PACKAGE MANAGER / DEPENDENCY INSTALLER

WHY DO WE NEED IT?

- Python is a popular language among data scientists due to its simplicity and the vast array of libraries available.
- However, managing these libraries can be a challenge!
- Two of the most popular tools for managing Python packages are “pip” and “conda”.

WHY ANACONDA AND NOT PIP?

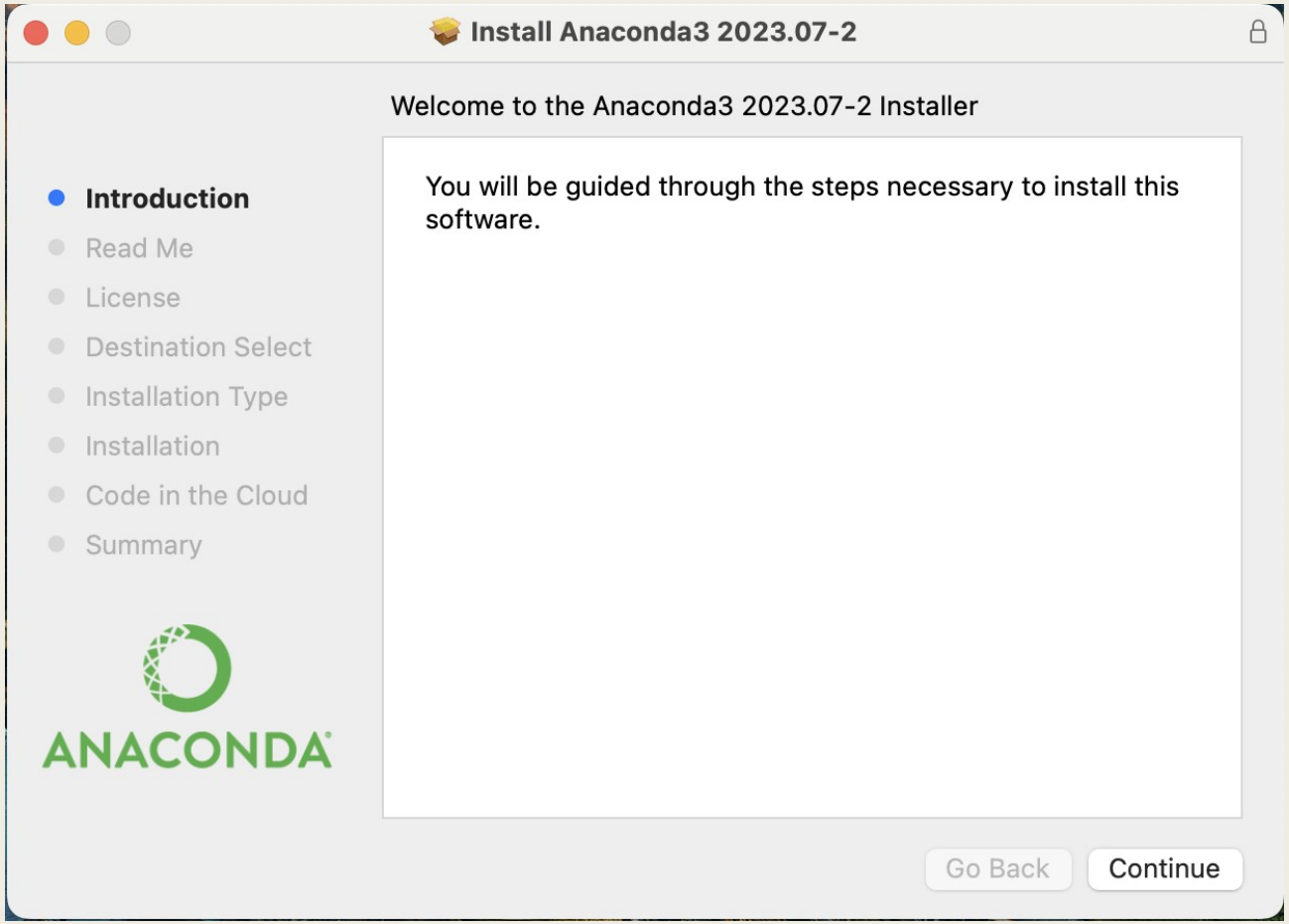
- Use pip if you are working with pure Python projects and need access to the vast array of libraries.
- Use Anaconda if you are working with projects that use multiple languages, need different versions of Python, or require complex binary dependencies.
- If you're a beginner in data science, use Anaconda;
- If you're more experienced with the command line and cannot find packages for your project (that can be outside the data science domain), then go for Python's pip
- So, for **Ease Of Use**, we use “Anaconda”. Also pip comes already installed with anaconda ᨒ

INSTALLING ANACONDA

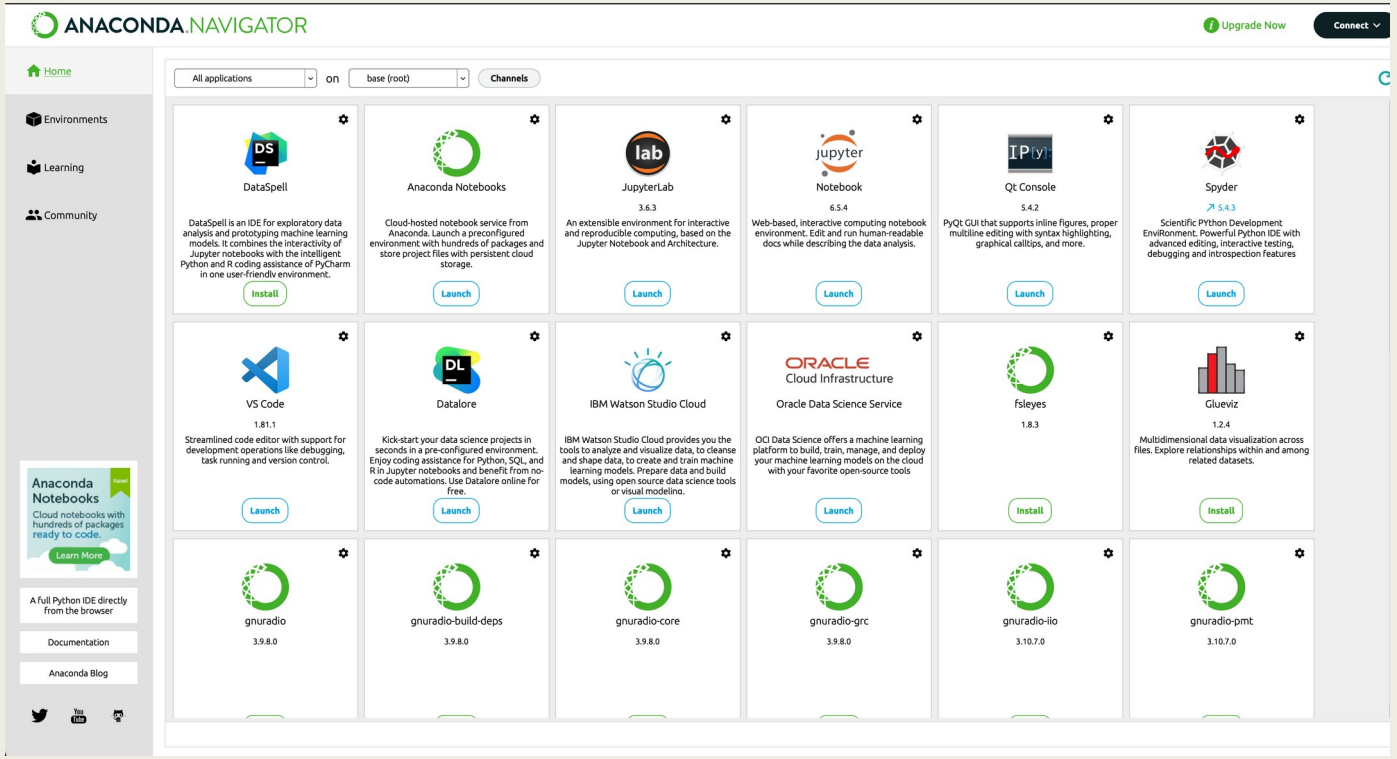
WEBSITE:

https://www.anaconda.com/products/individual

- Scroll down to near the bottom of the webpage until you see this.
- Click on the appropriate **64-bit graphical installer link** for your computer

