Problem Set #3

January 24, 2022

Grade: /22

Overview

Learning how to use git and GitHub can be intimidating. Problem sets this quarter will give you the opportunity to practice using git and GitHub both individually and in groups. In this problem set, you will work on becoming familiar with the basic git/GitHub workflow (e.g., adding, committing, and pushing changes). You'll also be asked to perform other tasks on the command line to help you feel more comfortable using it.

In this problem set, you will write most of your answers in the problemset3.Rmd file, including some questions that ask you to run commands on the Terminal and copy those commands and the resulting output into the .Rmd file; and some of the questions will ask you to add code to an .R script

To give you a sense of the arc, in this problem set you will:

- Create a remote repository on github and then clone it to your local machine
- Create some folders and files using the terminal/command line
- write a short script that downloads/unzips/reads-in data, and creates a graph
- perform git operations on Terminal to files you created above

Part I: Command line & setting up git repo

/1

1. First, create a new private GitHub repository in the `anyone-can-cook` organization:

    - Navigate to this page in your browser
    - Name your repo as `ps3_lastname_firstname` (fill in your name)
    - Select **private** repository
    - Check the `Add a README file` and `Add .gitignore` boxes and choose the `R` template for the `.gitignore` file

/1

2. Open your **Git Bash** (Windows) or **Terminal** (Mac) and navigate to where you want to clone your repository. (*Hint*: Change directories to where you want this repository to go) Then, obtain the URL for the GitHub repository you created and clone it to your local machine. Copy the commands you used for each step in the code block below:

```
# Command to change directories
pwd
cd desktop
pwd

# Command to clone repo
git init
git clone https://github.com/anyone-can-cook/ps3_nazario_paula
git status
```

/1

3. Next, turn this repository into an **RStudio Project** by opening your **RStudio** and creating a new project from an existing directory (i.e., `ps3_lastname_firstname`). Place this `problemset3.Rmd` file that you're working on into this project directory.

   Now that you have your **RStudio Project** set up, we recommend you use your **RStudio Terminal** moving forward, since it automatically starts up in your project directory.

/1

4. In your **RStudio Terminal**, run the command that lists out all the contents of your `ps3_lastname_firstname` directory, including hidden files and directories (i.e., entries starting with `.`) Copy the command you used as well as the **output** you see in the code block below:

```
# Command to list directory contents, including hidden files/directories:
ls
ls -a

# Output:
README.md               problemset3.Rmd         ps3_nazario_paula.Rproj

.                       .Rproj.user             README.md
..                      .git                    problemset3.Rmd
.Rhistory               .gitignore              ps3_nazario_paula.Rproj
```

/1.5

5. Still using the command line, create the following directories inside `ps3_lastname_firstname`: `scripts`, `data`, and `output`. Then create 2 additional subdirectories inside `output`: `output/files` and `output/plots`. Also create an (empty) R script called `ps3_script.R` inside the `scripts` folder. Your directory structure should look like the following. Copy the commands you used in the code block below.

```
ps3_lastname_firstname
|
|- data/
|- output/
  |- files/
  |- plots/
|- scripts/
  |- ps3_script.R
```

```
# Command to create new directories
mkdir -p data output/files output/plots scripts

# Command to create R script
touch scripts/ps3_script.R
```

