Introduction to Git and GitHub

Jay Urbain, PhD

Agenda

I. Introduction
II. Exploring GitHub
III. Using Git with GitHub
IV. Contributing on GitHub
V. Bonus Content

I. Introduction

Why learn version control?

- Version control is useful when you write code, and data scientists write code
- Enables teams to easily collaborate on the same codebase
- Enables you to contribute to open source projects
- Attractive skill for employment

What is Git?

- Version control system that allows you to track files and file changes in a repository (“repo”)
- Primarily used by software developers
- Most widely used version control system
  - Alternatives: Mercurial, Subversion, CVS
- Runs from the command line (usually), gui’s available
- Can be used alone or in a team

What is GitHub?

- A website, not a version control system
- Allows you to put your Git repos online
  - Largest code host in the world
  - Alternative: Bitbucket
- Benefits of GitHub:
  - Backup of files
  - Visual interface for navigating repos
  - Makes repo collaboration easy
- “GitHub like a Dropbox for Git”
- *Note: Git does not require GitHub*

Git can be challenging to learn

- Designed (by programmers) for power and flexibility over simplicity
- Hard to know if what you did was right
- Hard to explore since most actions are “permanent” (in a sense) and can have serious consequences
- We’ll focus on the most important 10% of Git

II. Exploring GitHub

GitHub setup

- Create an account at github.com
- There's nothing to install
  - "GitHub for Windows" & "GitHub for Mac" are GUI clients (alternatives to command line)

Navigating a GitHub repo (1 of 2)

- Example repo: https://github.com/jayurbain/DataScienceIntro
- Account name, repo name, description
- Folder structure
- Viewing files:
  - Rendered view (with syntax highlighting)
  - Raw view
- README.md:
  - Describes a repo
  - Automatically displayed
  - Written in Markdown

Navigating a GitHub repo (2 of 2)

- Commits:
  - One or more changes to one or more files
  - Revision highlighting
  - Commit comments are required
  - Most recent commit comment shown by filename
- Profile page

Creating a repo on GitHub

- Click “Create New” (plus sign):
  - Define name, description, public or private
  - Initialize with README (if you’re going to clone)
- Notes:
  - Nothing has happened to your local computer
  - This was done on GitHub, but GitHub used Git to add the README.md file

Basic Markdown

- Easy-to-read, easy-to-write markup language
- Usually (always?) rendered as HTML
- Many implementations (aka “flavors”)
- Let’s edit README.md using GitHub!
- Common syntax:
  - ## Header size 2
  - *italics* and **bold**
  - [link to GitHub](https://github.com)
  - * bullet
  - `inline code` and ```code blocks```
- Valid HTML can also be used within Markdown

