Lecture 3: Data in Linguistics, Git/GitHub

LING 1340/2340: Data Science for Linguists

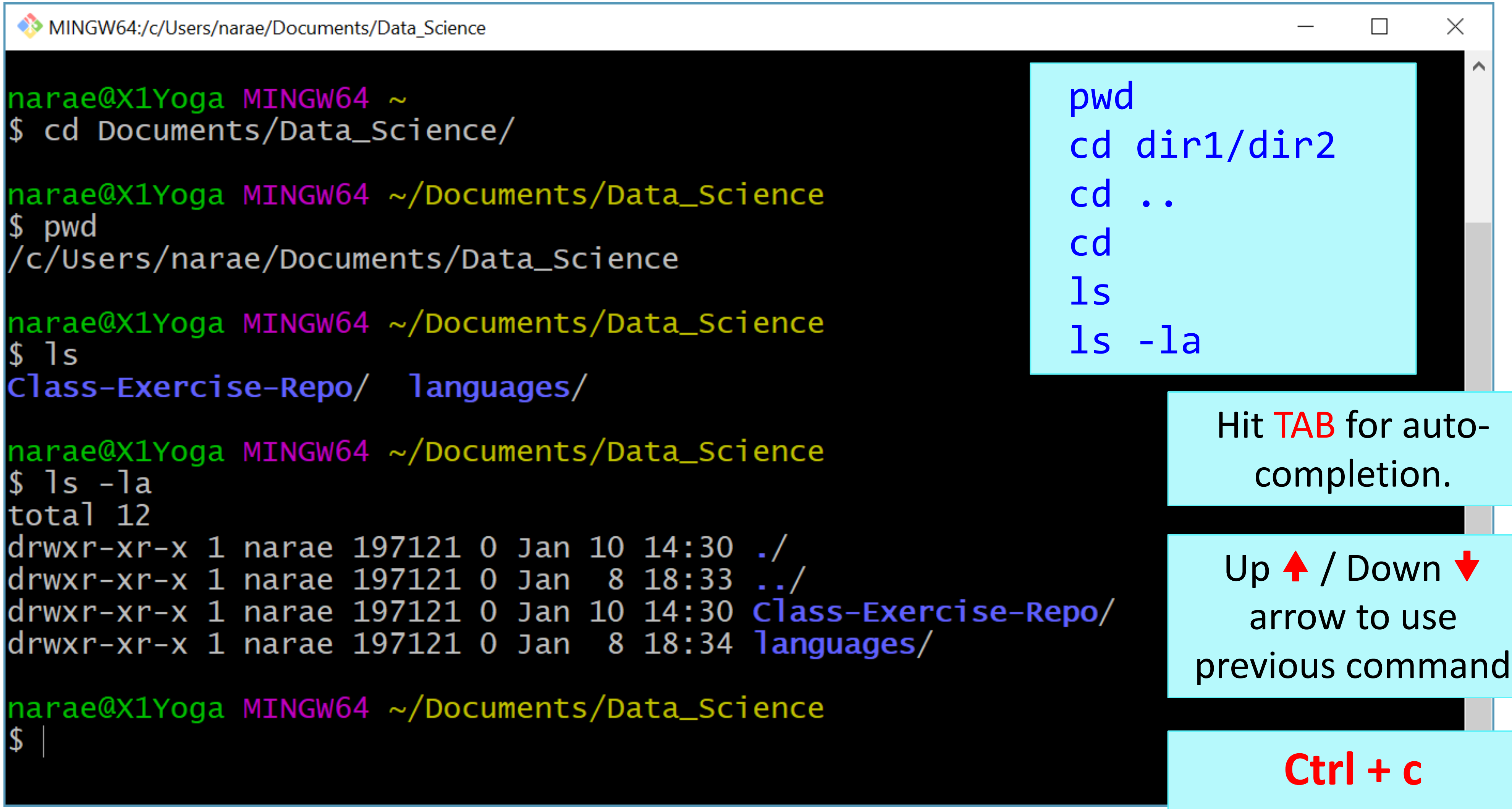
Na-Rae Han

Objectives

- To-do 2: What linguistic data did you find?
- GitHub: completing the fork triangle
- Tools:
  - Git and GitHub
  - Jupyter Notebook
  - Using DataCamp tutorials

You should be taking NOTES!

First thing to do every class



```
narae@X1Yoga MINGW64 ~
$ cd Documents/Data_Science/

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ pwd
/c/Users/narae/Documents/Data_Science

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ ls
Class-Exercise-Repo/  languages/

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ ls -la
total 12
drwxr-xr-x 1 narae 197121 0 Jan 10 14:30 ./
drwxr-xr-x 1 narae 197121 0 Jan  8 18:33 ../
drwxr-xr-x 1 narae 197121 0 Jan 10 14:30 Class-Exercise-Repo/
drwxr-xr-x 1 narae 197121 0 Jan  8 18:34 languages/

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$
```

```
pwd
cd dir1/dir2
cd ..
cd
ls
ls -la
```