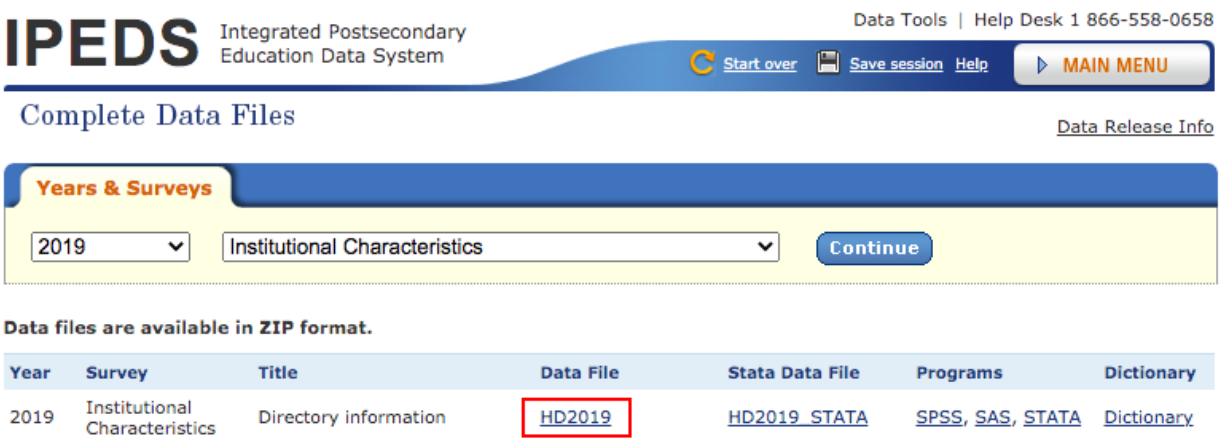
Part II: Working with data in R

/1

1. Open up the `ps3_script.R` file you created and copy paste the template from here in your `ps3_script.R` file. Load the `tidyverse` library and create the following 3 file path objects: `data_dir`, `plot_dir`, `file_dir` (*Hint*: Remember that when you run an R script inside an **RStudio Project**, the working directory will be your project directory, so make sure your file path objects are created relative to your project directory)

/1.5

2. Navigate to the IPEDS Data Center in your browser and search for the **2019 Institutional Characteristics** survey as seen below. Depending on your browser, you may need to first select **Complete data files** under **Survey Data** to get to this page.



Right click on the **HD2019** link as seen above and choose **Copy Link Address** to obtain the URL to the data file. In your `ps3_script.R`, use this URL to download the `HD2019` data file into `data_dir`. Then, also unzip the data file inside `data_dir`. (*Note*: Make sure to use R functions to download and unzip the data from your script. Do not download it from your browser.)

/1.5

3. Read in the CSV data to a dataframe called `hd` – but only read in the following variables and make sure they have the following data types (*Hint*: Use `cols_only()` to help you read in only a subset of the columns):

   - `UNITID`: character type
   - `INSTNM`: character type
   - `STABBR`: character type
   - `LONGITUD`: double type

- `LATITUDE`: double type

/1

4. Filter the `hd` dataframe for only California schools and save that to a new object called `hd_ca`. Then, copy the following code to your R script and run it to save a plot:

```
png(file.path(plot_dir, 'ca_univs.png'))
ggplot(hd_ca, aes(x = LATITUDE, y = LONGITUD)) +
  geom_point() +
  theme_minimal() +
  coord_fixed(ratio = 1.5)
dev.off()
```

/1

5. Finally, export your `hd_ca` dataframe to a CSV file called `hd2019_ca.csv` inside `file_dir`.

Part III: Git & GitHub

/1

1. Now, let's get some practice with the git/GitHub workflow! Since `ps3_lastname_firstname` is a git repository, you can run git commands in this directory. In your **RStudio Terminal**, run the command to check the current status of the repository. Write the command you used below and answer the following question:

```
# Command to check status of repository
pwd
cd desktop/ps3_nazario_paula
pwd
git status

# Looking at the output from your command, what heading are your files listed under?
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .DS_Store
        data/
        output/
        problemset3.Rmd
        ps3_nazario_paula.Rproj
        scripts/

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

/1

2. Add the contents of the `data`, `output`, and `scripts` folders to the staging area. Write the command you used below and answer the following question:

```
# Command to add ‘data‘, ‘output‘, and ‘scripts‘ files
git add data output scripts

# Check the status of your repository again - what heading are these files listed under now?
git status
# I received following after entering ’git status’:
On branch main
Your branch is up to date with ’origin/main’.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   data/HD2019.zip
        new file:   data/hd2019.csv
        new file:   output/.DS_Store
        new file:   output/files/hd2019_ca.csv
        new file:   output/plots/ca_univs.png
        new file:   scripts/ps3_script.R

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .DS_Store
        problemset3.Rmd
```

/1

3. Commit your changes with a message of your choice and write the command you used here:

```
# Command to commit changes
git commit -m "Commit Changes Using Git"

# I received the following after using the "git commit -m" command:
[main eb34076] Commit Changes Using Git
 Committer: Paula Nazario <paulanazario@Paulas-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 6 files changed, 7343 insertions(+)
 create mode 100644 data/HD2019.zip
 create mode 100644 data/hd2019.csv
 create mode 100644 output/.DS_Store
 create mode 100644 output/files/hd2019_ca.csv
```

```
 create mode 100644 output/plots/ca_univs.png
 create mode 100644 scripts/ps3_script.R
```

/1

4. Check the status of your repository again and you should notice you don't see the committed files listed anymore. At this point, make sure you've knitted this `problemset3.Rmd` file at least once. You should see the `.Rmd`, `.md`, and `.pdf` versions of this file listed when you checked the status. (*Note*: If you look at the YAML header of this `.Rmd`, we had specified to keep the intermediary `.md` file, which is why this exists in addition to the usual `.pdf` when you knitted)

   Let's say we don't want to track this intermediary `.md` file. Add this file to the `.gitignore` file. (In addition, if you're a Mac user and see the hidden file `.DS_Store` listed, add that to `.gitignore` as well. This file gets automatically created when you view the folder in Finder and does not need to be tracked.)

git status echo "problemset3.md" > .gitignore git status echo ".DS_Store" » .gitignore git status

/1.5

5. Check the status of your repository again – what heading is the `.gitignore` file listed under? Also run the git command to show the changes made to `.gitignore` and copy the output below:

