

Beginning your Data Science Life Cycle

Problem Formulation, Data Collection,
& Data Cleaning

January 22, 2025

Today's plan

- 1 Review: Problem Formulation + Data Collection**
- 2 Data Cleaning**
- 3 Introduction to Lab 1**
- 4 Setting up Git/GitHub**

Review: Problem Formulation + Data Collection

Review of last time

What do **problem formulation** and **data collection** look like in reality?


- 1 **Case Study 1: Cardiovascular Genomics**
- 2 **Case Study 2: COVID-19 PPE Resource Allocation**

Case Study: Cardiovascular Genomics



Euan Ashley, MD, PhD (Stanford University)

Which gene interactions are important drivers of heart disease?

- + How do you define “interaction”?
 - + What type of data do we have? Any limitations with the data?
 - + Which heart disease? How do you quantify that heart disease?
 - + Can you tell me more about the particular type of heart disease that you want to study?
 - + Why do you want to find these interactions? How many genes?
 - + ...
-  Communicate, communicate, communicate across the full scientific team!
- + Learn about the **data** and the **science**

Case Study: COVID-19 PPE Resource Allocation



Don Landwirth (Response4Life)

! Setting: beginning of March 2020

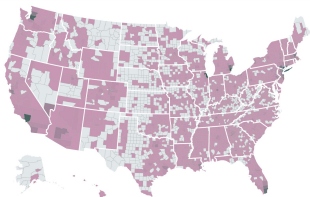
We have PPE to donate to hospitals. Which hospitals are in most need of the PPE so that we can send it to them?

- + What types of PPE?
 - + Where are the hospital regions of interest?
 - + How long does it take to deliver the PPE to various places?
 - + Loads of questions about PPE availability and the available hospital data...
- 🔑 No data → modify the problem formulation + **justify** your modifications

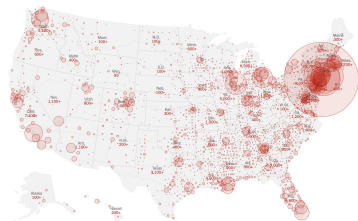
Data Collection (from **20+ sources**)

COVID-19 Cases/Deaths

USA FACTS



The New York Times



County-level Data

(Risk Factors, Demographics, SES, Social Mobility)



Centers for Disease Control and Prevention
CDC 24/7: Saving Lives, Protecting People™

Division for Heart Disease and Stroke Prevention



esri™

COVID-19 GIS Hub

County Health
Rankings & Roadmaps

Building a Culture of Health, County by County

USDSS

UNITED STATES
DIABETES
SURVEILLANCE SYSTEM
Division of Diabetes Translation, CDC



GHDx



CMS.gov

Centers for Medicare & Medicaid Services

United States®
Census
Bureau

SAFE GRAPH



kinsa®



STREETLIGHT



cuebiq



Introducing the Unacast

Social Distancing
Scoreboard

KHN
KAISER HEALTH NEWS



JOHNS
HOPKINS
UNIVERSITY

Google

Apple Maps Mobility Trends Reports

COVID-19 Community Mobility Reports

Hospital-level Data

(e.g., #ICU beds, staff)