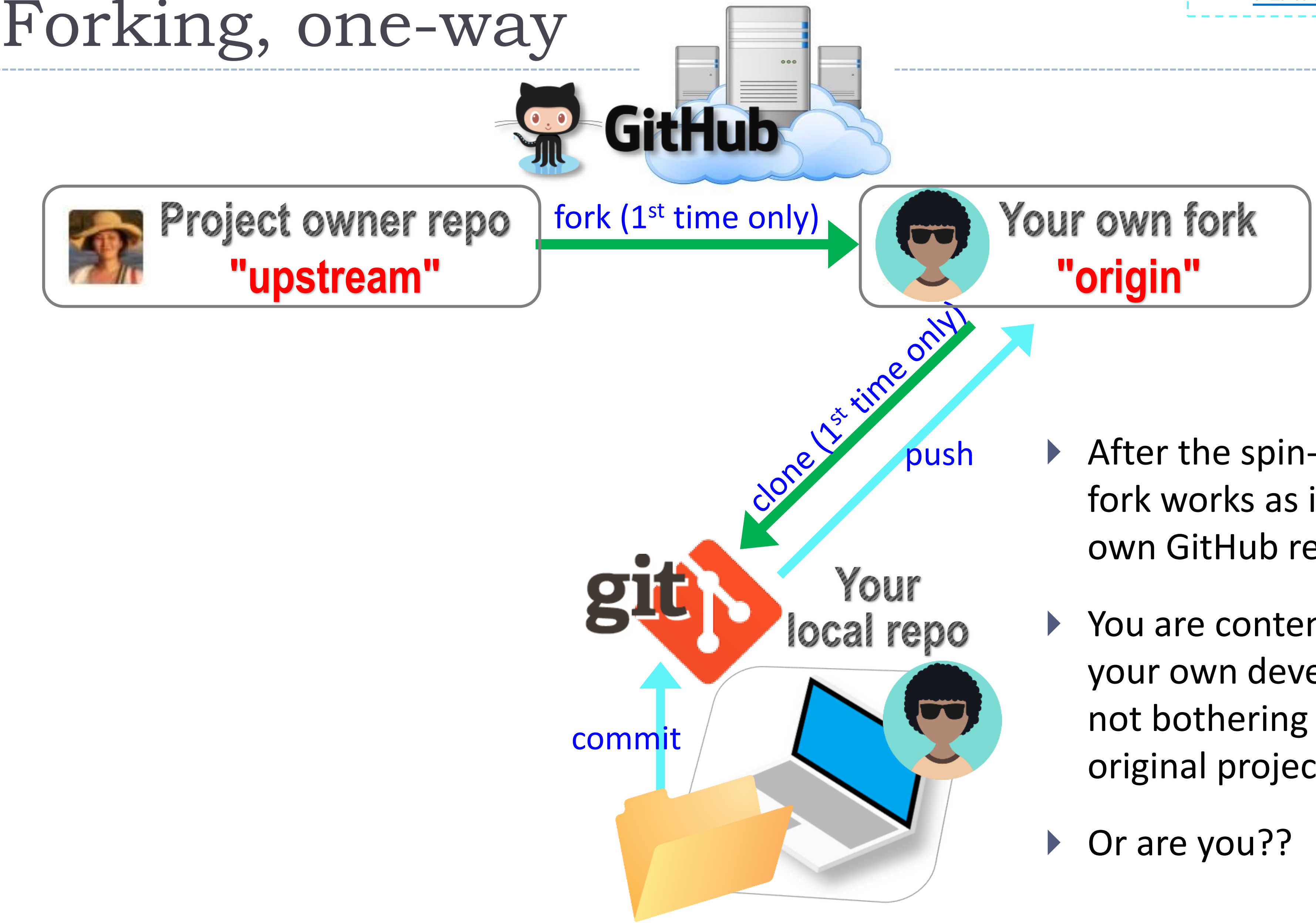
Hit TAB for auto-completion.

Up ↑ / Down ↓ arrow to use previous command

Ctrl + c
to cancel

Forking, one-way

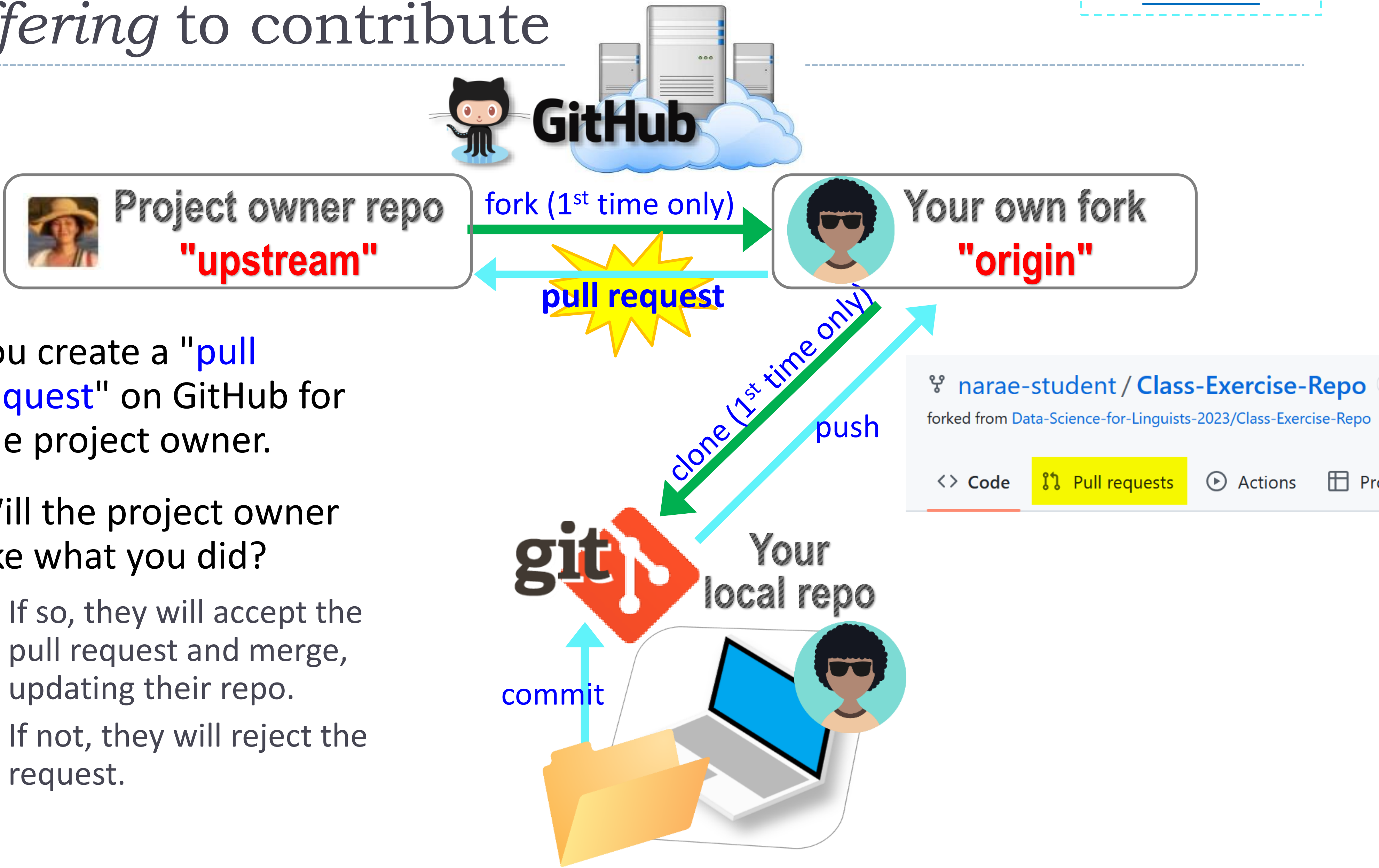
*Avatar icons by FLATICON



- After the spin-off, your fork works as if your own GitHub repo.
- You are content to do your own development, not bothering the original project owner…
- Or are you??