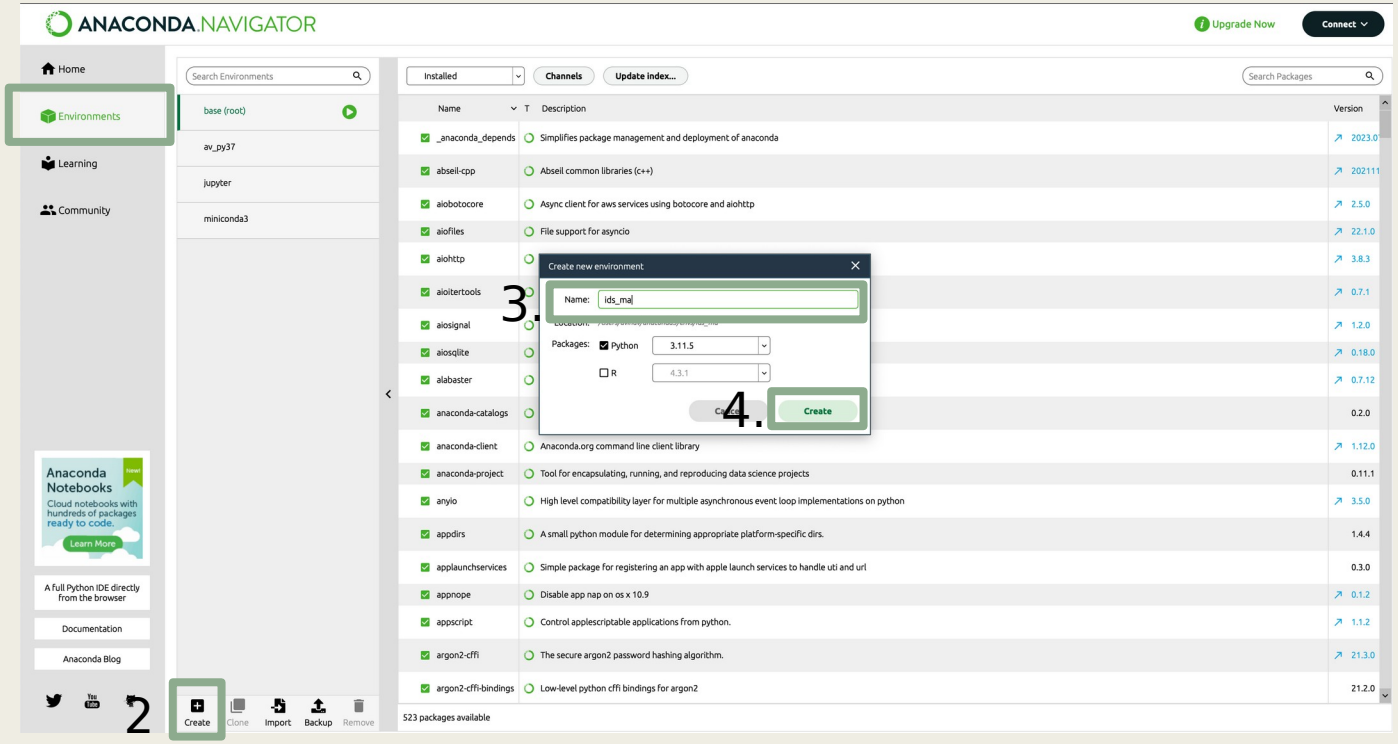


- Follow download instructions



- Locate ”Anaconda Navigator” in your applications folder
- When you open it, this is what it should look like

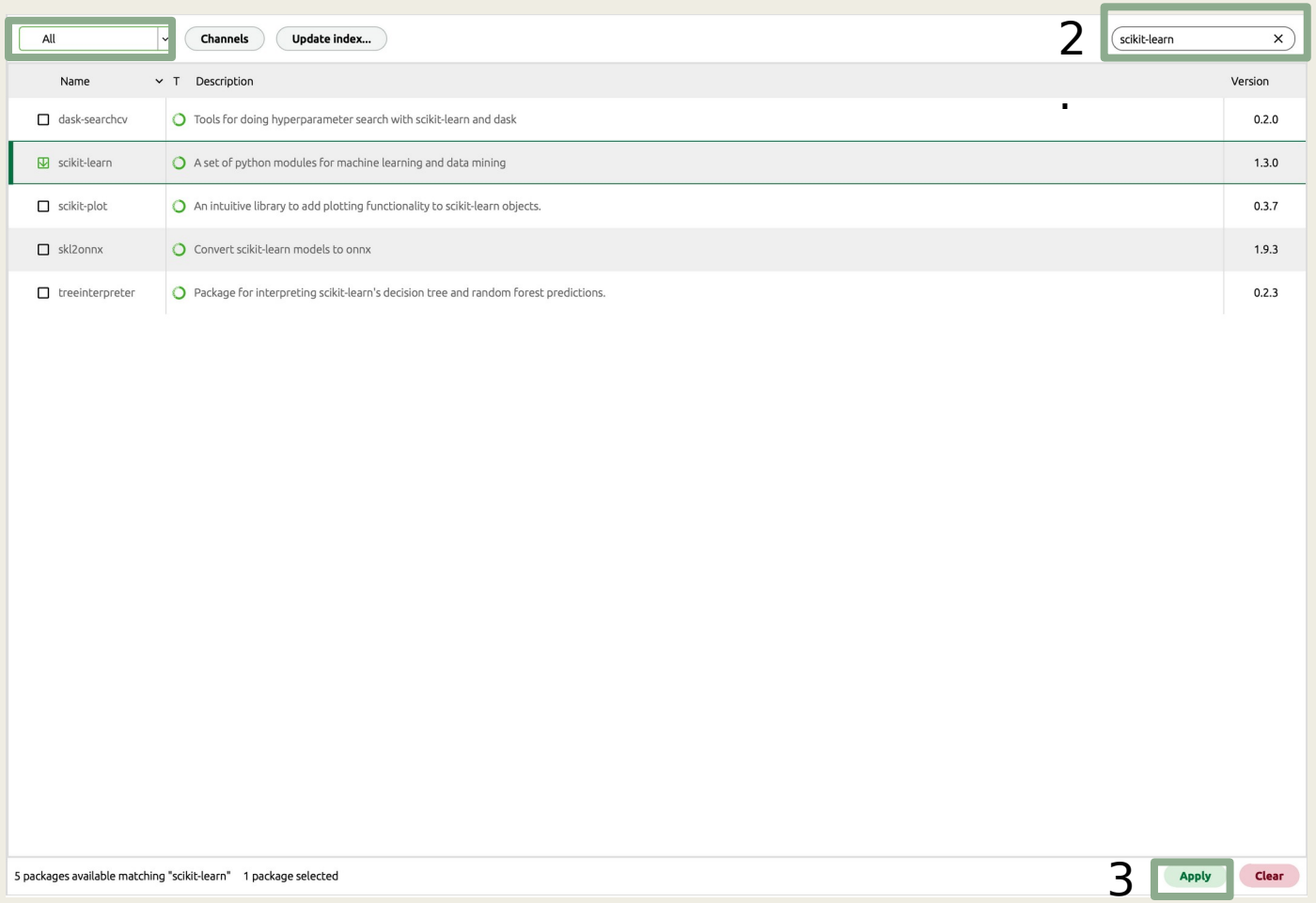
SETTING UP YOUR ENVIRONMENT



1. Go to “Environments”
2. Create a new Environment
3. Add a name for your new environment
4. Click Create and it’s done!

INSTALLING NEW PACKAGES

1.

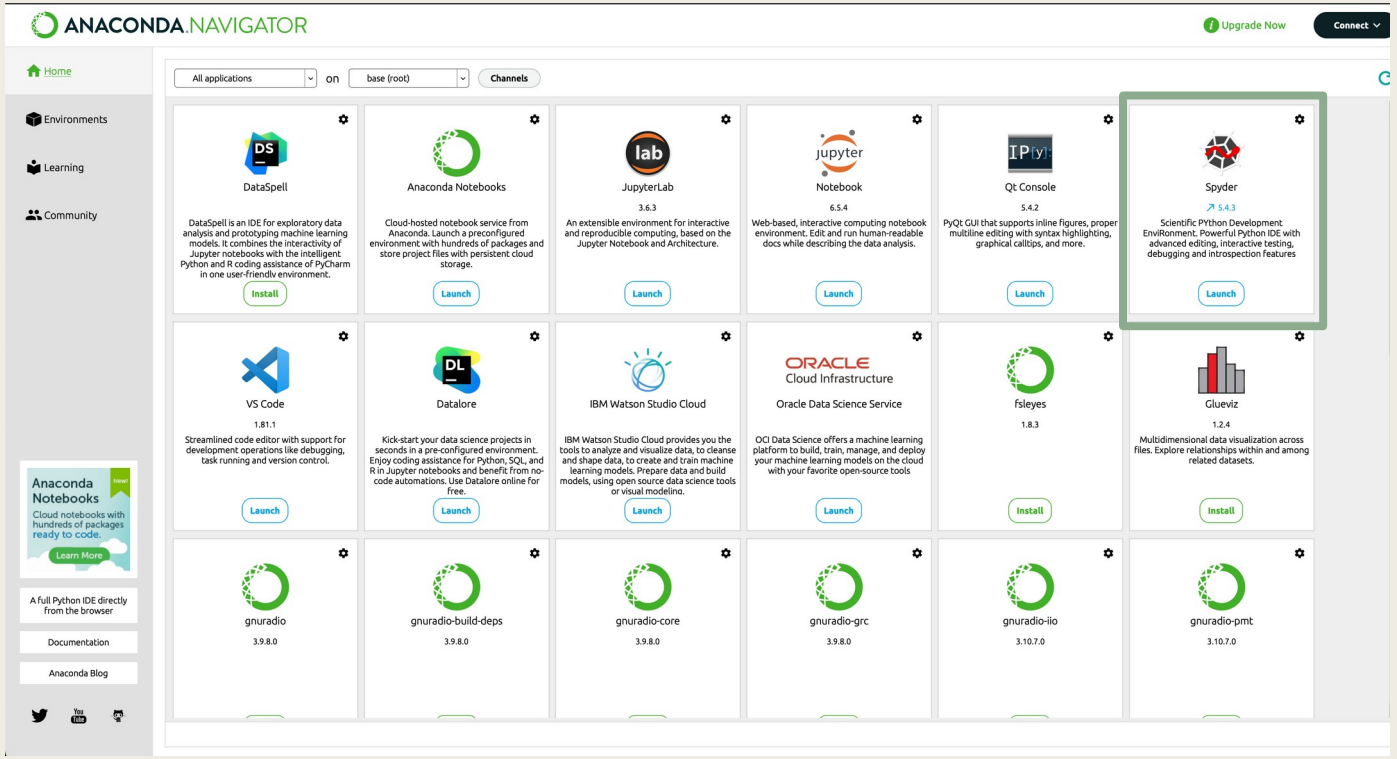


2.