HRSA
Health Resources & Services Administration



ArcGIS Hub



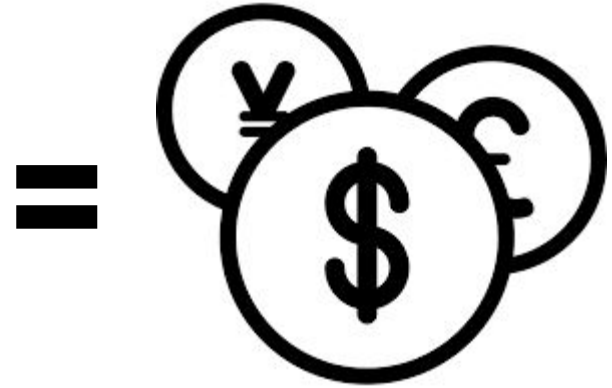
Samuel
Scarpino



Data is the real currency



Image credit: Dall-E

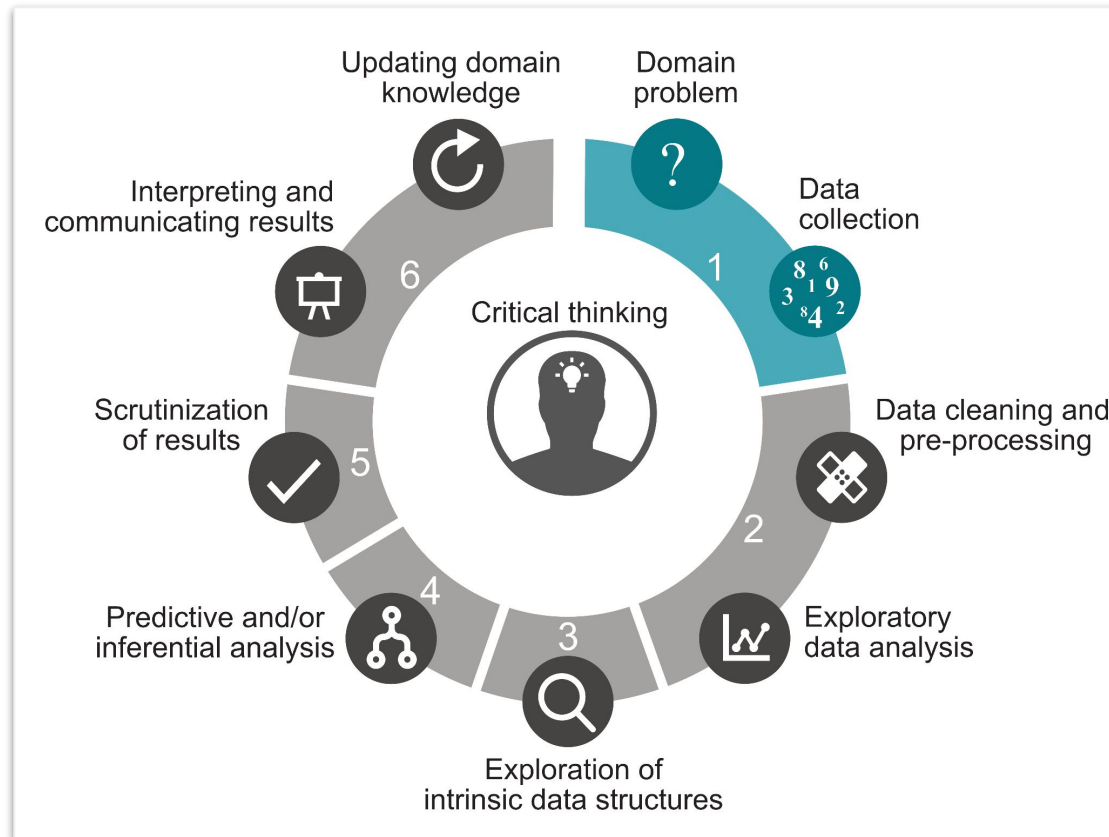


=

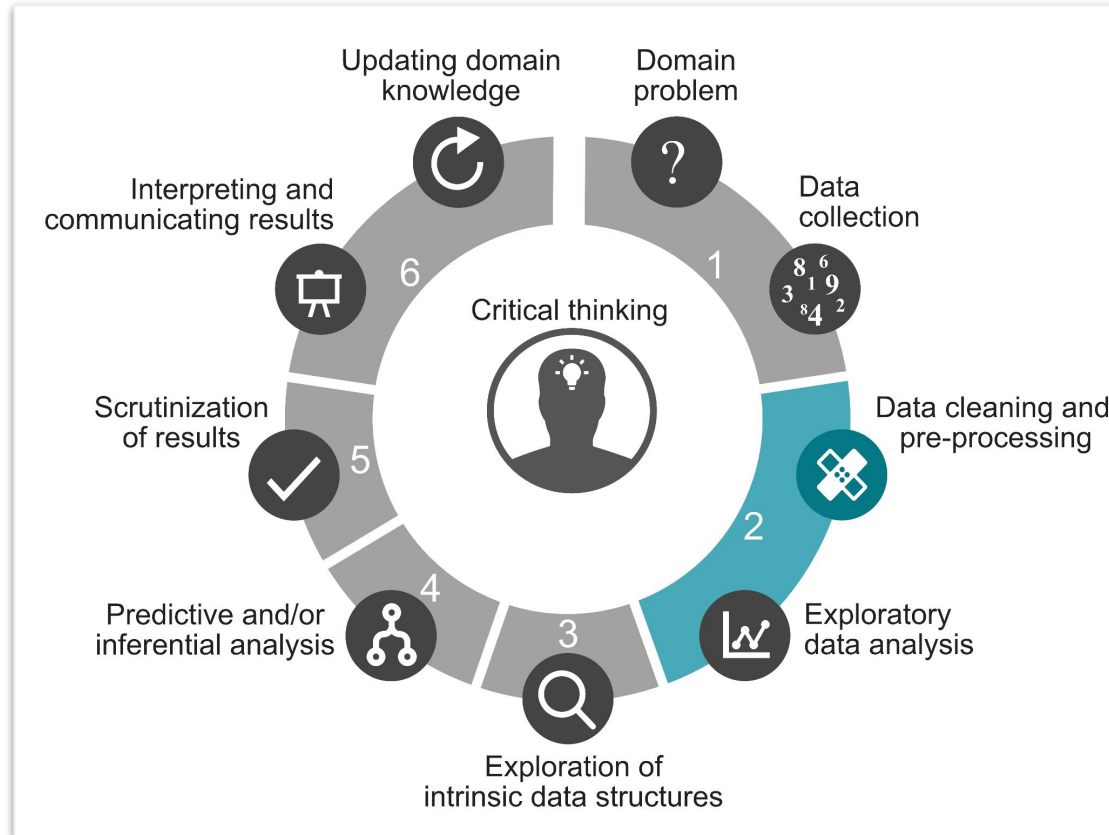
Lack of representation in training data reinforces societal biases