III. Using Git with GitHub

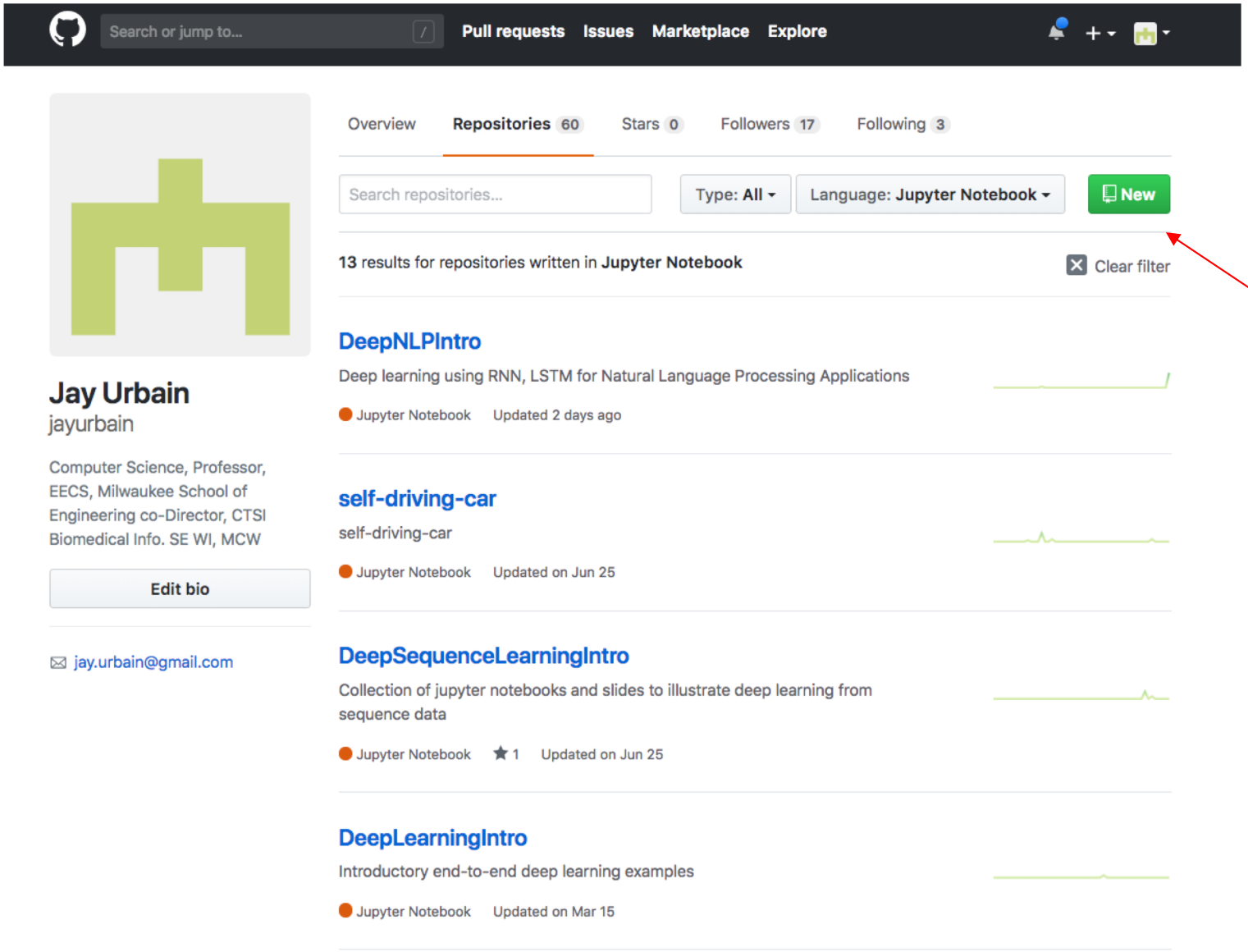
Git installation and setup

- Installation: tiny.cc/installgit
- Open Git Bash or Power Shell (Windows) or Terminal (Mac/Linux):
  - git config --global user.name “YOUR FULL NAME”
  - git config --global user.email “YOUR EMAIL”
- Use the same email address you used with your GitHub account