```
# What heading is `.gitignore` listed under?
git status
# I got the following after using 'git status' command:
no changes added to commit (use "git add" and/or "git commit -a")
Paulas-MacBook-Pro:ps3_nazario_paula paulanazario$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .gitignore
        modified:   scripts/ps3_script.R

# Command to show changes made to `.gitignore`
git status
git diff .gitignore

# Output
no changes added to commit (use "git add" and/or "git commit -a")
Paulas-MacBook-Pro:ps3_nazario_paula paulanazario$ git diff .gitignore
diff --git a/.gitignore b/.gitignore
index fae8299..c1e8cd8 100644
--- a/.gitignore
+++ b/.gitignore
@@ -1,39 +1,2 @@
-# History files
-.Rhistory
-.Rapp.history
-
```

```
-# Session Data files
-.RData
-
-# User-specific files
-.Ruserdata
-
-# Example code in package build process
-*-Ex.R
```

/1

6. Now, add your `.gitignore` file as well as the `ps3_lastname_firstname.Rproj` file that should have been created when you created the **RStudio Project**. Commit these files with a message and copy the commands you used below:

```
# Command to add ‘.gitignore‘ and ‘ps3_lastname_firstname.Rproj‘
git status
git add .gitignore ps3_nazario_paula.Rproj

# Command to commit changes
git commit -m "Commit Changes Using Git Part 2"
```

/0.5

7. Check the history of commits and write the command you used below. You should see your 2 commits from above as well as an initial commit that was made for you when you initialized the GitHub repository with a `README` and `.gitignore` in the first step.

```
# Command to check the history of commits
git log

# I got the following after using the 'git log' command:
no changes added to commit (use "git add" and/or "git commit -a")
Paulas-MacBook-Pro:ps3_nazario_paula paulanazario$ git log
commit cd33ec3c585936015badee4cf9744e3b7b8feb6b (HEAD -> main)
Author: Paula Nazario <paulanazario@Paulas-MacBook-Pro.local>
Date:   Tue Jan 25 16:54:54 2022 -0800

    Commit Changes Using Git Part 2

commit eb340763135405ad091b380ba8f8ce97e37cff9e
Author: Paula Nazario <paulanazario@Paulas-MacBook-Pro.local>
Date:   Tue Jan 25 15:57:29 2022 -0800

    Commit Changes Using Git

commit 2f890f8df52e3d5ce62ed2e22829c70bd176d691 (origin/main, origin/HEAD)
Author: nazapa <90589466+nazapa@users.noreply.github.com>
Date:   Sun Jan 23 22:00:05 2022 -0800

    Initial commit
```

/1

8. It's time to push your changes to the remote repository! But before you do, first check what remote repository you're connected to. Specify the option that will show more detailed info about the remote.

```
# Command to check the remote
git remote
git remote -v

# Output
Paulas-MacBook-Pro:ps3_nazario_paula paulanazario$ git remote
origin
origin  https://github.com/anyone-can-cook/ps3_nazario_paula (fetch)
origin  https://github.com/anyone-can-cook/ps3_nazario_paula (push)
```

/0.5

9. Now, push your changes and you should be able to see them on the GitHub repository in your browser.

```
# Command to push to remote
git push
```

Part IV: GitHub issue

/2

- Go to the class repository and create a new issue.
- You can either:
  - Ask a question that you have about this problem set or the course in general. Make sure to assign the instructors (@ozanj, @lizachavac, @briannawright135) and mention your team (e.g., @anyone-can-cook/your_team_name).
  - Share something you learned from this problem set or the course. Please mention your team (e.g., @anyone-can-cook/your_team_name).
- You are also required to respond to at least one issue posted by another student.
- Paste the url to your issue here: https://github.com/anyone-can-cook/rclass2_w22_student_issues/issues/83
- Paste the url to the issue you responded to here: https://github.com/anyone-can-cook/rclass2_w22_student_issues/issues/77

Knit to pdf and submit problem set

Knit to pdf by clicking the "Knit" button near the top of your RStudio window (icon with blue yarn ball) or drop down and select "Knit to PDF"

You will submit this problem set by pushing it to your repository. Follow the same steps you used above to add, commit, and push both the `.Rmd` and `.pdf` files.

Submitting problem set using the add, commit, and push functions

git status git add ps3_nazario_paula.Rmd ps3_nazario_paula.pdf git commit -m “Git Command Part 3” git push