*Avatar icons by FLATICON

Offering to contribute



narae-student / Class-Exercise-Repo

forked from Data-Science-for-Linguists-2023/Class-Exercise-Repo

- You create a "pull request" on GitHub for the project owner.
- Will the project owner like what you did?
  - If so, they will accept the pull request and merge, updating their repo.
  - If not, they will reject the request.

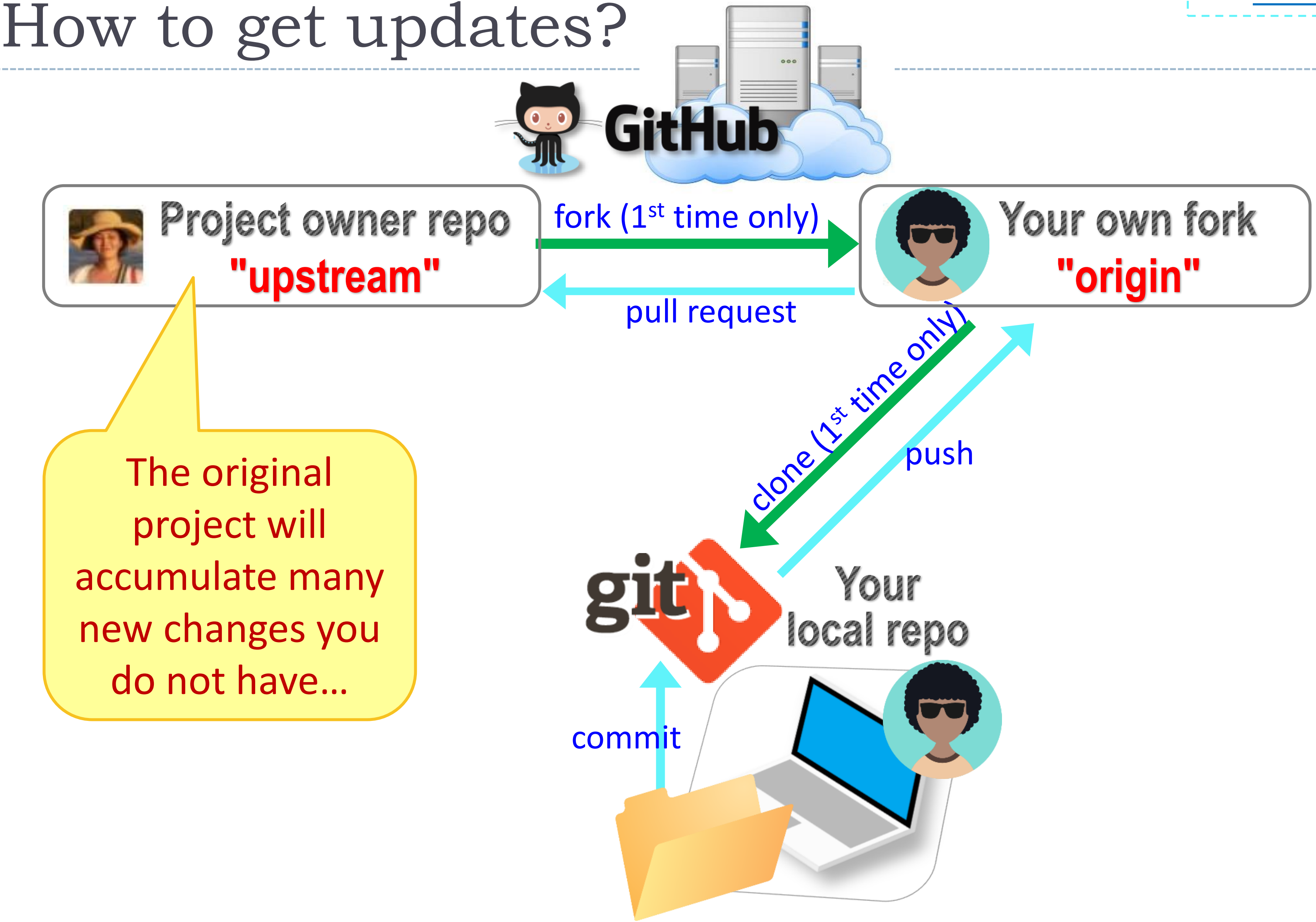
To-do #2: Linguistic Datasets

https://github.com/Data-Science-for-Linguists-2024/Class-Exercise-Repo

- Your To-do #2 submissions
  - Lots of files! I have merged in everyone's contributions.
- What linguistic data sets did you look at?
  - Corpus data? Non-corpus? (What's the distinction?)
  - Non-English data? Speech data? Social media data? Interviews? With linguistic annotation? Format -- Raw? XML? Spreadsheet?
- **Wait! Your own fork does not have everyone's files...**

*Avatar icons by FLATICON

How to get updates?



