REMINDER: videos on linear algebra

The Essence of Linear Algebra series (3blue1brown) linked below is an excellent resource for reviewing (or learning) basic concepts of linear algebra. The most important concepts to review are vectors (Chapter 1-2), linear transformations and matrices (Chapter 3), matrix multiplication (Chapter 4), non-square matrices as transformations between dimensions (Chapter 8), and dot products (Chapter 9).

Please review these over the next few weeks.

https://www.youtube.com/playlist?list=PLZHQObOWTQDPD3MizzM2xVFitgF8hE_ab

REMINDER: help sessions

Mondays 3:50-5:00
this room (or WH 113)

except for this week, which will be Wed after class

REMINDER: Piazza

- piazza.com/vanderbilt/fall2022/psy42196219

(part of) HOMEWORK 1:

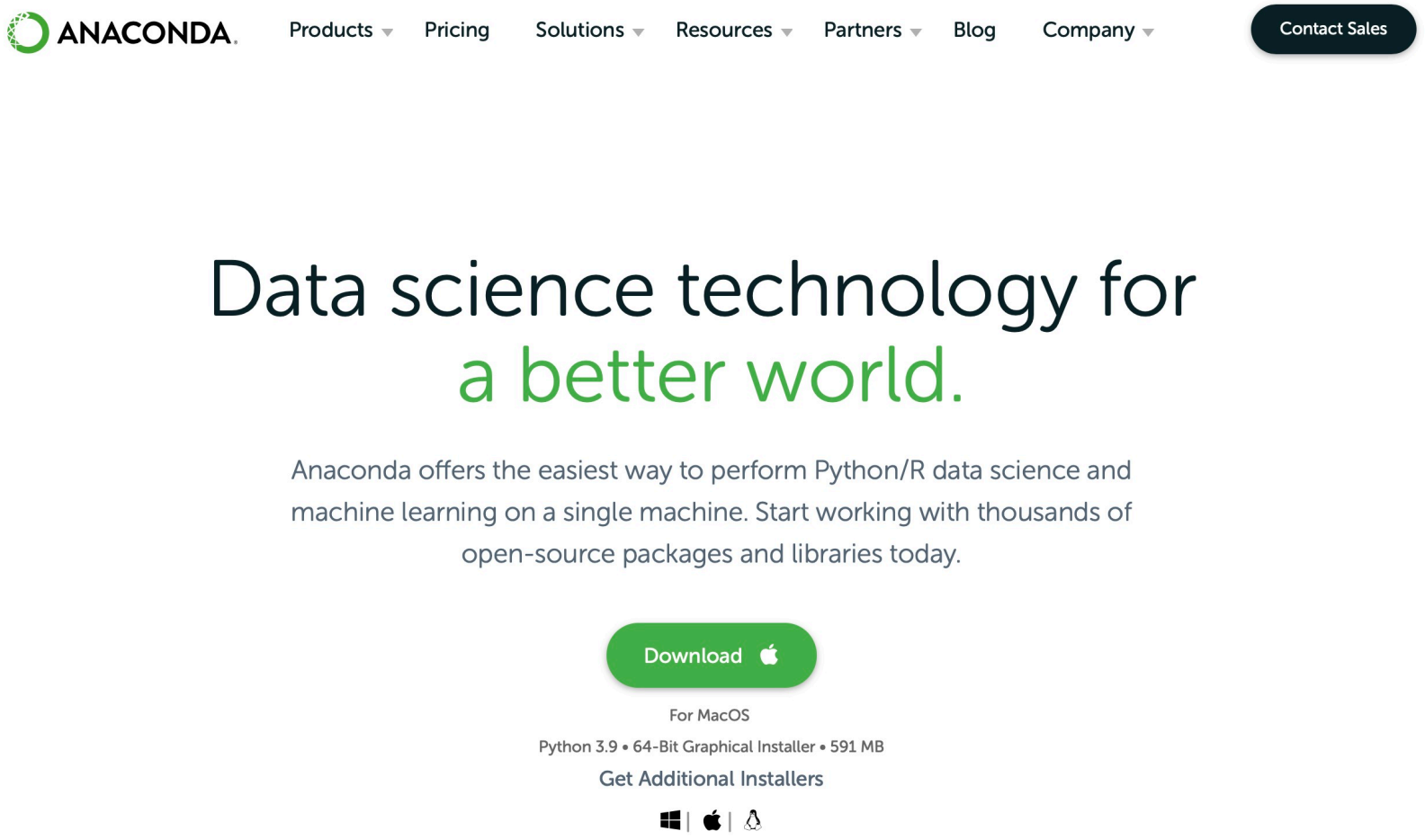
Create a Post and **Respond to a Post** on Piazza (under the "welcome" tag) - share something about yourself (your interests and hobbies, or your future plans, or what you hope to learn in the class, or really anything you're willing to share)

I shared a (long) post with a bit about myself - after you create a post for yourself, you can respond to the one I wrote or respond to another post

Installing and Configuring Python

installing Python

- if you have never installed Python before, go to https://www.anaconda.com, download, and install



if you already have Python

- if you have Python, check that you have at least version 3.8.x - within Python, run the following code snippet:

```
import sys

print(sys.version_info)
```

- if your version of Python is old, the simplest thing to do is download and install Anaconda (latest version)

  if you have 3.9.x you might also have 3.8.x installed

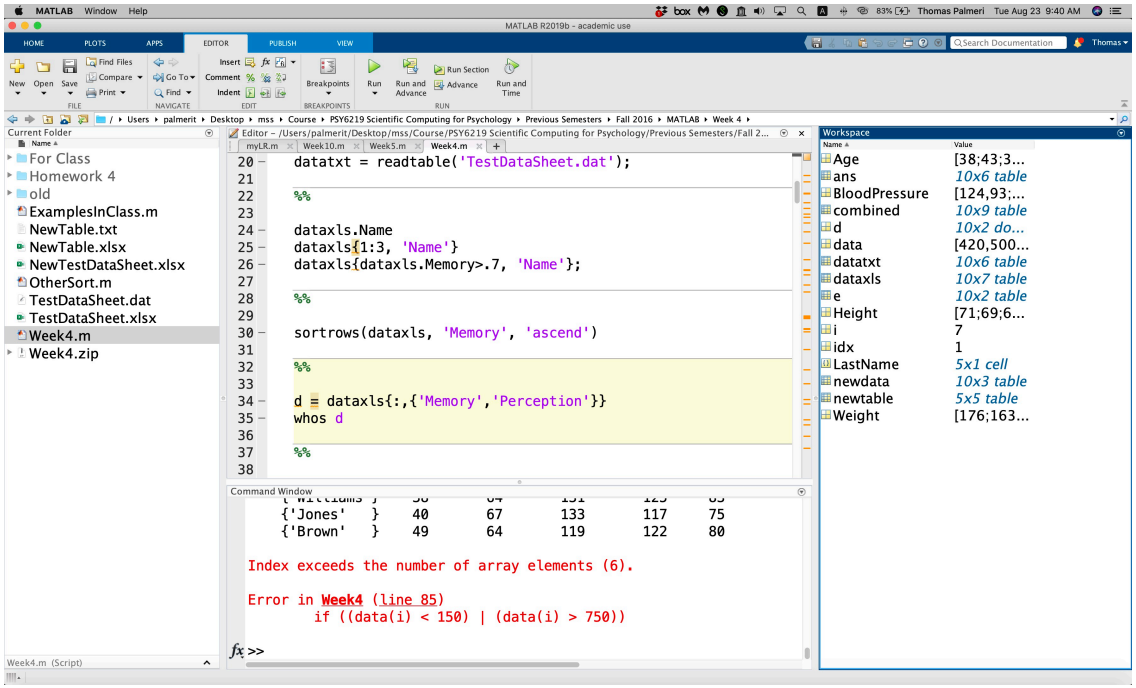
if you already have Python

- you don't have to use the Anaconda build of Python, or download the full (bloated) Anaconda, but that is the (simplest) approach we describe in class

  some might need to use a different installation for various reasons (Jason can help if it is necessary to get Python working on your computer, or you can explore other options on your own)

navigating the pieces of your installation

how Python differs from Matlab



Mathworks is the company that owns Matlab. When you buy Matlab, you get the programming language, any toolboxes you purchase along with base Matlab, and the integrated development environment (IDE).

install "Matlab" and you get the language, the toolboxes, and the IDE.

what does a Python installation look like?

Python the language

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image

def myfilter(x, alpha=1., beta=2., gam=.5, lam=.1):
    F = 1 - np.exp(-(x/alpha)**beta)
    psi = gam + (1. - lam - gam)*F
    return (psi)

alpha = 100
beta  = 3
gam   = 0.48
lam   = 0.48

x = np.arange(0, 255, .1)
psi = myfilter(x, alpha, beta, gam, lam)
plt.plot(x, psi, 'r-');
plt.ylim((0,1))
```

Python is open source, freely usable and distributable, even for commercial use - Python's license is administered by the Python Software Foundation

you can download Python (for free) directly from python.org (all releases going back to Python 1.1)

what does a Python installation look like?

Python the language

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image

def myfilter(x, alpha=1., beta=2., gam=.5, lam=.1):
    F = 1 - np.exp(-(x/alpha)**beta)
    psi = gam + (1. - lam - gam)*F
    return (psi)

alpha = 100
beta  = 3
gam   = 0.48
lam   = 0.48

x = np.arange(0, 255, .1)
psi = myfilter(x, alpha, beta, gam, lam)
plt.plot(x, psi, 'r-');
plt.ylim((0,1))
```

Python is open source, freely usable and distributable, even for commercial use - Python's license is administered by the Python Software Foundation

you can download Python (for free) directly from python.org (all releases going back to Python 1.1)

with the most basic Python installation, you edit Python .py files (text files) using a basic text editor, or a text editor designed for editing code (Emacs, Vim, BBedit, and many others)

run Python code from the command line / terminal

```
Last login: Sun Aug 21 15:41:34 on ttys001

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Tom-MacBook-Pro-2020:~ palmerit$ python mycode.py
```

what does a Python installation look like?

Python the language

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image

def myfilter(x, alpha=1., beta=2., gam=.5, lam=.1):
    F = 1 - np.exp(-(x/alpha)**beta)
    psi = gam + (1. - lam - gam)*F
    return (psi)

alpha = 100
beta  = 3
gam   = 0.48
lam   = 0.48

x = np.arange(0, 255, .1)
psi = myfilter(x, alpha, beta, gam, lam)
plt.plot(x, psi, 'r-');
plt.ylim((0,1))
```

Python packages / modules

(a package is collection of modules)
(like toolboxes in Matlab, libraries in C)

`numpy, scipy, matplotlib, seaborn, pandas, pillow, scikit-learn, psychopy, keras`

what does a Python installation look like?

Python the language

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image

def myfilter(x, alpha=1., beta=2., gam=.5, lam=.1):
    F = 1 - np.exp(-(x/alpha)**beta)
    psi = gam + (1. - lam - gam)*F
    return (psi)

alpha = 100
beta  = 3
gam   = 0.48
lam   = 0.48

x = np.arange(0, 255, .1)
psi = myfilter(x, alpha, beta, gam, lam)
plt.plot(x, psi, 'r-');
plt.ylim((0,1))
```

Python packages / modules

(a package is collection of modules)
(like toolboxes in Matlab, libraries in C)

`numpy, scipy, matplotlib, seaborn, pandas, pillow, scikit-learn, psychopy, keras`

you can download packages from pypi.org
using pip from the command line / terminal

what does a Python installation look like?

Anaconda is a company that provides a distribution of Python and 100s of packages (instead of python.org and pypi.org)

