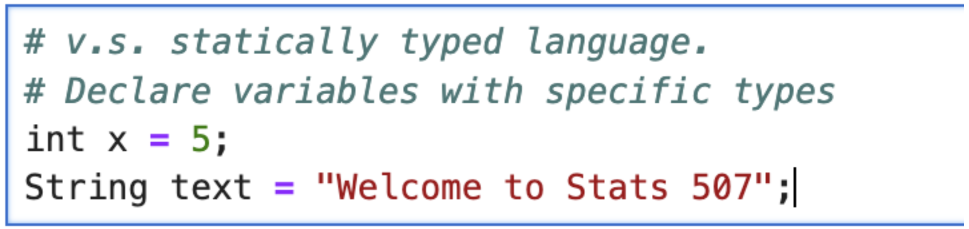
In many languages, when you declare a variable, you must specify the variable's **type** (e.g., int, double, Boolean, string). Python does not require this, the type of a variable is defined at **runtime**.

v.s. statically typed, flexible yet more error-prone



```
x = 5                          # x is now an integer
x = "Welcome to Stats 507"     # x is now a string
type(x)
```

```
str
```



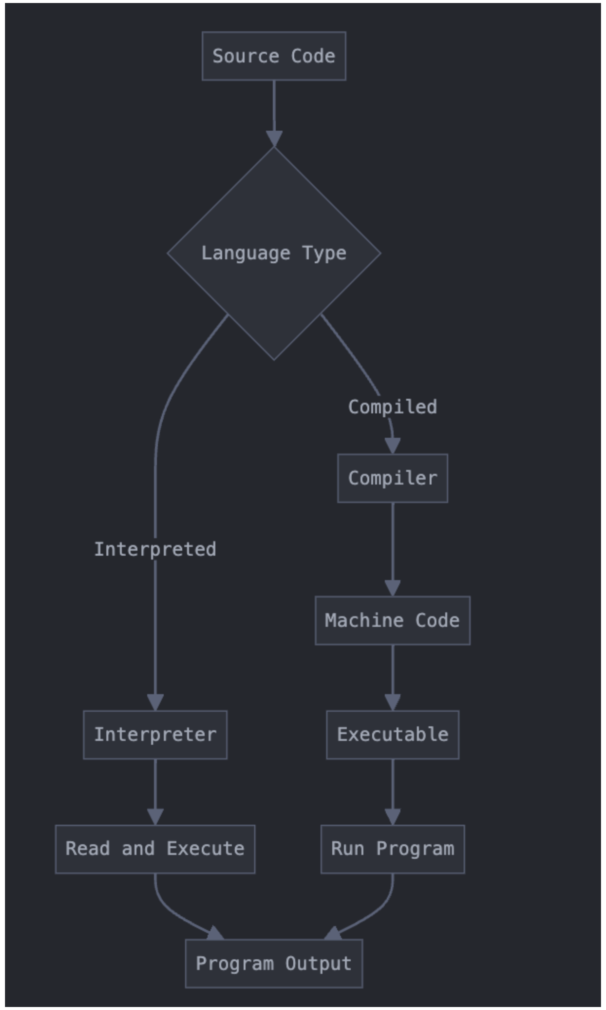
```
# v.s. statically typed language.
# Declare variables with specific types
int x = 5;
String text = "Welcome to Stats 507";
```

An **interpreted** programming language

Interpreted

Some languages (e.g. C/C++ and Java) are compiled: we write code, from which we get a runnable grogram via **compilation**. In contrast, Python is **interpreted**: a program, called the **interpreter**, runs our code directly, line by line.

v.s compiled: simple yet slower



Compiled: Write code -> Compile -> Run executable
Interpreted: Write code -> Run directly with interpreter

Writing and Running Python

Python syntax differs from R, Java, C/C++, MATLAB

- Whitespace delimited
- Limited use of brackets, semicolons, etc.

More on this later…

Running Python

Your HWs and projects must be handed in as **Jupiter** notebooks.

We'll be using Jupyter notebook through Anaconda.