GitHub

Project owner repo
"upstream"

fork (1st time only)

pull request

Your own fork
"origin"

clone (1st time only)

push

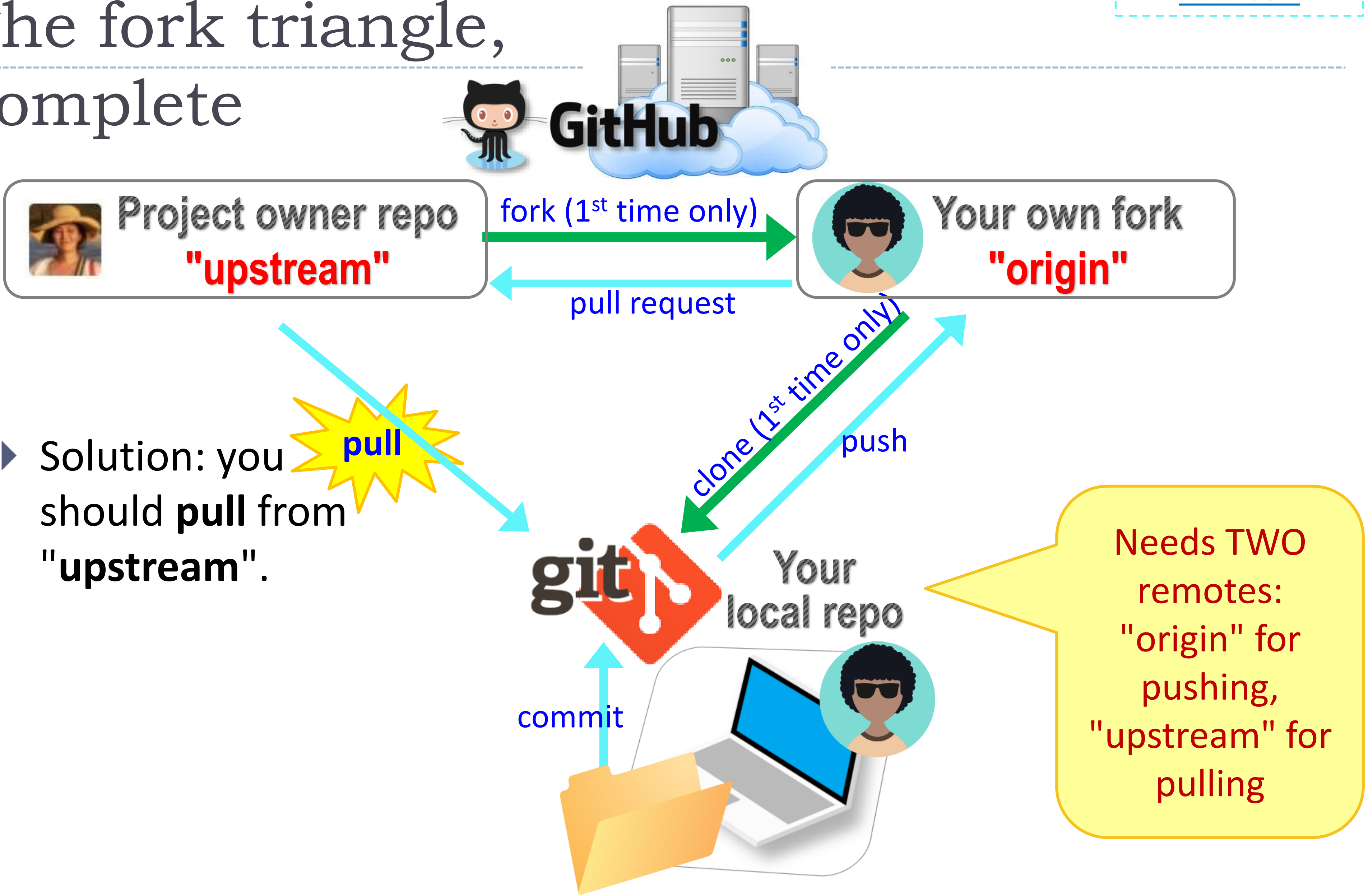
Your local repo

commit

The original project will accumulate many new changes you do not have…

The fork triangle, complete

*Avatar icons by FLATICON

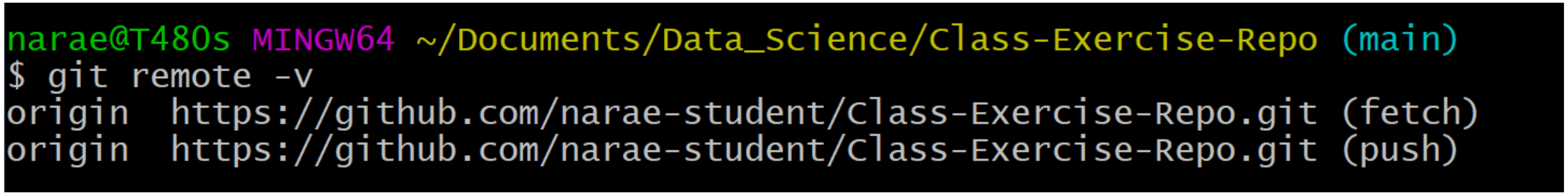


- Solution: you should **pull** from "**upstream**".

Needs TWO remotes: "origin" for pushing, "upstream" for pulling

Keeping your fork up-to-date

- The original repo ("upstream") will have new changes from other users.
  - How to keep your copies (GitHub fork and local repo) up-to-date?
- Cloning already configured your GitHub fork as "origin":



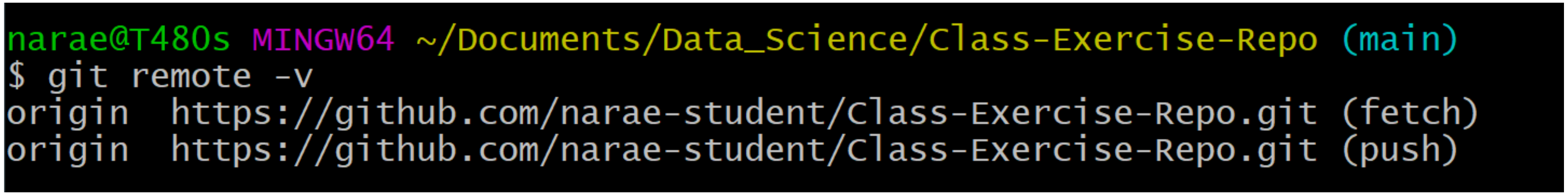
```
narae@T480s MINGW64 ~/Documents/Data_Science/Class-Exercise-Repo (main)
$ git remote -v
origin  https://github.com/narae-student/Class-Exercise-Repo.git (fetch)
origin  https://github.com/narae-student/Class-Exercise-Repo.git (push)
```