3.

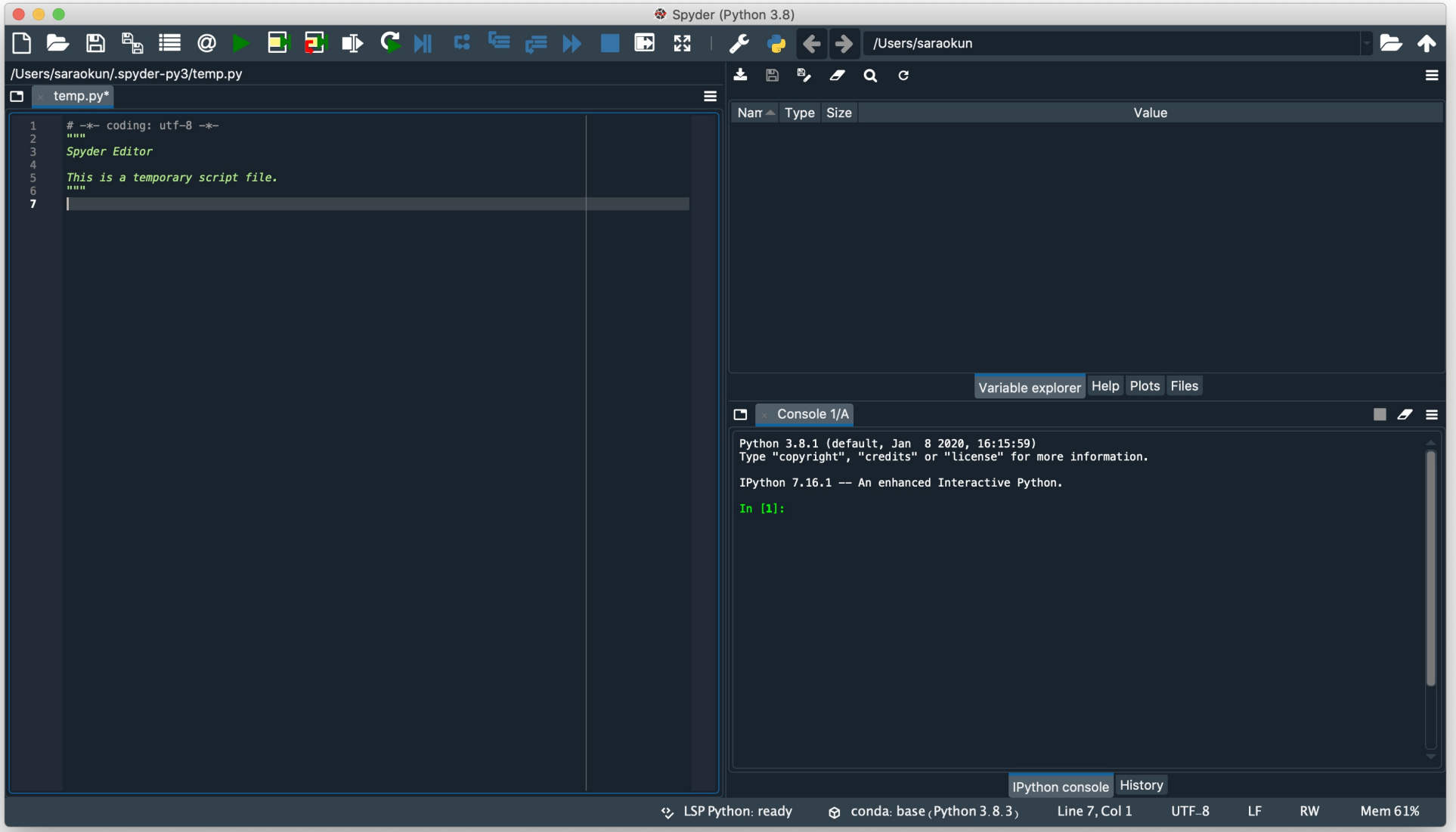
1. Set Package Discovery to all
2. Search for a new Environment, example scikit-learn
3. Click on apply
4. And viola! it installs the new package

USING IDE
(INTEGRATED DEVELOPMENT ENVIRONMENT)



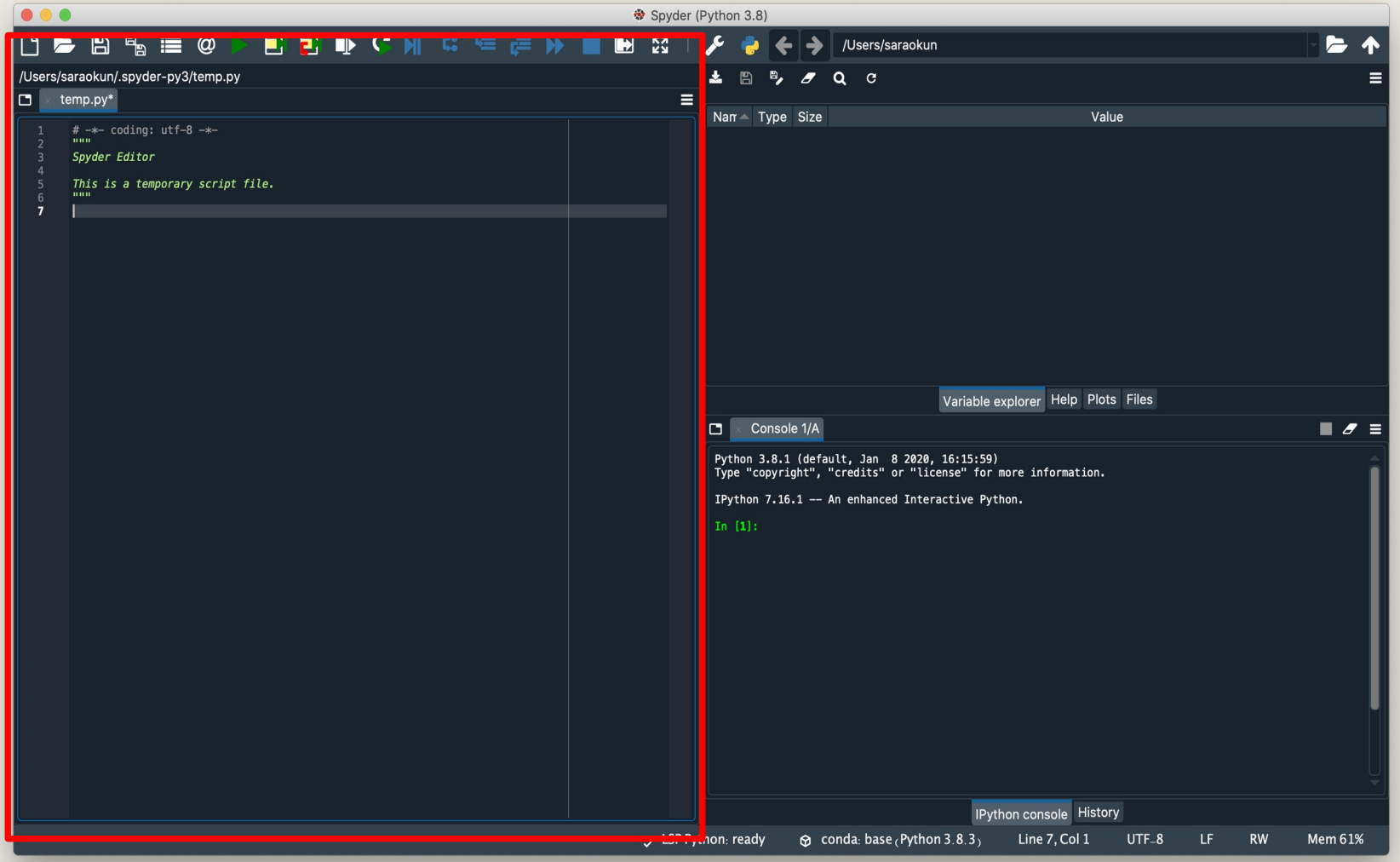
- We will be using Spyder
- If Not installed, Install Spyder(You can find it if you scroll down)
- Click “launch” to open Spyder

This is what Spyder looks like!