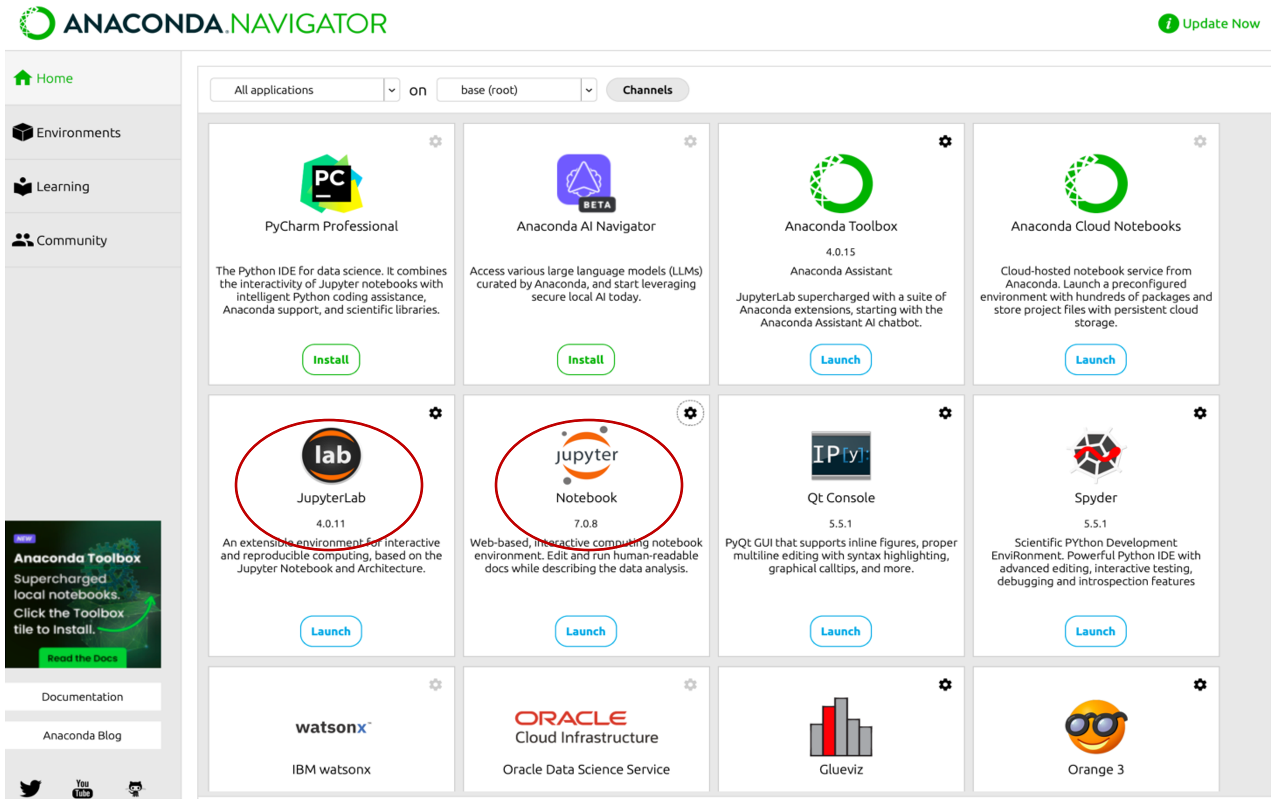
Install Anaconda: https://docs.anaconda.com/

Anaconda: A distribution of Python that includes many useful libraries for data science and scientific computing.

Jupyter Notebook: An interactive web-based environment for creating and sharing documents containing live code, equations, visualizations, and text…

Install Anaconda and launch Jupyter

https://docs.anaconda.com/anaconda/install/



The Python IDE for data science. It combines the interactivity of Jupyter notebooks with intelligent Python coding assistance, Anaconda support, and scientific libraries.

Access various large language models (LLMs) curated by Anaconda, and start leveraging secure local AI today.

JupyterLab supercharged with a suite of Anaconda extensions, starting with the Anaconda Assistant AI chatbot.

Cloud-hosted notebook service from Anaconda. Launch a preconfigured environment with hundreds of packages and store project files with persistent cloud storage.

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

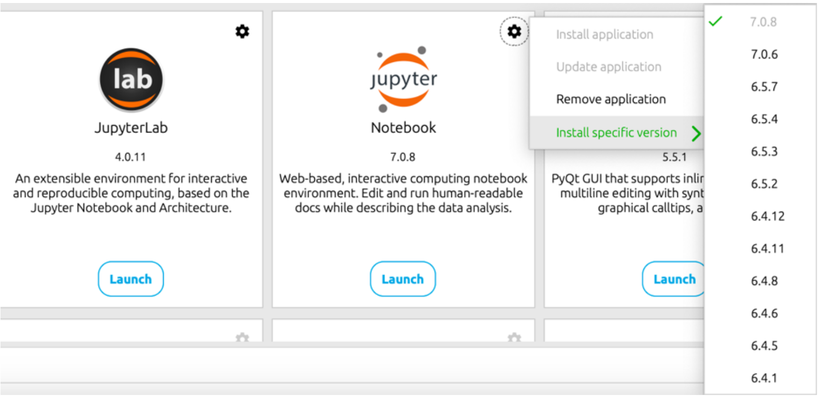
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Anaconda Toolbox Supercharged local notebooks. Click the Toolbox tile to install.