- Configure the original repo as another remote: "upstream"
  - `git remote add upstream <GitHub-repo-URL.git>`
- When it's time to sync, pull from upstream:
  - `git pull upstream main`
- Pushing should be done to your GitHub fork ("origin").
  - `git push`

Same as
git push origin main

Keeping your fork up-to-date

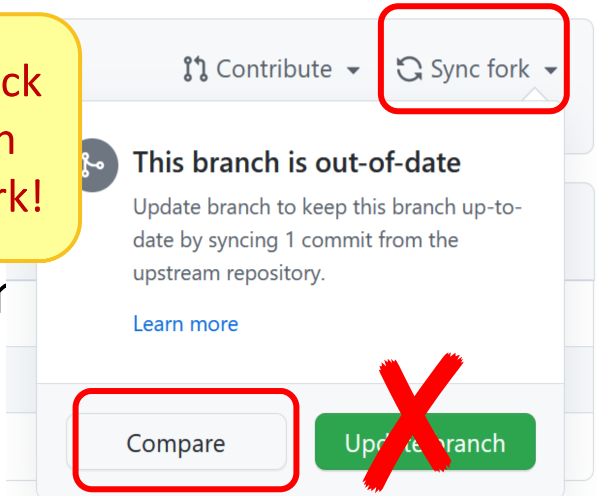
- The original repo ("upstream") will have new changes from other users.
  - How to keep your copies (GitHub fork and local repo) up-to-date?
- Cloning already configured your GitHub fork as "origin":



```
narae@T480s MINGW64 ~/Documents/Data_Science/Class-Exercise-Repo (main)
$ git remote -v
origin  https://github.com/narae-student/Class-Exercise-Repo.git (fetch)
origin  https://github.com/narae-student/Class-Exercise-Repo.git (push)
```

- Configure the original repo as another remote: "upstream"
  - `git remote add upstream <GitHub-repo-URL.git>`
- When it's time to sync, pull from upstr
  - `git pull upstream main`

Before this, check for **conflicts** on your GitHub Fork!



This branch is out-of-date

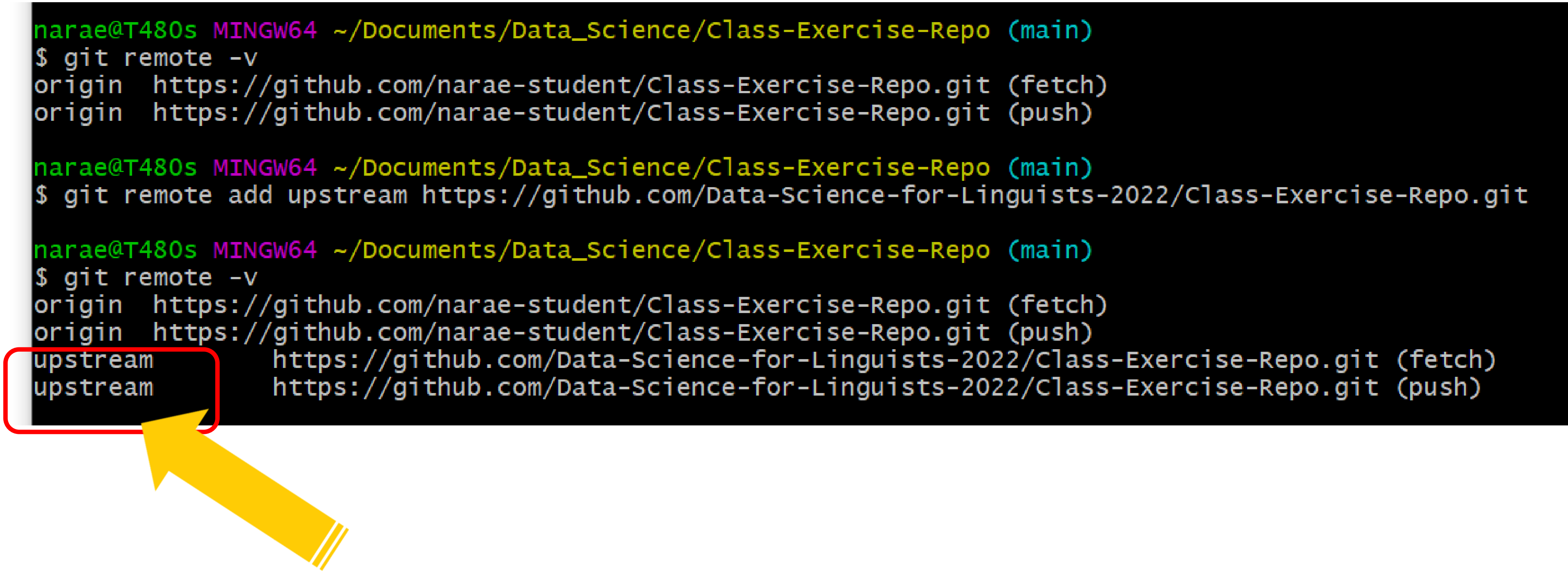
Update branch to keep this branch up-to-date by syncing 1 commit from the upstream repository.

Learn more

- Pushing should be done to your GitHub fork ("origi
  - `git push`

Same as
git push origin main

Two remotes: "origin", "upstream"



```
narae@T480s MINGW64 ~/Documents/Data_Science/Class-Exercise-Repo (main)
$ git remote -v
origin  https://github.com/narae-student/Class-Exercise-Repo.git (fetch)
origin  https://github.com/narae-student/Class-Exercise-Repo.git (push)

narae@T480s MINGW64 ~/Documents/Data_Science/Class-Exercise-Repo (main)
$ git remote add upstream https://github.com/Data-Science-for-Linguists-2022/Class-Exercise-Repo.git

narae@T480s MINGW64 ~/Documents/Data_Science/Class-Exercise-Repo (main)
$ git remote -v
origin  https://github.com/narae-student/Class-Exercise-Repo.git (fetch)
origin  https://github.com/narae-student/Class-Exercise-Repo.git (push)
upstream        https://github.com/Data-Science-for-Linguists-2022/Class-Exercise-Repo.git (fetch)
upstream        https://github.com/Data-Science-for-Linguists-2022/Class-Exercise-Repo.git (push)
```

Git and GitHub are complicated.