- + **Google Photos tagging:** found to be racially biased
 - + Mistakenly tagged Black people as gorillas [[Forbes report](#) (2015)]
 - + “Google’s Photo App Still Can’t Find Gorillas. And Neither Can Apple’s.” [[NYT](#) (2023)]
- + **Facial Recognition:** typically higher error rates for women and people of color
- + **Algorithmic biases in healthcare:**
 - + Many dermatological AI tools are trained predominantly on images of lighter skin tones
 - + Pulse oximeters are known to be less accurate in individuals with darker skin
 - + During COVID-19 pandemic, this led to under-detection of hypoxemia in Black patients
 - + Very classical heart disease diagnosis algorithms were developed in trials with majority white males → misdiagnosis and/or delayed treatment for minority populations

We've got the data. What's next?



We've got the data. What's next?



Data Cleaning

USAFacts COVID-19 County-level Case & Death Counts Data

**Got the data from
website**



**Q: What are potential data
issues to look out for?**

countyFIPS	County Name	State	stateFIPS	1/22/2020	1/23/2020
0	Statewide	AL	1	0	0
1001	Autauga Co	AL	1	0	0
1003	Baldwin Co	AL	1	0	0
1005	Barbour Co	AL	1	0	0
1007	Bibb Count	AL	1	0	0
1009	Blount Cou	AL	1	0	0
1011	Bullock Cou	AL	1	0	0
1013	Butler Cour	AL	1	0	0
1015	Calhoun Co	AL	1	0	0
1017	Chambers C	AL	1	0	0
1019	Cherokee C	AL	1	0	0
1021	Chilton Cou	AL	1	0	0
1023	Choctaw C	AL	1	0	0
1025	Clarke Cour	AL	1	0	0
1027	Clay County	AL	1	0	0
1029	Cleburne C	AL	1	0	0
1031	Coffee Cou	AL	1	0	0
1033	Colbert Cou	AL	1	0	0
1035	Conecuh C	AL	1	0	0
1037	Coosa Cour	AL	1	0	0
1039	Covington C	AL	1	0	0
1041	Crenshaw C	AL	1	0	0
1043	Cullman Co	AL	1	0	0
1045	Dale Count	AL	1	0	0
1047	Dallas Cour	AL	1	0	0
1049	DeKalb Cou	AL	1	0	0
1051	Elmore Cou	AL	1	0	0

USAFacts COVID-19 County-level Case & Death Counts Data

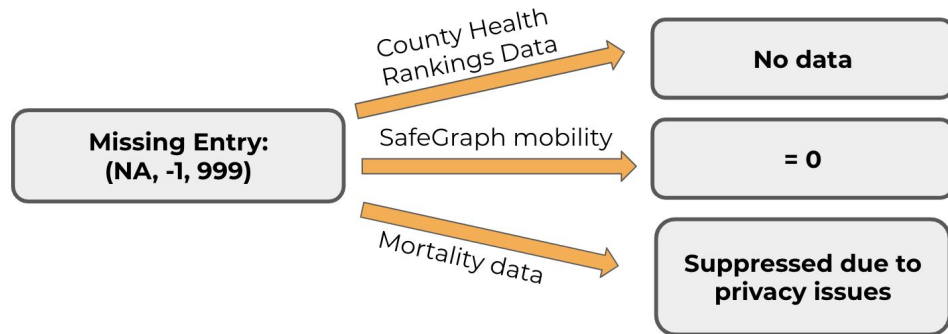
Issues found from an **examination** of the data + (basic) **domain knowledge**:

- **Irrelevant entries:** remove state entry if interested in counties only
- **Duplicates:** multiple entries for the same county
- **Invalid values:** *cumulative* deaths counts sometimes decrease
- **Missingness:** some cases/deaths cannot be allocated to a county
- **Data revisions:** some case/death counts are revised days after posted
 - Also a major change in the reporting “probable” cases/deaths ~ April 14

Some issues are *fixable* while some aren't, but we do what we can and **document** as much as possible.

