LEARN CODING

ale66

github.com/ale66/learn-coding

EXAM PROJECT

TYPE: A REPLICATION STUDY

We often read of data-driven Social science analyses

Let’s replicate such studies with our own coding

and, possibly, use it as a base for further discoveries

TOPIC:

Can you convincingly replicate the findings of

Chan et al., *Four best practices for measuring news sentiment using ‘off-the-shelf’ dictionaries: a large-scale p-hacking experiment*

Comp. Communication Res., 2021

code is available at github.com/chainsawriot/ots

N.B. not all findings need replication: precise instructions will follow

Operations will be on GitHub: sps-unimi-it-c4css-2025-26

EVALUATION METRICS:

- correctness (syntactical & semantical)
- code quality: is it clear? Is it easy to reuse/maintain?
- presentation: graphics etc.
- veracity: can the student explain their code line-by-line?

HOW TO: GITHUB CLASSROOM

GIT: A METHOD

coding is not even half of the story

debugging requires deep understanding of

- the computational problem at hand
- the data
- the tools (Python)

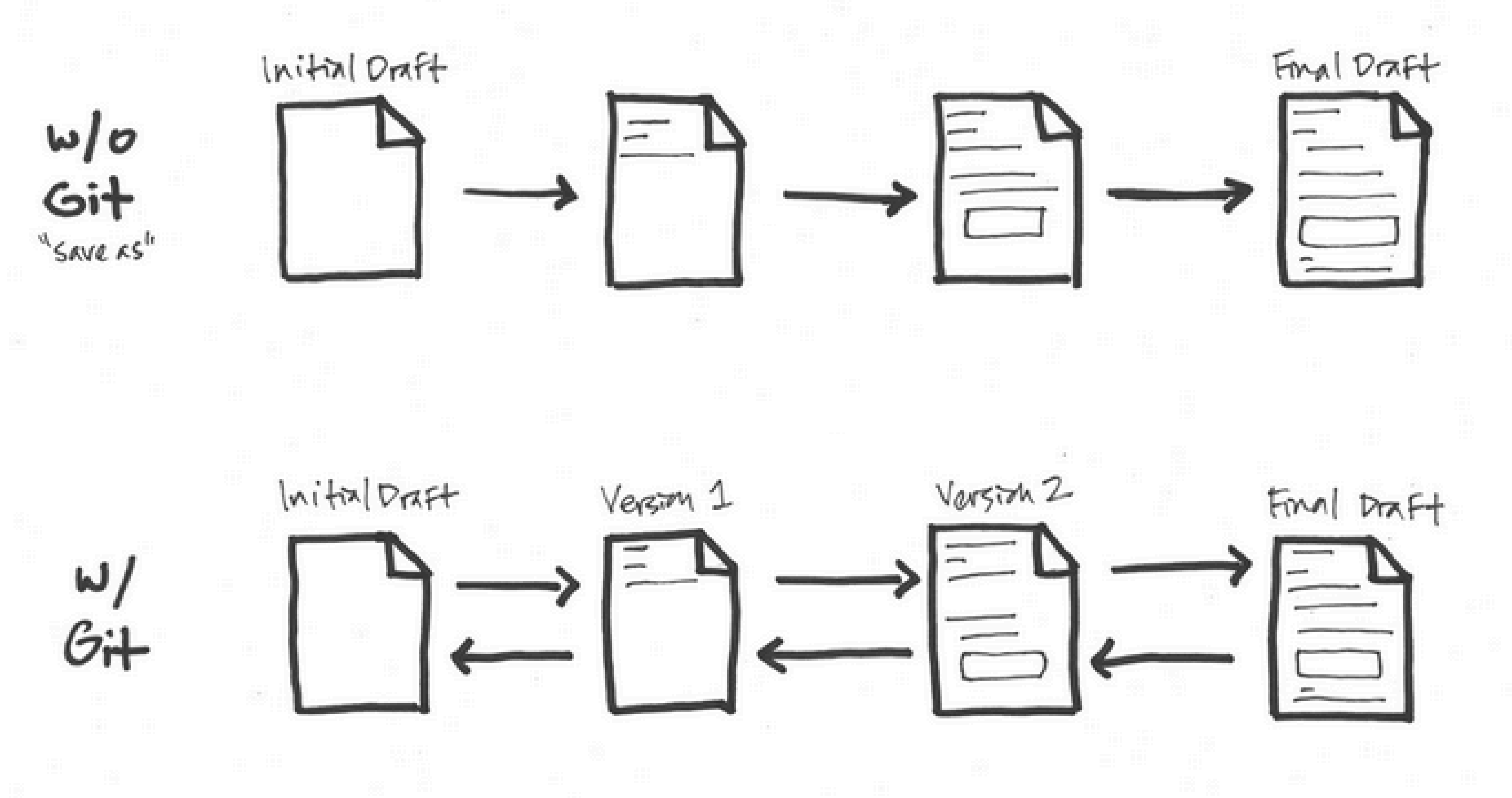
maintaining also takes time

Regression: it used to work but not anymore

Git supports collaboration in software

the codebase is centralised and receives contributions

one can *branch* a codebase to develop and test ideas.