- They are powerful tools.
- There are a lot of abstract, high-level concepts involved.
- Concepts do not make sense before you get hands-on.
- You cannot get hands-on without the right context.
- Successful collaboration hinges on everyone doing their part.

← We will learn slowly, learning various pieces as we go.

← You need to be patient, careful and methodical. Make sure you don't rush, and follow instructions.

Git and GitHub are complicated.

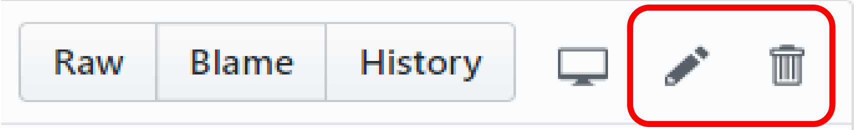
- We will follow some **ground rules**.

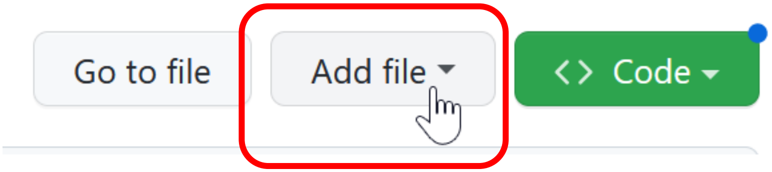
- Don't accidentally commit a file! Be mindful of what you add.
  Do NOT use:
  - `git add .`
  - `git add *`

- For now, do not **delete** or **re-name** any previously committed file.
  - If you must: use `git rm` (to delete) or `git mv` (to move file or rename)

Git and GitHub are complicated.

- We will follow some **ground rules**.
- DO NOT EDIT A REPOSITORY'S CONTENT THROUGH GITHUB.

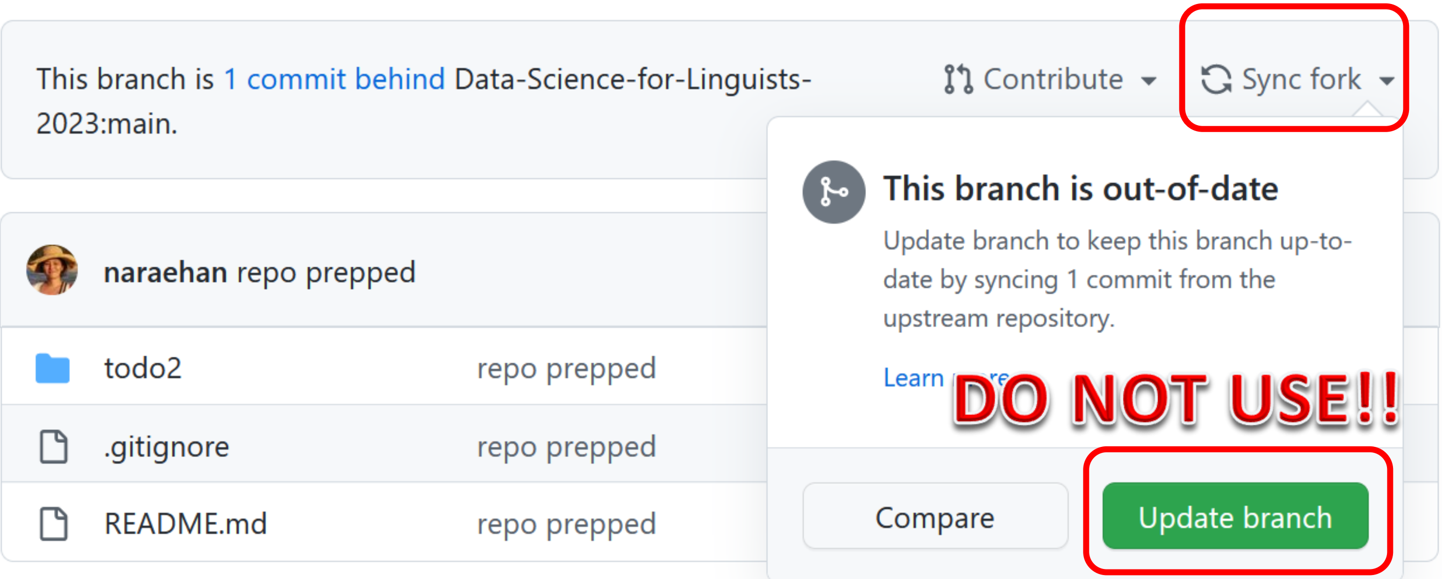




DO NOT USE!!

DO NOT USE!!

- Do not sync your fork through GITHUB. Instead, use command line to pull directly from upstream.



This branch is 1 commit behind Data-Science-for-Linguists-2023:main.

