DSC Capstone Sequence

Lecture 02
Using Servers

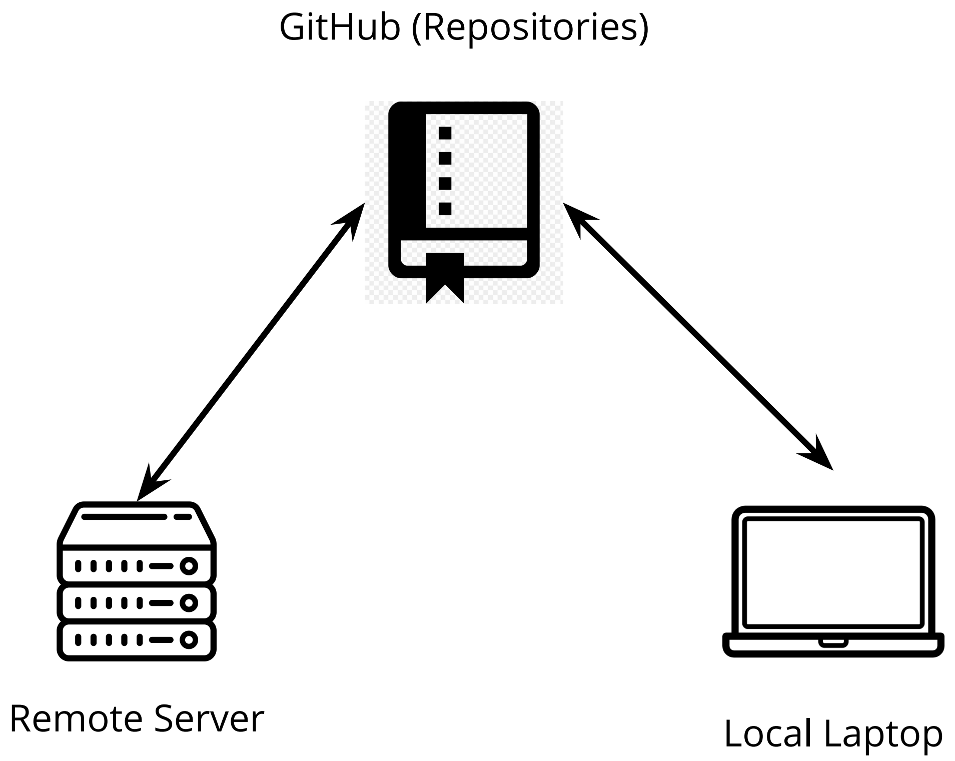
Lecture Outline

- Local vs. Remote Development
- Using the campus cluster

Local vs Remote Development

The Data Scientist's Work Environment

- Local Development on laptop
  - Cheap and convenient
  - Higher efficiency in your 'home' environment
- Development on Server
  - More RAM and CPU available
  - Long running programs
- Need ability to do both!
  - Can run code in both places *simultaneously!*
  - However, code must run both places



The local + remote setup example

Setup: need to train a model on 1 TB data. Est time ~24 hours.

- The bad way:
  - Provision a large, expensive gpu cluster (remote) - run `train.py`.
  - Wait to finish; realize code was incorrect. (this *will* happen to you!)
  - Restart process; waste weeks of time and compute.
  - Same holds true with 10 GB data and your laptop!
- The good way:
  - Develop your code locally on small, realistic *test data*.
  - Push changes to GitHub from laptop; pull changes from GitHub to gpu cluster.
  - run `train.py` *once* and be happy with the results!
- Moral: don't run under-tested code in 'production'!

Required tools for dual development

Moving between local and remote development requires these skills/tools:

- Requires keeping project code up-to-date in both places:
  - Diligent use of git, with remote repositories on GitHub. Branches will help keep code in sync between the two places.
- Requires use of the command-line (bash):
  - Remote servers don't have desktop GUIs (only Juypter notebooks, at best)
- Ability to keep a consistent compute environment between local/remote
  - We will learn to use Docker for this later (also, package your own python module)