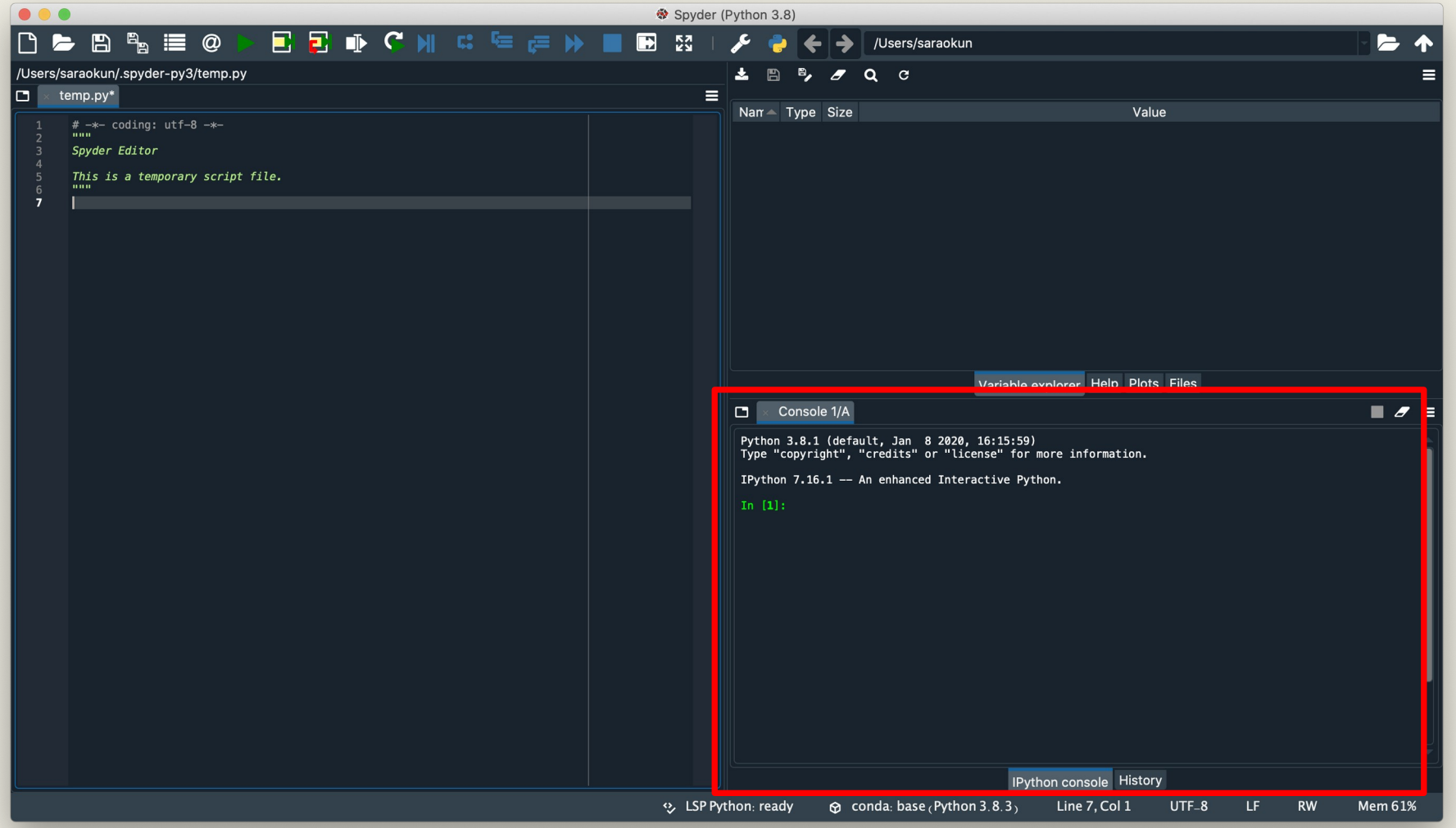
Editor

- This is where you can write python scripts and save your work as a file.