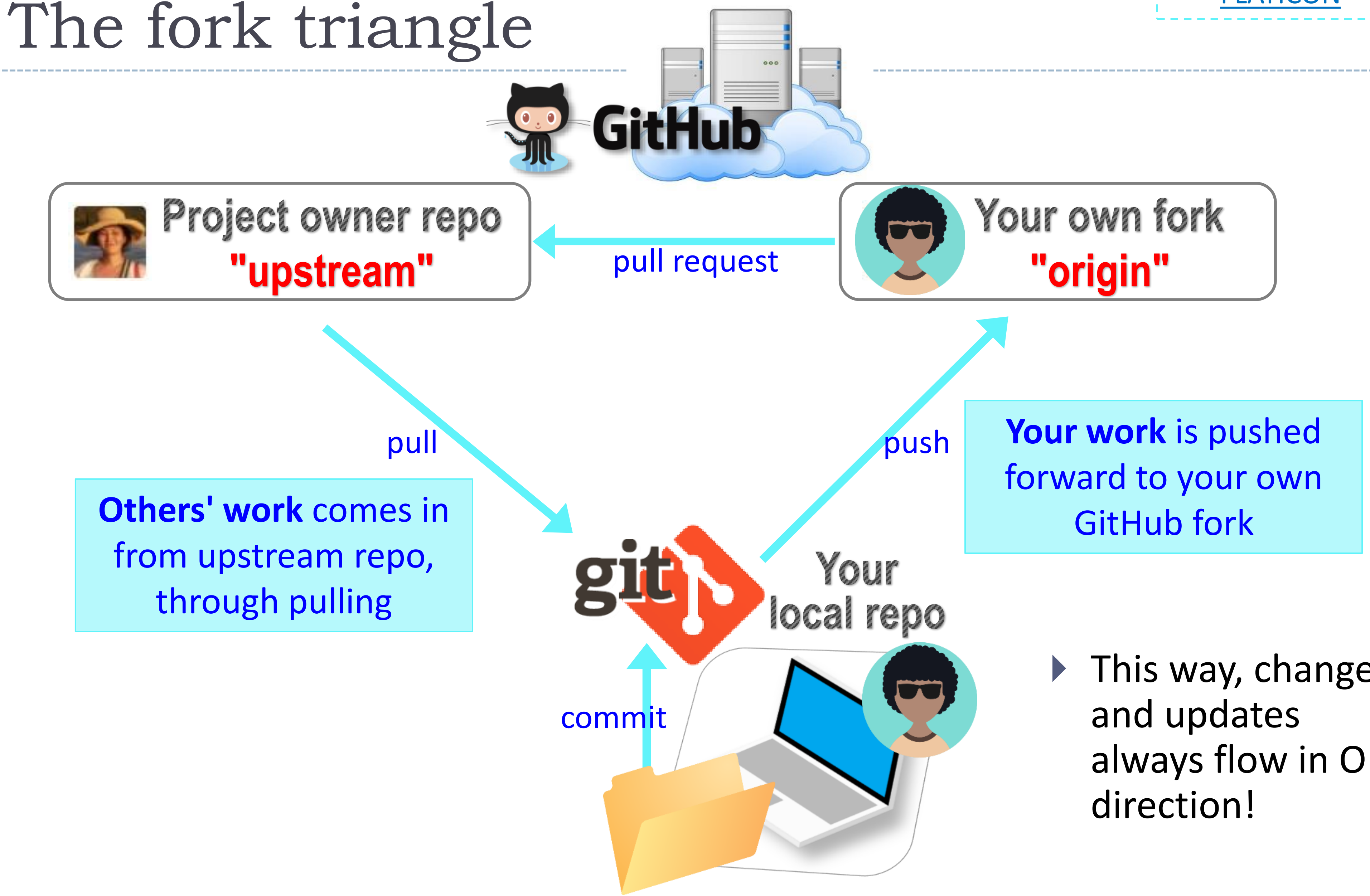
This branch is out-of-date

Update branch to keep this branch up-to-date by syncing 1 commit from the upstream repository.

DO NOT USE!!

*Avatar icons by FLATICON

The fork triangle

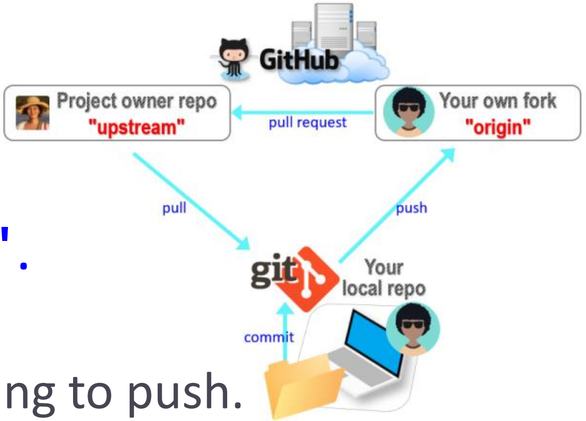


Others' work comes in from upstream repo, through pulling

Your work is pushed forward to your own GitHub fork

- This way, changes and updates always flow in ONE direction!