More Data Cleaning!

Improperly coded missing values



Read all available documentation!

More Data Cleaning!

Different ways of encoding the same text

- IN = Indiana = indiana
- St. Joseph = St. Joseph County = St Joseph County = st joseph county
- “01234” zip = 1234 zip
- If there are two Joe Smith’s, are they the same person or different?

Data encoding choices for non-numerical data

- Categorical variables → dummy or one-hot encoding
- Text → text embeddings (e.g., word2vec)
- Graph → adjacency matrix or graph embeddings
- Images/videos → matrix or tensor of pixel/RGB values

More Data Cleaning!

Data transformations

- Using log-, sqrt-, or Box-Cox-transformations to transform highly-skewed data to more Gaussian-like data
- Should we center and scale the data so that each feature's values have mean 0 and variance 1?

Row/column filtering

- Feature selection: which features do we keep?
- Outlier removal: which samples do we remove?

Feature engineering

- Can we create new features that might be helpful for our task (maybe based on domain knowledge)?

Tips + Takeaways from data cleaning

“The data cleaning procedure will involve learning about the data collection process, obtaining domain knowledge, examining the data, asking and answering questions, checking assumptions, and identifying and documenting any issues and inconsistencies...”

- [Veridical Data Science \(VDS\) Textbook](#) (ch 4)

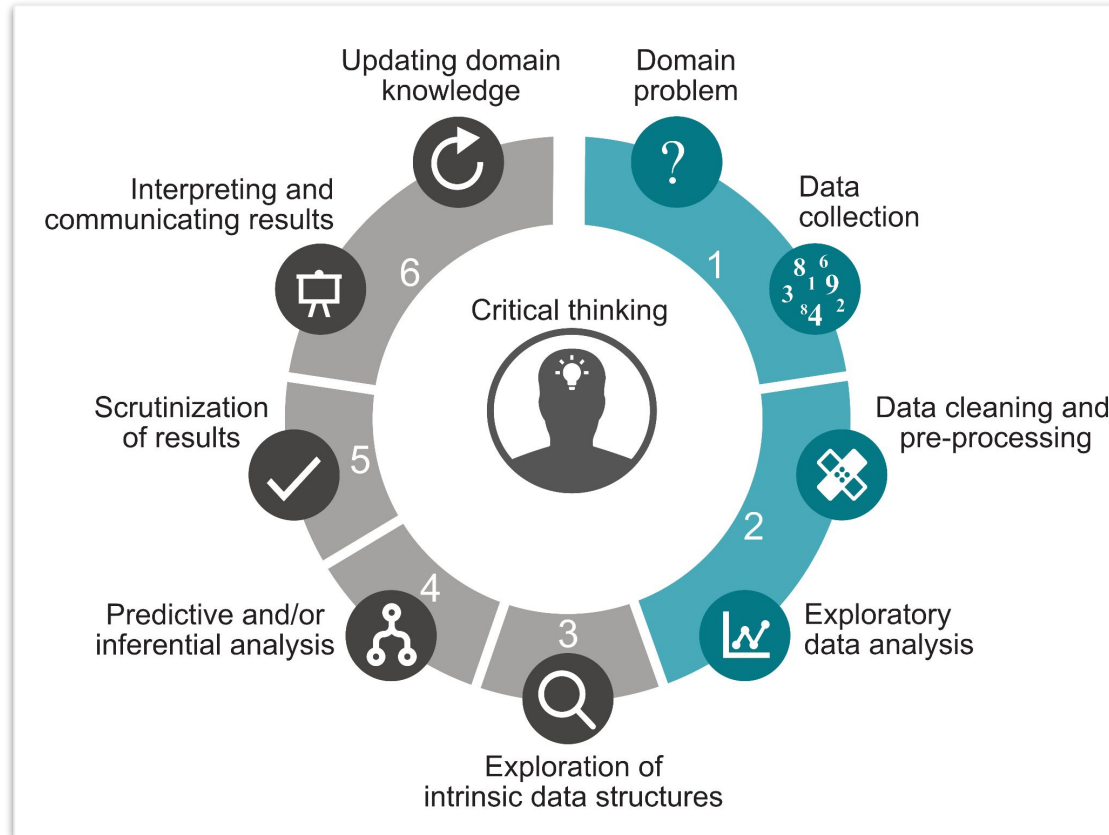
It's an iterative and interactive process!