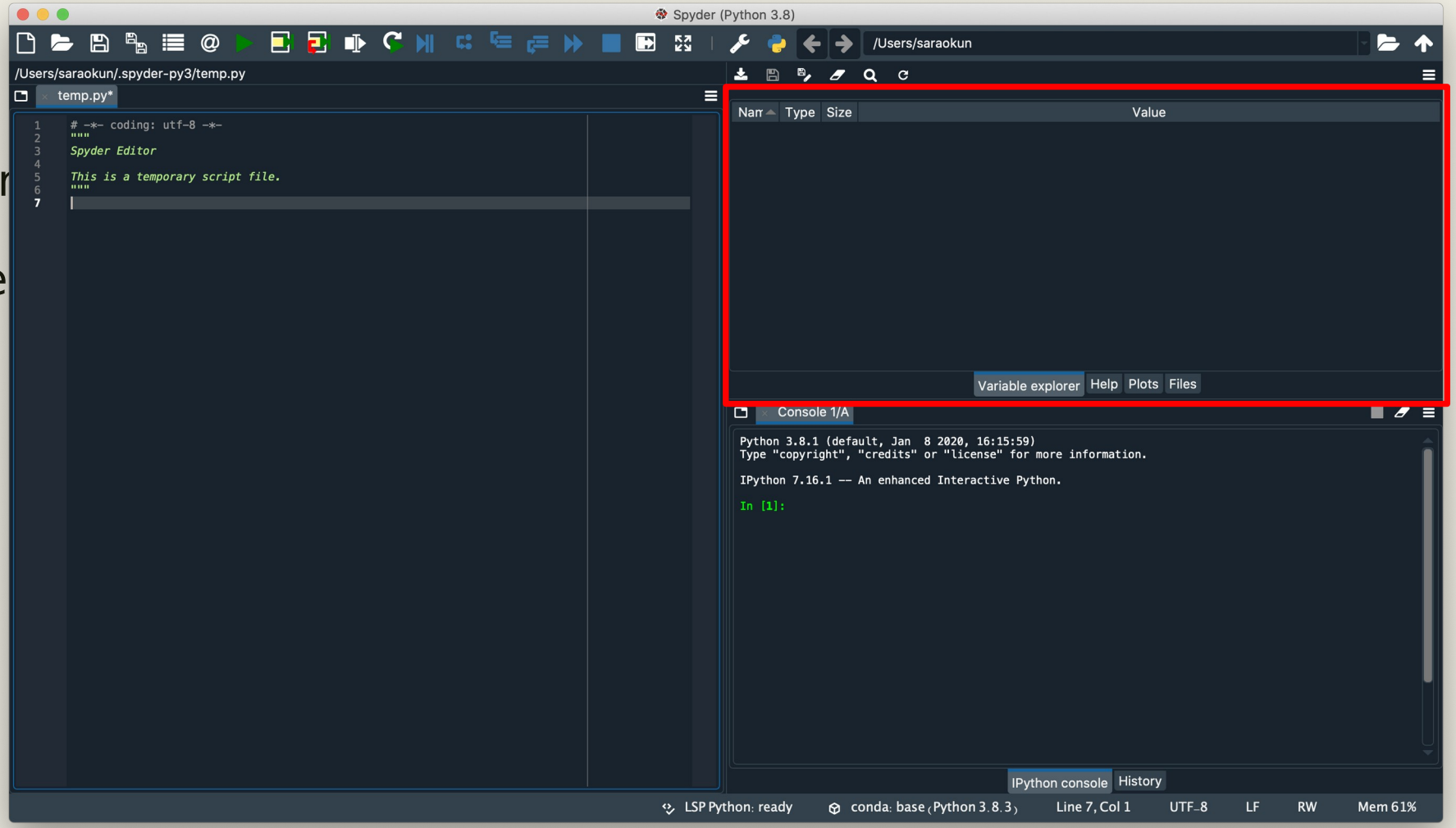
Console

- This is where you will see your code output

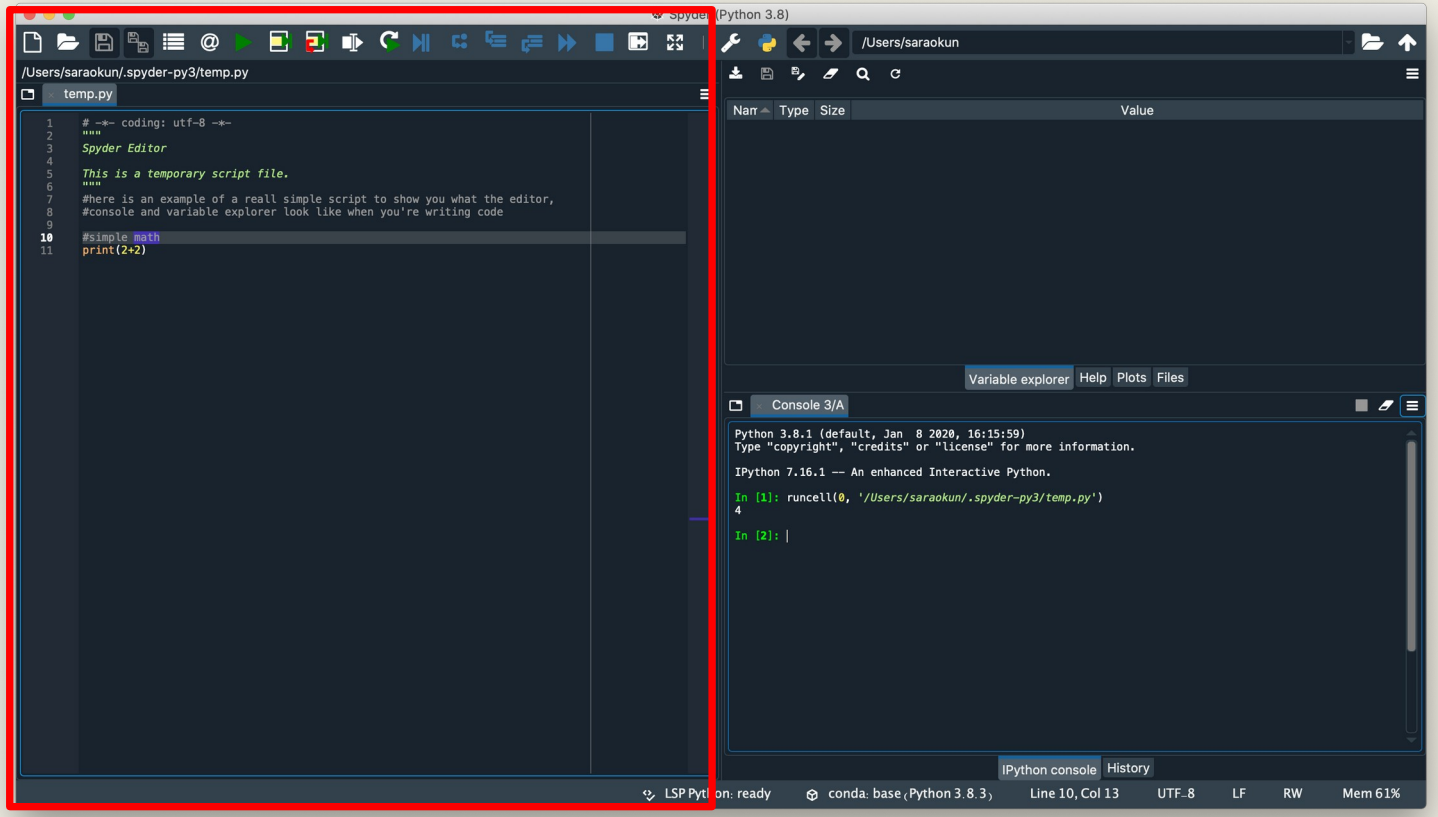


Variable explorer

- The variable explorer tab will show you all the objects you have created

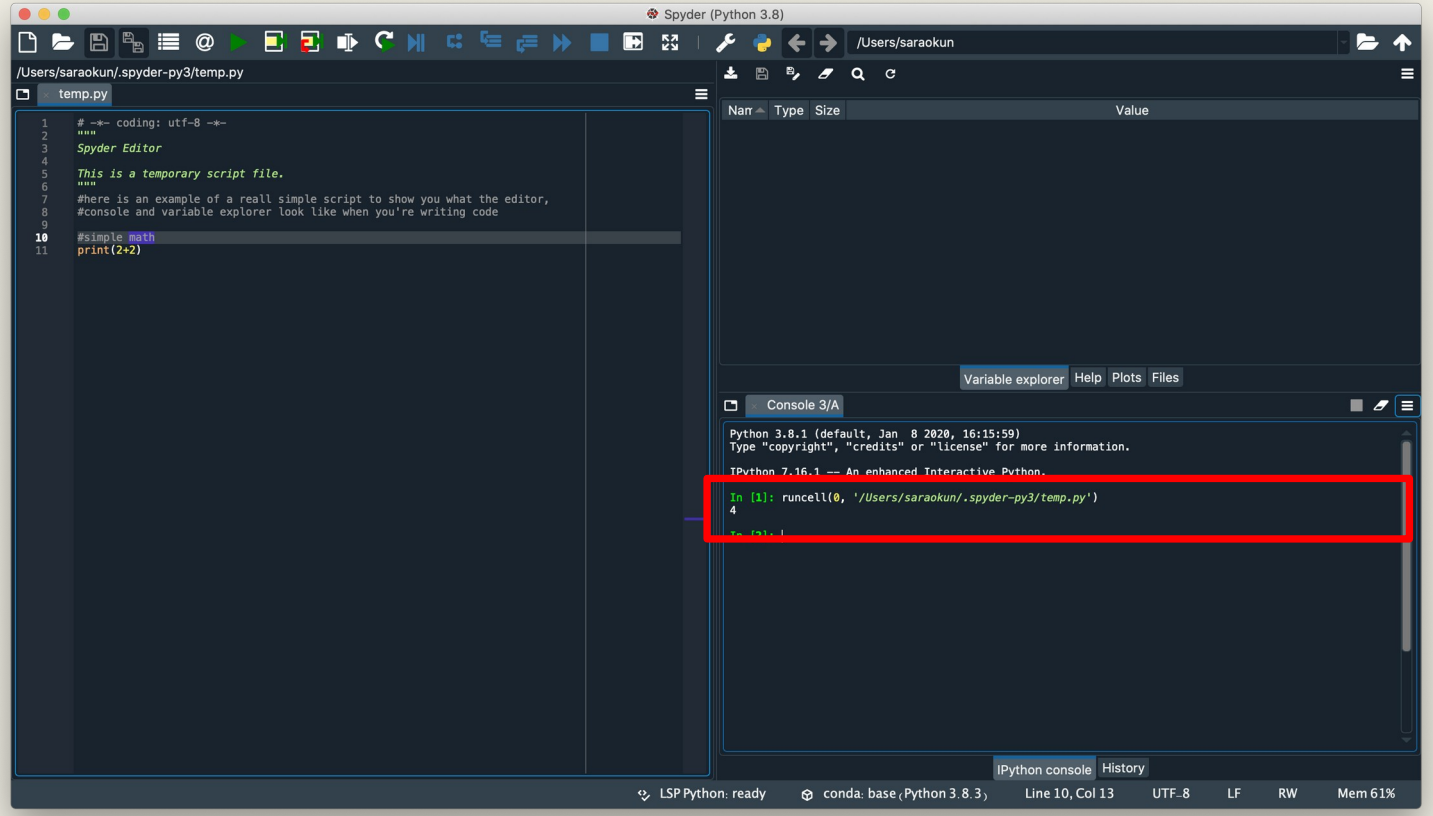


Simple example



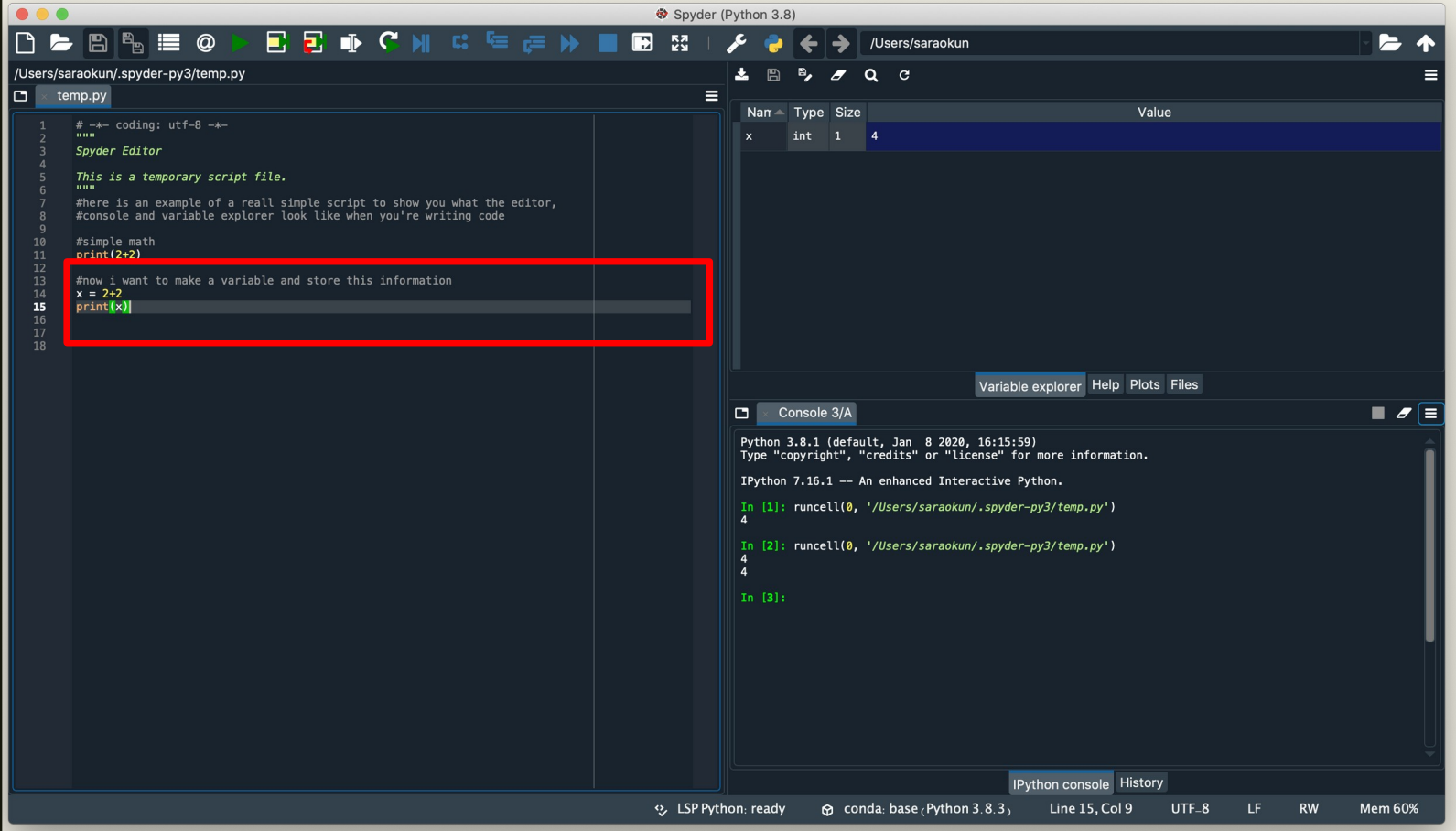
- Here is what my environment looks like when I run my code
- I am writing a script here so that I can later save all of my code