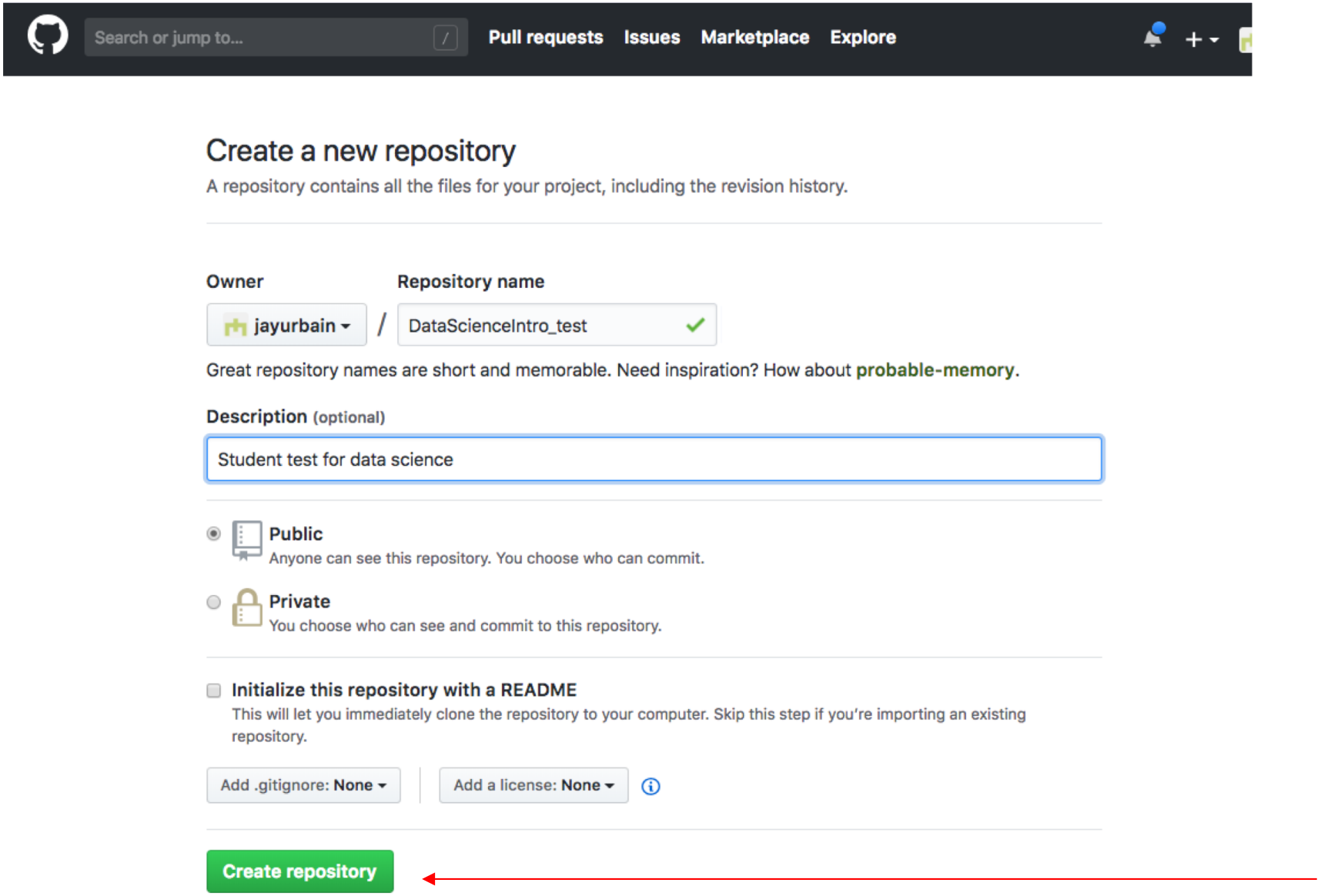
Preview of what you're about to do

- Copy your new GitHub repo to your computer
- Make some file changes locally
- Save those changes locally ("commit" them)
- Update your GitHub repo with those changes