Summary of the data science life cycle so far

- + **Problem formulation:** includes formulation of both the *statistical* and the *substantive* problem
- + **Data collection:** garbage in, garbage out
- + **Data cleaning:** a highly iterative process
 - + Choices must be made. Justify them whenever possible.
- + Ask questions, use common sense, and **document** everything

Introduction to Lab 1: Redwood Trees

Lab 1: Redwood Trees



A Macroscope in the Redwoods [[Tolle et al. \(2005\)](#)]

+ Coastal redwood trees

- Tallest trees in the world (>350ft or 115m)
- Incredibly old species (pre-dating humans, spiders, and flowers, first appearing over 240 million years ago during the time of the dinosaurs)



A Macroscope in the Redwoods [[Tolle et al. \(2005\)](#)]

- + 44-day study in Sonoma, California
(April 27, 2004 5:10pm - June 10, 2004 2pm)

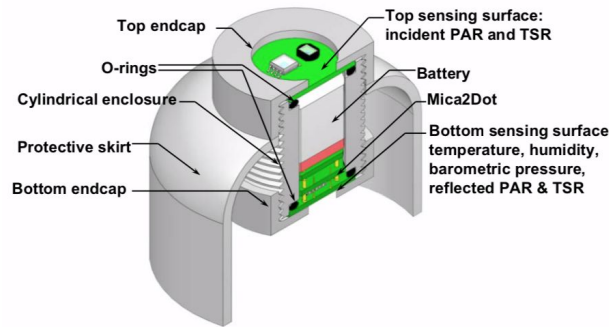


Figure 2: Sensor node and packaging

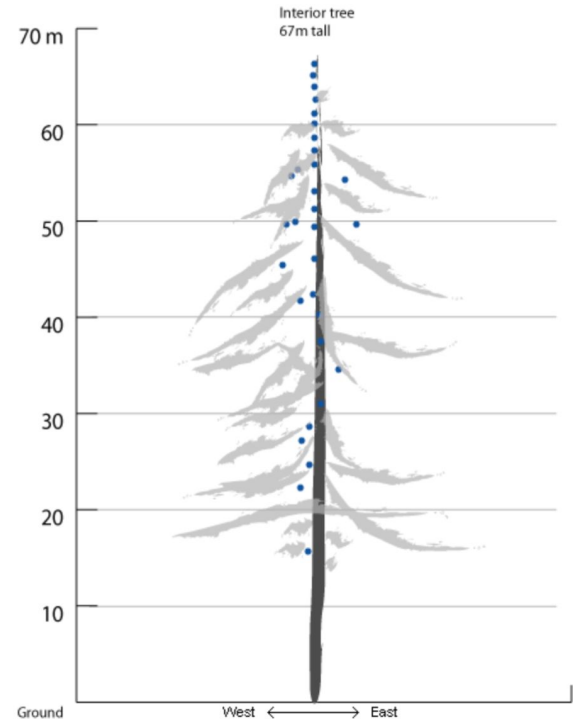


Figure 1: The placement of nodes within the tree

Introduction to Git/GitHub

Tools for today: Git and GitHub

Make sure that you have installed **git** on your computer:

<https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

If you haven't already, **please sign up for GitHub** (<https://github.com>)

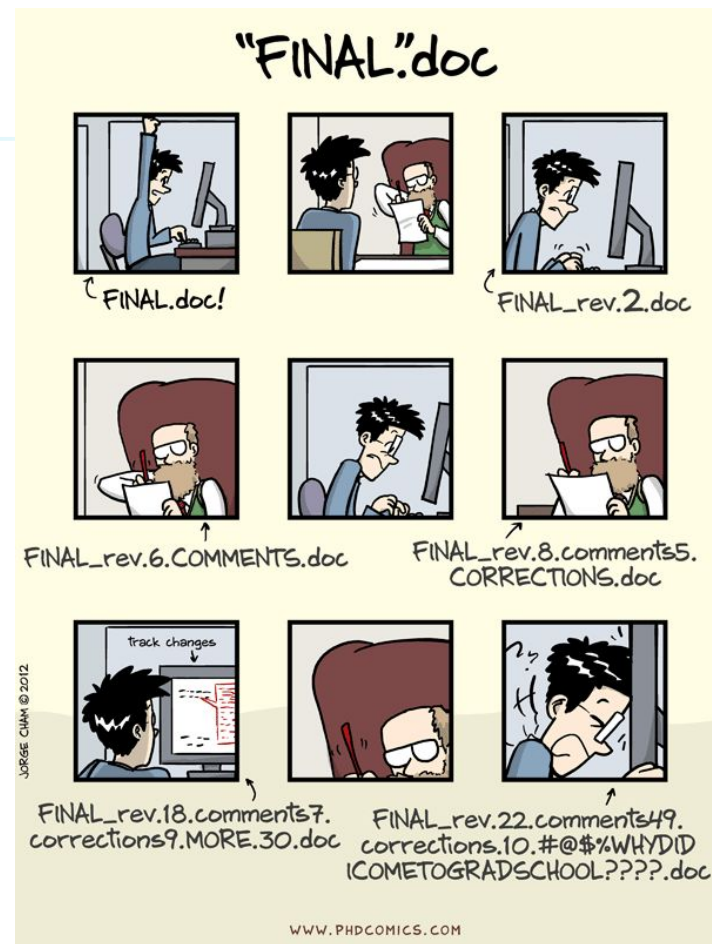
- + Sign up for the student pack (<https://education.github.com/>) to get unlimited private repositories. You are a "student" and want an "individual account".