Simple example



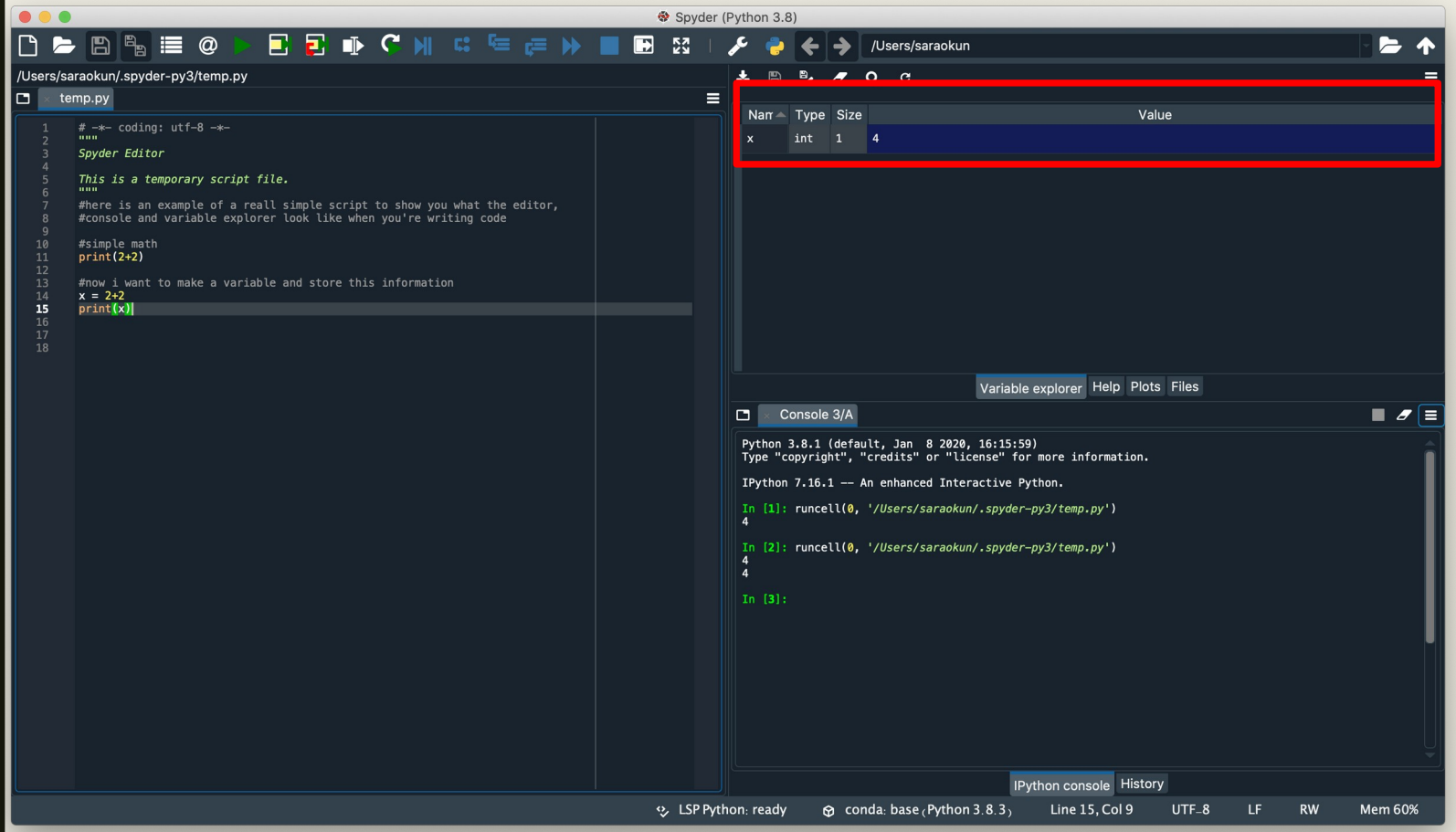
- Here is the output of my code

Variables!



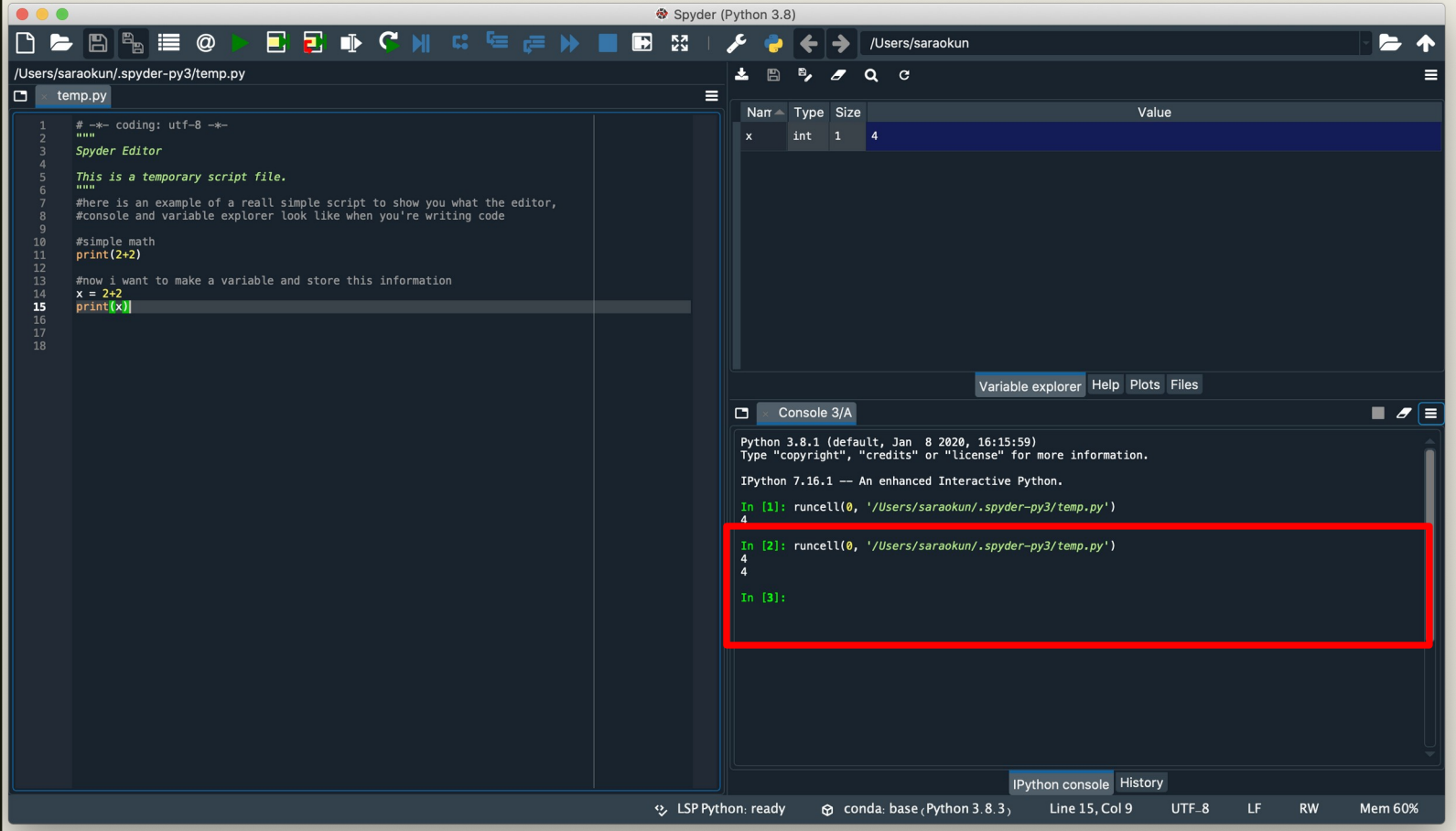
- I then created a variable in my script

Variables!