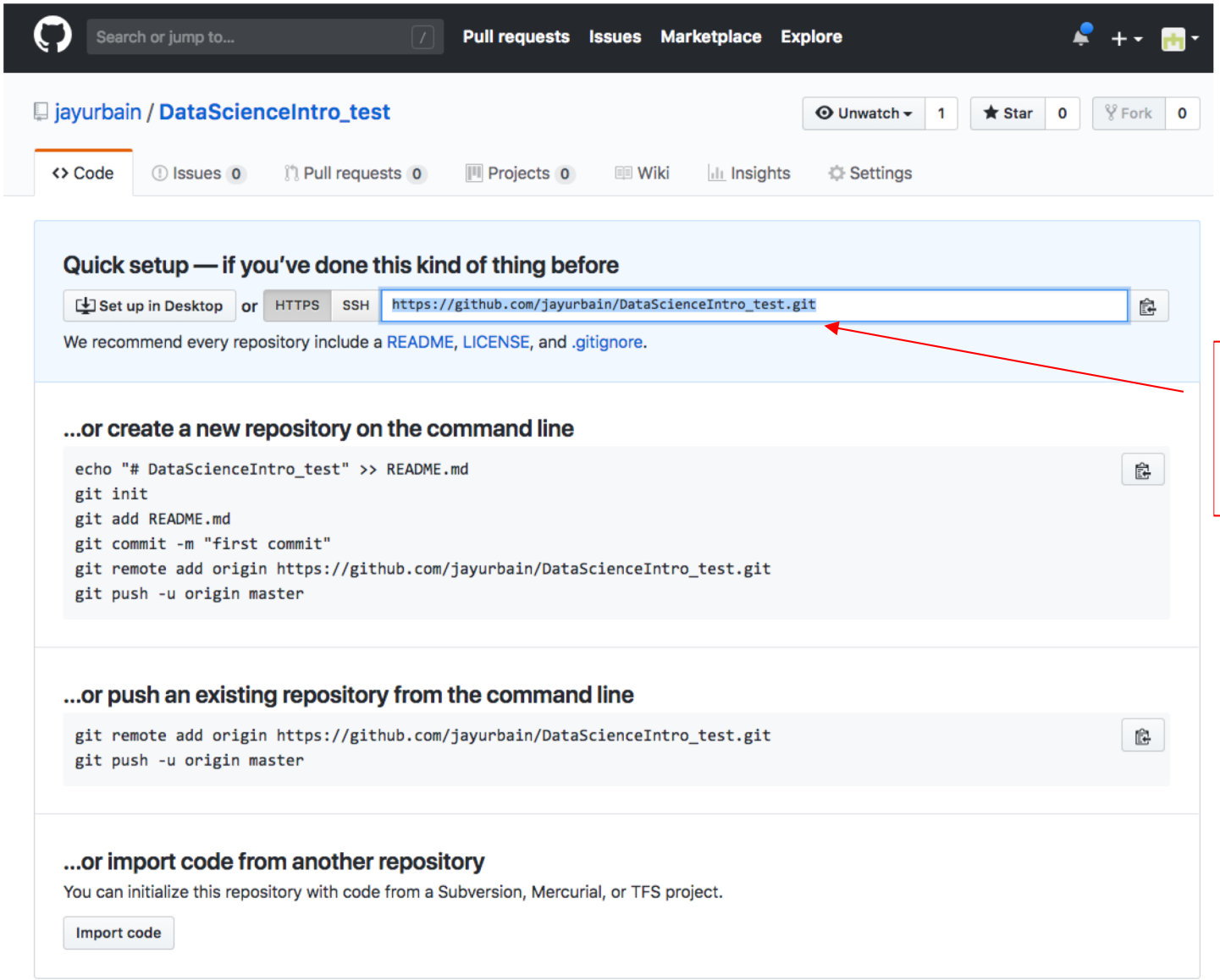
Create repo on github



Create repo on github



Create repo on github



Copy Repository url

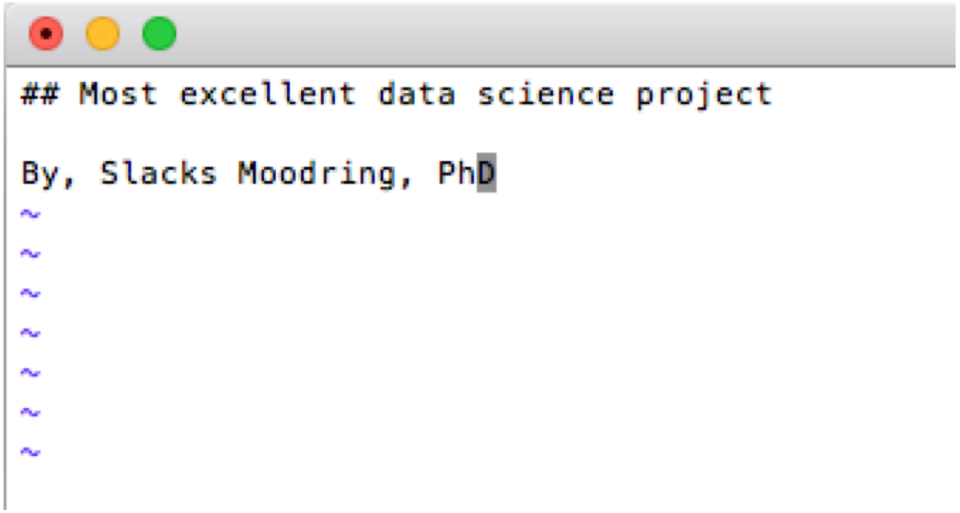
Cloning a GitHub repo

- Cloning = copying to your local computer
  - Like copying your Dropbox files to a new machine
- First, change your working directory to where you want the repo you created to be stored: cd
- Then, clone the repo: git clone <URL>
  - Get HTTPS or SSH URL from GitHub (ends in .git)
  - Clones to a subdirectory of the working directory
  - No visual feedback when you type your password
- Navigate to the repo (cd) then list the files (ls)

```
jays-mbp-2:DataScienceIntro jayurbain$ git clone https://github.com/jayurbain/DataScienceIntro_test.git
Cloning into 'DataScienceIntro_test'...
warning: You appear to have cloned an empty repository.
jays-mbp-2:DataScienceIntro jayurbain$
```

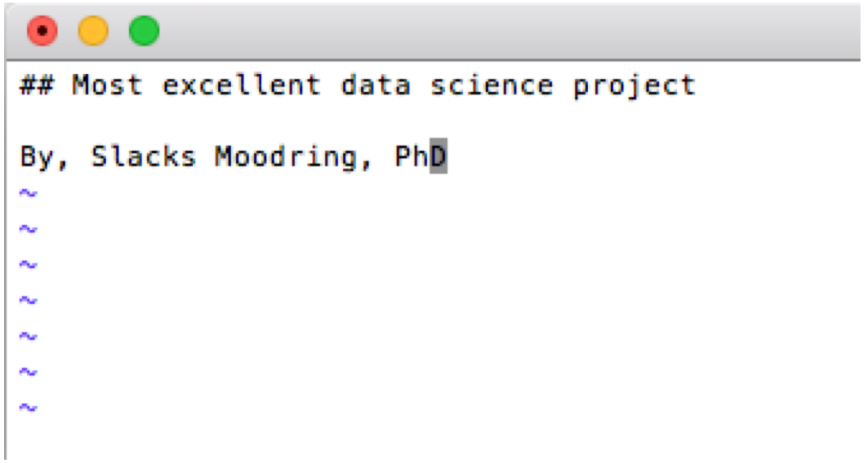
Making changes

- Change directories into your cloned repository.
- Create and add something to README.md



Making changes, checking your status

- Making changes:
  - Modify README.md in any text editor
  - Create a new file: touch <filename>



Making changes, checking your status

- Check your status:
  - git status
- File statuses (possibly color-coded):
  - Untracked (red)
  - Tracked and modified (red)
  - Staged for committing (green)
  - Committed

```
jays-mbp-2:DataScienceIntro_test jayurbain$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        README.md

nothing added to commit but untracked files present (use "git add" to track)
```