While optional, I highly recommend downloading the following software:

- + **GitHub Co-pilot:** <https://github.com/features/copilot>
 - AI code completion tool developed by GitHub and OpenAI
- + **GitHub Desktop:** <https://desktop.github.com/download/>
 - GUI for interacting with GitHub (as opposed to command line only)
- + **GitKraken:** <https://www.gitkraken.com/>
 - A fancier and nicer alternative to GitHub Desktop

What is git?

- + A 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



Git vs. GitHub

Local Git Repository

- + You have a local version of the folder on your computer
- + History stored in .git file
- + Only you can see the changes made in the local version



Remote GitHub Repository

- + A remote version of the folder is hosted on the GitHub website
- + Everyone can see these changes (if repository is public)



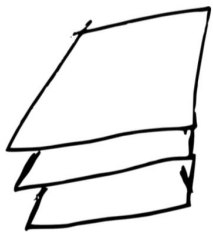
Why do we need Git/GitHub?

- + Imagine working on a project with several collaborators...
- + Using Git/GitHub allows everyone to have their own local version of the project while still maintaining a “main” version of the project, hosted remotely on GitHub
- + You can make changes freely without people seeing what you are doing
 - You can thoroughly test your changes before adding to the master copy
- + Version control!!
 - Especially great if your changes create bugs because you can backtrack/revert

Typical Git/GitHub Pipeline

(2) make local changes

(e.g., create file called filename.txt)



**LOCAL
REPOSITORY**



(3) git add filename.txt

(changes are staged/waiting to be committed)

(4) git commit -m "[description of changes]"

(commit when you have made some changes and want to be able to save your current checkpoint as a snapshot)

(1) git pull

(to retrieve the most recent version from the server)



(5) git push

(make changes available to everyone with access to the repo)

**REMOTE
REPOSITORY**



Warning: remember to “git pull” before “git push” to mitigate potential merge conflicts

Let's set up a GitHub repository for this class

Setting up a GitHub repository

1. Three (of many) possible starting points:
 - a. Start with an *existing* remote repository from GitHub
 - b. Create a *new* remote repository on GitHub
 - i. I usually initialize a **private** repo with a **README** file and with a **license** (e.g., MIT)
 - c. Convert local folder into a git/GitHub repository

Setting up a GitHub repository

1. Three (of many) possible starting points:
 - a. Start with an *existing* remote repository from GitHub
 - b. Create a *new* remote repository on GitHub
 - i. I usually initialize a **private** repo with a **README** file and with a **license** (e.g., MIT)
 - c. Convert local folder into a git/GitHub repository
2. Clone remote repository to your local computer via one of the following:
 - a. **Command Line:**
`git clone https://github.com/[username]/[repositoryname]`
 - b. **GitHub Desktop:** click green “Code” button > “Open with GitHub Desktop”

Setting up a GitHub repository

1. Three (of many) possible starting points:
 - a. Start with an *existing* remote repository from GitHub
 - b. Create a *new* remote repository on GitHub
 - i. I usually initialize a **private** repo with a **README** file and with a **license** (e.g., MIT)
 - c. Convert local folder into a git/GitHub repository
2. Clone remote repository to your local computer via one of the following:
 - a. **Command Line:**
`git clone https://github.com/[username]/[repositoryname]`
 - b. **GitHub Desktop:** click green “Code” button > “Open with GitHub Desktop”

You should now see a local folder named `repositoryname/`

Making changes to your repository

3. Make changes to your repository, e.g.,

- a. Create a new file called info.txt
- b. In your info.txt file, add the following two lines:
 name = "Jane Smith"
 github_name = "janesmith"