Python the language

+

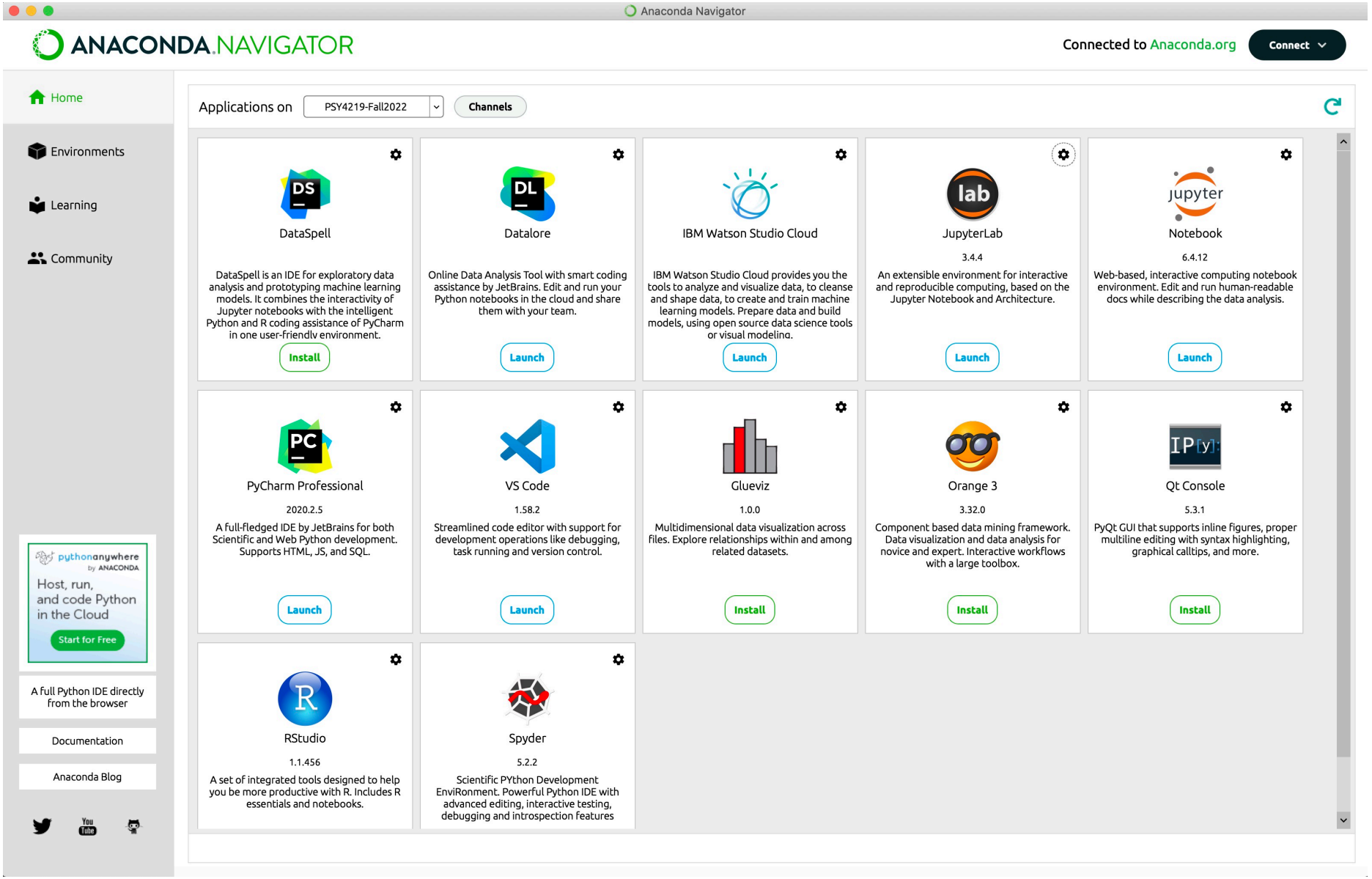
Python packages / modules

$$$ for businesses

free for education

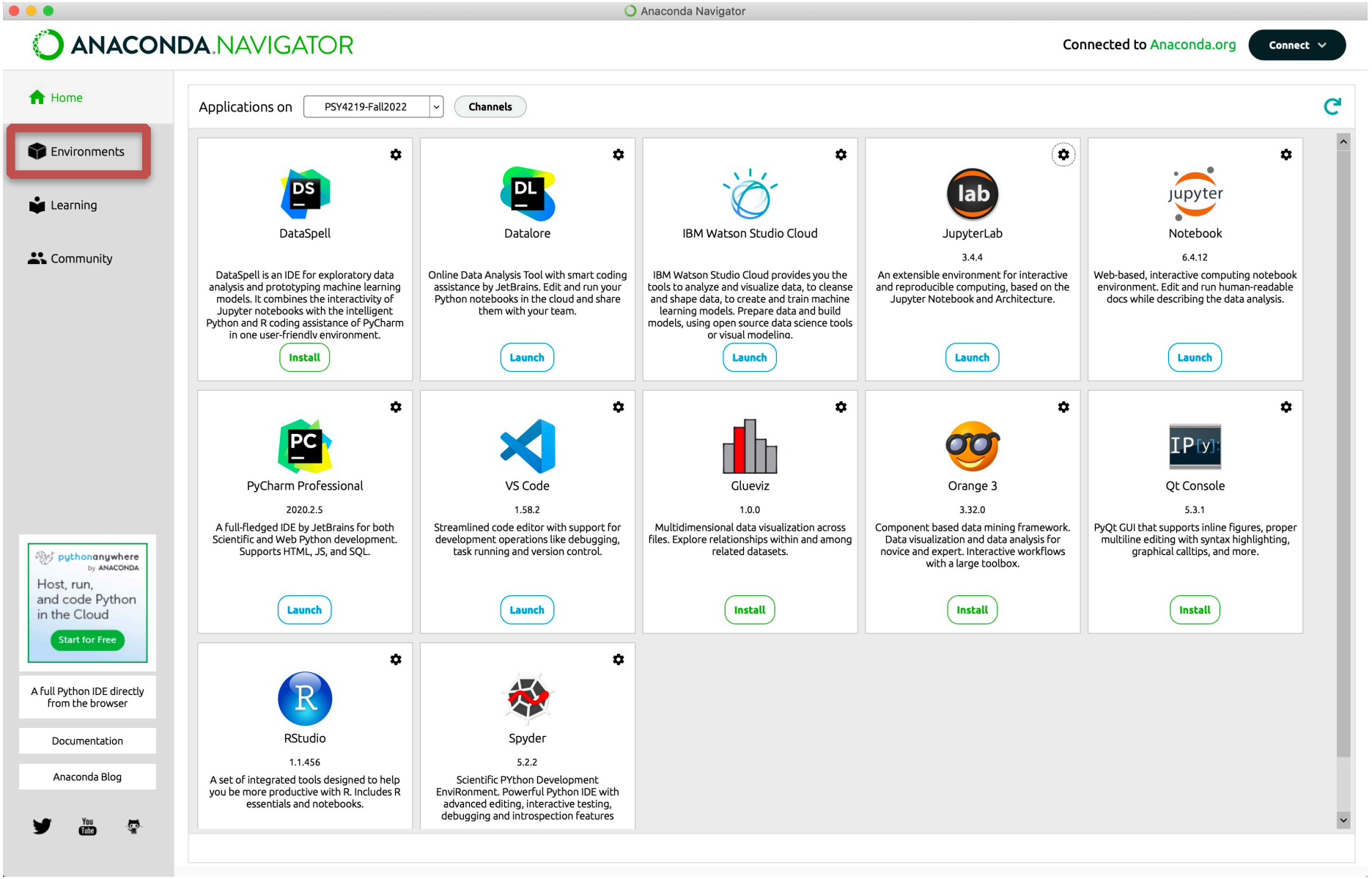
what does a Python installation look like?

the Anaconda installation comes with Anaconda Navigator, which lets you launch applications, and manage packages and environments



what does a Python installation look like?

the Anaconda installation comes with Anaconda Navigator, which lets you launch applications, and manage packages and environments



what does a Python installation look like?

in this class, you need Python 3.8.x to use the psychopy package, and that package depends on particular versions of other packages

in computational neuroscience, you may need a different version of Python to use tensorflow, which in turn depends on other versions of various packages

in your lab research, you use an old analysis package that depends on an old different version of Python and old versions of other packages

what does a Python installation look like?

in this class, you need Python 3.8.x to use the psychopy package, and that package depends on particular versions of other packages

in computational neuroscience, you may need a different version of Python to use tensorflow, which in turn depends on other versions of various packages

in your lab research, you use an old analysis package that depends on an old different version of Python and old versions of other packages

tools to create virtual environment allow you to have all of these different configurations of Python at once

`virtualenv` and `conda` are two popular tools

Anaconda Navigator creates conda environments and lets you easily manipulate them and use them

what does a Python installation look like?

in this class, you need Python 3.8.x to use the psychopy package, and that package depends on particular versions of other packages

in computational neuroscience, you may need a different version of Python to use tensorflow, which in turn depends on other versions of various packages

in your lab research, you use an old analysis package that depends on an old different version of Python and old versions of other packages

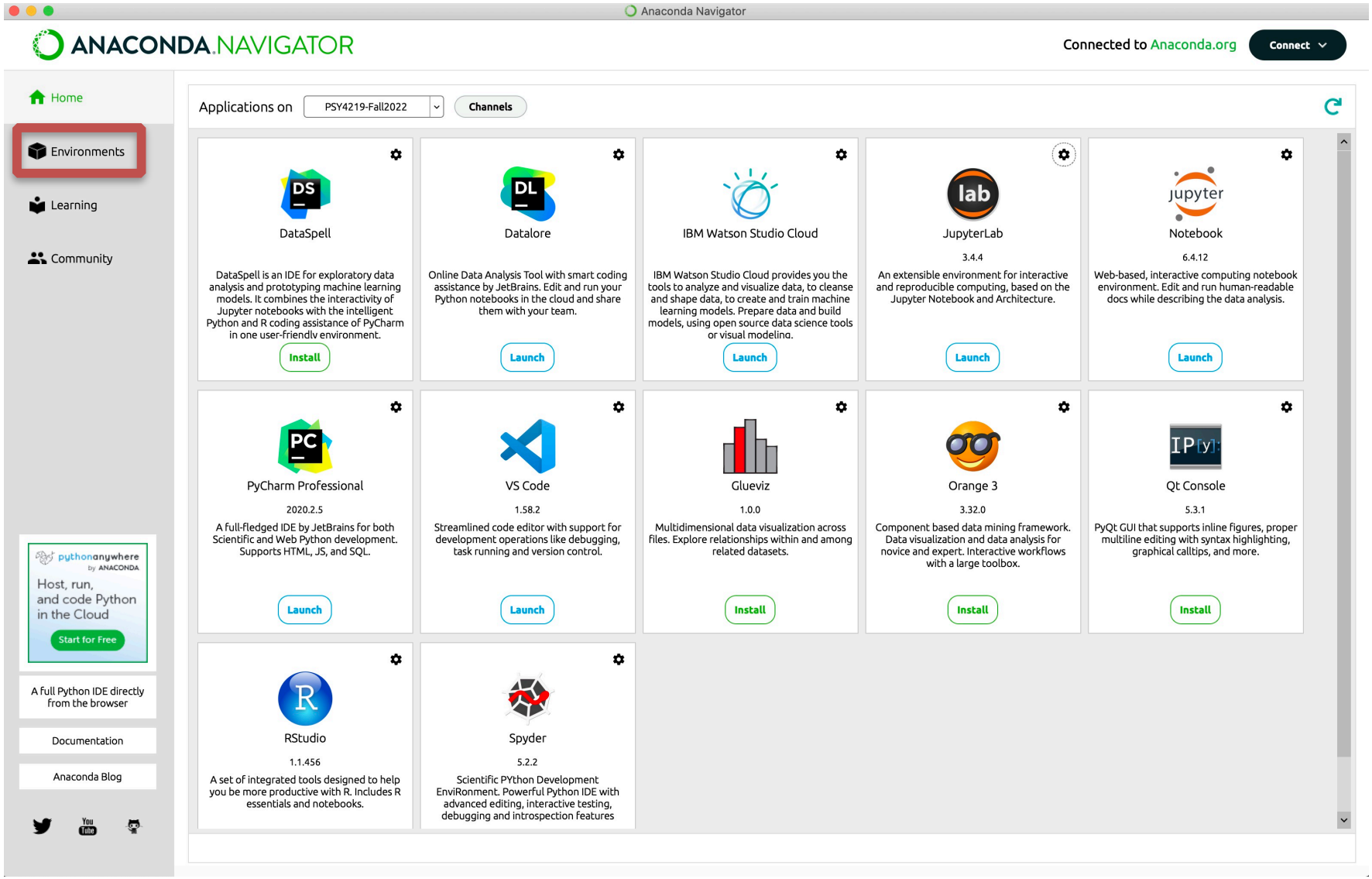
(more extreme versions are docker or singularity containers, which encapsulate a configured version of an entire operating system, not just Python)

tools to create virtual environment allow you to have all of these different configurations of Python at once

`virtualenv` and `conda` are two popular tools

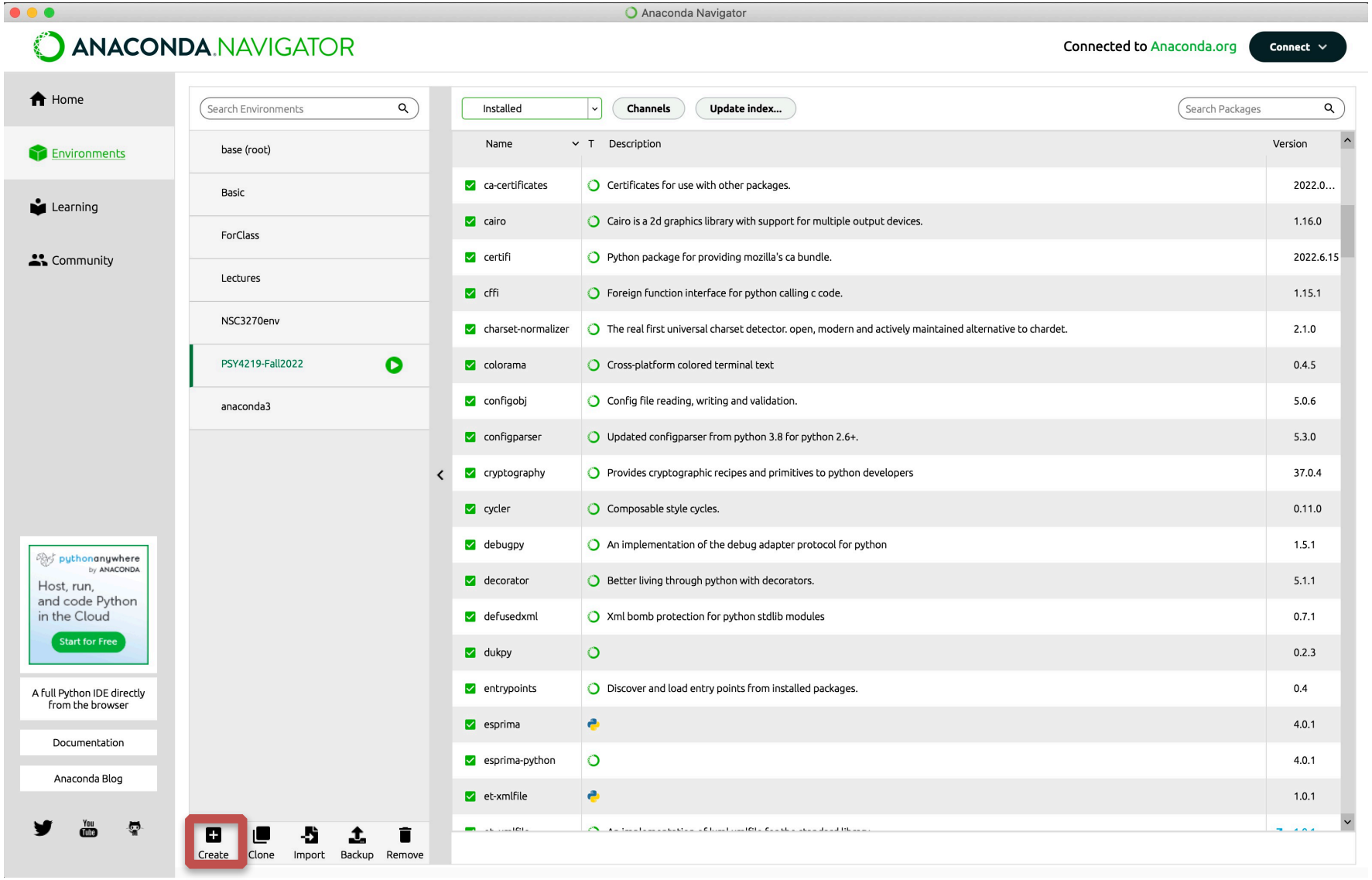
Anaconda Navigator creates conda environments and lets you easily manipulate them and use them

