conda Environment Basics

DATA ANALYTICS BOOTCAMP

Objectives

1. Understand what a conda environment is
2. Create a new environment from a yaml file
3. Learn commands associated with conda environments

What is a conda environment?

From the conda docs:

"A conda environment is a directory that contains a specific collection of conda packages that you have installed. For example, you may have one environment with NumPy 1.7 and its dependencies, and another environment with NumPy 1.6 for legacy testing. If you change one environment, your other environments are not affected. You can easily activate or deactivate environments, which is how you switch between them. You can also share your environment with someone by giving them a copy of your environment.yaml file."

tl;dr

A conda environment is the **specific set of conda packages, and their specific versions,** you have installed.

Why do we need to think about environments?

- Libraries release new versions, which add, deprecate, or remove functions
  - This can impact can significantly impact how code runs, and, in the case of removing a function, may result in code not running at all
- Particular versions of different libraries/packages may not work well together
- When sharing code with others, if they have a different environment, code that was working beautifully for you may not function for them unless they create an environment that matches yours

Package Managers

The two most common package managers for installing third party packages you'll use are **conda** and **pip**

- **conda:** A package manager designed for use by data analysts/data scientists that installs packages from Anaconda's repository. When installing, will analyze the current environment and ensure that there are no conflicts between package versions.
- **pip:** The standard package manager for Python. Installs packages from the Python Package Index, aka PyPI, aka the Cheese Shop, the official third-party repository for Python.
- It is recommended that you ***install packages using conda whenever possible*** and use pip only when a package is not available from conda. You can learn more here: https://www.anaconda.com/blog/using-pip-in-a-conda-environment

More About conda

conda is not just a package manager, but is also an environment manager, meaning that you can create separate environments containing files, packages, their dependencies, and their own versions of the Python interpreter.

This serves two purposes:

- Isolates your projects
- Makes it easier to share your work and allows for reproducibility

Learn more about why using environments is a good idea:

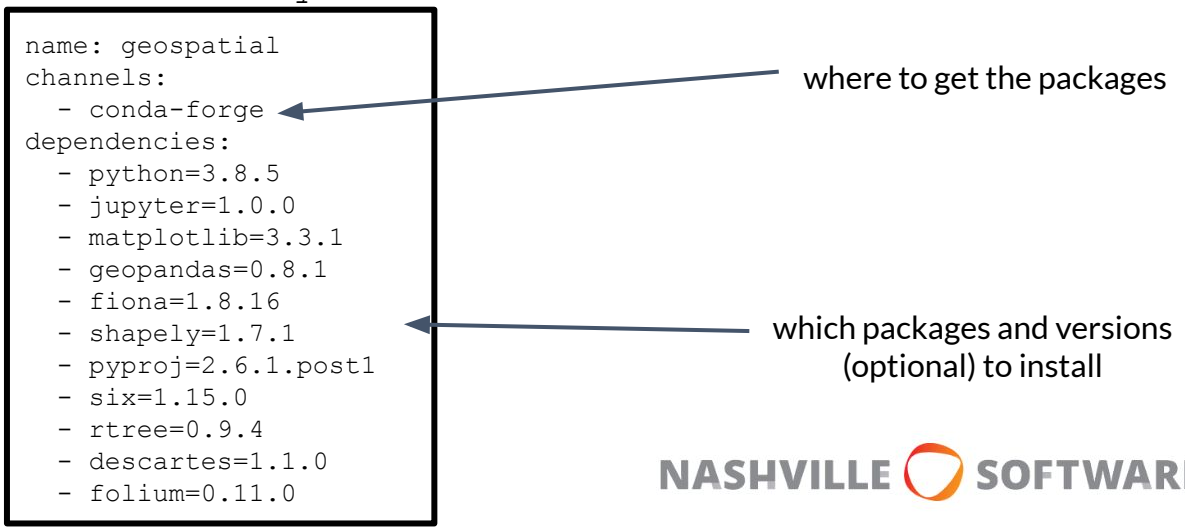
- https://www.freecodecamp.org/news/why-you-need-python-environments-and-how-to-manage-them-with-conda-85f155f4353c/
- https://gavh3.github.io/posts/why-you-need-conda-envs/

Creating a conda environment

There are two ways to create conda environments:

1. From scratch
2. Using a YAML file (this is what we will do)
    a. The YAML file contains a list of packages and their versions that should be installed

`environment.yaml`



NASHVILLE SOFTWARE SCHOOL

Create a conda environment - YAML file

1. Open Anaconda Navigator
2. Launch Powershell Prompt
3. Navigate to the folder containing your YAML file (likely the data folder within your Python folder)
    - Useful commands to know for Windows:
        - **`cd`** to change directory
            - **`cd ..`** will take you up a level in your file directory
            - **`cd foldername`** will take you into that folder
        - **`dir`** to get a list of available folders to cd into
    - Useful commands to know for Mac:
        - **`cd`** to change directory
        - **`ls`** to get a list of available folders to cd into
4. Once you are in the correct folder, use this command:
    - `conda env create -f environment.yaml`