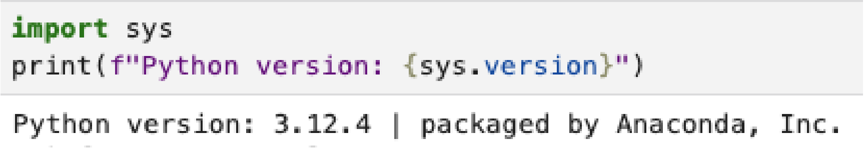
Make sure to install the most recent version:



An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

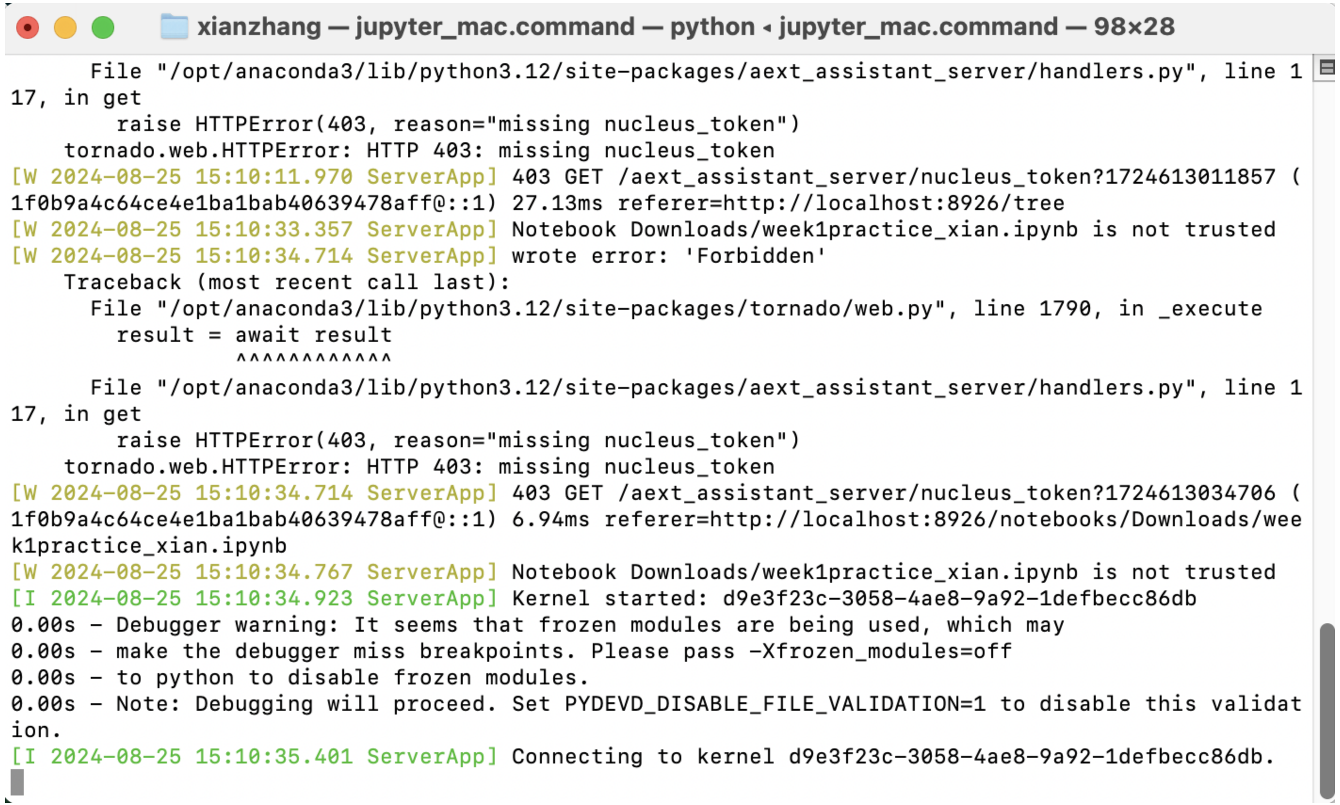
PyQt GUI that supports inlir multiline editing with synt graphical calltips, a



```python
import sys
print(f"Python version: {sys.version}")
```

```
Python version: 3.12.4 | packaged by Anaconda, Inc.
```