virtual environments in Anaconda Navigator



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator



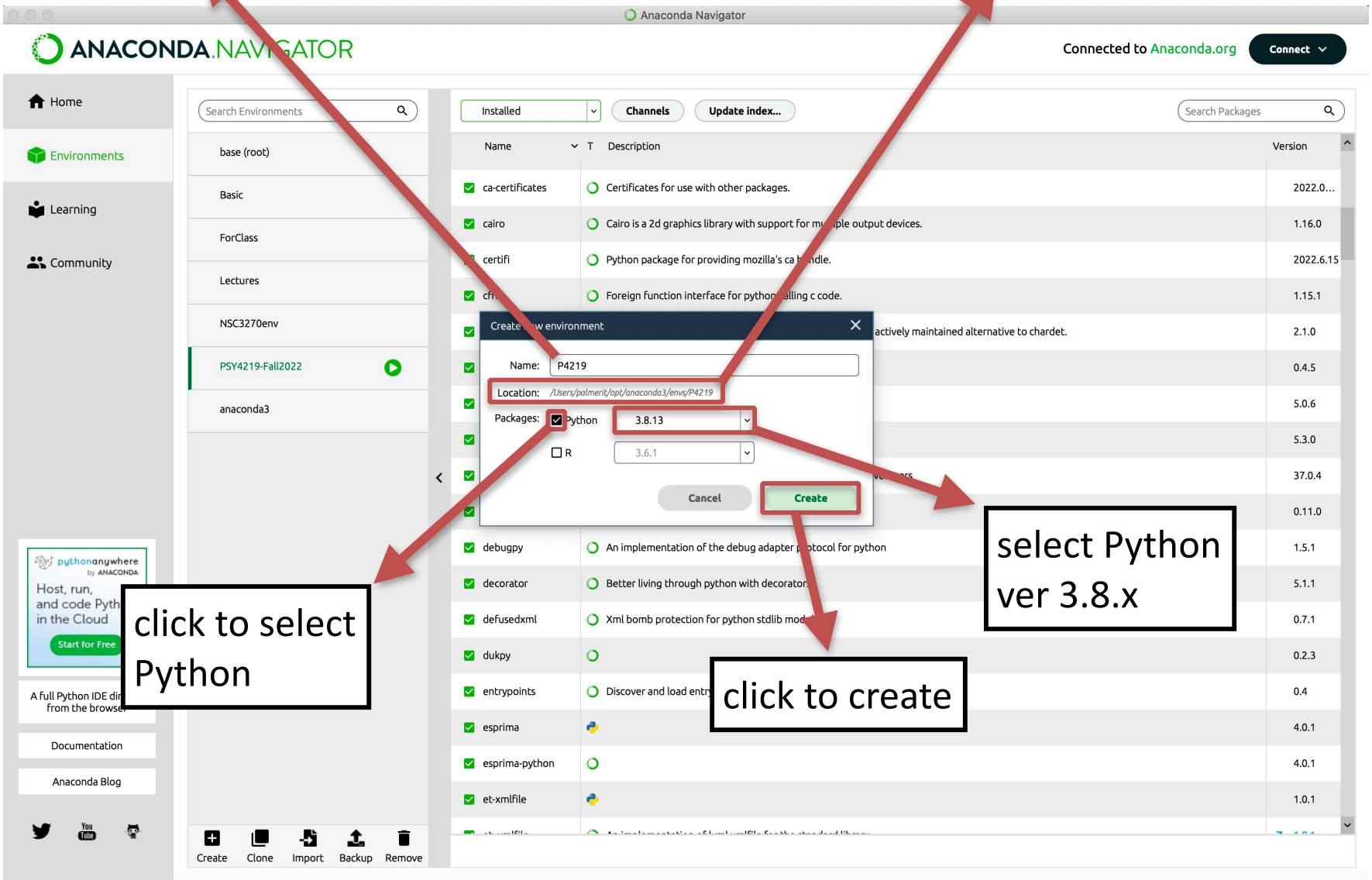
https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

name of the environment

write down the location of the environment

e.g., `/Users/palmerit/opt/anaconda3/envs/P4219`

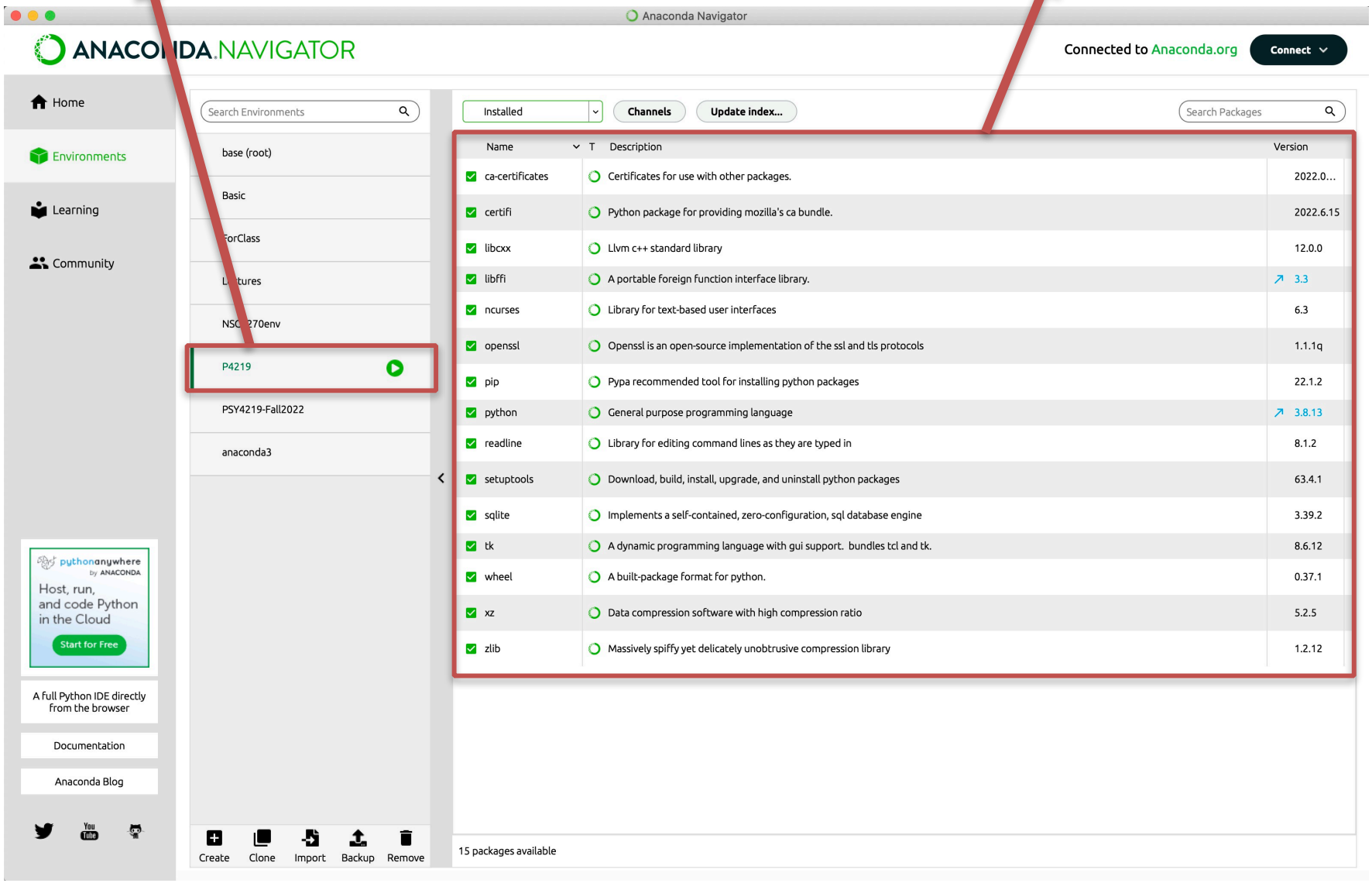


https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

new environment is created

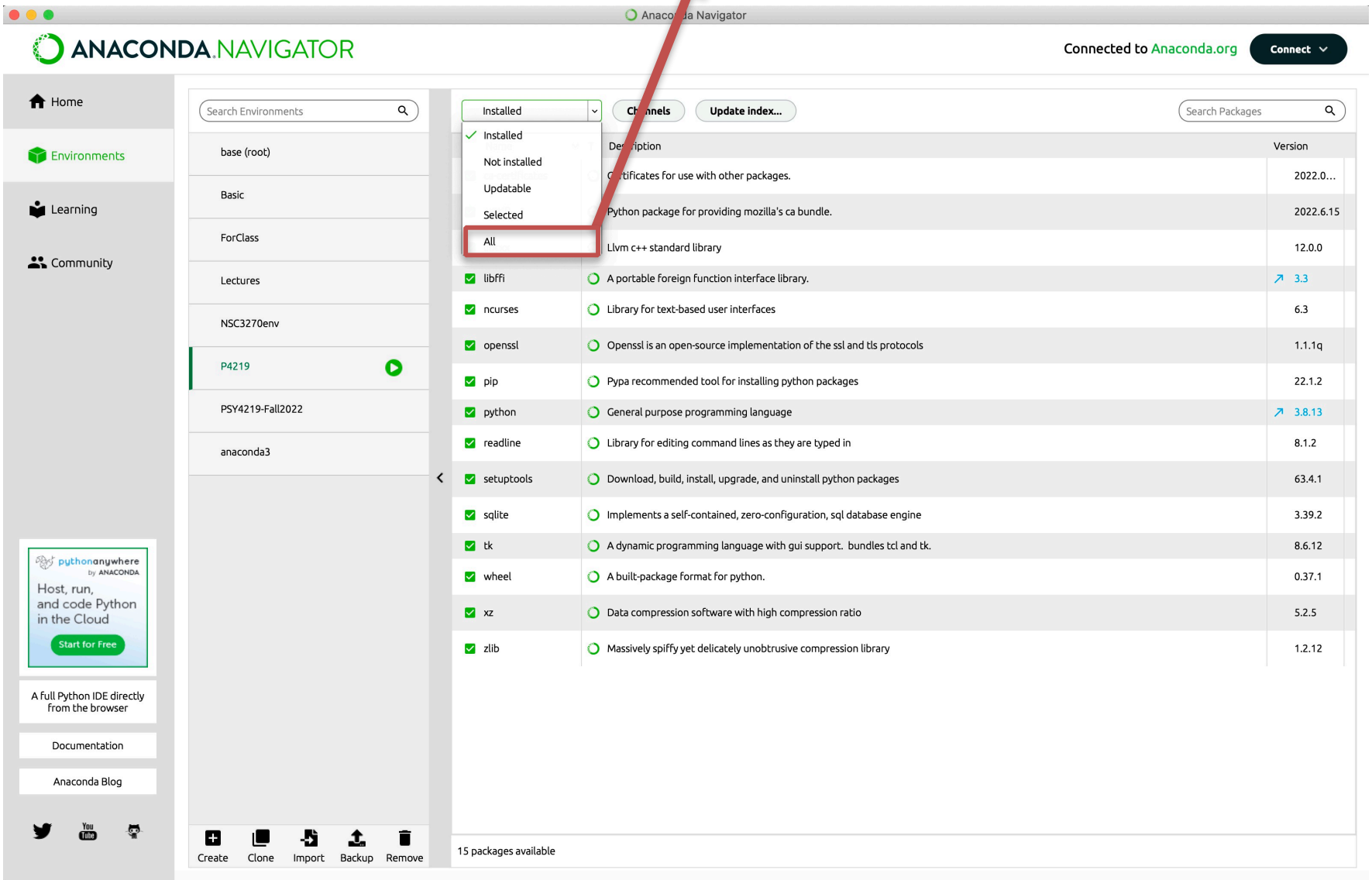
very few packages installed



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

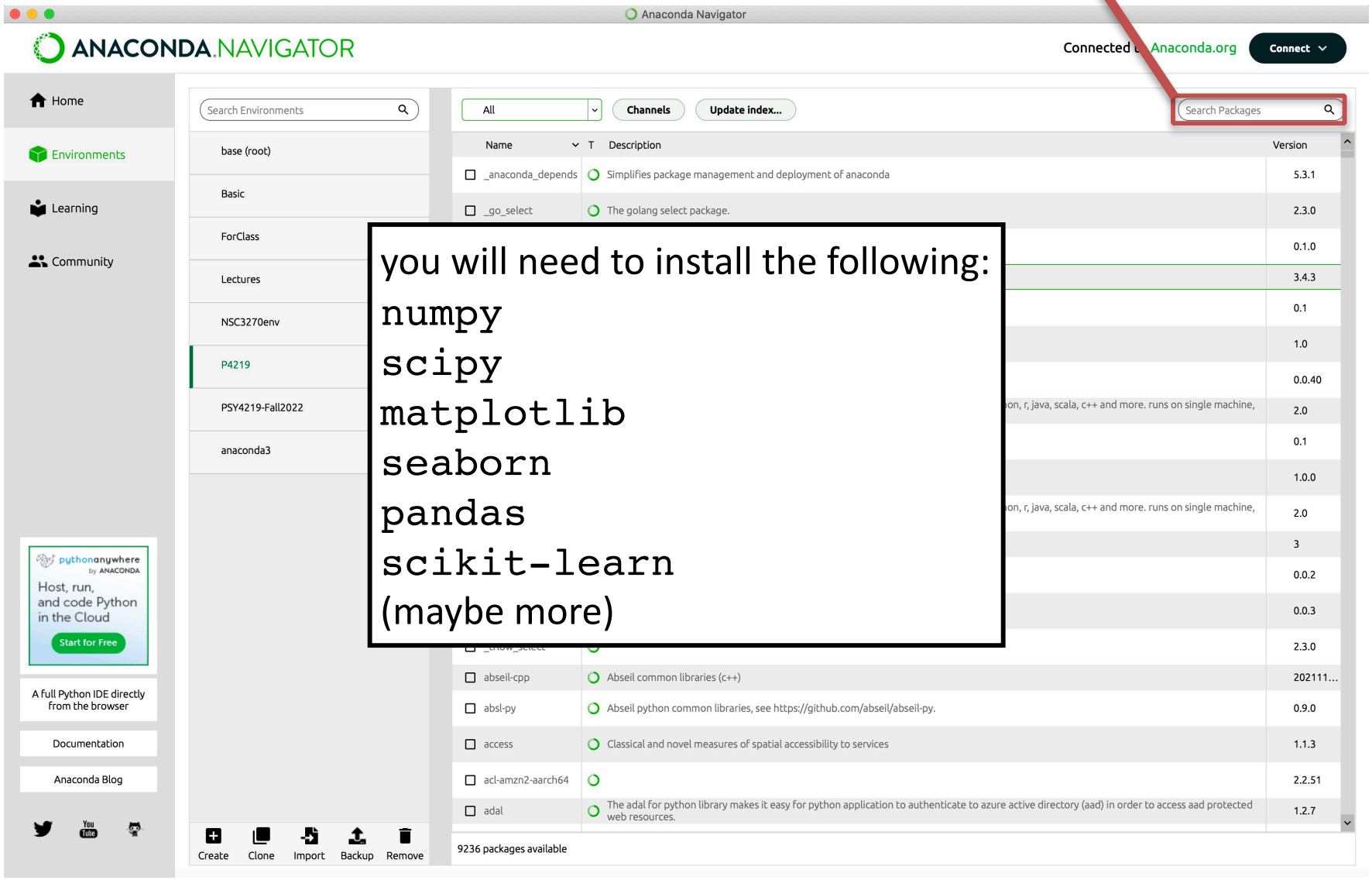
select "All" to see all available packages