Learn more about Unix commands: https://astrobiomike.github.io/unix/unix-intro

NASHVILLE SOFTWARE SCHOOL

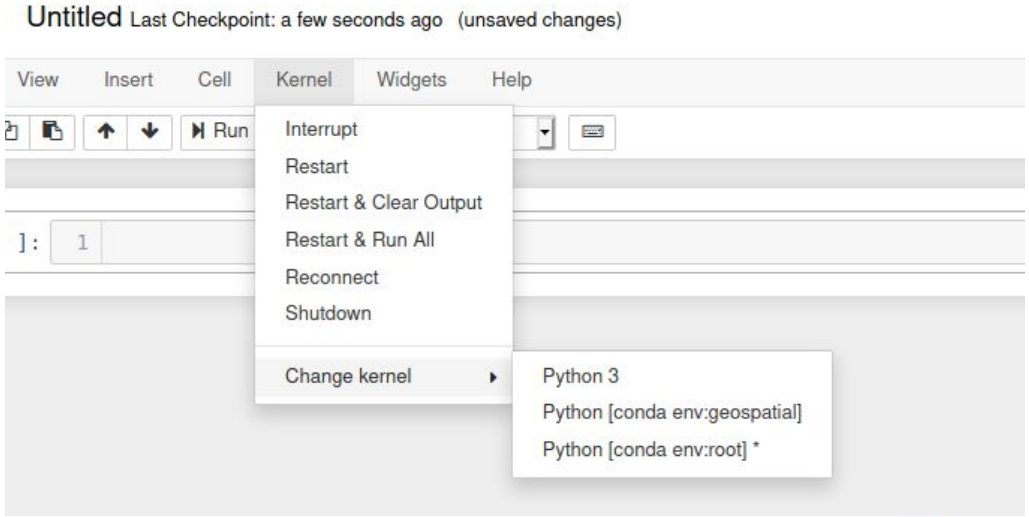
Commands related to conda environments

- `conda env list`
  - Use this to show a list of all of your conda environments
  - An asterisk (*) is place next to the active environment
- `conda activate environment_name`
  - e.g. `conda activate geospatial`
  - Use this to switch to a different environment
- `conda deactivate`
  - Use this to return to the base environment
- `jupyter notebook`
  - Once you've activated the environment you want, use this command to launch Jupyter Notebook (you must launch Jupyter Notebook this way when working in any environment other than the base)
  - **After launching Jupyter Notebook, you need to leave the Powershell Prompt open and running when you're in an environment other than your base environment**

Additional Considerations

You may also want to install the nb_conda_kernels package in your base environment, which lets you choose the kernel from within jupyter using this command (when you're in your base environment):

```
conda install nb_conda_kernels
```



Installing Packages

First, check whether the package of interest is available:

- `conda search packagename`
  - e.g. `conda search plotly`

```
(base) michael@michael-HP-Pavilion-Laptop-15-cs3xxx ~ $ conda search plotly
Loading channels: done
# Name                       Version           Build  Channel
plotly                        2.0.15  py27h139127e_0  pkgs/main
plotly                        2.0.15  py35h43bf465_0  pkgs/main
plotly                        2.0.15  py36hd032def_0  pkgs/main
plotly                         2.1.0  py27h77e25ac_0  pkgs/main
plotly                         2.1.0  py35hac5c16f_0  pkgs/main
plotly                         2.1.0  py36h56a57e5_0  pkgs/main
plotly                         2.2.2  py27hb784091_0  pkgs/main
plotly                         2.2.2  py35h6d67e38_0  pkgs/main
plotly                         2.2.2  py36hd7be514_0  pkgs/main
plotly                         2.4.0          py27_0  pkgs/main
plotly                         2.4.0          py35_0  pkgs/main
plotly                         2.4.0          py36_0  pkgs/main
plotly                         2.4.1          py27_0  pkgs/main
plotly                         2.4.1          py35_0  pkgs/main
plotly                         2.4.1          py36_0  pkgs/main
plotly                         2.5.1          py27_0  pkgs/main
```

Installing Packages

- To install a package in the **current environment**:
  - `conda install packagename`
    - e.g. `conda install plotly`
- To install a specific version of a package in the current environment:
  - `conda install packagename=versionnumber`
    - e.g. `conda install scipy=0.15.0`

Creating a YAML file

If you want to share your environment so that others can recreate it, you need to create an environment.yaml file. Two ways to do this are

1. Do it by hand by creating a .yaml file and listing the packages and versions needed
2. Create one automatically by using the conda env export command and redirecting the output into an environment.yaml file
    - `conda env export > environment.yaml`
    - Remember to cd into the folder where you want this new YAML file to be saved ***before*** running this command

Additional Resources

- Conda cheatsheet https://docs.conda.io/projects/conda/en/4.6.0/_downloads/52a95608c49671267e40c689e0bc00ca/conda-cheatsheet.pdf
- https://www.youtube.com/watch?v=1VVCd0eSkYc