Committing changes

- Stage changes for committing:
  - Add a single file: git add <filename>
  - Add all “red” files: git add . # stages all but not deleted
  - Add all “red” files: git add –A # stages all

- Check your status:
  - Red files have turned green
- Commit changes:
  - git commit -m “message about commit”
- Check your status again!
- Check the log: git log

Committing changes

- Stage changes for committing:
  - Add a single file: git add <filename>
  - Add all “red” files: git add . # stages all but not deleted
  - Add all “red” files: git add –A # stages all

- Check your status:
  - Red files have turned green

```
jays-mbp-2:DataScienceIntro_test jayurbain$ git add README.md
jays-mbp-2:DataScienceIntro_test jayurbain$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   README.md
```

Committing changes

- Commit changes:
  - git commit -m “message about commit”
- Check your status again!
- Check the log: git log

```
jays-mbp-2:DataScienceIntro_test jayurbain$ git commit README.md -m "initial commit"
[master (root-commit) 96e0cf4] initial commit
 1 file changed, 3 insertions(+)
 create mode 100644 README.md
jays-mbp-2:DataScienceIntro_test jayurbain$ git status
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean
jays-mbp-2:DataScienceIntro_test jayurbain$ git log
commit 96e0cf4d2db42ae2fbf3eea259e785b46cbb6cc8 (HEAD -> master)
Author: jayurbain <jay.urbain@gmail.com>
Date:   Thu Aug 2 09:23:32 2018 -0500

    initial commit
```