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

search for packages to install



you will need to install the following:

```
numpy
scipy
matplotlib
seaborn
pandas
scikit-learn
```

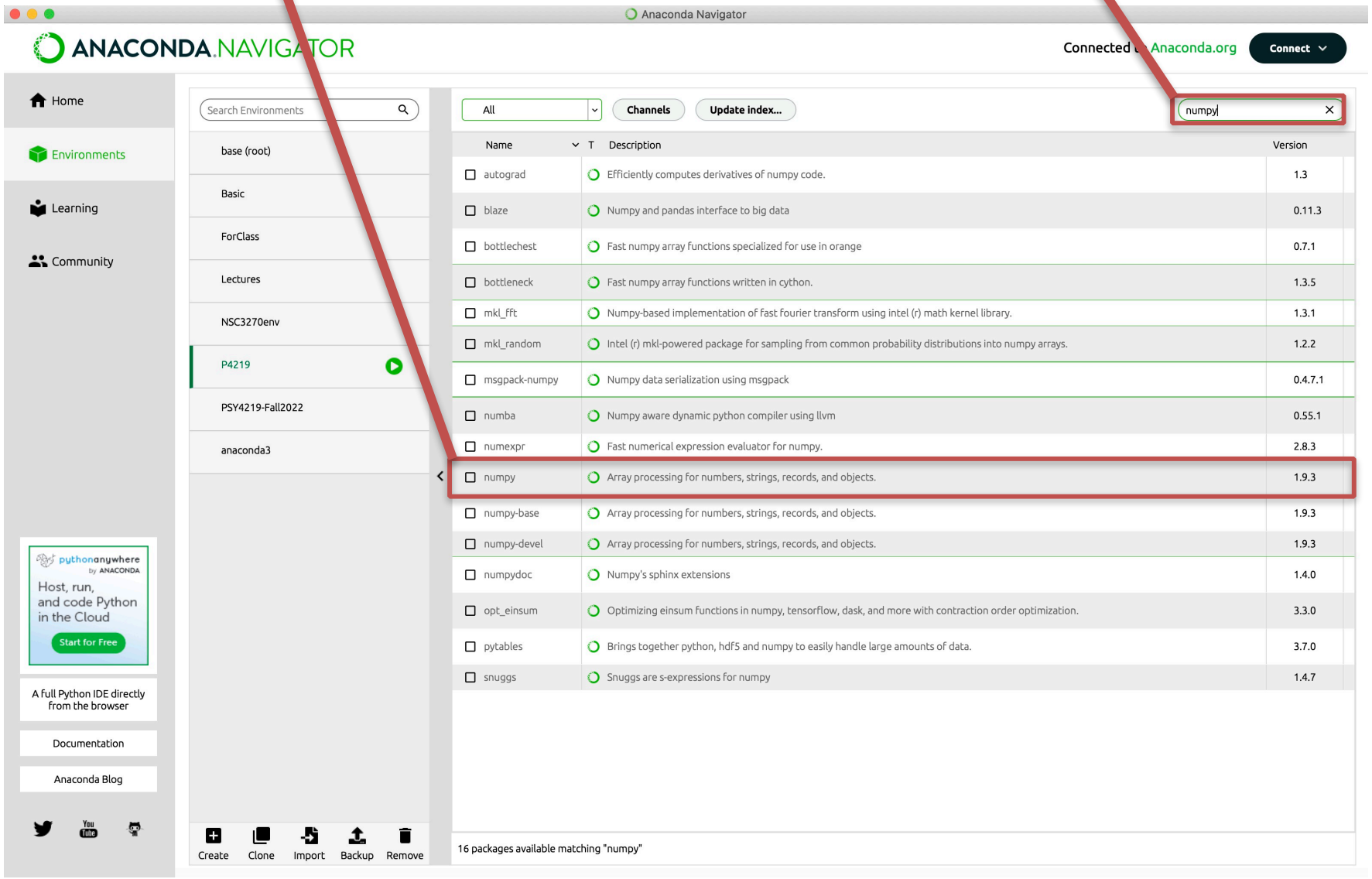
(maybe more)

https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

numpy (and other things) shows up

e.g., type in `numpy`

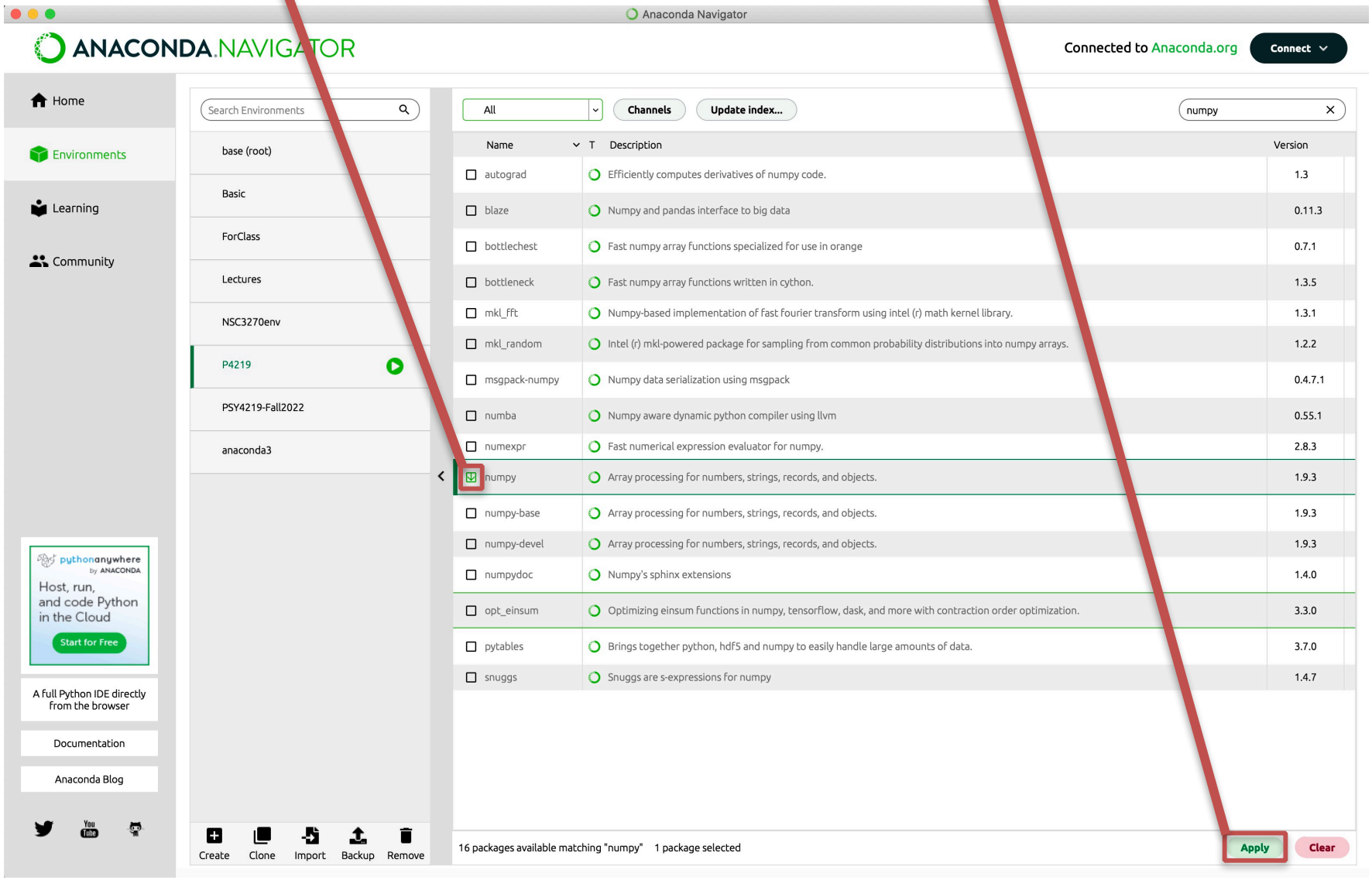


https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

click on numpy (and only numpy)

and click "Apply"