- I then created a variable in my script
- After running the script, you can now see a variable and its value in the variable explorer

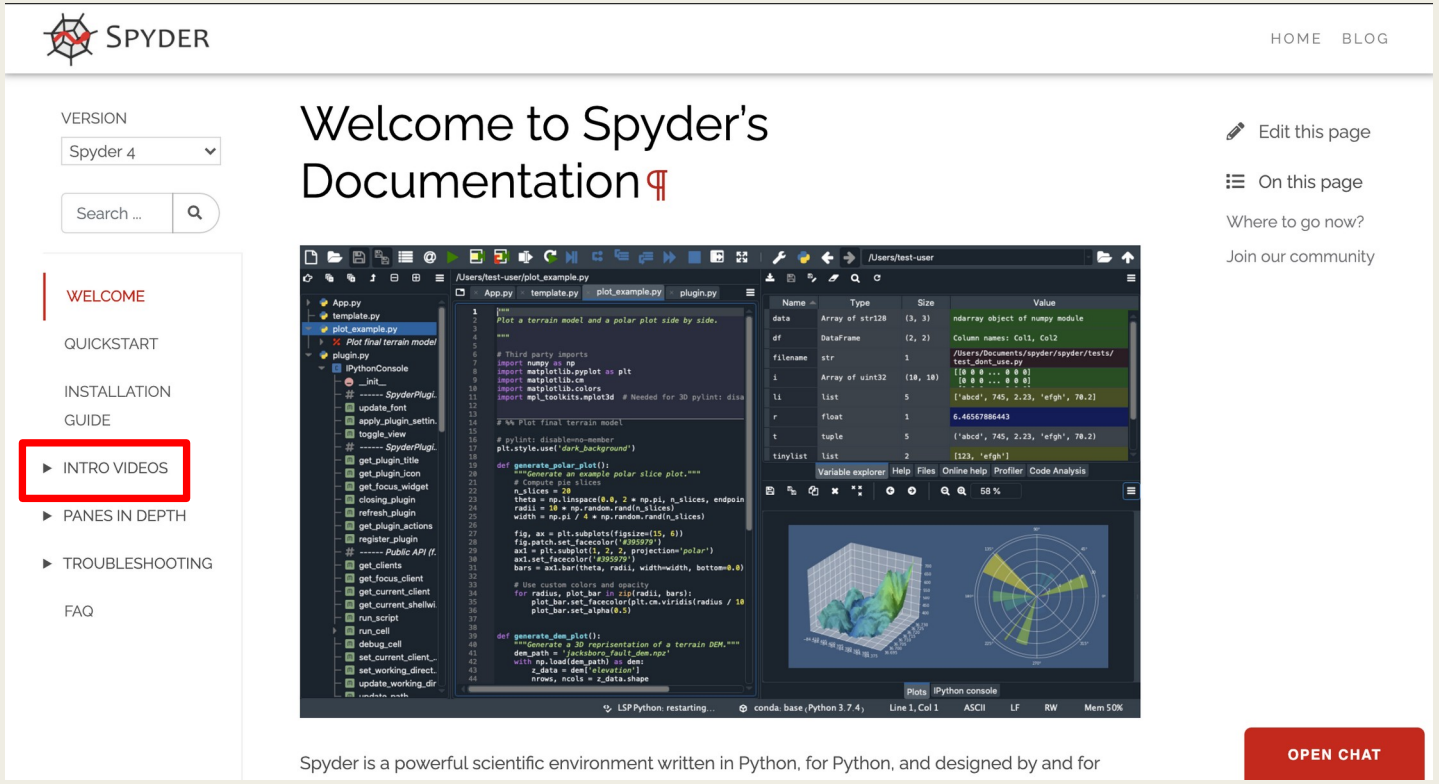
Variables!



- I then created a variable in my script
- After running the script, you can now see a variable and its value in the variable explorer
- Here is the output the second time I ran the code, after creating the variable

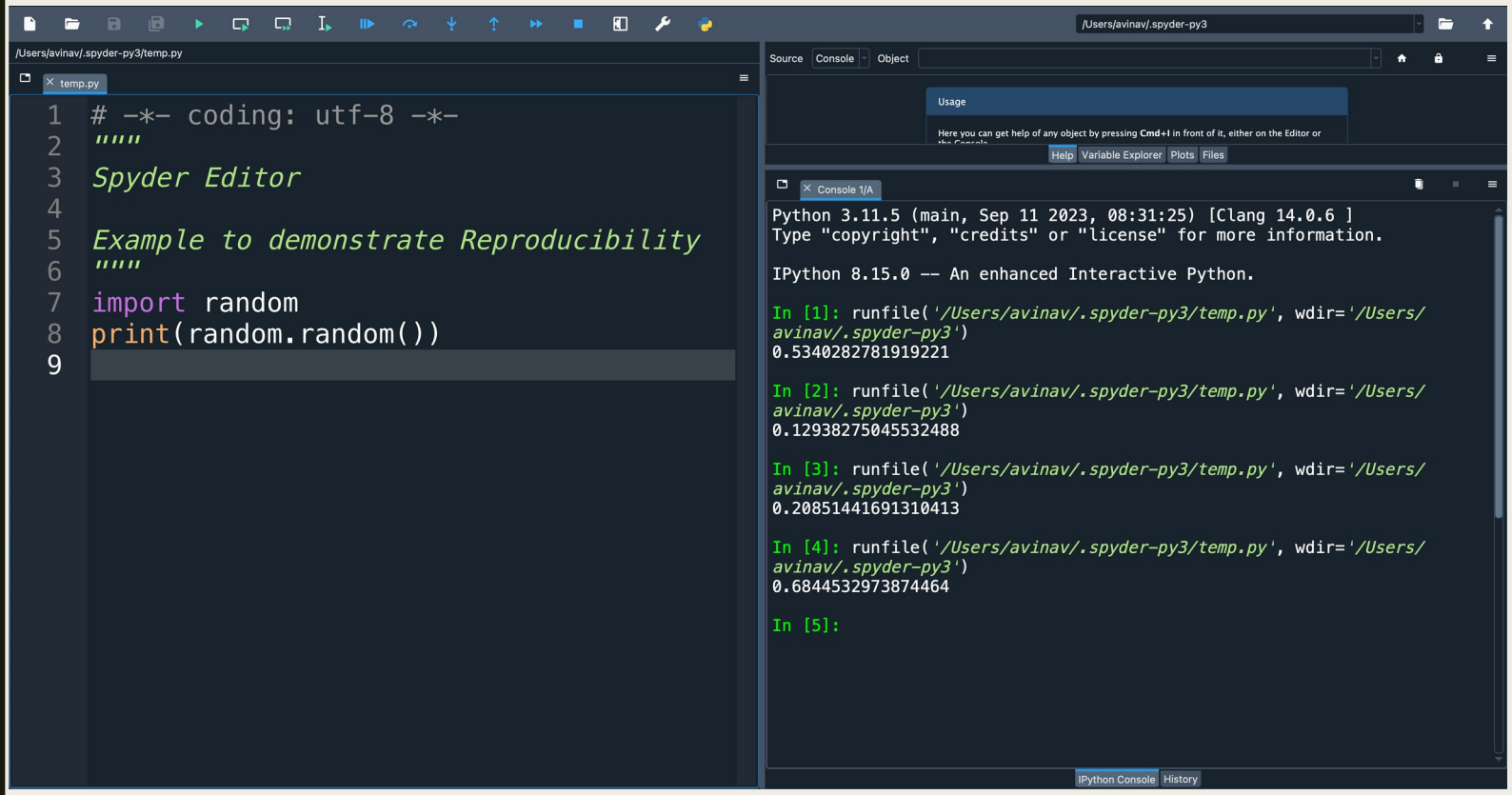
Helpful links

- https://docs.spyder-ide.org/current/index.html
  - *There are some helpful intro videos I would recommend going through if you get confused*
  - *You can find them under "Intro Videos"*



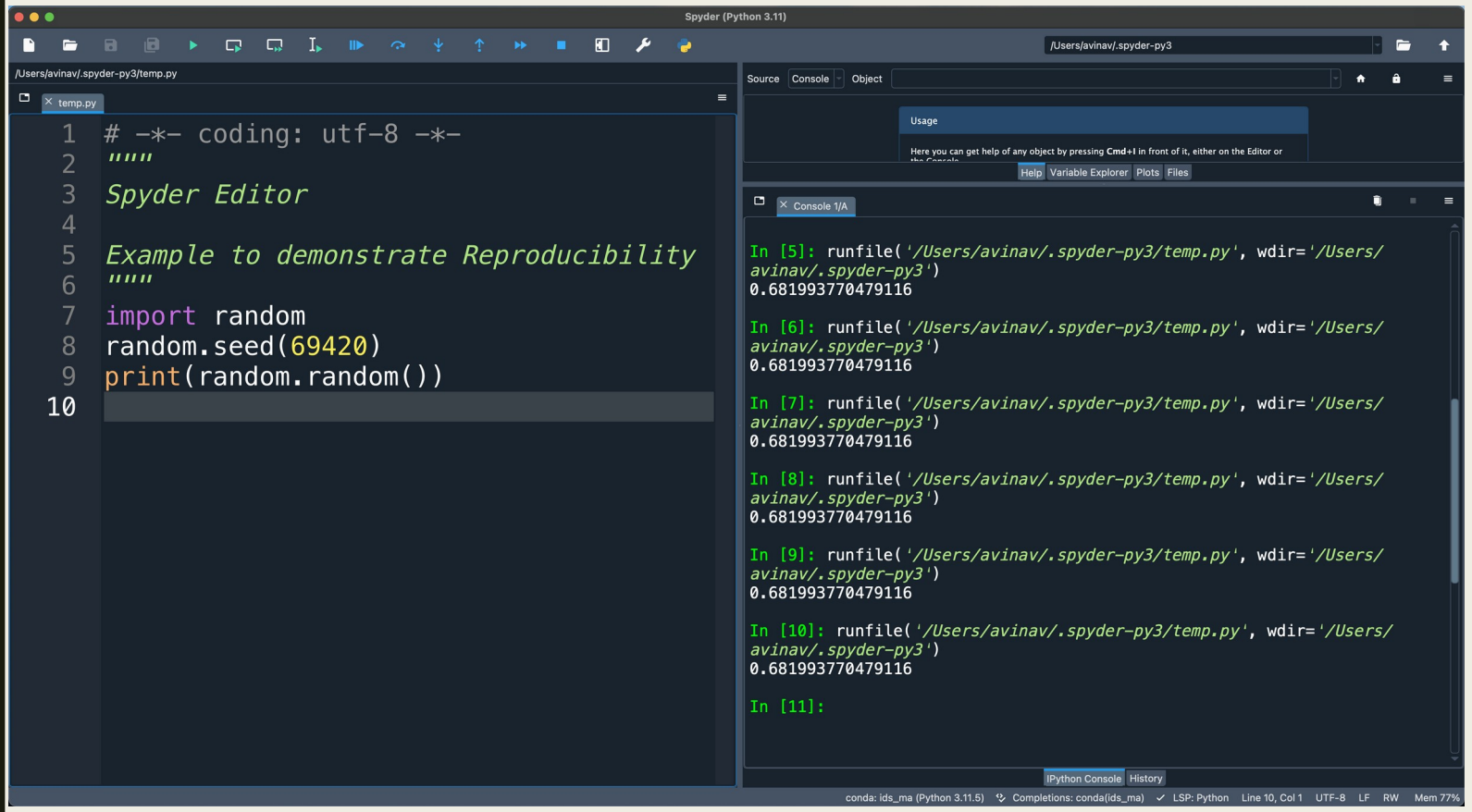
REPRODUCIBILITY EXAMPLE

Example of Random Variable!