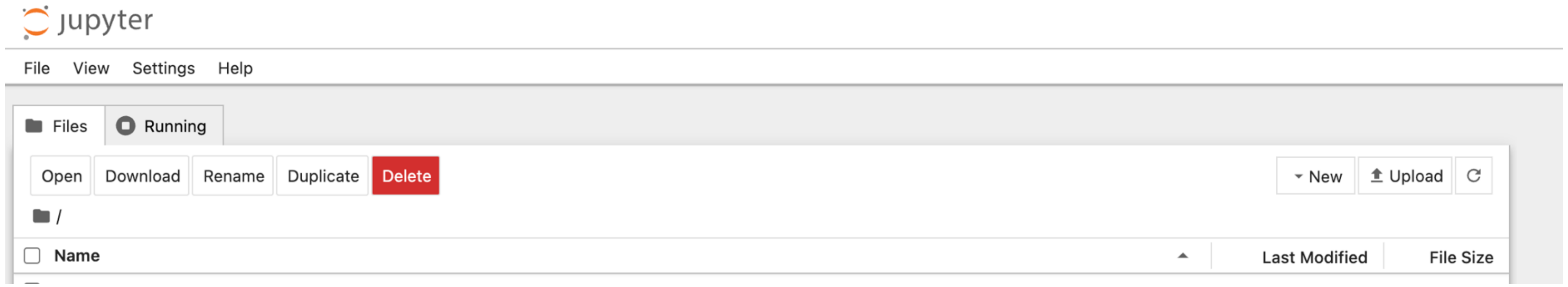
Python in Jupyter

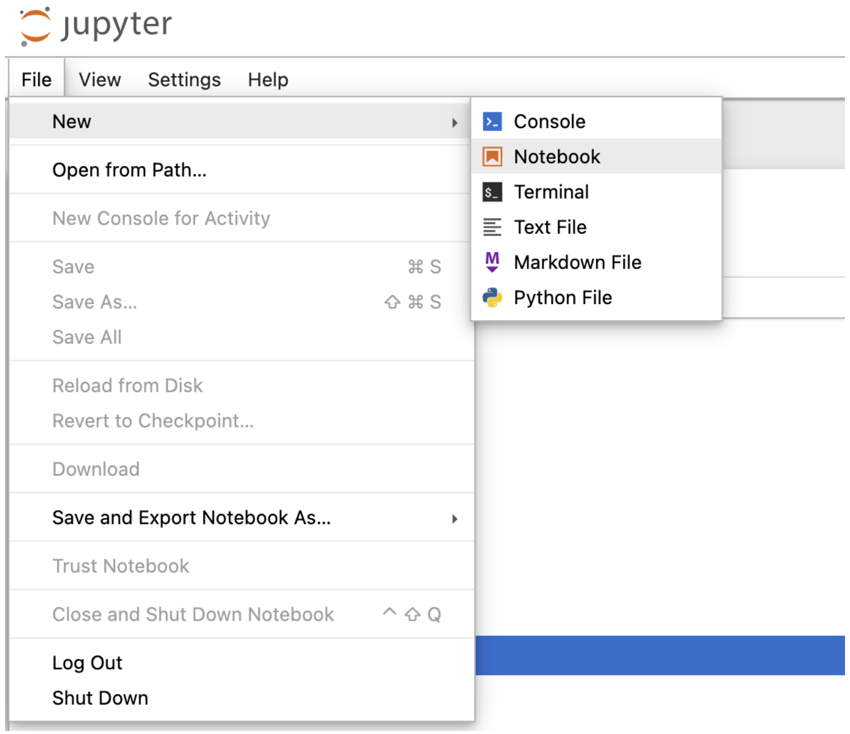

xianzhang — jupyter_mac.command — python ◂ jupyter_mac.command — 98×28

```
    File "/opt/anaconda3/lib/python3.12/site-packages/aext_assistant_server/handlers.py", line 1
17, in get
      raise HTTPError(403, reason="missing nucleus_token")
  tornado.web.HTTPError: HTTP 403: missing nucleus_token
[W 2024-08-25 15:10:11.970 ServerApp] 403 GET /aext_assistant_server/nucleus_token?1724613011857 (
1f0b9a4c64ce4e1ba1bab40639478aff@::1) 27.13ms referer=http://localhost:8926/tree
[W 2024-08-25 15:10:33.357 ServerApp] Notebook Downloads/week1practice_xian.ipynb is not trusted
[W 2024-08-25 15:10:34.714 ServerApp] wrote error: 'Forbidden'
  Traceback (most recent call last):
    File "/opt/anaconda3/lib/python3.12/site-packages/tornado/web.py", line 1790, in _execute
      result = await result
               ^^^^^^^^^^^^
    File "/opt/anaconda3/lib/python3.12/site-packages/aext_assistant_server/handlers.py", line 1
17, in get
      raise HTTPError(403, reason="missing nucleus_token")
  tornado.web.HTTPError: HTTP 403: missing nucleus_token
[W 2024-08-25 15:10:34.714 ServerApp] 403 GET /aext_assistant_server/nucleus_token?1724613034706 (
1f0b9a4c64ce4e1ba1bab40639478aff@::1) 6.94ms referer=http://localhost:8926/notebooks/Downloads/wee
k1practice_xian.ipynb
[W 2024-08-25 15:10:34.767 ServerApp] Notebook Downloads/week1practice_xian.ipynb is not trusted
[I 2024-08-25 15:10:34.923 ServerApp] Kernel started: d9e3f23c-3058-4ae8-9a92-1defbecc86db
0.00s - Debugger warning: It seems that frozen modules are being used, which may
0.00s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.00s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validat
ion.
[I 2024-08-25 15:10:35.401 ServerApp] Connecting to kernel d9e3f23c-3058-4ae8-9a92-1defbecc86db.
```