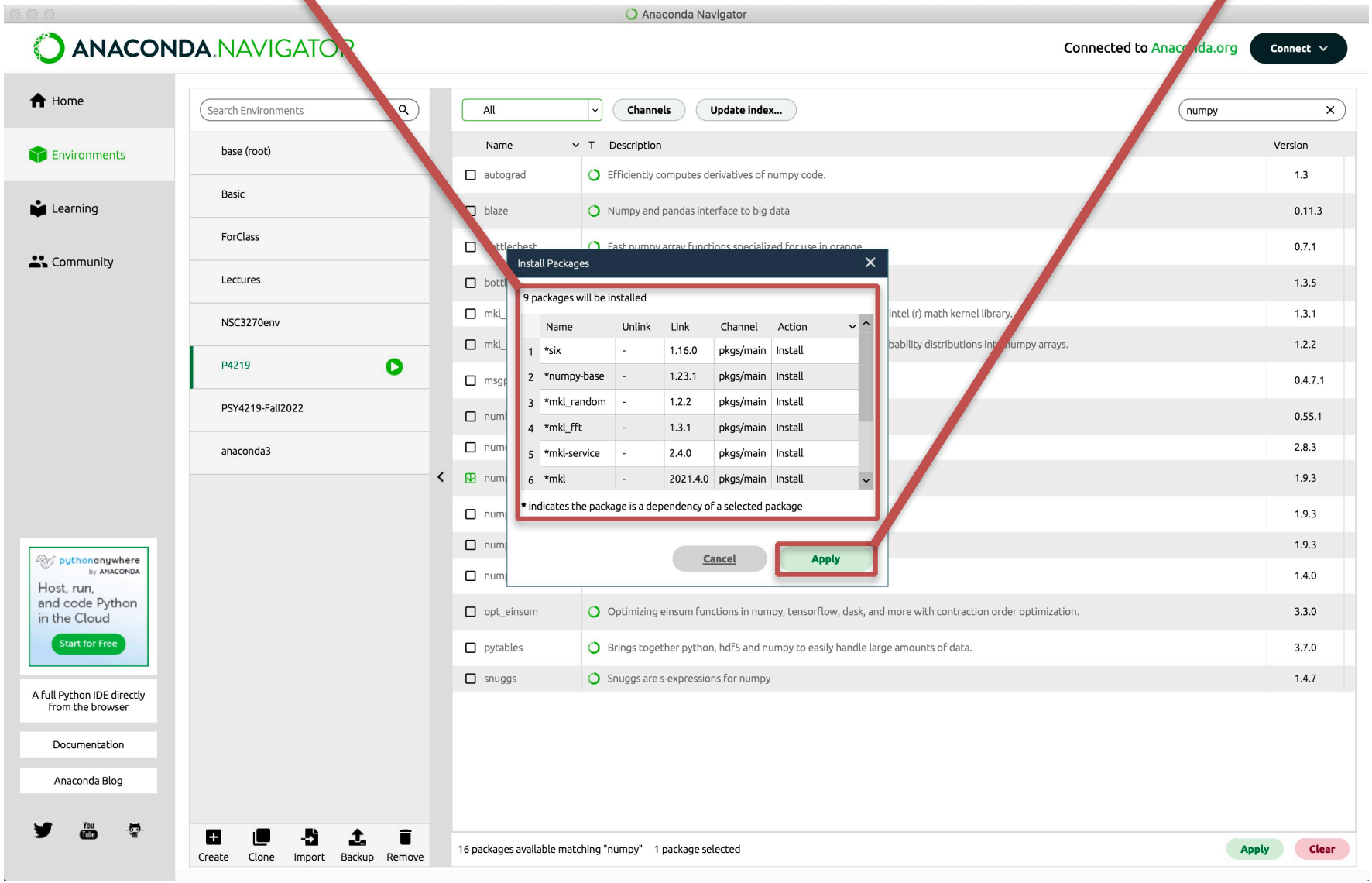


https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

numpy and its dependencies being installed

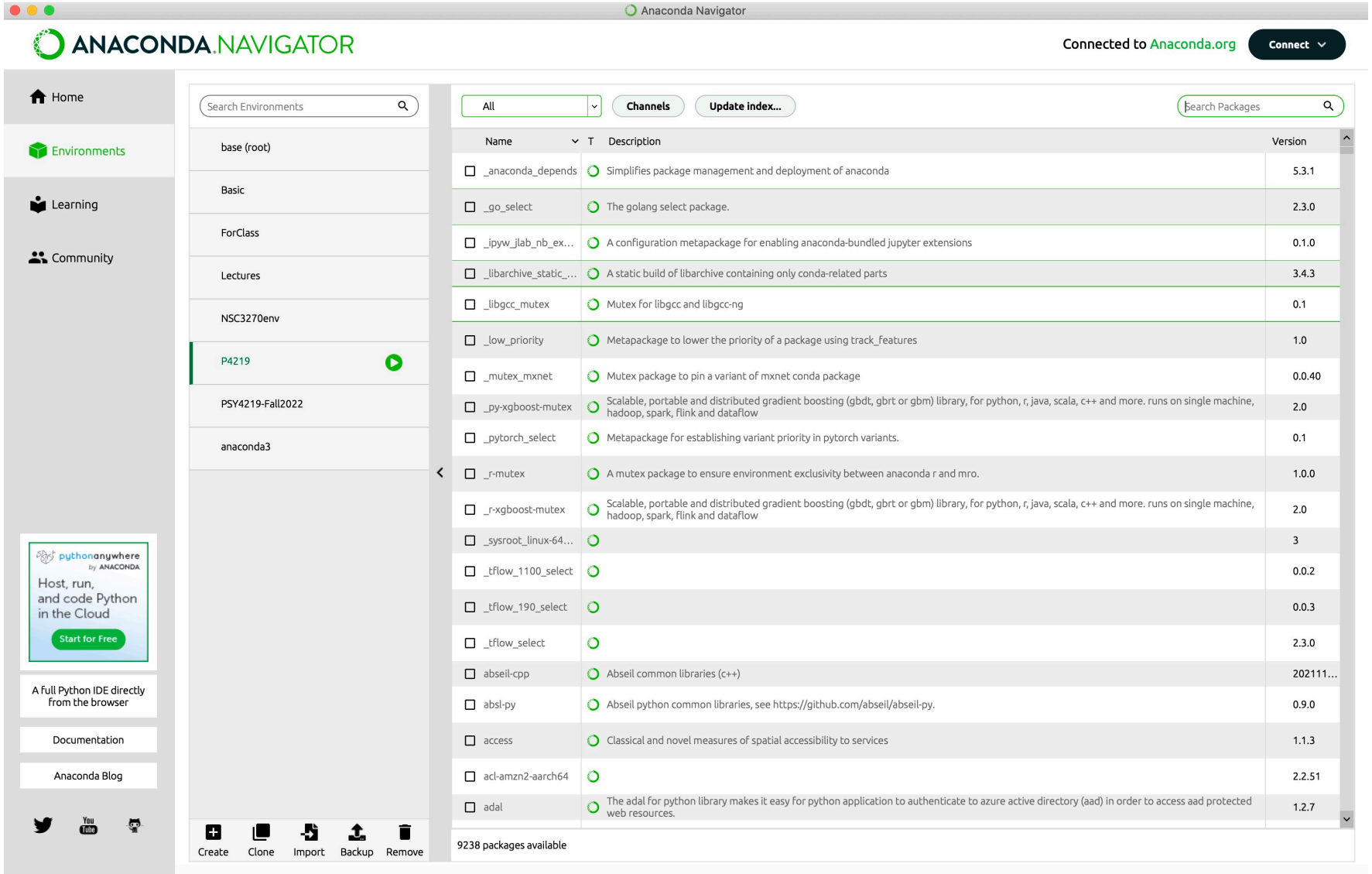
click "Apply"



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

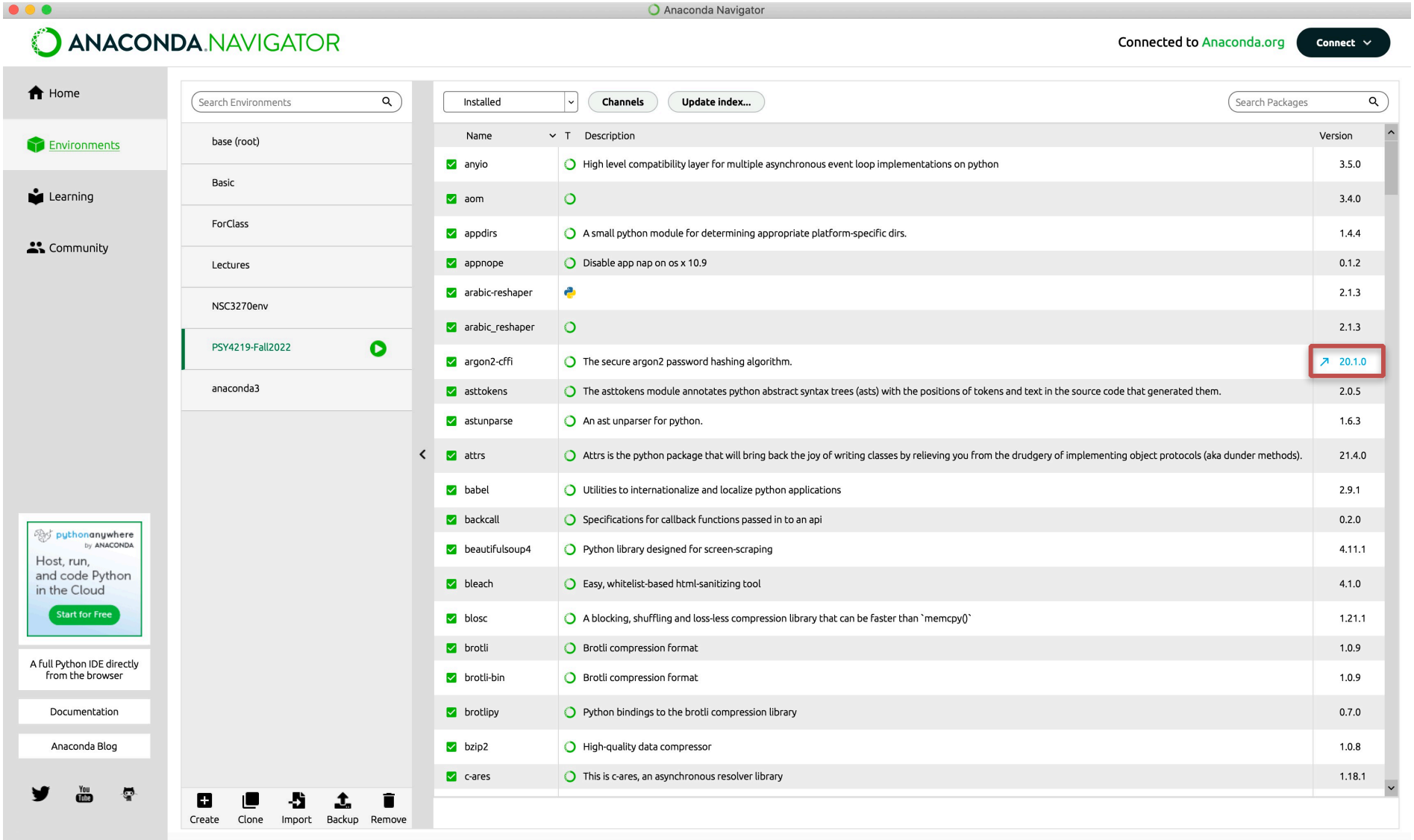
do the same thing for `scipy`, `matplotlib`, `seaborn`, `pandas`, `scikit-learn`



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

DO NOT update packages (unless you know what you are doing)



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

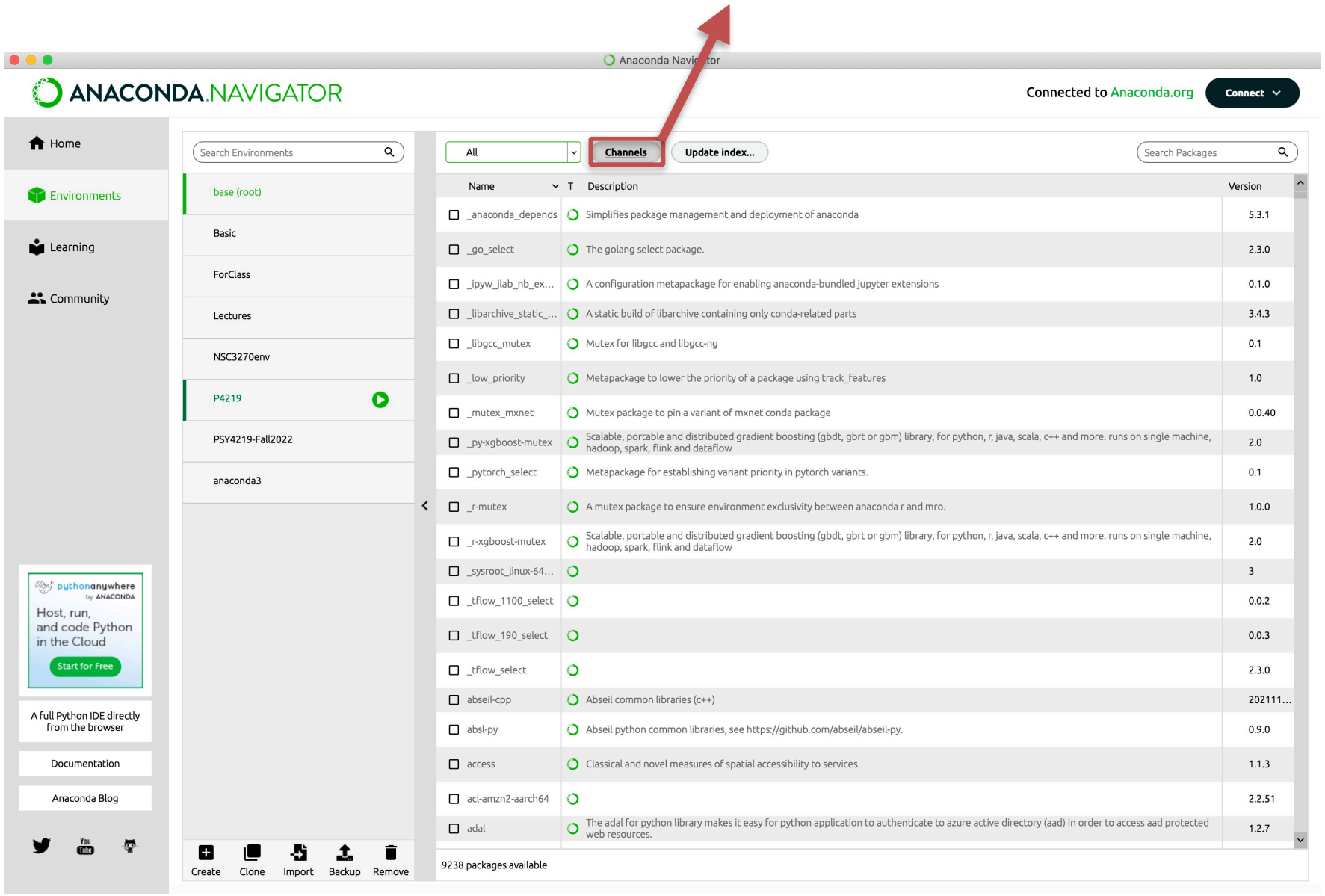
next, you need to install psychopy, which is not part of the anaconda distribution (most specialized discipline-specific packages are not part of anaconda)

in some cases, you would need to install these specialized packages using conda from the command line / terminal, but psychopy (like some specialized packages) has a distribution on `conda-forge`

https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

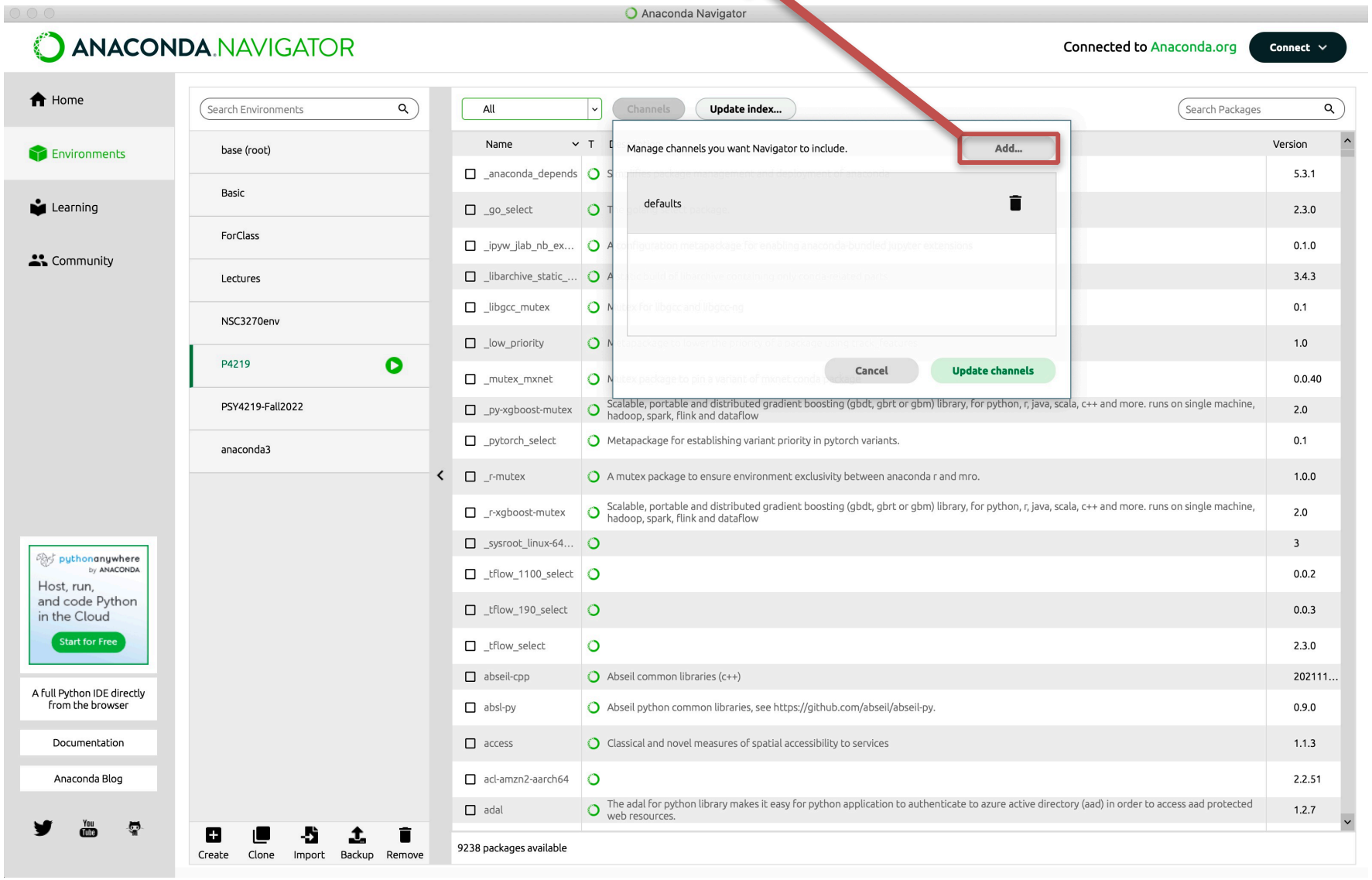
need to see if conda-forge is on your "Channels" (places Anaconda checks for packages)