4. Follow "Typical GitHub pipeline": in command line,

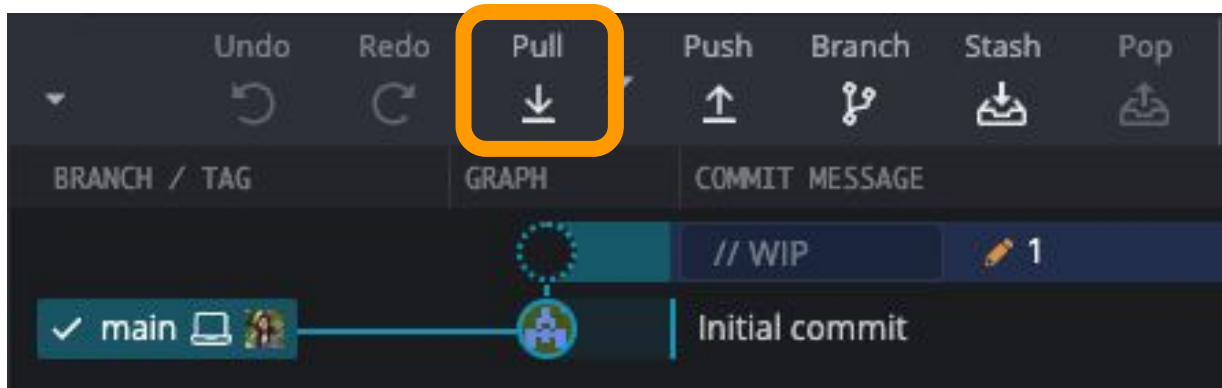
- a. `git pull` # Retrieve latest version of repository from remote
- b. `git add info.txt` # Stage changes (can add/remove multiple files)
- c. `git commit -m "added info"` # Commit staged changes to save a snapshot
- d. `git push` # Push from local to remote repository

Alternatively, can accomplish all of these steps using GUIs like GitHub Desktop or GitKraken.

Managing your repository using GitKraken

3. Open README.md and edit file
4. Follow “Typical GitHub pipeline”: in GitKraken,

(a) `git pull` # Retrieve latest version of repository from remote

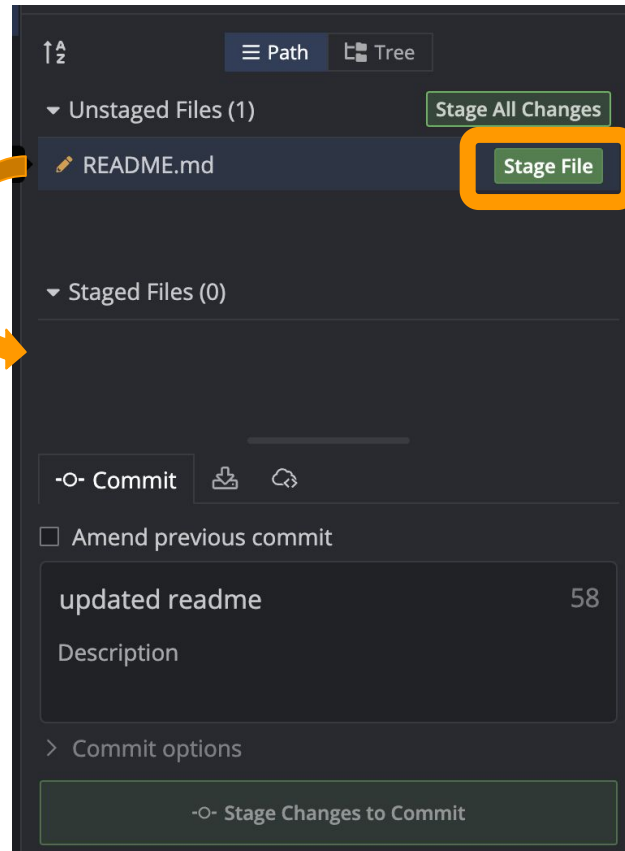


Managing your repository using GitKraken

3. Open README.md and edit file
4. Follow “Typical GitHub pipeline”: in GitKraken,

(b) `git add README.md`

Stage changes (can add/remove multiple files)