MAIN COMMANDS

```
git create myrepo
```

```
git clone mycompanyrepo
```

More than one copy of the repo can be cloned in different parts of the same computer

```
git brach mycompanyrepo
git pull
```

reads a ‘secret’ `.git` folder inside the top folder of the repo

```
git add newfile.py
```

from now on, also `newfile.py` will be versioned

```
git status
```

what has been changed since the last version?

THE KEY COMMANDS

```
git commit -m "This is what I did: cleaned some code and added newfile.py"
```

cristallises a new version

```
git push
```

update the codebase to my current version

To protect the codebase, destructive updates are treated by creating a branch

```
git merge
```

Proceed with caution…

GITHUB

GITHUB

A cloud platform to support collaborative coding

in principle, all code is public

We can clone, make changes, commit them in our local repository then send a *pull request* to the owner of the repo for our changes to be incorporated.

Knowing Git/GH is a critical skill today

A GH *portfolio* speak for our skills

GITHUB CLASSROOM

GH Classroom helps learning by

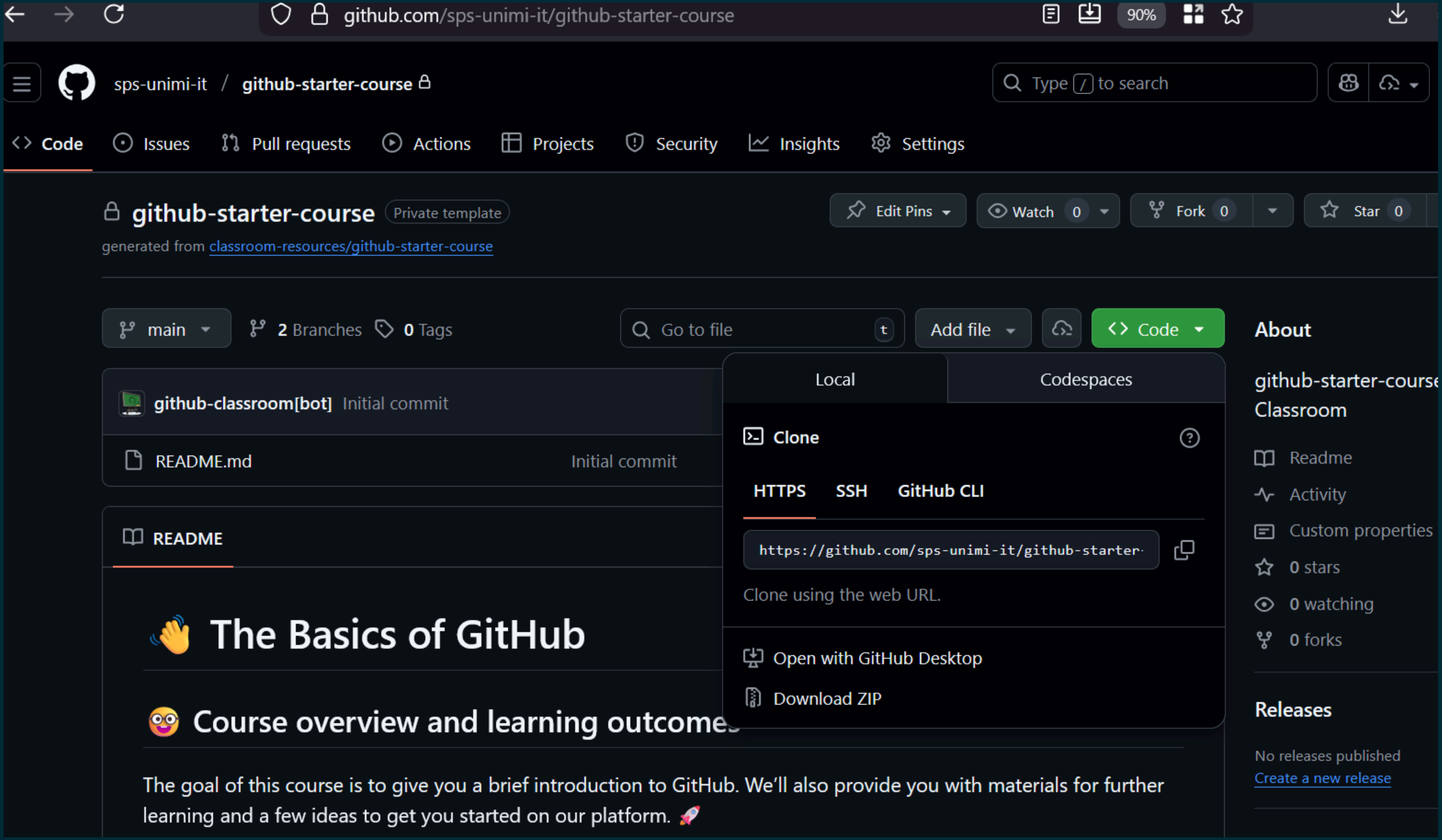
- distributing coding assignments
- managing student repositories automatically
- providing feedback through GitHub
- tracking progress