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

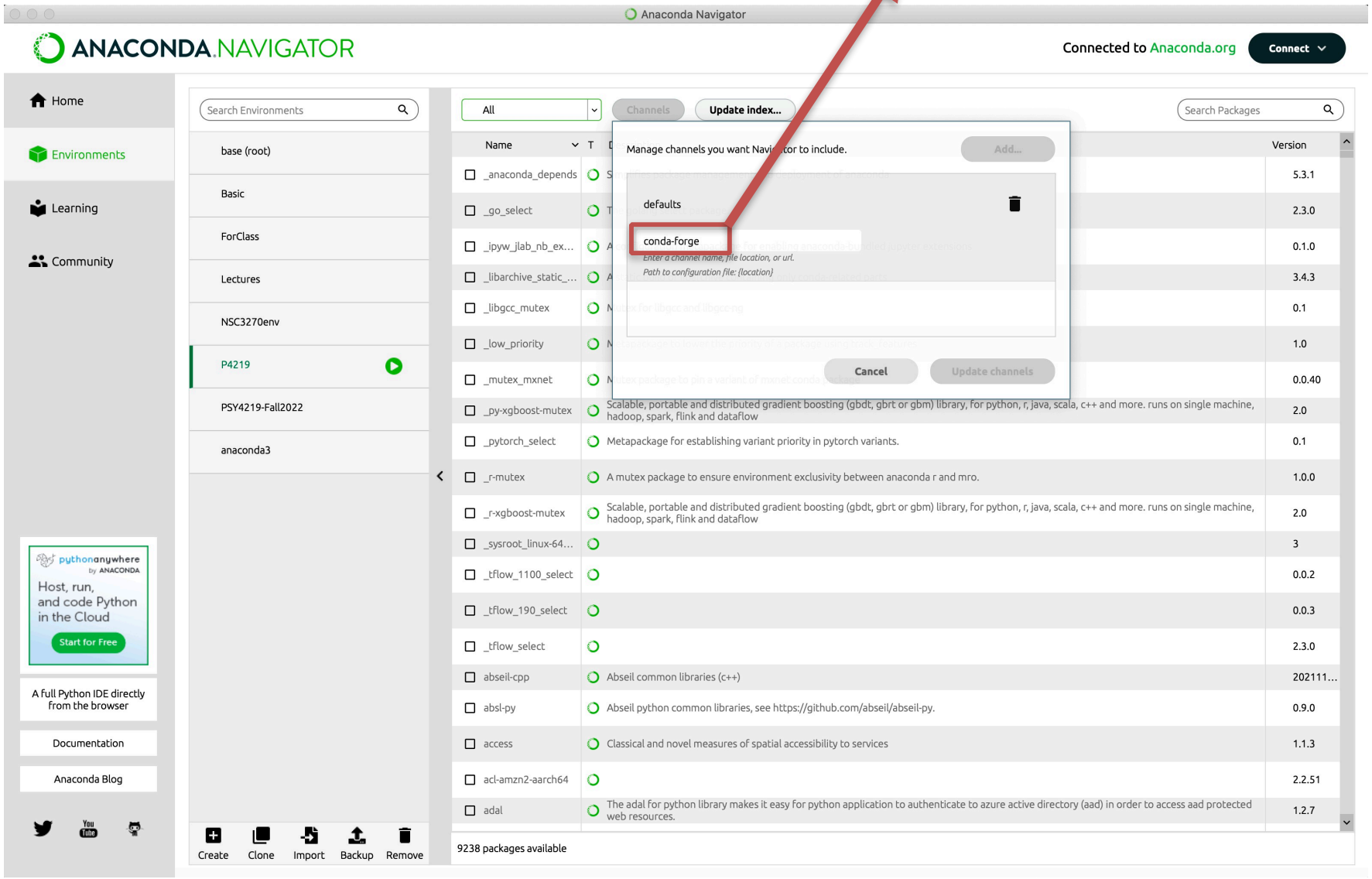
if conda-forge is not listed, click "Add..."



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

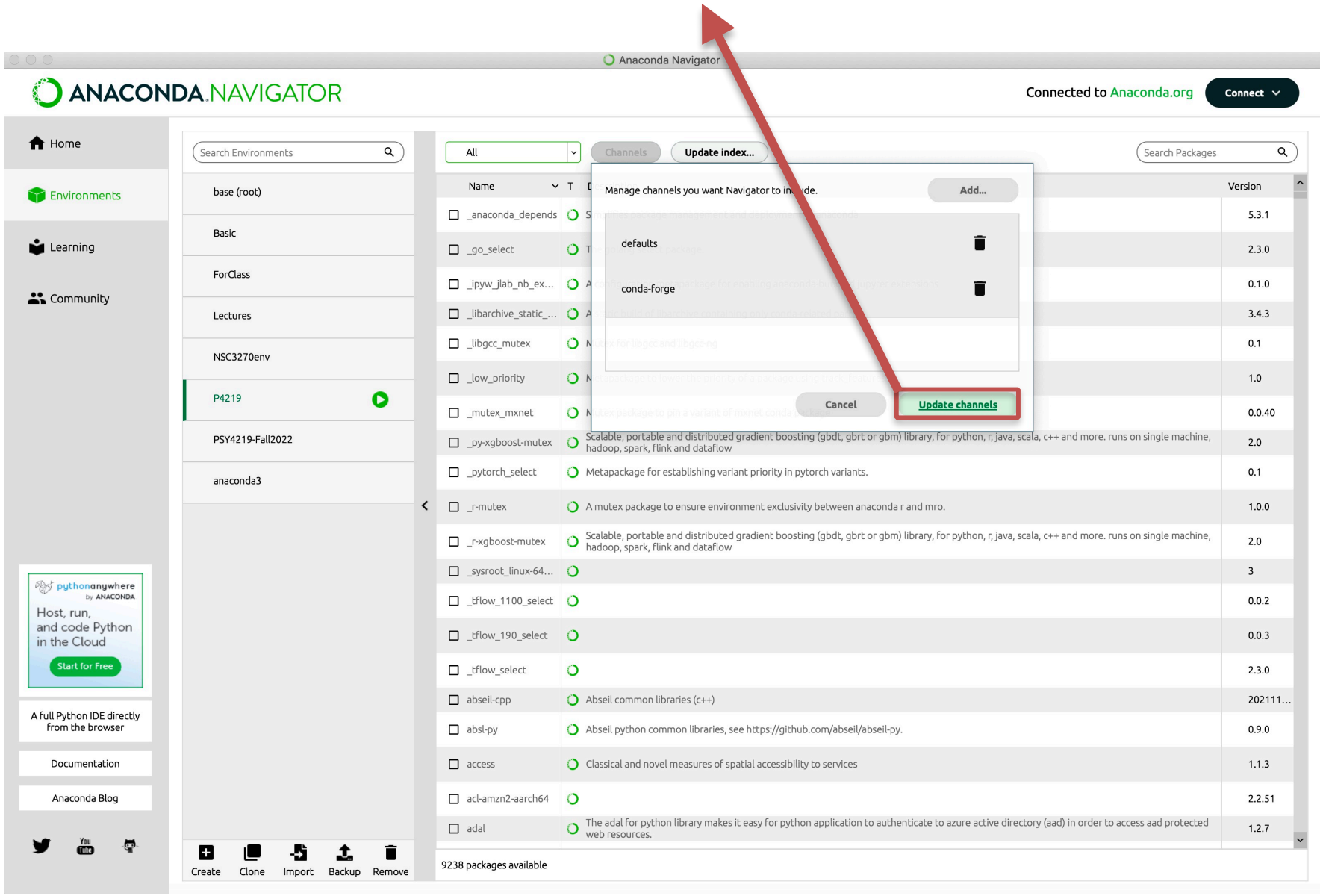
type in conda-forge and hit enter/return



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

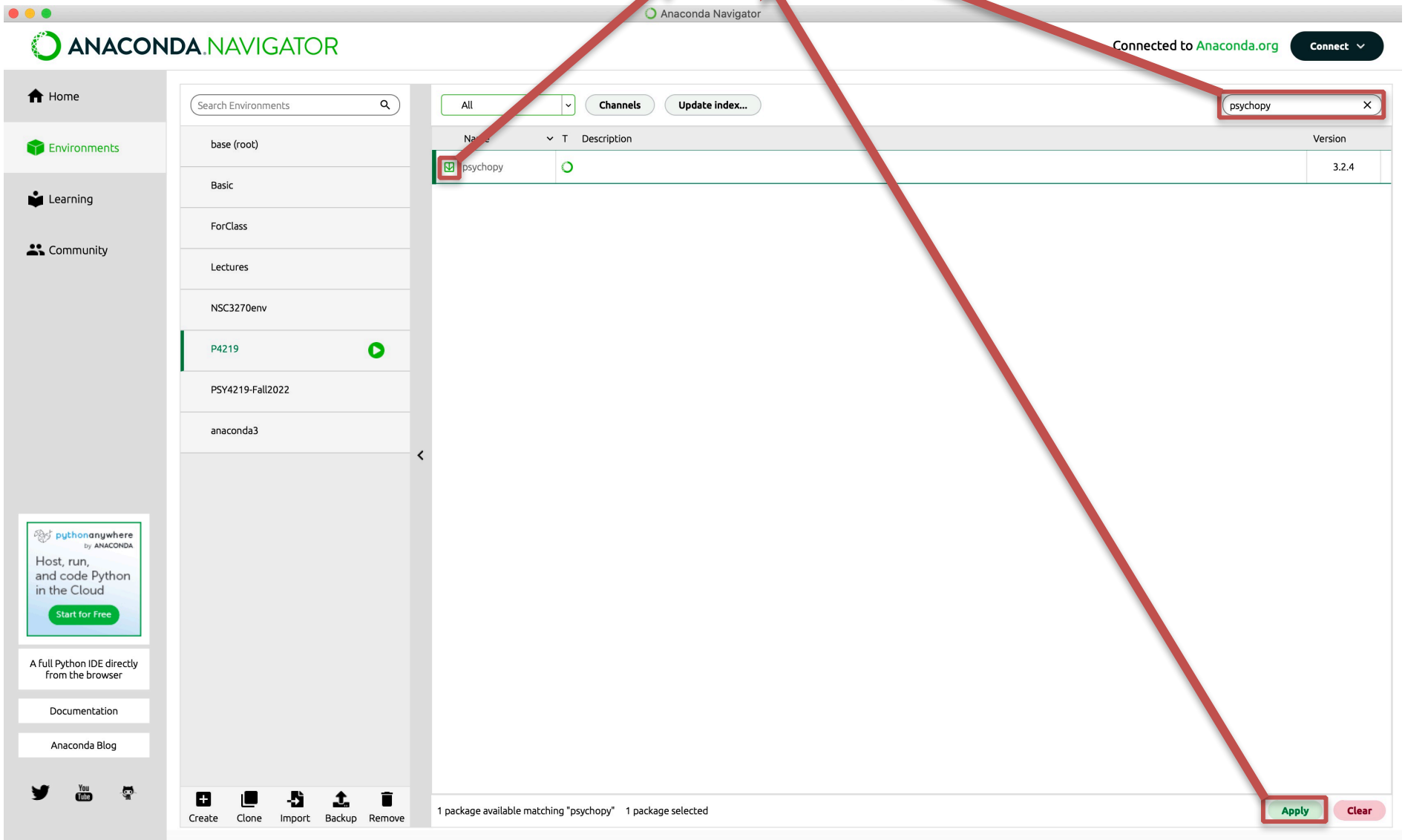
click on "Update channels"



https://docs.anaconda.com/anaconda/navigator/

virtual environments in Anaconda Navigator

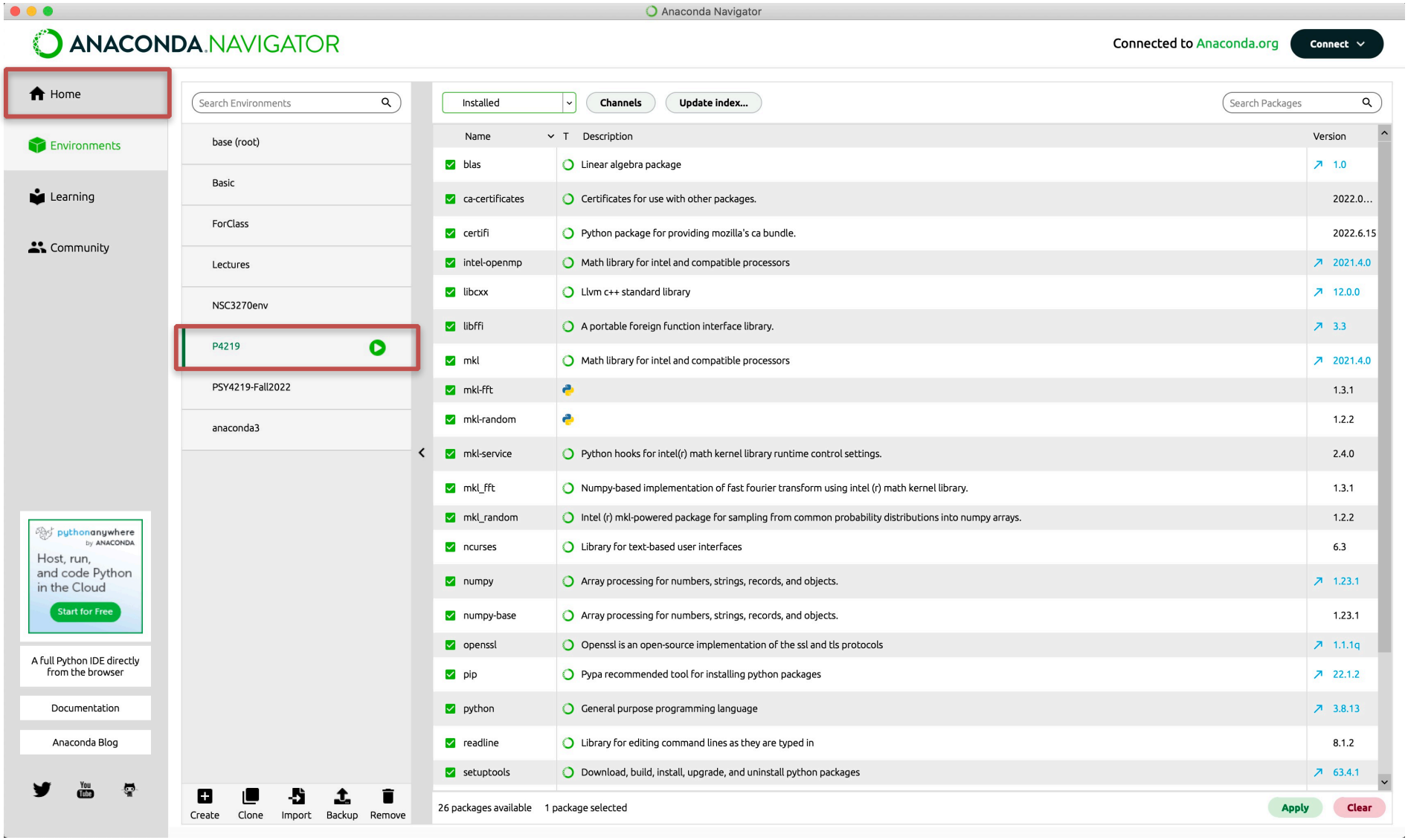
type in psychopy, click on it, and click "Apply"