Managing your repository using GitKraken

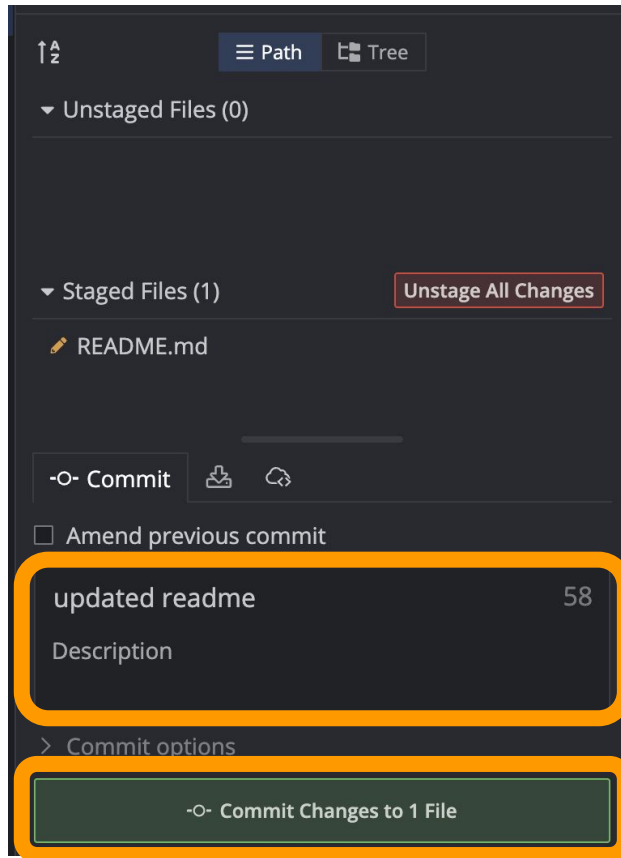
3. Open README.md and edit file
4. Follow “Typical GitHub pipeline”: in GitKraken,

(c) `git commit -m “updated readme”`

Commit staged changes to save a snapshot

(i) add description

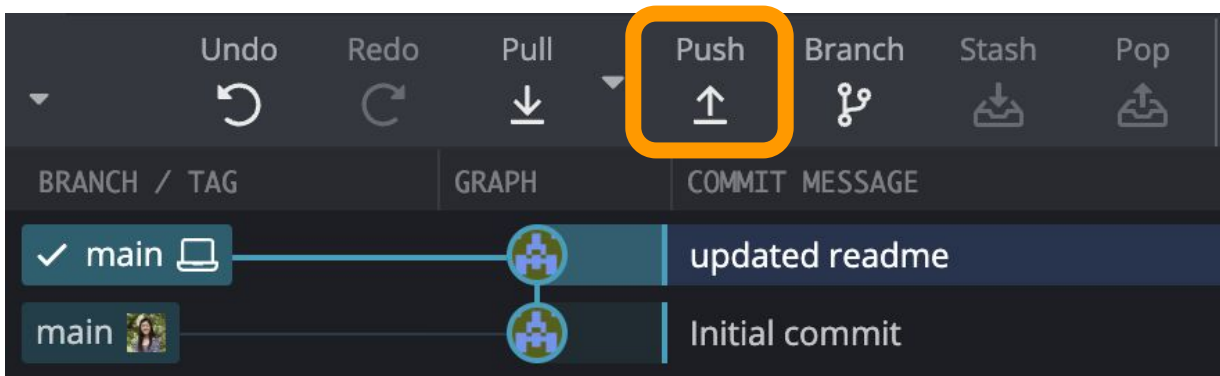
(ii) click commit



Managing your repository using GitKraken

3. Open README.md and edit file
4. Follow “Typical GitHub pipeline”: in GitKraken,

(d) `git push` # Push from local to remote repository



Utilizing .gitignore

- + Often, we do not want to track changes and push all files to GitHub
- + Some types of files or folders that we usually do not want to track:
 - __pycache__/
 - .DS_Store
 - data/
- + We can add these files to the .gitignore file:

```
1 .Rproj.user
2 .Rhistory
3 .RData
4 .Ruserdata
5 .DS_Store
6
7 __pycache__/*
8 data/*
```

Git Cheat Sheet

- + To check the status of working directory and staging area: **git status**
- + To see what is different/changed in file(s) but not staged: **git diff**
- + To create a new branch at the current commit: **git branch [branch-name]**
- + To switch to another branch: **git checkout**
- + To save modified and staged changes (e.g., in order to change branches or pull from remote): **git stash**
- + To retrieve previous stash: **git stash pop**

Full cheat sheet: <https://education.github.com/git-cheat-sheet-education.pdf>

Adding collaborators to your GitHub repository

On GitHub, go to:

Settings > Collaborators > Add “tiffanymtang”

Getting started with Lab 1

- + Instructions, redwood paper, and data found on Canvas
- + Carefully read through Tolle et al.
- + You should be able to start thinking about and answering the problem formulation/data collection questions in Part 1

Recap + What's Coming Up

Recap

- + **Git** is a version control system. **GitHub** hosts git repositories remotely + does more.
- + **Typical GitHub workflow:** git pull, add, commit, push

Coming up...

- + Reproducible environments with renv in R and conda in python
- + Hands-on practice with data cleaning + exploratory data analysis
[\[chapters 4 and 5 from VDS textbook\]](#)