Common Data Science Tools

* Interacting With Computers: The Command Line Interface
  + Whenever we use a computer, we need a way to "talk to it" so that we can tell it what to do.
  + The way you're probably most familiar with is through a graphical user interface (GUI).
  + GUI (often pronounced as "gooey") describes any system that allows you to interact with information by clicking, dragging, or otherwise arranging and selecting objects, typically using a mouse and a keyboard.
  + GUIs are visually appealing and easy to navigate.
* Navigating in the CLI: Paths
  + Every file or folder in a file system can be read, written, and deleted by referencing its position inside the file system.
  + When we talk about the position of a file or a folder in a file system, we refer to its "path." There are a couple of different paths we can use to refer to a file: an absolute path and a relative path.
    - Absolute path: The specific location of a file or folder from the root directory, typically shown as /. The root directory is the starting point from which all other folders are defined. It is not normally the same as your home directory, which is usually found at /Users/[Your Username].
    - Relative path: A reference to a file or folder relative to your current position or the present working directory (pwd).
* In the CLI:
  + pwd (which stands for "print working directory") shows you the absolute path of your current directory.
  + cd is the command to "change directory."
    - By itself, cd takes you back to the home directory.
    - cd + path will navigate from wherever you are to the new path. So, if you're in folder A and want to navigate into folder B, type cd B.
    - cd .. will navigate you one directory back up the path.
    - To indicate the absolute path (and jump directly to any directory without navigating through each folder), add a / to the beginning of the path. Without a /, the CLI will interpret the path as looking for a subfolder.
  + ~ (tilde) is a reference to the home directory, so cd ~ takes you "home," and ~/directoryname takes you to a directory branching off of the home directory.
* Below are some examples that use relative and absolute paths to complete the same action:
  + Our present location is /folder/subfolder and now we want to change our directory to /folder. In other words we want to navigate up a level
    - Using a relative path: cd..
    - Using an absolute path: cd /folder
  + Our present location is /folder/subfolder and we want to switch our location to /folder/othersubfolder. IN other words, we want to change the directory ot a different subfolder
    - Using an absolute path: cd / folder/othersubfolder.
    - *Note: “Directory” is an important term that is often used interchangeably with “folder.” Although they are not always the same, when we say “navigate to your project directory,” think of it as “navigate to your project folder.”*
  + Useful Commands in the CLI
    - [Cheat Sheet](https://www.cheatography.com/davechild/cheat-sheets/linux-command-line/#downloadsr)
    - The following are some useful commands to use within the CLI.
      * To stop a script that's currently running, press ctrl-c. (Try this out: Type ping 127.0.0.1, which basically sends a message to your computer asking if it's awake. It will keep pinging until you type ctrl-c.)
      * To open a file, use open and then the filename (or path, if you're not already in that folder): open file.txt or open subfolder/file.txt.
      * ls lists the files and directories in the current folder (or any folder if you list the path as well: ls /Applications).
      * ls -a also shows the files that are currently hidden.
      * Create a new file with touch filename.
      * Remove a file with rm filename.
      * Create a new directory with mkdir foldername.
      * Remove a folder directory with rm -r foldername.
        + The -r tells the computer to remove the folder AND any files/folders it contains.
    - *Note: Be careful using rm or rm -r. When you delete a file using the CLI, it cannot be restored! In other words, the file you remove doesn’t go into the "trash" directory, it is actually removed from the computer’s memory.*
* Text Editors
  + There are limits to what you can accomplish directly through the command line. Programers and data scientiss who want to complex scripts typically compose their code using text editors.
  + So what exactly is a text editor?
    - It’s an interface for creating, viewing, and modifying text files
  + Text editors are handy programs. They can:
    - Open a file or directory.
    - Understand context.
    - Offer context-sensitive help.
    - Highlight errors or bad practices in your code.
    - Adapt to different file formats.
    - Provide syntax highlighting.
  + Types of Text Files
    - Text editors can understand, interpret, and edit a wide variety of text files. Examples include:
      * Plaintext (.txt).
      * Markdown (.md).
      * CSV (.csv).
    - Text editors allow you to write in various programming languages, each of which have their own sets of rules, keywords, operators, and syntax. With text editors, you can compose scripts for languages such as:
      * Python.
      * HTML/CSS.
      * JavaScript.
      * Ruby.
      * SQL.
* Open Source Tools
  + An open-source tool is one whose source code is freely distributed to the public. Anyone can download, install, run, make copies, or suggest improvements to the code.
  + Open-source tools used by most data scientists are **Git** and **GitHub**. Git is a framework used to help simplify version control, while GitHub is an open-source platform built on Git in order to make it easier to use. Programmers and data scientists use GitHub to contribute and collaborate in an organized way.
  + We'll become more familiar with Git and GitHub later in the course. For now, think of them as ways to organize and share files. With Git and GitHub, file history is always preserved, making it easier to undo specific changes made at specific points in time.
  + Another go-to, open-source tool for data scientists is Jupyter Notebook. Found in the Anaconda suite of tools, Jupyter Notebook allows you to create your code in organized "Notebooks," which can intersperse code chunks (that can run individually in any order) with markdown text, allowing for easy annotation and clear, organized scripts.
    - Jupyter is an integrated development environment (IDE). An IDE is like a super-charged text editor that makes it easy to write and execute scripts.
    - Throughout this course, we'll present materials in Jupyter Notebooks running Python 3.6 to allow for easy demonstration of code, as well as to offer well-formatted explanatory text, images, and videos.