Introduction to Git and GitHub

Version Control for Research and Data Analysis

Melle Mendikowski

Department of Economics

June 17, 2026

Who Am I?

Melle Mendikowski

PhD Candidate in Computer Science at the University of Hamburg

Dissertation Project: *New Approaches for High-Quality and Privacy-Preserving Tabular Data Synthesis*

Why Am I Here?

- started an interdisciplinary side project on research narratives in AI papers with Johanna
- gave a short Git introduction for our project
- This lecture expands that introductory workshop

Before We Start

Please make sure you have:

- A GitHub account: `https://github.com/signup`
- Git installed:
  - Windows: `https://git-scm.com/download/win`
  - macOS: `https://git-scm.com/download/mac`
  - Linux: `https://git-scm.com/download/linux`

What We Will Do Today

By the end of this session, you will be able to:

- Explain what Git and GitHub are
- Create a local Git repository
- Make commits
- Push your work to GitHub
- Understand the basic collaboration workflow

The Problem

Many research projects eventually look like this:

- `analysis_final.do`
- `analysis_final_v2.do`
- `analysis_final_v2_revised.do`
- `analysis_FINAL_really_final.do`

This creates problems:

- Which version is correct?
- What changed?
- Who changed it?
- Can we recover an older version?

What Is Git?

Git is a version control system.

It records:

- What changed
- When it changed
- Who changed it
- Why it changed

Think of Git as a complete history of your project.

What Is GitHub?

GitHub is an online platform for Git repositories.

- Git tracks changes on your computer
- GitHub stores and shares your repository online
- GitHub supports collaboration, review, and publication

Git is the tool. GitHub is the online platform.

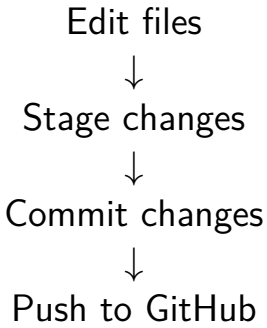
Why This Matters for Economics and Social Science

Git and GitHub are useful for:

- Replication packages
- Data cleaning scripts
- Statistical analysis
- Collaborative research papers
- Policy analysis
- Transparent and reproducible workflows

The Basic Workflow

The basic Git workflow is:



Key Concepts

- **Repository**: a project folder tracked by Git
- **Commit**: a saved snapshot of changes
- **Branch**: an alternative line of work
- **Remote**: an online copy of the repository
- **Push**: upload changes to GitHub
- **Pull**: download changes from GitHub

Check That Git Works

Open Terminal, Git Bash, or another command line and type:

```
git --version
```

You should see something like:

```
git version 2.xx.x
```

Configure Git

Set your name and email address:

```
git config --global user.name "Your Name"
git config --global user.email "the-email-you-use-on-github@example.com"
```

Check your settings:

```
git config --global --list
```

Create a Project Folder

Create a new folder for today's exercise:

```
mkdir github-workshop
cd github-workshop
```

Create a README file:

```
echo "# My First GitHub Repository" > README.md
```

Initialise a Git Repository

Turn the folder into a Git repository:

```
git init
```

Check the status:

```
git status
```

Git now tracks this folder.

Make the First Commit

Stage the file:

```
git add README.md
```

Commit the file:

```
git commit -m "Initial commit"
```

A commit is a saved snapshot of your project.

Create a Repository on GitHub

Now go to GitHub:
`https://github.com`

Click:

New repository

Use this repository name:

`github-workshop`

Important:

- Do not add a README on GitHub
- Keep the repository public or private as you prefer

Connect Local Git to GitHub

GitHub will show you a repository URL. Use the HTTPS URL.
Add it as the remote:

```
git remote add origin https://github.com/YOUR_USERNAME/github-workshop.git
```

Rename the main branch:

```
git branch -M main
```

Push to GitHub:

```
git push -u origin main
```

First push

When you push to GitHub for the first time, GitHub may ask you to sign in. A browser window or Git Credential Manager prompt may appear. Follow the prompts and sign in to GitHub.

Milestone

You now have:

- A Git repository on your computer
- A GitHub repository online
- Your first commit uploaded to GitHub

Refresh your GitHub page and check that your README file appears.

Make a Second Change

Open `README.md` and add one sentence, for example:

```
This repository was created during the Git and GitHub workshop.
```

Then check what changed:

```
git status
```

Commit and Push Again

Stage the changed file:

```
git add README.md
```

Commit the change:

```
git commit -m "Add workshop description"
```

Push the change:

```
git push
```

View Project History

To see your commit history:

```
git log
```

A shorter version:

```
git log --oneline
```

Git keeps a complete record of the project.

Good Commit Messages

Good commit messages are:

- Short
- Clear
- Specific

Good examples:

- `Add data cleaning script`
- `Fix typo in README`
- `Update regression table`

Bad examples:

- `stuff`
- `changes`
- `final version`

Graphical Interfaces

You do not always need to use the command line.

Two common graphical options are:

- **Visual Studio Code**: editor with built-in Git tools
- **GitHub Desktop**: simple graphical app for GitHub workflows

However, the command line has many advantages.

Collaboration on GitHub

GitHub supports collaborative work through:

- Shared repositories
- Issues
- Branches
- Pull requests
- Code review

This is useful for research teams, teaching projects, and replication packages.

What Is a Pull Request?

A pull request is a proposal to add changes to a project.

Typical workflow:

1. Someone makes a change
2. They open a pull request
3. Others review the change
4. The change is accepted and merged

Branches

A branch is a separate line of work.
Create a new branch:

```
git switch -c new-feature
```

Make a change, then commit:

```
git add README.md
git commit -m "Add new feature description"
```

Branches are useful for experiments.

Return to Main Branch

Return to the main branch:

```
git switch main
```

Merge the branch:

```
git merge new-feature
```

Push the result:

```
git push
```

Common Problems

- Git is not installed correctly
- The wrong folder is open
- The repository URL was copied incorrectly
- GitHub asks for authentication
- A file was changed but not staged

Useful command:

```
git status
```

When in doubt, check the status.

Best Practices

- Commit often
- Use clear commit messages
- Keep one project per repository
- Always include a README file
- Push regularly
- Do not upload confidential or sensitive data

What You Can Use GitHub For

GitHub is useful for:

- Research code
- Data documentation
- Teaching materials
- Reproducibility packages
- Collaborative writing support
- Public project portfolios

Takeaways

Today you learned how to:

- Create a Git repository
- Track file changes
- Make commits
- Push work to GitHub
- Understand branches and pull requests
- Use GitHub for research collaboration

Thank you.