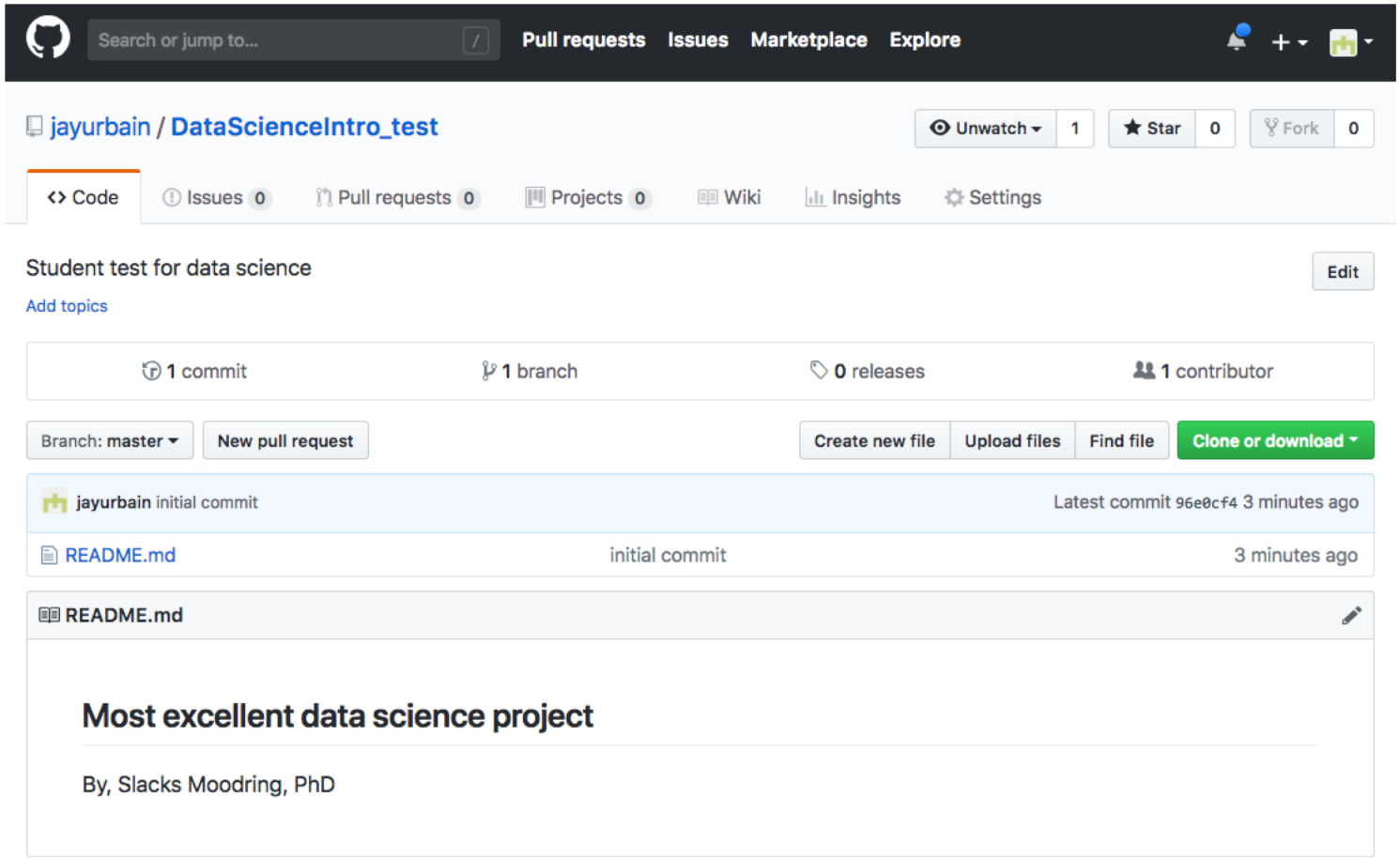
Pushing to GitHub

- Everything you've done to your cloned repo (so far) has been local
- You've been working in the "master" branch
- Push committed changes to GitHub:
  - Like syncing local file changes to Dropbox
  - git push <remote> <branch>
  - Often: git push origin master
- Refresh your GitHub repo to check!

```
jays-mbp-2:DataScienceIntro_test jayurbain$ git push origin master
Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 275 bytes | 275.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/jayurbain/DataScienceIntro_test.git
 * [new branch]      master -> master
jays-mbp-2:DataScienceIntro_test jayurbain$
```

Pushing to GitHub

- Refresh repository on github



Quick recap of what you've done

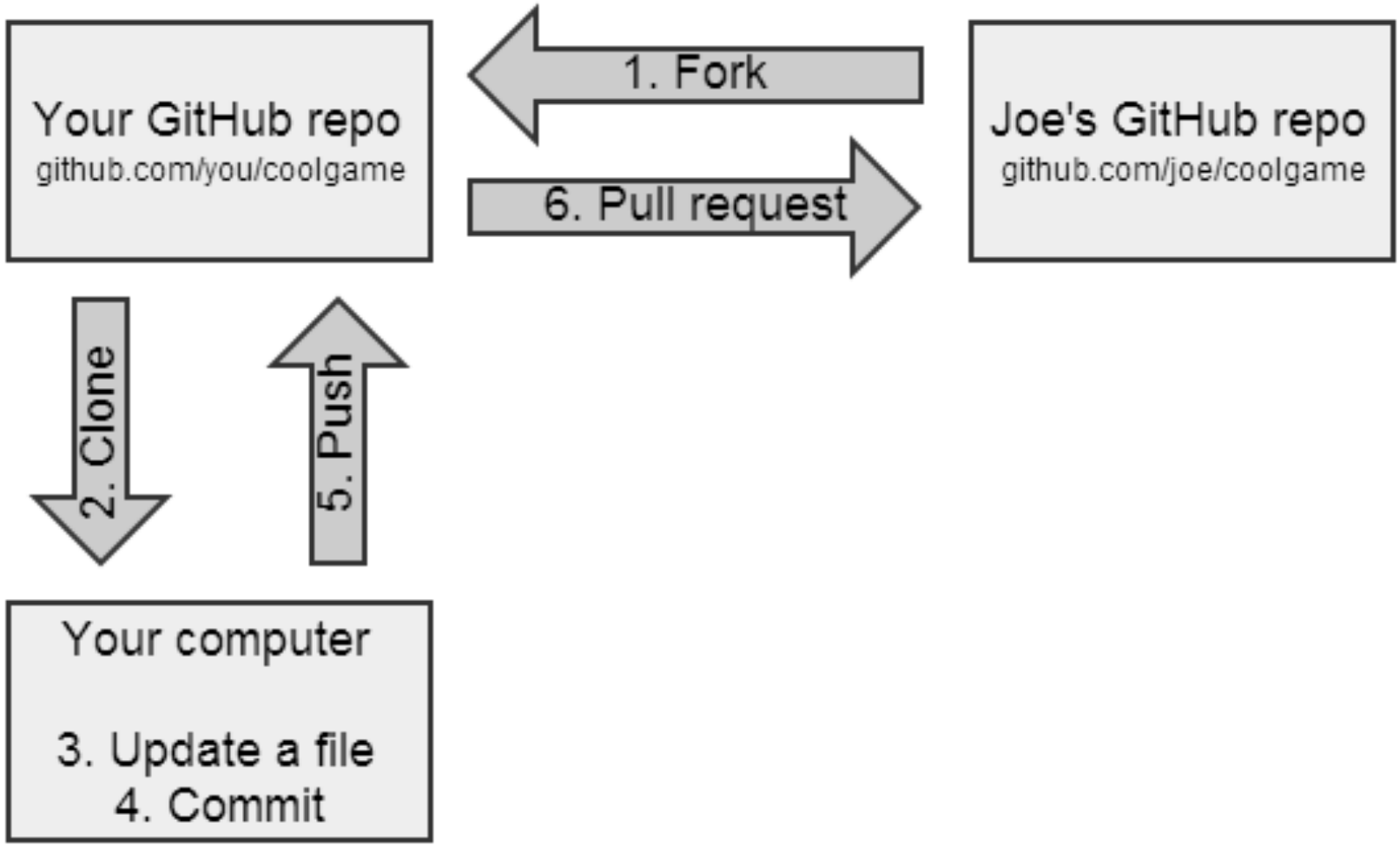
- Created a repo on GitHub
- Cloned repo to your local computer (git clone)
  - Automatically sets up your "origin" remote
- Made file changes
- Staged changes for committing (git add)
- Committed changes (git commit)
- Pushed changes to GitHub (git push)
- Inspected along the way (git remote, git status, git log)