THE GH CLASSROOM WORKFLOW

1. instructor creates an assignment
2. student receives an invitation link
3. s. accept the assignment → GitHub auto-creates his/her own repo
4. s. clones the repo and work on it
5. s. pushes their changes (the submission)
6. i. reviews work on GitHub

Try the starter course:

```
git clone https://github.com/sps-unimi-it/github-starter-course
```



generated from classroom-resources/github-starter-course

github-classroom[bot] Initial commit

README.md Initial commit

https://github.com/sps-unimi-it/github-starter-

Clone using the web URL.

github-starter-course Classroom

README

👋 The Basics of GitHub

🤓 Course overview and learning outcomes

The goal of this course is to give you a brief introduction to GitHub. We'll also provide you with materials for further learning and a few ideas to get you started on our platform. 🚀

github.com/ale66/learn-coding

FOR REFERENCE: STUDENT INSTRUCTIONS

The following material is from the GitHub Classroom admin pages

ACCEPTING AN ASSIGNMENT

instructor will share a link like:

`https://github.com/sps-unimi-it/c4css-2025-26`

when student clicks, it:

1. authorizes GitHub Classroom (first time only)
2. accepts the assignment
3. creates a personal repo on GH classroom

The repo name is usually: `c4css-2025-26-yourGitHubUsername`

FIRST STEPS

After accepting, you’ll see a repository URL:

```
git clone https://github.com/sps-unimi-it/c4css-2025-26-yourname.git

cd c4css-2025-yourname
```

The repository contains:

- Assignment instructions (usually in README.md)
- Starter code or templates
- Test files (sometimes)

WORKING ON YOUR ASSIGNMENT

You already know how to:

- `git clone` - get the repository, *una tantum*
- `git pull` - get updates from instructors

Now you’ll work locally, editing files in your favorite editor or IDE.

SAVING YOUR WORK: THE GIT WORKFLOW

To submit your work, you need to send your changes back to GitHub.

This involves three new commands:

1. `add` - Stage your changes
2. `commit` - Save a snapshot with a message
3. `push` - Send commits to GitHub

STEP 1: `git add`

selecting files to be included the next save point.

```
# Add a specific file
git add homework.ipynb
```

```
# Add a specific file
git add homework.ipynb

# Add all changed files
git add .

# or
git add --all
```

`git status` shows what’s staged and what’s not.

STEP 2: `git commit`

A commit is like a save point in a game - it records your changes with a descriptive message.

```
git commit -m "Complete question 1 and 2"
```

The message should briefly describe what you changed.

Good messages:

- “Fix bug in calculation function”
- “Add solutions for problems 3-5”

Meh messages:

- “stuff”
- “changes”

STEP 3: `git push`

This uploads your commits from your computer to GitHub.

```
git push
```

That’s it: the new version now visible to the instructors

N.B. submitting = the last push before the deadline

COMPLETE WORKFLOW EXAMPLE

```
# Make changes to your files in your editor
# Then in terminal:

git status                    # See what changed
git add solution.py          # Stage the file
git commit -m "Add solution to problem 1"
git push                     # Upload to GitHub

# Continue working...
git add tests.py readme.md
git commit -m "Add tests and update readme"
git push
```

BEST PRACTICES

Commit Often - Make small, logical commits - Don’t wait until everything is done

Push Regularly - Backup your work to GitHub - Avoid last-minute technical issues

Write Clear Messages - Helps you track your progress - Helps instructors understand your work

COMMON SCENARIOS

Getting instructor updates:

```
git pull
```

Made changes but want to see status:

```
git status
```

Forgot to push before deadline?

- commits are timestamped locally
- push ASAP and notify instructors

TROUBLESHOOTING TIPS

"Your branch is behind…" - Someone else pushed changes - Run `git pull` first, then push

Forgot to commit? - Your changes are local only - `git add` and commit before pushing

Need help? - Check `git status` for hints - Ask your instructor or TA - GitHub Classroom interface shows submission status

VIEWING YOUR SUBMISSION

After pushing, visit your repository on GitHub:

`https://github.com/sps-unimi-it/c4css-2025-26-yourname.git`

- see all your files and chronology of commits
- check the timestamp of your last push
- read instructors’ feedback (in Issues or PR comments)

DEADLINES AND SUBMISSIONS

Key points:

- your latest push before the deadline = your submission
- all commits are timestamped
- you can push updates until the deadline
- for simple tasks there is auto-grading (instant feedback)

QUICK REFERENCE

```
# Get the assignment
git clone <repo-url>

# Check status
git status

# Save and submit workflow
git add <files>
git commit -m "descriptive message"
git push

# Get updates
git pull
```