Common Git Commands for Aspiring Data Scientists

July 17, 2020

In a series of a vidoes that I created on getting started with Git and Github for Data Science, I covered a number of Git commands in the final video of the series. You can click here to access the Mac version of the series, or click here to access the Windows version.

However, for an aspiring data scientist who is just starting, it can be overwhelming to be working with a number of different commands, so I am creating here a list of the Git commands that I cover in my series, so that it can serve as a reference to anyone who is aspiring to become a data scientist until they are comfortable with Git and Github.

git status

***Usage*: git status**

A Git command that displays the status of a Git repository. It displays what files in a repository are being tracked, what files were added, what files underwent changes, what changes have been staged, and whether the local copy of the repository is in-sync with the remote repository or not.

git diff

***Usage*: git diff <filename_1> <filename_2> . . .**

A Git command that displays what changes were made to files. For each file, it would display deletions in red and additions in green. You can run the command for more than one file at the same time. See *Usage* above.

git add

***Usage*: git add <filename_1> <filename_2> . . .**

A Git command that represents the first step in saving changes to a repository. This command stages or identifies the files or the changes that you wish to save. Just like the git diff command, you can stage more than one file at the same time. See *Usage* above.

git commit

***Usage*: git commit -m "<message>"**

A Git command that saves any staged changes. This is typical run after a git add command. It also displays a summary of the changes that get saved, for example, the number of files, and how many additions or deletions in the files that are committed.

git reset

***Usage*: git reset**

A powerful Git command that essentially undoes changes to a Git repository. When run alone, the command is used to unstage changes that were staged using the git add command.

git reset

***Usage*: git reset —hard HEAD~<# of commits>**

The command can also be used to delete commits from the repository tree when used with the hard option and the number of commits to be deleted. See *Usage* above.

git log

***Usage*: git log**

A Git commands that displays a log of the commits made so far or the depth of the repository tree since each commit represents a node in the repository tree.