Note: Checking your remotes

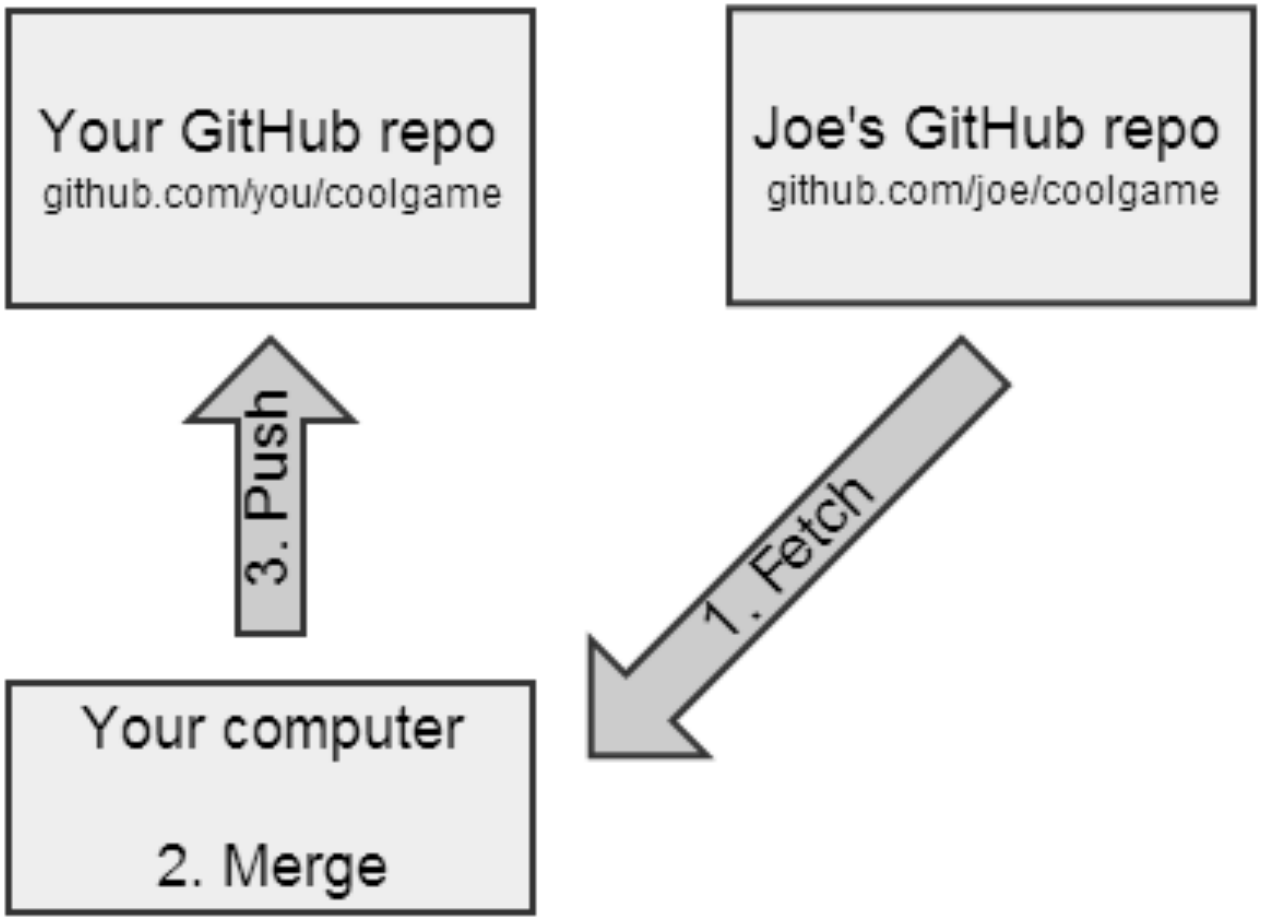
- A “remote alias” is a reference to a repo not on your local computer
  - Like a connection to your Dropbox account
- View remotes: git remote -v
- “origin” remote was set up by “git clone”
- Note: Remotes are repo-specific

```
jays-mbp-2:DataScienceIntro_test jayurbain$ git remote -v
origin  https://github.com/jayurbain/DataScienceIntro_test.git (fetch)
origin  https://github.com/jayurbain/DataScienceIntro_test.git (push)
jays-mbp-2:DataScienceIntro_test jayurbain$
```