Jupyter provides some information about its startup process, and then…

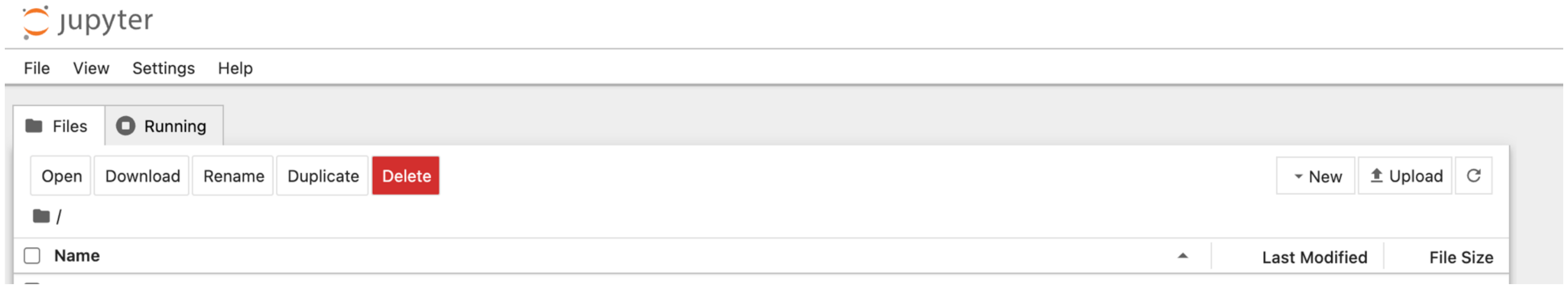
Python in Jupyter

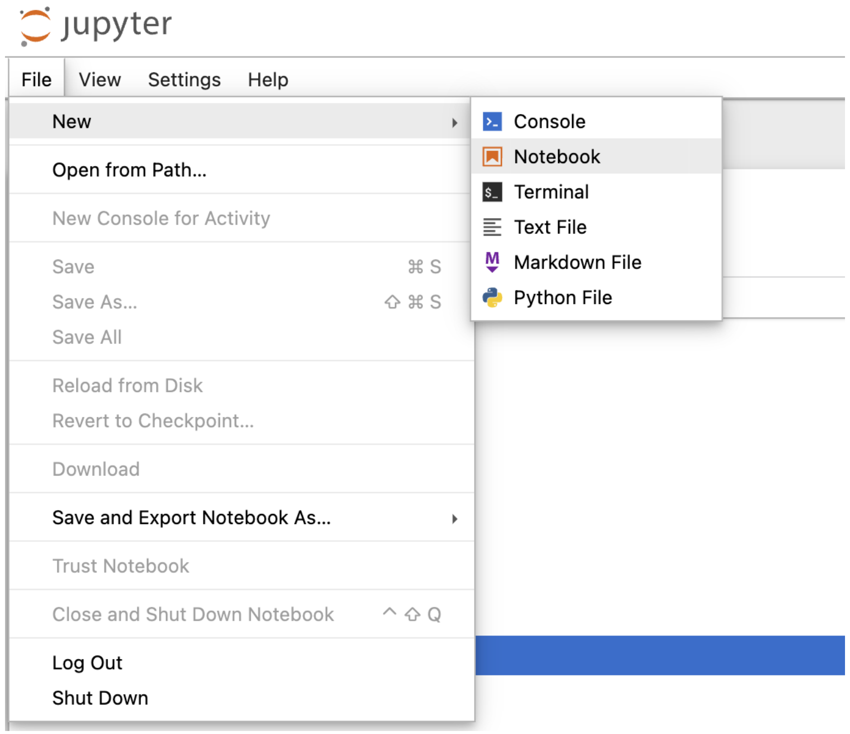




…Jupyter opens a browser window in which you can launch a new notebook or open an existing one…