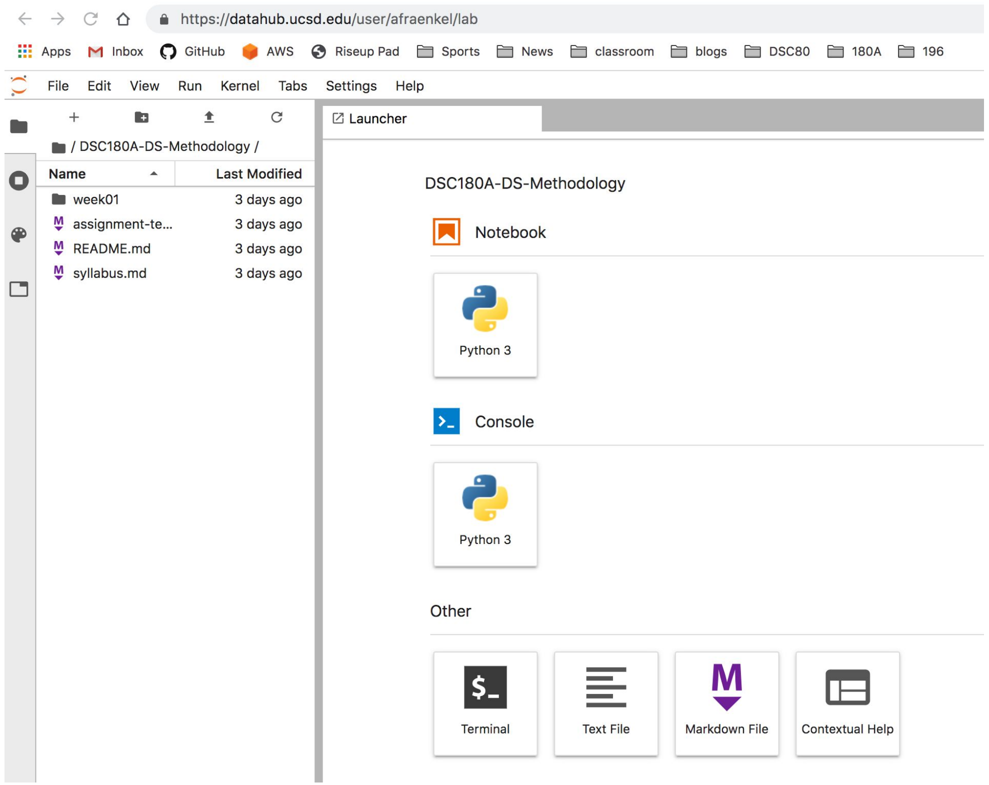
Have a course Piazza meant for help with topics!

Using the Campus Cluster

Remote Server: DataHub

- Jupyter portal just a server.
- Open a JupyterLab IDE by:
  - datahub.ucsd.edu/user/<NAME>/lab
- Open a Terminal from Jupyter to pull from git, etc…
- DataHub shuts down after 30 minutes of client inactivity =(
- Only small RAM/CPU

We *won't* be using datahub!



Name	Last Modified
week01	3 days ago
assignment-te...	3 days ago
README.md	3 days ago
syllabus.md	3 days ago

Remote Server: DSMLP

Accessing DataHub DSMLP through the Terminal.

- Like DataHub, but accessed via command-line.
  - use *ssh* to log-in to an initial server.
  - launch a default environment much like datahub…
  - launch a custom environment, with custom software.
  - specify RAM, CPU/GPU up to limits larger than your laptop + long running jobs.
- Use your own personal server; up your cloud-computing skills!
  - DSMLP is a Kubernetes cluster on which you can run processes in Docker containers.
  - Similar skills to AWS/GCP/Azure.
  - Will learn more about this usage later!

See usage refs here: https://dsc-capstone.github.io/resources/computing/

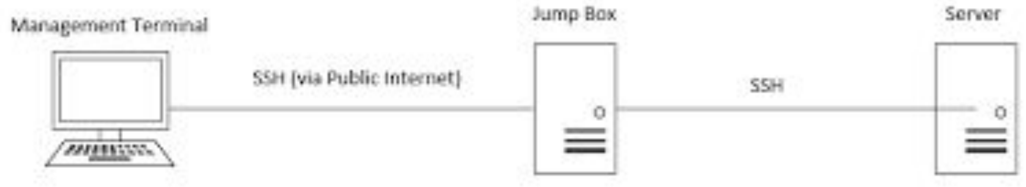
Remote Server Filesystem

Once logged onto the campus cluster (either DSMLP or DataHub)...

- You log into your *home directory*.
  - It is private (only you have permission to read/write)
  - It has small storage
  - Keep and run your code from your home dir.
  - Place your customization files (e.g. `.bashrc`) here.
- DSC 180AB has a shared Volume (external disk) for sharing data
  - Located in `/teams` (not the leading slash!)
  - You have permission to read/write in the subdirectory for your section only!
  - Store/use data in this location (huge storage)
  - If you don't want others reading and/or writing over your data, you can create a directory for yourself and change the permissions

Logging into DSMLP Servers

- `ssh user@dsmlp-login.ucsd.edu` (your school username)
  - Logs you into your home directory in a jump-box.
- `launch-scipy-ml.sh` (launches a 'pod' with 8GB RAM)
  - Your home directory is also available from here
  - Other scripts `launch-XXX-XXX.sh` launch different server configurations
  - Can open a Jupyter Notebook from this server, if on campus network or VPN
- The launch scripts can take a Dockerfile that configures the environment.



DSMLP Tips

- `Set-up ssh keys to avoid always typing your password.`
  - `For logging-in, for pulling/pushing to GitHub`
- `launch-XXX-XXX.sh` takes useful configuration flags:
  - Specify RAM
  - Specify number of CPUs/GPUs
  - Specify timeout
  - Specify Docker Image (environment setup)
- Detaching from your server (long running jobs) [to come]
- Creating your own Docker Image [to come]

Learning basic bash commands/concepts will be helpful!