https://docs.anaconda.com/anaconda/navigator/

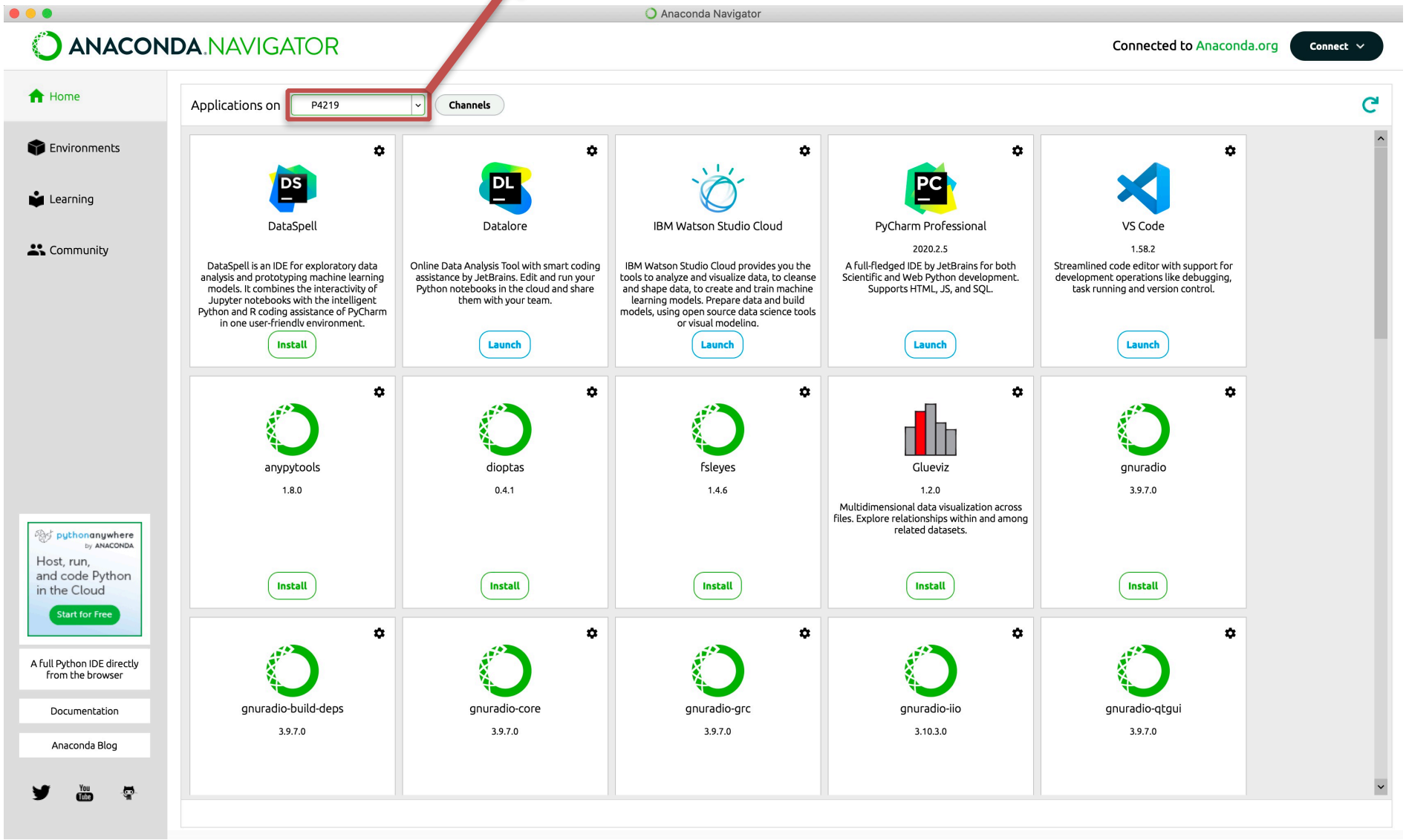
how do you use the virtual environment?

assuming it's selected, click on Home



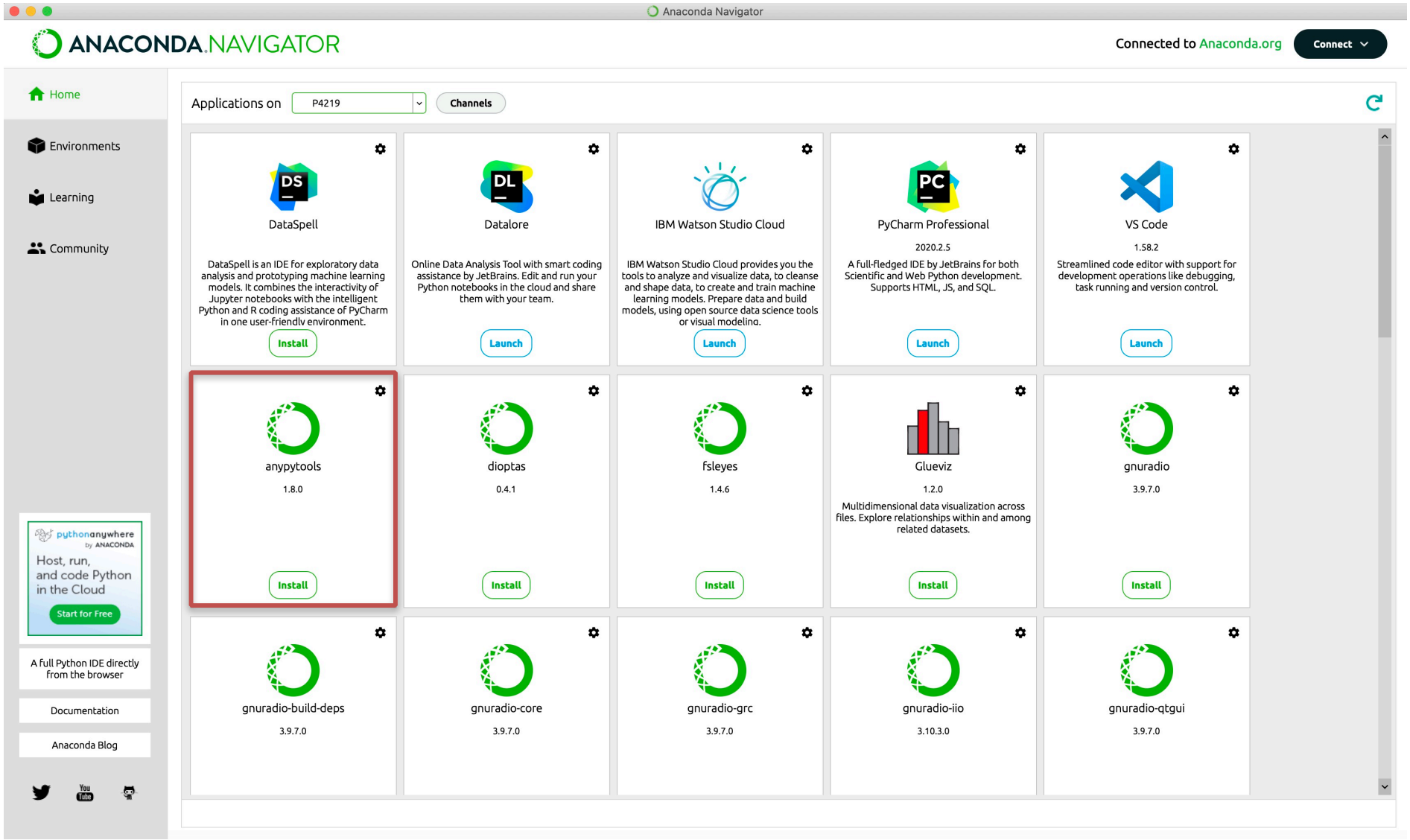
how do you use the virtual environment?

can see here that your new environment is selected (can also select others)



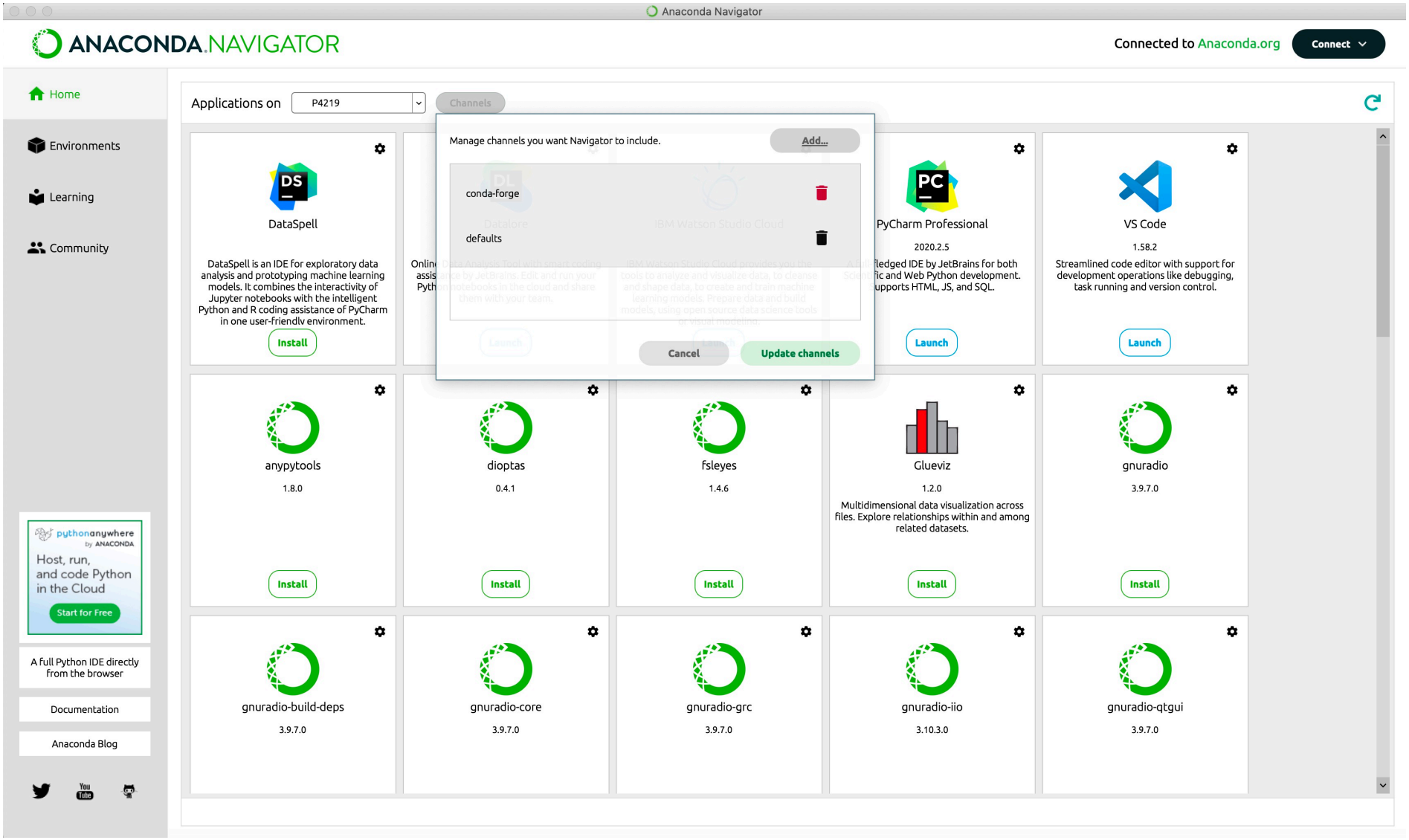
how do you use the virtual environment?

you might also see a bunch of other stuff on the dashboard (from conda-forge)