Python in Jupyter



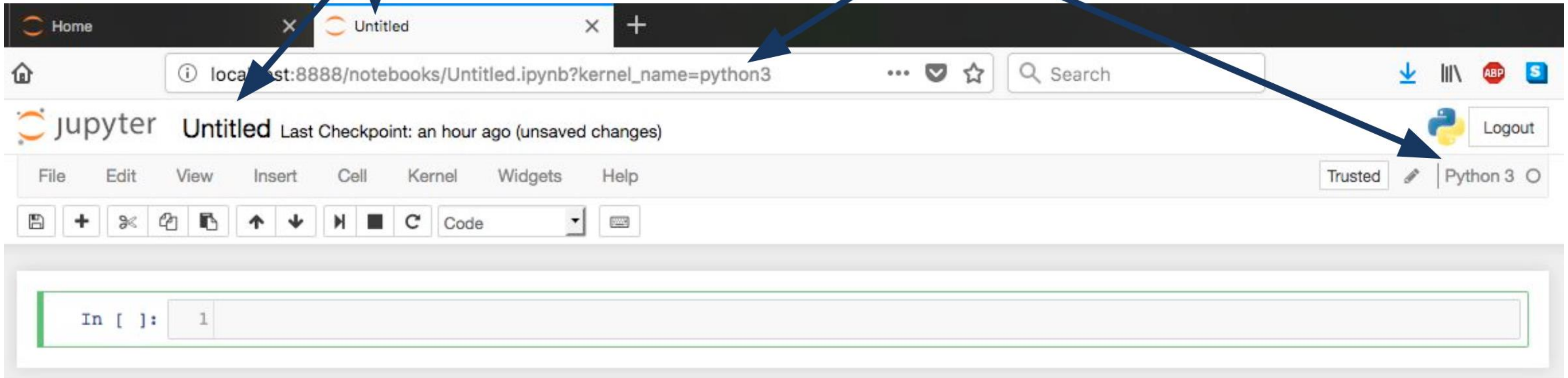


...Jupyter opens a browser window in which you can launch a new notebook or open an existing one...

Python in Jupyter

Notebook doesn't have a title, yet.

Running Python 3


Home
Untitled
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
Search
jupyter
Untitled
Last Checkpoint: an hour ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Code
In [ ]:
1

I'll leave it to you to learn about the other features by reading the documentation. For now, the green-highlighted box is most important. That's where we write Python code.

Write code in the highlighted box, then press shift+enter to run the code in that box...

Demo with in-class practice

Resources and support

Example notebook:
https://nbviewer.jupyter.org/github/jrjohansson/ scientific-python-lectures/blob/master/Lecture-4-Matplotlib.ipynb

Clean, well-organized presentation of code, text and images, in one document

Good tutorials:
https://www.datacamp.com/tutorial/tutorial-jupyter-notebook
https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/execute.html

1. Syllabus

2. Intro to Python and Jupyter

3. Intro to Version Control Systems (git)

What is version control system and git?

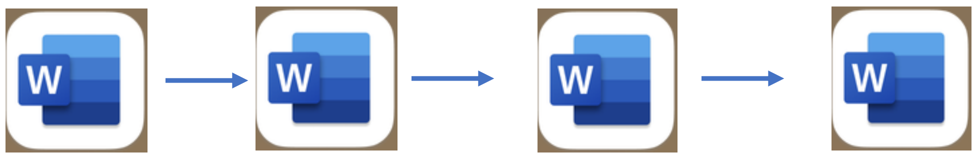
VCS: Tools used to track changes to source code (or other collections of **files** and **folders**). (who wrote what at when + message…)

Ad-hoc (centralized) version control

Created By Xian

Edited By Tim

Edited By Bob



v1

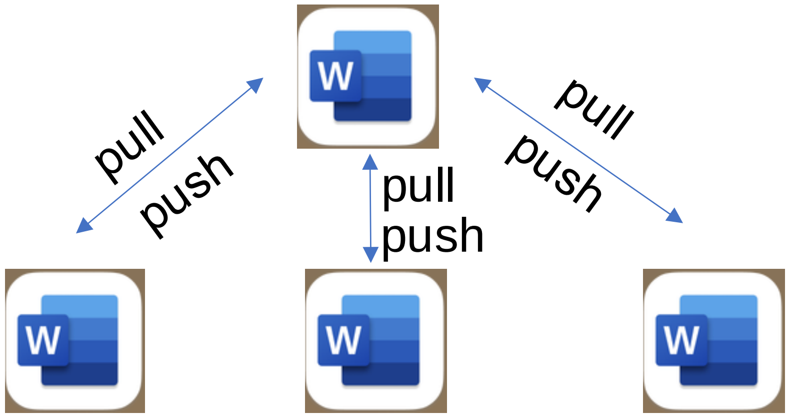
v2

v3

Final version

Add version notes…

Distributed version control



Local branch Xian

Local branch Tim

Local branch Bob

Git is the de facto standard for distributed version control.

Why we need git?

1. By yourself, small projects

- Track history and keep a log of changes
- Parallel branches of development
- Build project portfolios (HWs and projects…)

2. Collaboration, larger projects

- Who wrote this module, when, why…
- Resolving conflicts
- Can easily roll back to previous versions (features were broken…)

How git works…

Git models the history of a collection of **files** and **folders** within some top-level directory as a series of snapshots of history.

