STAT 215A Fall 2018

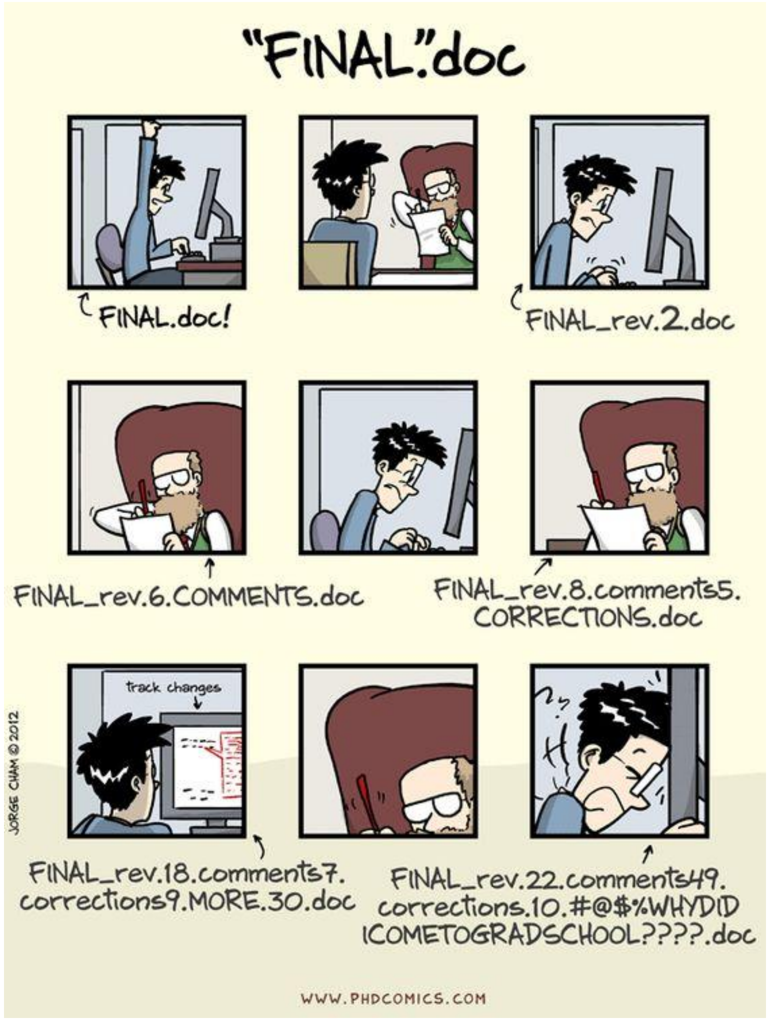
Week 1 - Zoe Vernon

Thanks to Rebecca Barter for sharing her slides

Goals for the day

1. Discuss Git and Github
    a. Understand the basics of Git and Github
    b. Set up GItHub repository for everyone (with me as a collaborator)
2. R and the “tidyverse”
    a. We will use R Sweave or R Markdown to create reports
    b. Give a quick introduction to tidyverse if time

Git and GitHub


"FINAL".doc
FINAL.doc!
FINAL_rev.2.doc
FINAL_rev.6.COMMENTS.doc
FINAL_rev.8.comments5.CORRECTIONS.doc
track changes
FINAL_rev.18.comments7.corrections9.MORE.30.doc
FINAL_rev.22.comments49.corrections.10.#@$%WHYDID ICOMETOGRADSCHOOL????.doc
JORGE CHAM © 2012
WWW.PHDCOMICS.COM

What is Git?

- ❑ Version control system
- ❑ Stores data as a series of snapshots
- ❑ If files have not changed it will simply access the file from a previous commit instead of saving it again
- ❑ Allows access to all the committed steps along the way of your project

Git vs. GitHub

Local Git repository

- ❑ Local version of the folder
- ❑ History stored in a .git file
- ❑ Only you can see the changes that are made in the local version

Remote GitHub repository

- ❑ Remote version of the folder
- ❑ Everyone can see these changes (if repository is public)

Communicating between the two

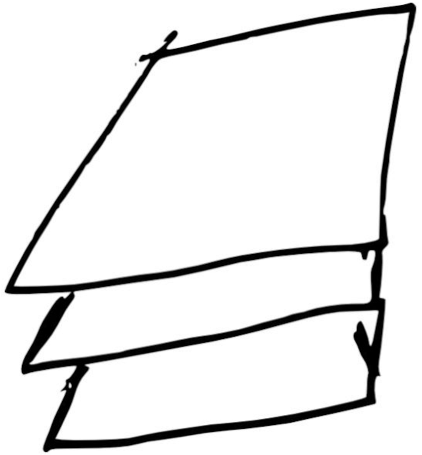
Everyone can have their own local version of a project

To make sure you have the most recent version, you should always pull from the server first.

