Data Science for Public Policy

Alex Engler & Aaron Williams - Georgetown University

Command Line Interface and Git

Common CLI Commands

CLI stands for 'Command Line Interface', which is a common way to interact with a computer through code, rather than pointing and clicking.

pwd - Print Working Directory. Prints to the CLI your working directory, which you can think of as where you `are`.

ls - List. Lists all files in the current working directory. The flag (or option) `-a` changes the behavior of `ls` to include hidden files.

cd - Change Directory. This allows you to more the current working directory.

mkdir - Make Directory. This allows you to create new directories (aka folders).

mv - Move. Moves a file from one directory to another. This command takes two inputs - the first is the source (file to be moved) and the second is the destination folder.

touch - Touch. Creates a new file with the name given.

cat - Concatenate. Prints file contents to the CLI.

CLI Examples

Create a new folder named 'test', then move the current working directory into test, then print that directory. The next line moves back to the parent directoty (.. refers to the parent directory) and then prints the working directory again - which is now the original working directory.

```
mkdir test
cd test
pwd
cd ..
pwd
```

The first of these two lines moves a dataset into the test folder. The second line moves the file back to the parent folder of test.

```
mv fips.csv test/
mv test/fips.csv ../
```

The snippet below check the current working directory, see its contents, then create a new text file. Finally, use ls to see the new text file you've created.

```
pwd
ls
touch file.txt
ls
```

The snippet below prints the contents of your new file (it starts empty), then adds a line of text, then prints the contents of that file to the CLI.

```
cat file.txt
echo "Start of my file" >> file.txt
echo "Second line of my file" >> file.txt
cat file.txt
```

The code snippet below creates a new folder, called `test`. Then it moves the text file you created above into the folder, using the `-v` flag (for verbose) to print to CLI any files moved.

```
mkdir test
mv -v file.txt test
ls
```

The code snippet below moves the current working directory into the new folder, prints that new location to the CLI, lists the files in that folder, and then prints the contents of file.txt (which you had moved there in the snippets above).

```
cd test
pwd
ls
cat file.txt
```

Common Git Commands

git init – starts a new git repository including everything in the folder you're currently in.

git status – examines the status of the current repository. This will tell you if files are tracked or untracked (you can track untracked files with git add) and if there are changed ready to be committed or pushed

git add – track files that are currently untracked. This tells git to pay attention to certain files within the repository. Without using add, git does not automatically pay attention to new or changed files.

git add –all – tracks all files within the folder that you have not explicitly told git to ignore.

git commit –m "A descriptive commit message" – groups all the changes you've made (to currently tracked files) since your last commit into one bundle of changes. The commit is that bundle of changes.

git commit –am "A descriptive commit message" – combines the adding of untracked files and the committing of changes into one line.

git log - shows the history of commits made to this repository.

git diff commitid - shows the difference between the commit identified (in the first argument) and the current state of the git repository.

Hit the 'q' key to exit from the `git diff` view.

git remote add origin https://github.com/username/repositoryname - the git command here is `remote add`, while `origin` and the url provided are arguments for that command. Adding a remote means linking this local git repository to a repository on GitHub.com. You only need to do this once per repository. The first argument is the name of the remote repository, which is `origin` by convention, the second argument is the url for the repository you created on GitHub (you need to do this separately by a point-and-click process here: https://github.com/new)

git push origin master – this pushes your commits to the GitHub repository on the internet, assuming you've established the relationship between your local repository and the remote repository.

git pull origin master – this gets any changes from the GitHub repository on the internet, assuming you've established the relationship between your local repository and the remote repository.