(root) (tree)

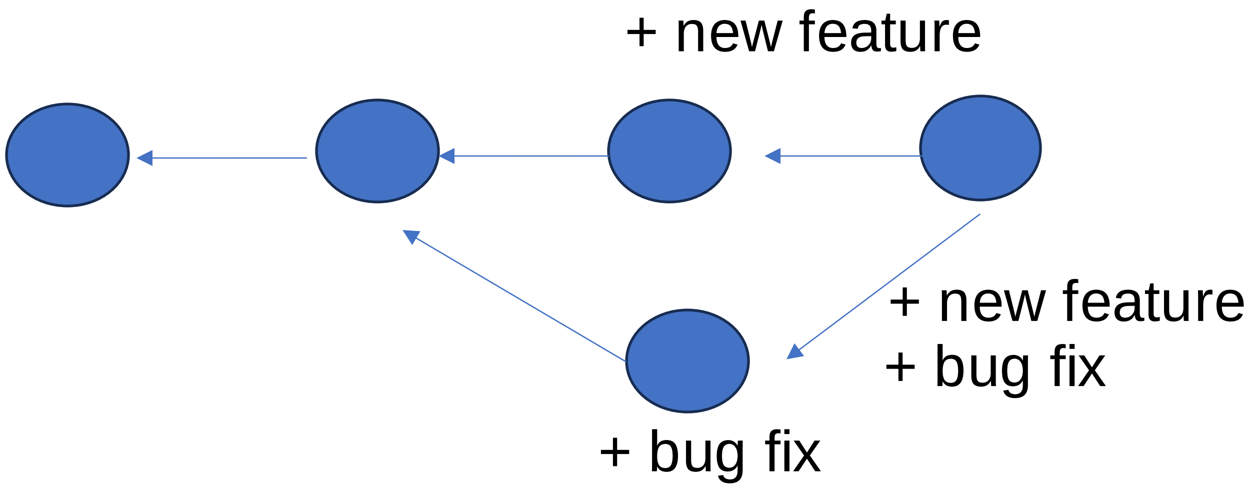
HW1 (tree)

+ hello.txt

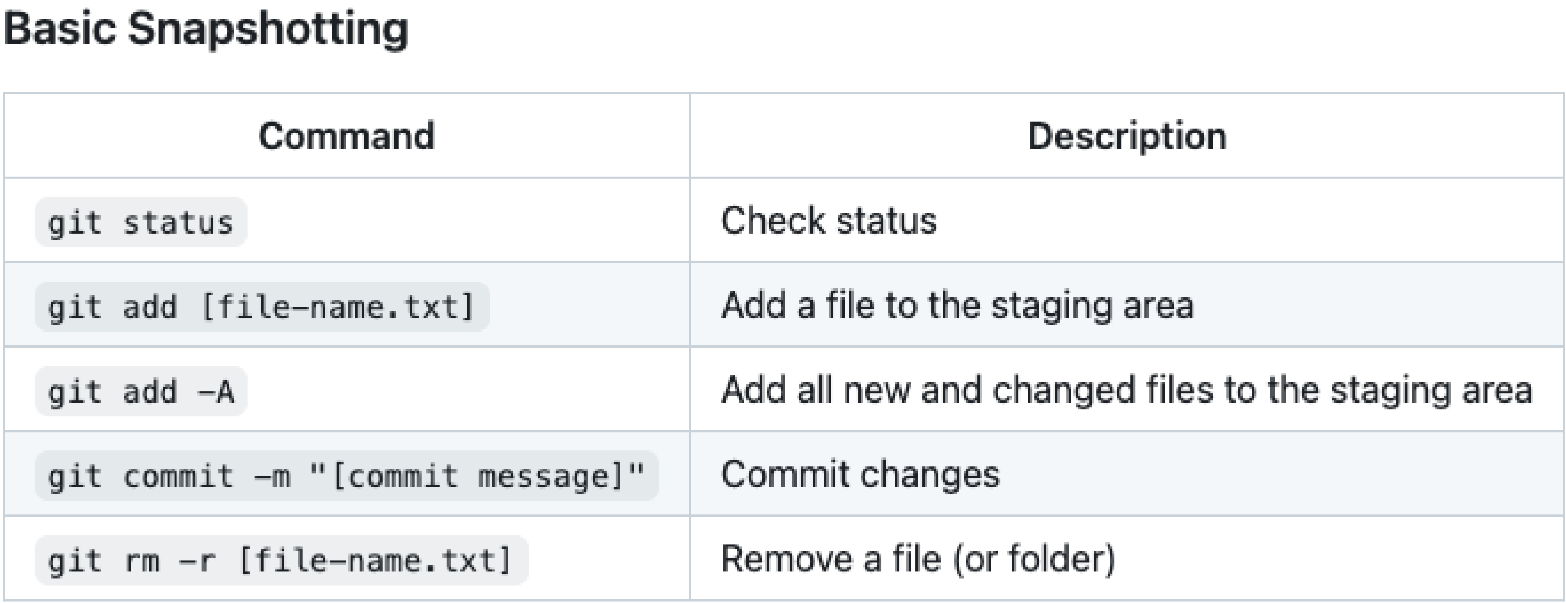
(blob, contents = “Welcome to Stats 507”)

+ readme.txt

(blob, contents = “This is Xian’s repo for stats 507”)