You can freely make changes on your local version without other people seeing what you are doing

Commiting changes

When you have made some changes and you want to be able to save your current checkpoint as a snapshot you can add and commit



```
git add important_file.txt
```

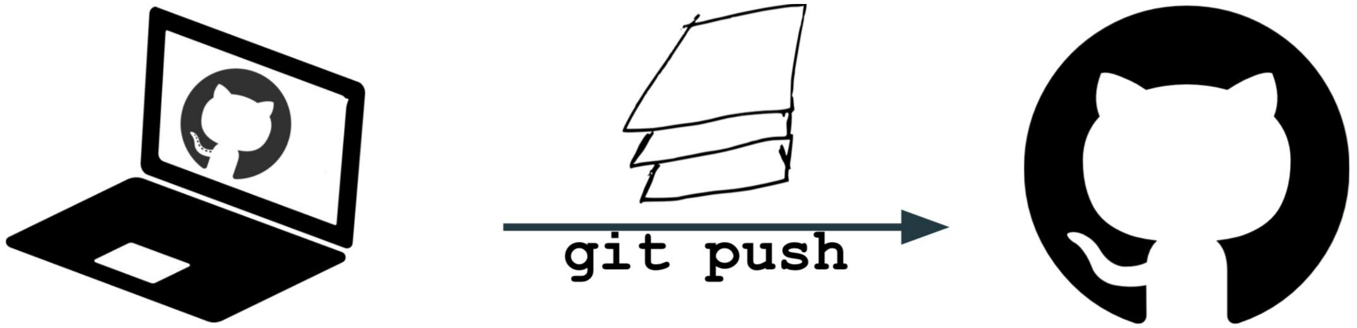
```
git commit -m “added a
description of the
clustering algorithm”
```

Pushing changes

If you are ready for your the snapshots you committed to be available to everyone with access to your GitHub repository.

First, **`git pull`** so you don't create conflicts.

Then **`git push`** your commits to the server



Summary

1. Pull most current content from remote GitHub repository

```
git pull
```

2. Make local edits interesting_file.txt and take a snapshot

```
git add interesting_file.txt
git commit -m “clarified explanation of clustering”
```

3. Push to GitHub

```
git push
```

GitHub repositories for this class

Class materials: https://github.com/zoevernon/STAT-215A-Fall-2018

Repository to clone to submit your project:
https://github.com/zoevernon/stat-215-a

Submitting your projects

1. Clone my stat-215-a repository: `git clone https://github.com/zoevernon/stat-215-a`
2. Create your own version of the repository (more instructions to follow)
3. Add me as a collaborator
4. Add, commit, and push the files

Setting up the repositories

Install Git on your system:
https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

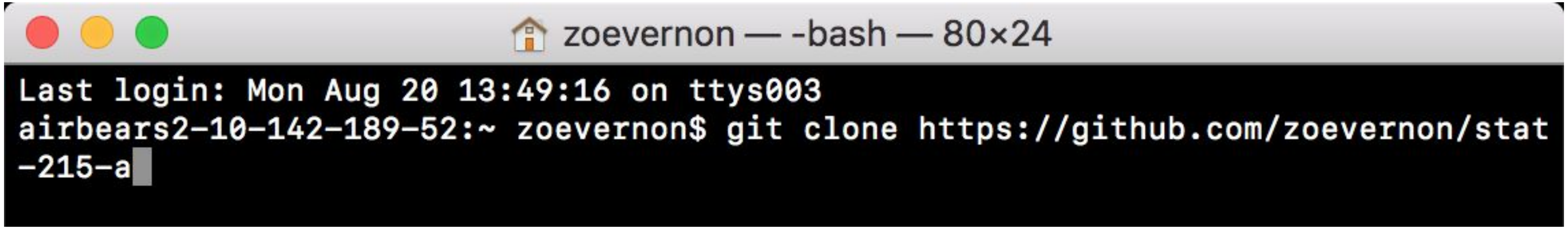
Sign up for GitHub: https://github.com/

Go to https://education.github.com/ and sign up for the student pack to get unlimited private repositories. You are a “student” and you want an “individual account”

Setting up the repositories

Create a copy of my stat-215-a repository on your own computer:

```
git clone https://github.com/zoevernon/stat-215-a
```