GitHub flow for contributing



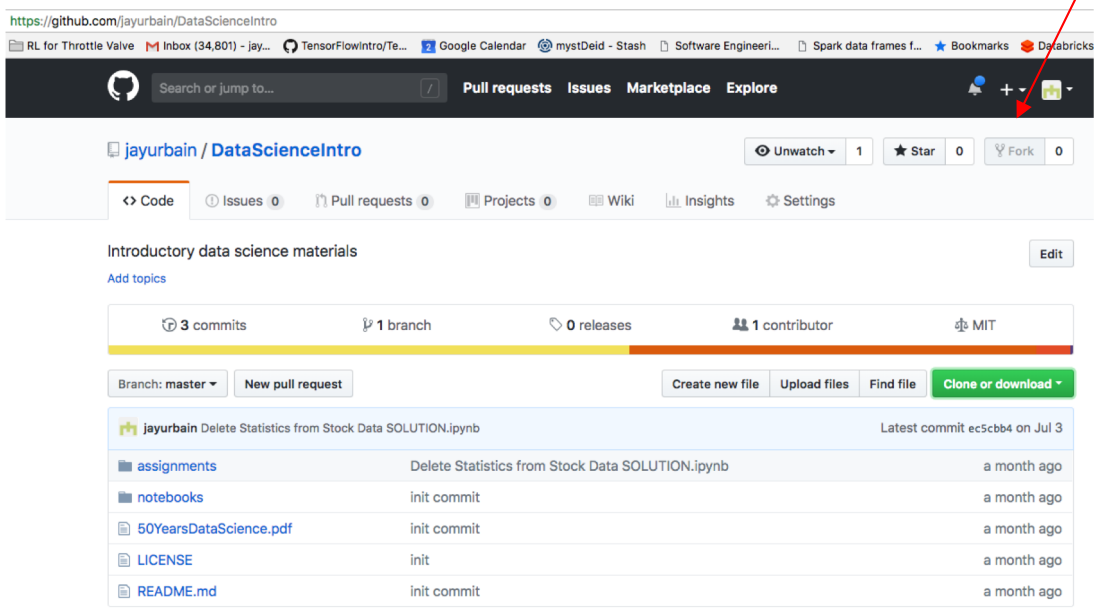
GitHub flow for syncing a fork



Forking a repo on GitHub

- What is forking?
  - Copy a repo to your account (including history)
  - Does not stay in sync with the “upstream”
- Why fork?
  - You want to make modifications
  - You want to contribute to the upstream

Fork repository



Forking a repo on GitHub

- Clone **your** fork: git clone <your URL>

$ git clone https://github.com/slacksmoodring/datascienceintro_test

  - Note: Don't clone inside your other local repo

Recipe for submitting homework

First:
git clone https://github.com/jayurbain/datascienceintro

For each homework:

1. cd datascienceintro
2. git pull # for each home work, make sure your local repository is up to date
3. Work on your homework files
4. git add . or git add <filename> # only if you need to add new files
5. git status
6. git commit -m “message”
7. Submit files, typically *.ipynb + *.html, feedback, and other items to Blackboard!

Note: If you would like to suggests about the course repository:

- Make comments in the issues section of: https://github.com/jayurbain/datascienceintro

Two ways to initialize Git

- Initialize on GitHub:
  - Create a repo on GitHub (with README)
  - Clone to your local machine
- Initialize locally:
  - Initialize Git in existing local directory: git init
  - Create a repo on GitHub (without README)
  - Add remote: git remote add origin <URL>

Deleting or moving a repo

- Deleting a GitHub repo:
  - Settings, then Delete
- Deleting a local repo:
  - Just delete the folder!
- Moving a local repo:
  - Just move the folder!

Excluding files from a repo

- Create a ".gitignore" file in your repo: touch .gitignore
- Specify exclusions, one per line:
  - Single files: pip-log.txt
  - All files with a matching extension: *.pyc
  - Directories: env/
- Templates: github.com/github/gitignore

Gists: lightweight repos

- You have access to Gist: gist.github.com
- Add one or more files
- Supports cloning, forking, commenting, committing
- Can be public or secret (not private)
- Useful for snippets, embedding, iPython nbviewer, etc.

Useful to learn next

- Working with branches
- Rolling back changes
- Resolving merge conflicts
- Fixing LF/CRLF issues