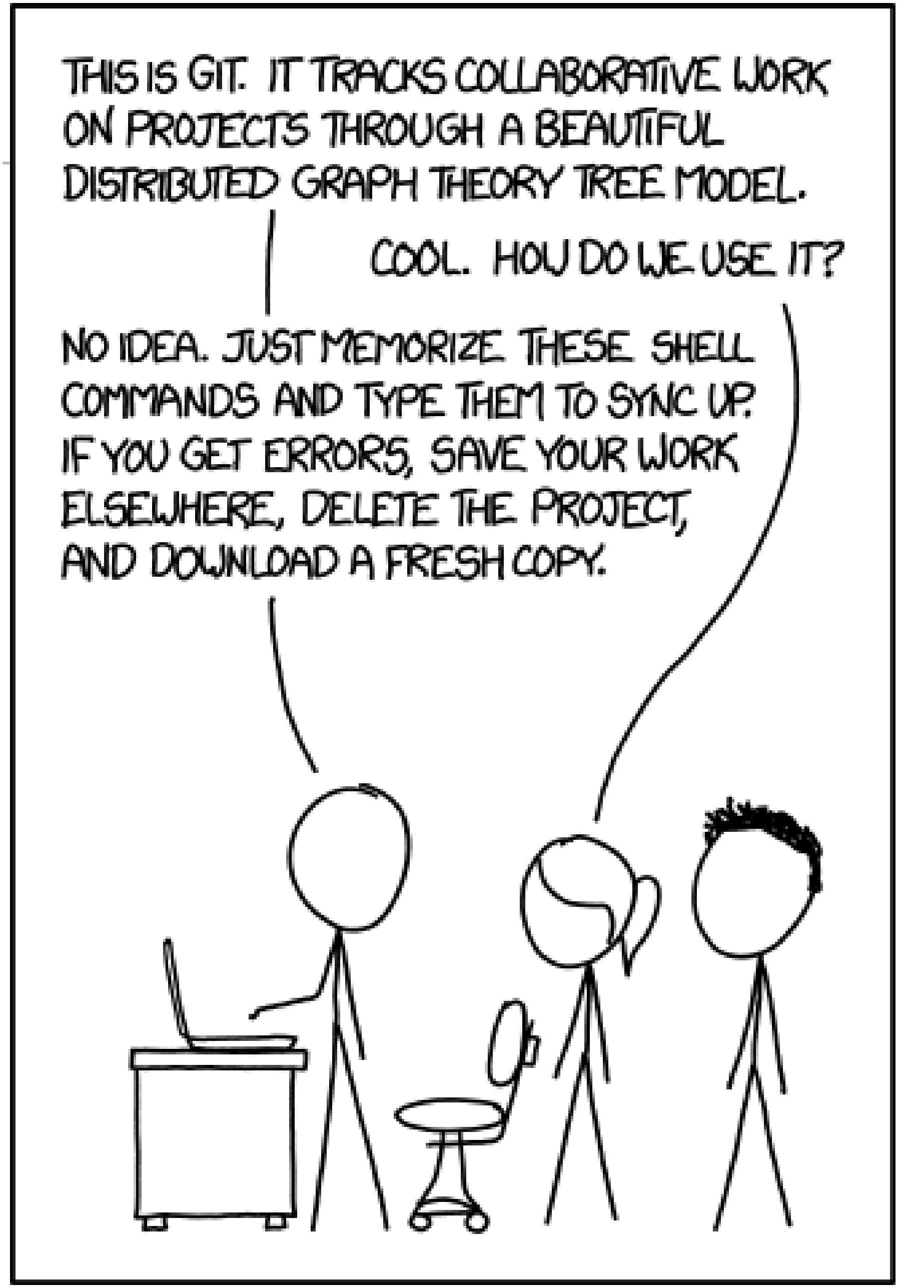
Your workflow



1. **Housekeeping**: Check YOUR WORK via "`git status`".
   - Your local repo is clean: no unsaved/uncommitted work.
   - Your GIHUB fork already has your latest commit: there's nothing to push.
2. **Housekeeping**: Bring in updates from OTHERS.
   - On your **GitHub fork**, check what updates have accumulated in the upstream repo.
   - Through "Sync fork → Compare", make sure those updates don't have conflicts with your fork. Don't press that green "Update Branch" button!
   - Back on **command line**, pull from upstream. Now your local repo is synced with the original repo.
   - Finally, sync your GitHub fork by pushing. The universe is in order now!
3. Work on your homework, to-do, etc.
   - *Now* start your homework. Make some commits along the way.
   - Push to your GitHub fork for one last time.
   - Submission time: Create a **pull request**. Make sure your pull request doesn't have conflicts.

Uh-oh, conflicts!

- Don't panic! Take note of which files are in conflict.
- Chances are you made changes to someone else's file and committed them by accident.
  - Walk back the changes in your fork. That will resolve the conflict.
  - If you're unsure, ask for help!
- If the problem is on the upstream's end (Na-Rae might have let something slip through…), let me know.


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

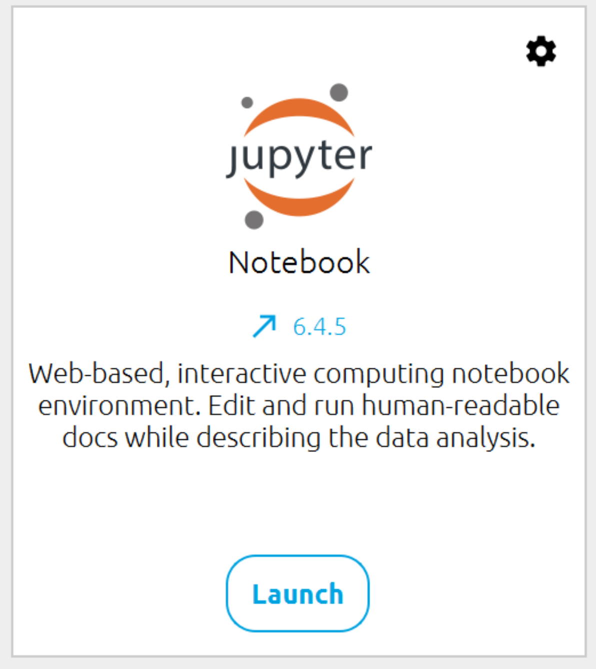
https://xkcd.com/1597

Jupyter Notebook

- Allows you to create and share documents that contain live code cells, output, equations, visualizations and explanatory text.
- Learn how to use it. Your Python code should be in the Jupyter Notebook format:
  - xxxx.ipynb
- You can launch it from the command line.
  - Move into the desired directory, and then execute

    ```
    jupyter notebook &
    ```

    ← '&' is not necessary, but it lets you keep using the terminal
  - If it doesn't work, then edit your system's path variable or just use a shortcut provided by your OS.

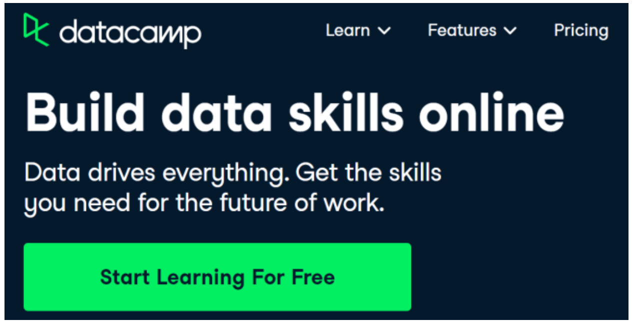


Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Wrapping up

- Homework #1 is out (due Wed): process a linguistic dataset of your choice in Python, using Jupyter Notebook
  - Don't be too ambitious! This HW is about taking stock of what you already know and where to go from there. And also new tools.

- Office hours
  - Na-Rae and Ashley's hours posted on home page.
  - We both have hours on Tue: can help with HW #1.
- Learn:
  - Get started with numpy and pandas.
  - DataCamp has good tutorials.



Build data skills online

Data drives everything. Get the skills you need for the future of work.