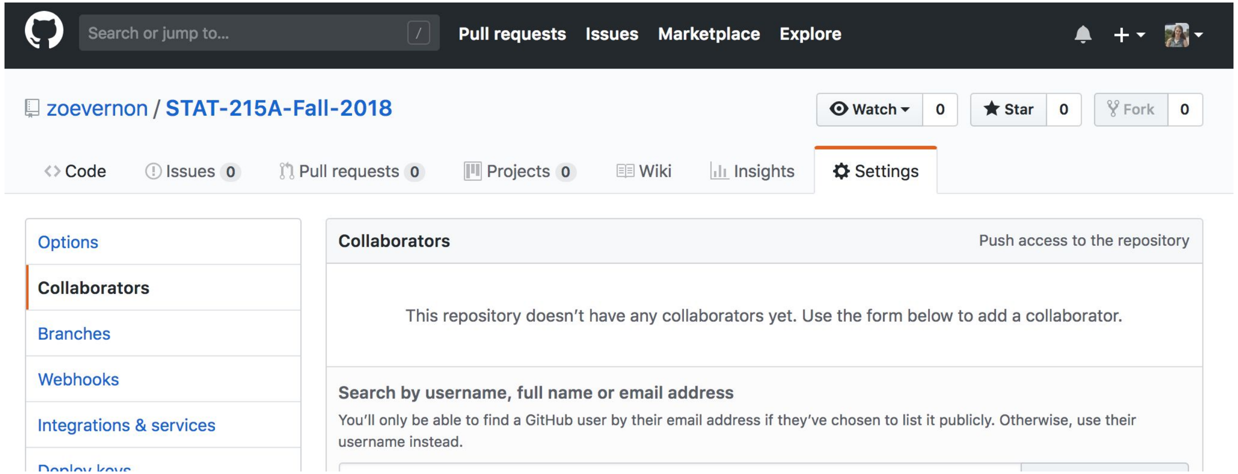


```
Last login: Mon Aug 20 13:49:16 on ttys003
airbears2-10-142-189-52:~ zoevernon$ git clone https://github.com/zoevernon/stat
-215-a
```

Setting up the repositories

On the GitHub website, login and create a **private** repository called stat-215-a

Add me (zoevernon) as a collaborator for this repository. Click "Settings" and navigate to "Collaborators"



Collaborators — Push access to the repository

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Setting up the repositories

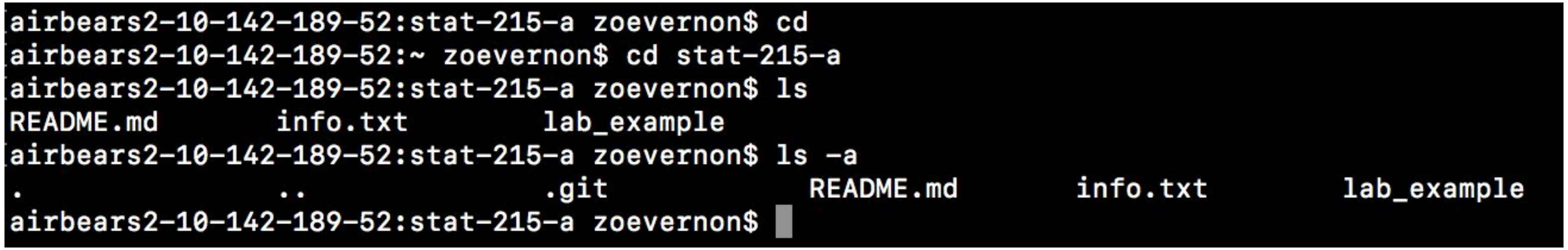
Locally, on your machine, set the origin of your local repository to be the remote repo that you just created:

```
git remote set-url origin
https://github.com/USERNAME/stat-215-a.git
```

This tells git which remote repository to push and pull from

Setting up the repositories

In your new stat-215-a folder there is a file named *info.txt*



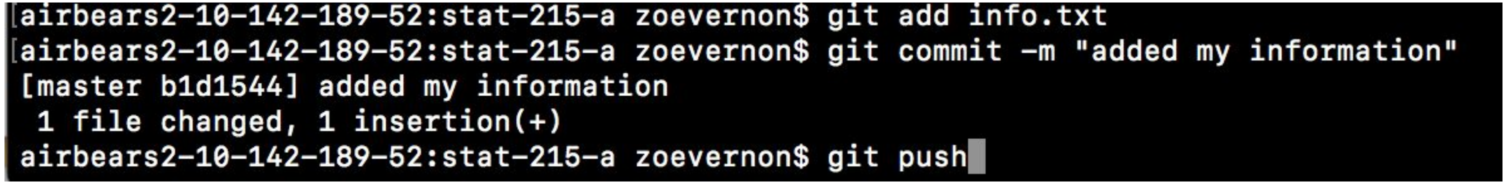
```
airbears2-10-142-189-52:stat-215-a zoevernon$ cd
airbears2-10-142-189-52:~ zoevernon$ cd stat-215-a
airbears2-10-142-189-52:stat-215-a zoevernon$ ls
README.md       info.txt        lab_example
airbears2-10-142-189-52:stat-215-a zoevernon$ ls -a
.               ..              .git            README.md       info.txt        lab_example
airbears2-10-142-189-52:stat-215-a zoevernon$ 
```