Git commands



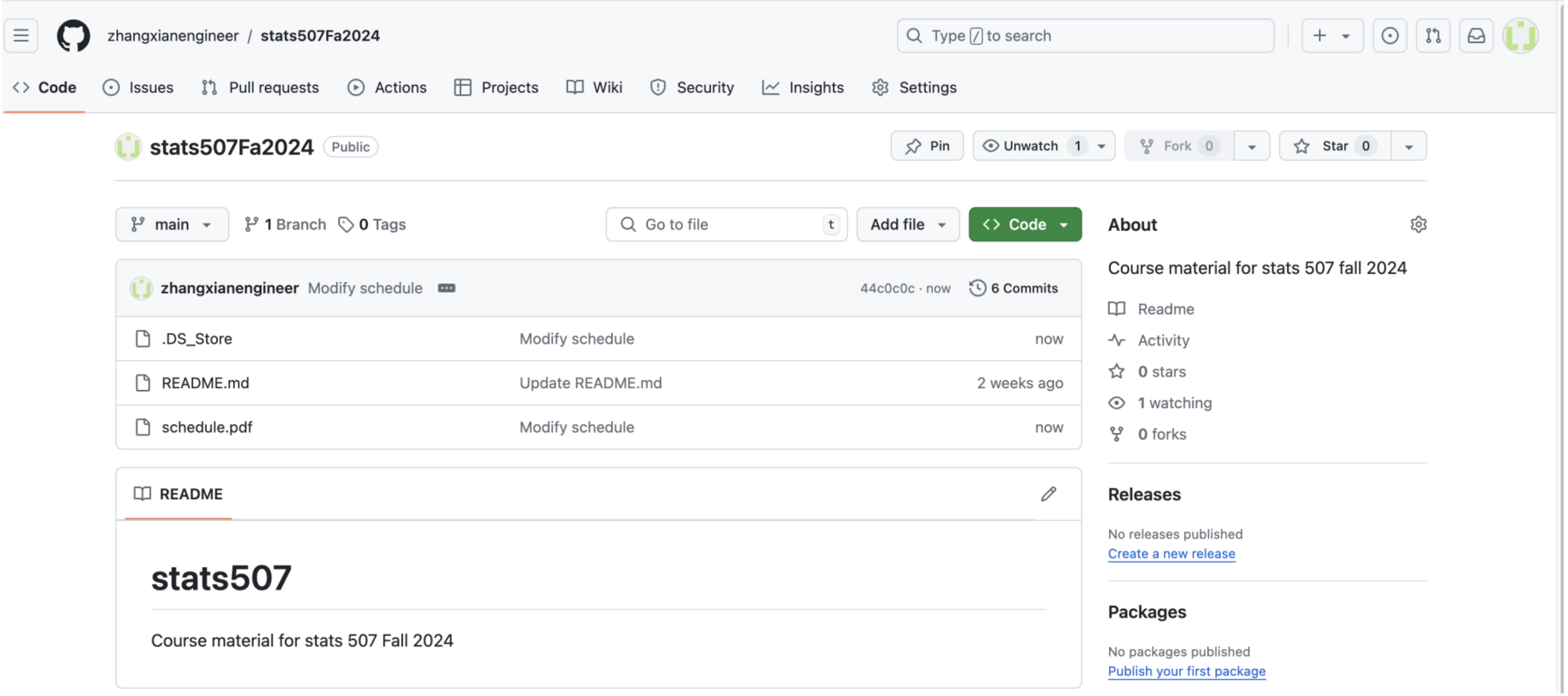
Basic Snapshotting

Command	Description
`git status`	Check status
`git add [file-name.txt]`	Add a file to the staging area
`git add -A`	Add all new and changed files to the staging area
`git commit -m "[commit message]"`	Commit changes
`git rm -r [file-name.txt]`	Remove a file (or folder)

More on: https://github.com/joshnh/Git-Commands

MIT missing semester version control lecture: https://missing.csail.mit.edu/2020/version-control/

Using github for STATS 507



zhangxianengineer Modify schedule 44c0c0c · now 6 Commits

.DS_Store	Modify schedule	now
README.md	Update README.md	2 weeks ago
schedule.pdf	Modify schedule	now

stats507

Course material for stats 507 Fall 2024

About

Course material for stats 507 fall 2024

- Readme
- Activity
- 0 stars
- 1 watching
- 0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

https://github.com/zhangxianengineer/stats507Fa2024

HW: sign up for github and create a stats 507 repo. You should NOT make your HWs public before the due date…

Week 1 practice problem

You will find them on Canvas in “Files/in-class practice”

First try it on your own, then we’ll discuss it.

Things to do:

Install Anaconda and jupyter

Familiarize yourself with jupyter:
https://docs.jupyter.org/en/latest/start/index.html

Sign up for github and create your own stats 507 repo.

Read ch 1-6 in Python 4 Everybody

Other things

HW1 posted on canvas.

If you run into trouble, attend GSI office hours for help.

- You can also post to the canvas discussion board.
- If you’re having trouble, at least one of your classmates is, too.
- You’ll learn more by explaining things to teach other than by reading posts.