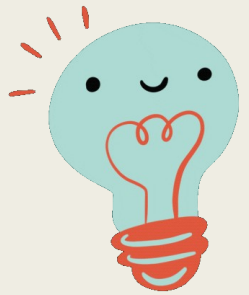
- Random is used in almost every Data Science Library, Machine Learning on the inside(hidden)
- So every time you run a function(maybe ML/Data Science), you might get a different value
- How to solve this problem then?

Example of Random Variable!



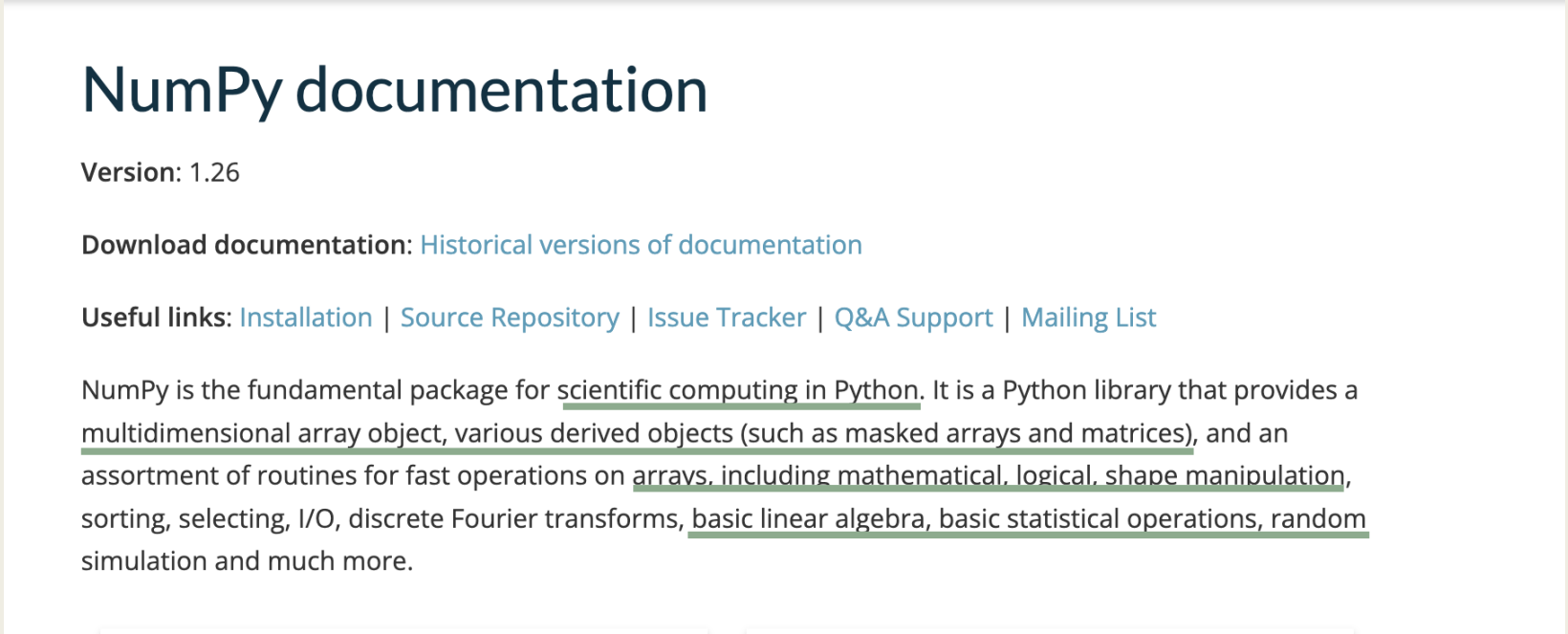
- Let’s set the “seed”!
- The seed() method is used to initialize the random number generator. The random number generator needs a number to start with (a seed value), to be able to generate a random number.
- By default the random number generator uses the current system time.

IS ANYTHING IN COMPUTERS REALLY “RANDOM”?



FREQUENTLY USED PACKAGES

NUMPY



NumPy documentation

Version: 1.26

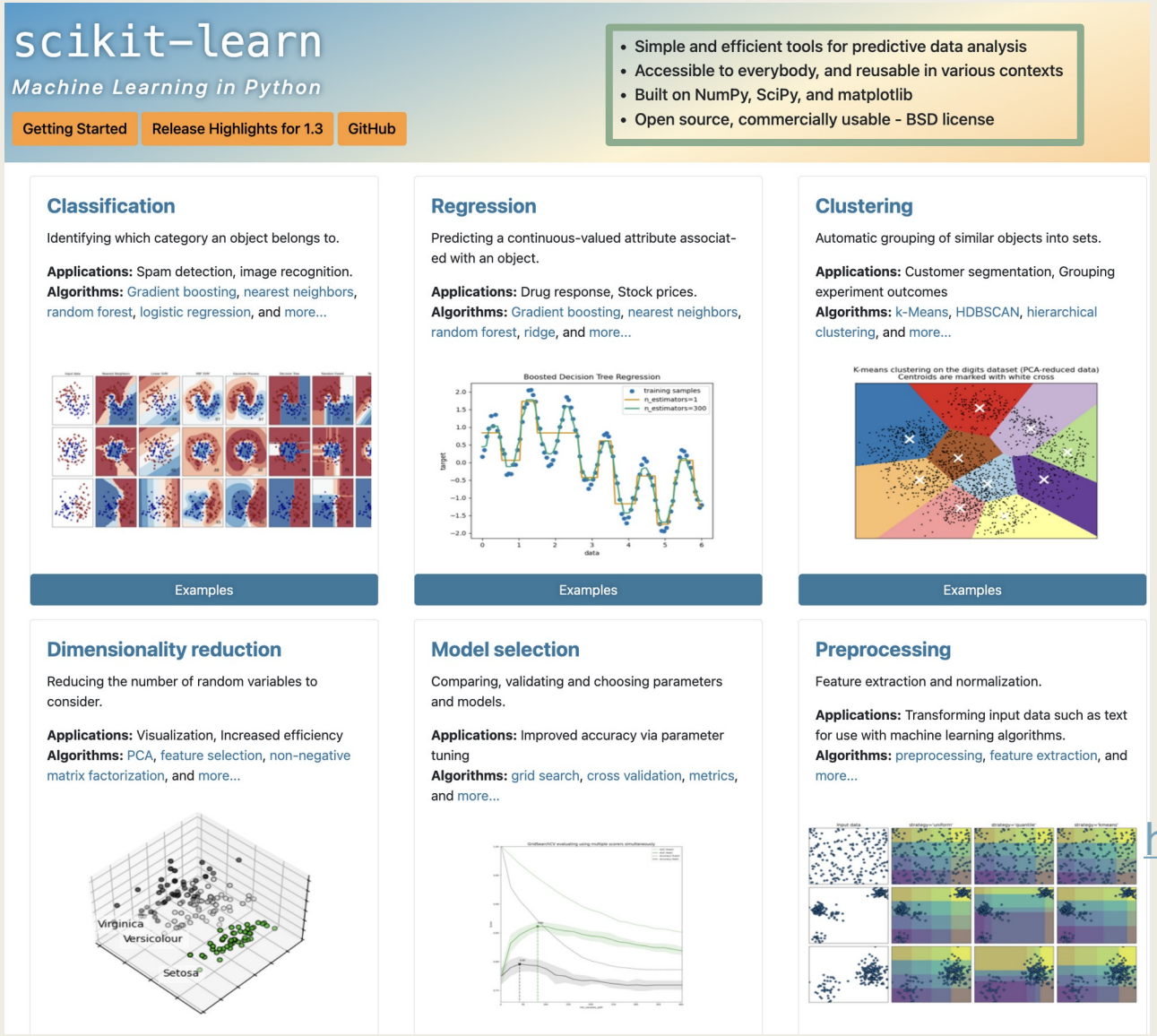
Download documentation: Historical versions of documentation

Useful links: Installation | Source Repository | Issue Tracker | Q&A Support | Mailing List

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

https://numpy.org/doc/stable/

SCIKIT-LEARN



https://scikit-learn.org/stable/