Edit this file to reflect your own information. Type `nano info.txt` (or whatever text editor you are most comfortable with)

Type `git status` and you should see that *info.txt* is modified

Setting up the repositories

Add, commit, and push these changes to your remote repository



```
airbears2-10-142-189-52:stat-215-a zoevernon$ git add info.txt
airbears2-10-142-189-52:stat-215-a zoevernon$ git commit -m "added my information"
[master b1d1544] added my information
 1 file changed, 1 insertion(+)
airbears2-10-142-189-52:stat-215-a zoevernon$ git push
```

R and tidyverse

R Markdown and R Sweave

File types for combining text, code, and output

R Markdown:

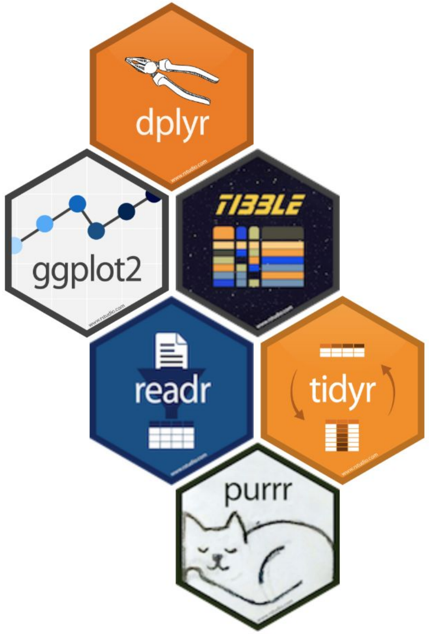
- ❏ Essentially a markdown document
- ❏ Difficult to add `LaTex` preamble, but create `LaTex` equations
- ❏ Better for creating html output

R Sweave:

- ❏ Essentially a `LaTex` document (need to install `LaTex` on your computer)
- ❏ Can add `LaTex` preamble and create `LaTex` equations
- ❏ Better for creating pdf output

Tidyverse

tidyverse
www.rstudio.com


dplyr
ggplot2
TIBBLE
readr
tidyr
purrr

Tidy data

1. Each variable forms a column
2. Each observation forms a row
3. Each type of observational unit forms a table

Resources:

https://cran.r-project.org/web/packages/tidyr/vignettes/tid%20y-data.html

http://vita.had.co.nz/papers/tidy-data.pdf

dplyr: "Grammar of data manipulation"

Contains functions for editing the variables in a data frame or tibble

e.g. `filter(), select(), mutate(), group_by(), summarize(), arrange()`

The first argument for each function is a data frame (or tibble)

The result is a new data frame (or tibble)

dplyr: piping (%>%)

Pronounce “%>%” as “then”

Piping is a way of chaining together functions so that you don’t have to keep redefining variables

Piping always puts the object being piped into the first argument of the function

dplyr: basic functions

`filter()`: find rows were the specified condition is true (see also `filter_all()`, `filter_if()`, and `filter_at()`)

`select()`: keep or remove only certain variables (see also `select_all()`, `select_if()`, and `select_at()`)

`mutate()`: create new variables consisting of functions of existing variables (see also `mutate_all()`, `mutate_if()`, and `mutate_at()`)

- `ifelse()` and `case_when()` allow you to create more complicated conditions

`group_by()`: change the scope of each function from operating of the entire dataset to operating on groups

`summarize()`: reduces multiple values down to a single summary

tidyr: "Grammar of tidy data"

Contains functions for changing the shape of the data from long-form to wide-form

Two main functions

❏ `spread()`: spreads a key-value pair across multiple columns
❏ `gather()`: takes multiple columns and collapses into key-value pairs, duplicating all other columns as needed

ggplot2: "Grammar of graphics"

The base layer is `ggplot()` which always has a data frame as the first argument

Functions are linked together by + (similar to piping `%>%`) each adding another layer to the plot

e.g. `geom_point()`, `geom_histogram()`, `geom_text()`, `theme()`, `scale_color_continuous()`

For next week

Lab 0 will be due next Friday (8/31).

Not for grade, but you must at least submit an empty project so we can ensure your GitHub is working.