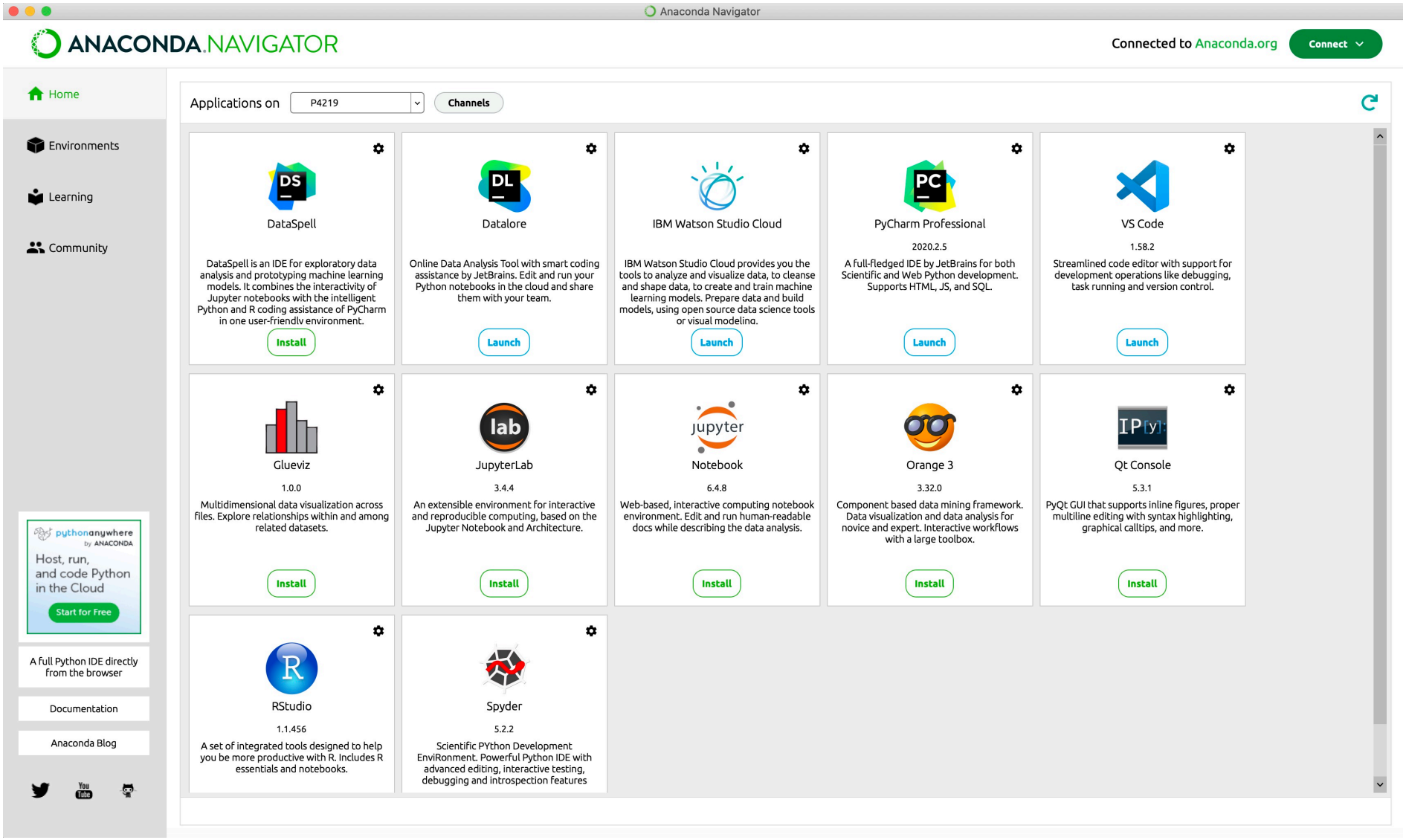
how do you use the virtual environment?

go into channels and delete conda-forge (trash can) and "Update channels"



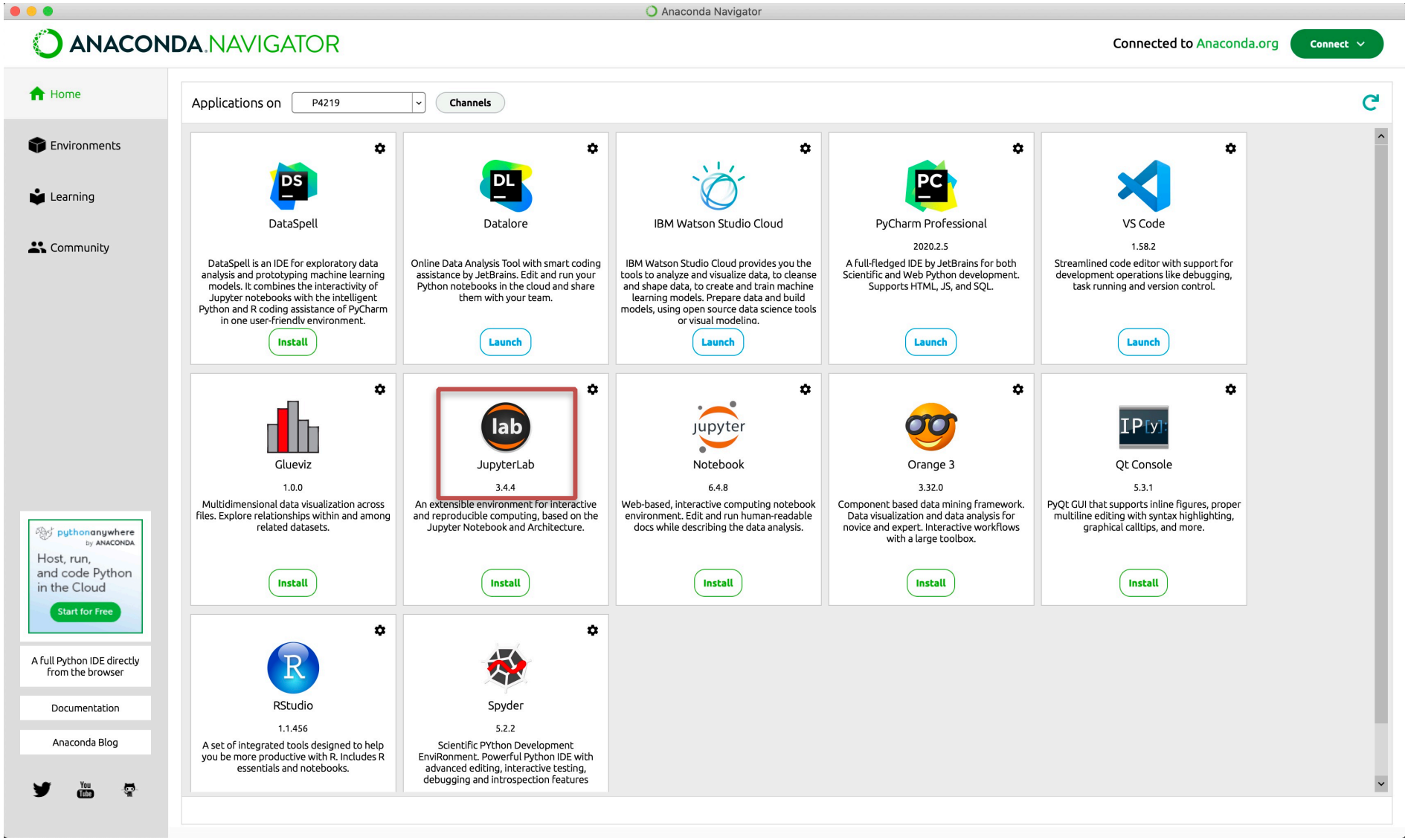
how do you use the virtual environment?

and they'll go away



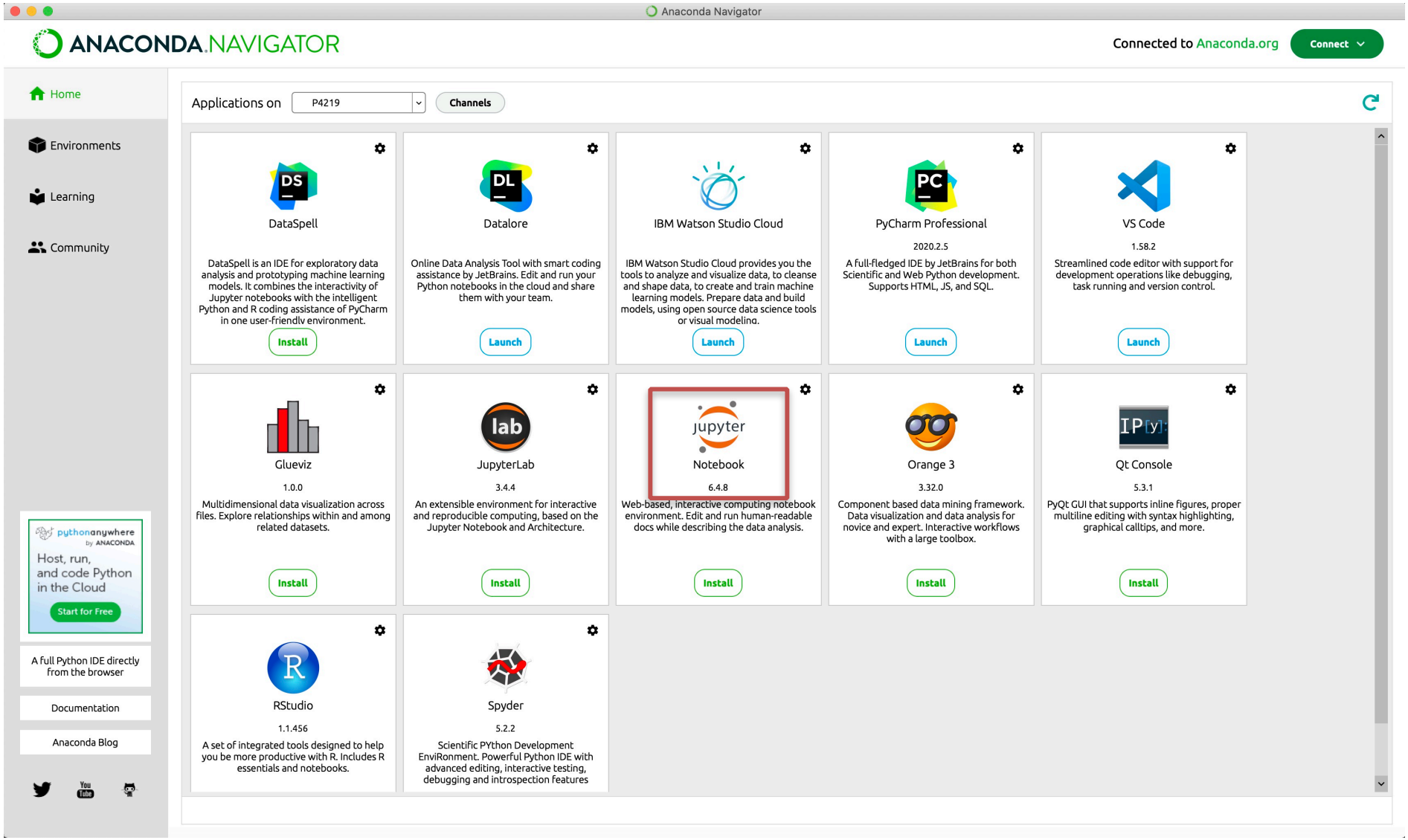
how do you use the virtual environment?

I'll be using JupyterLab



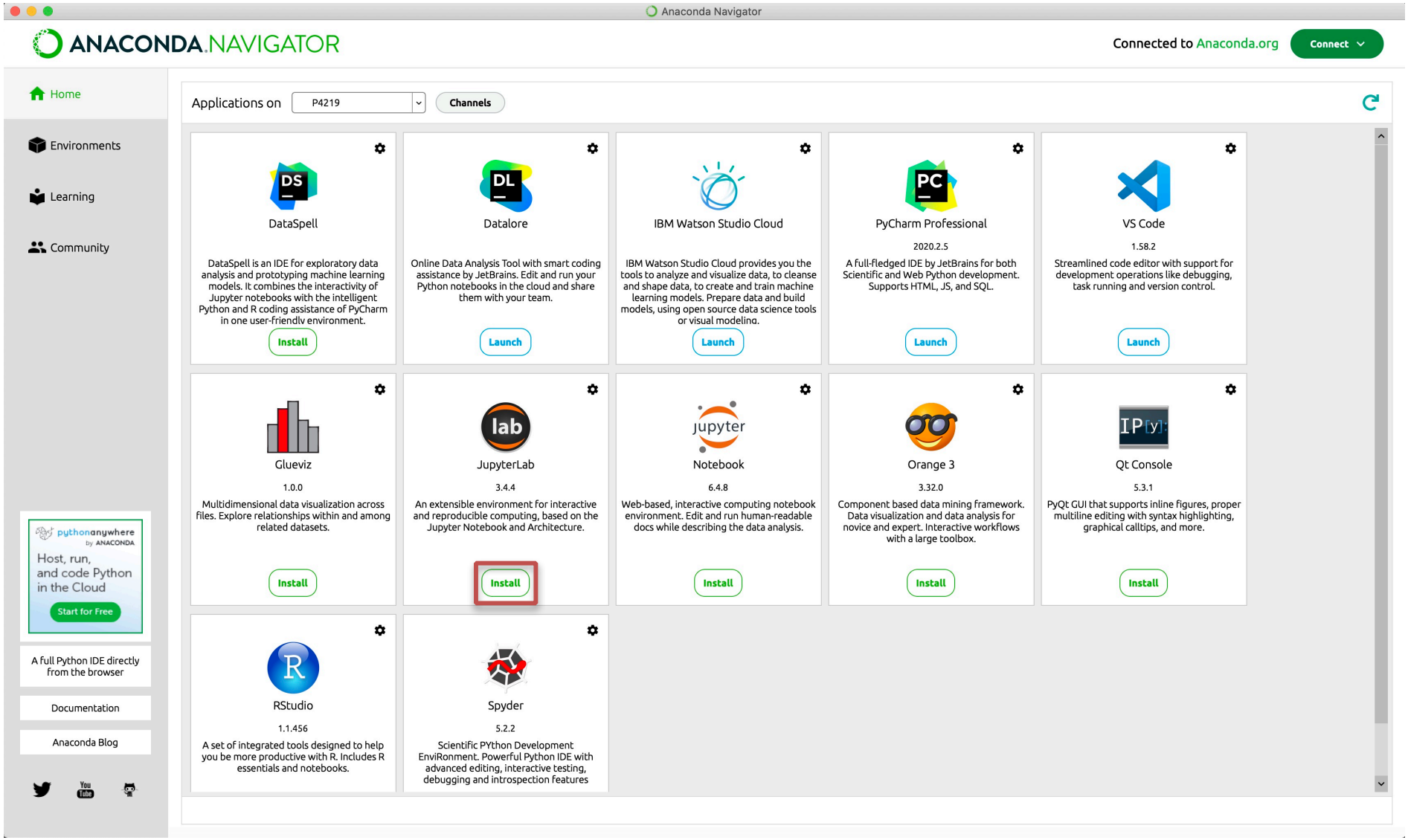
how do you use the virtual environment?

if you prefer Jupyter Notebook you can use that (JupyterLab has a bit more functionality)



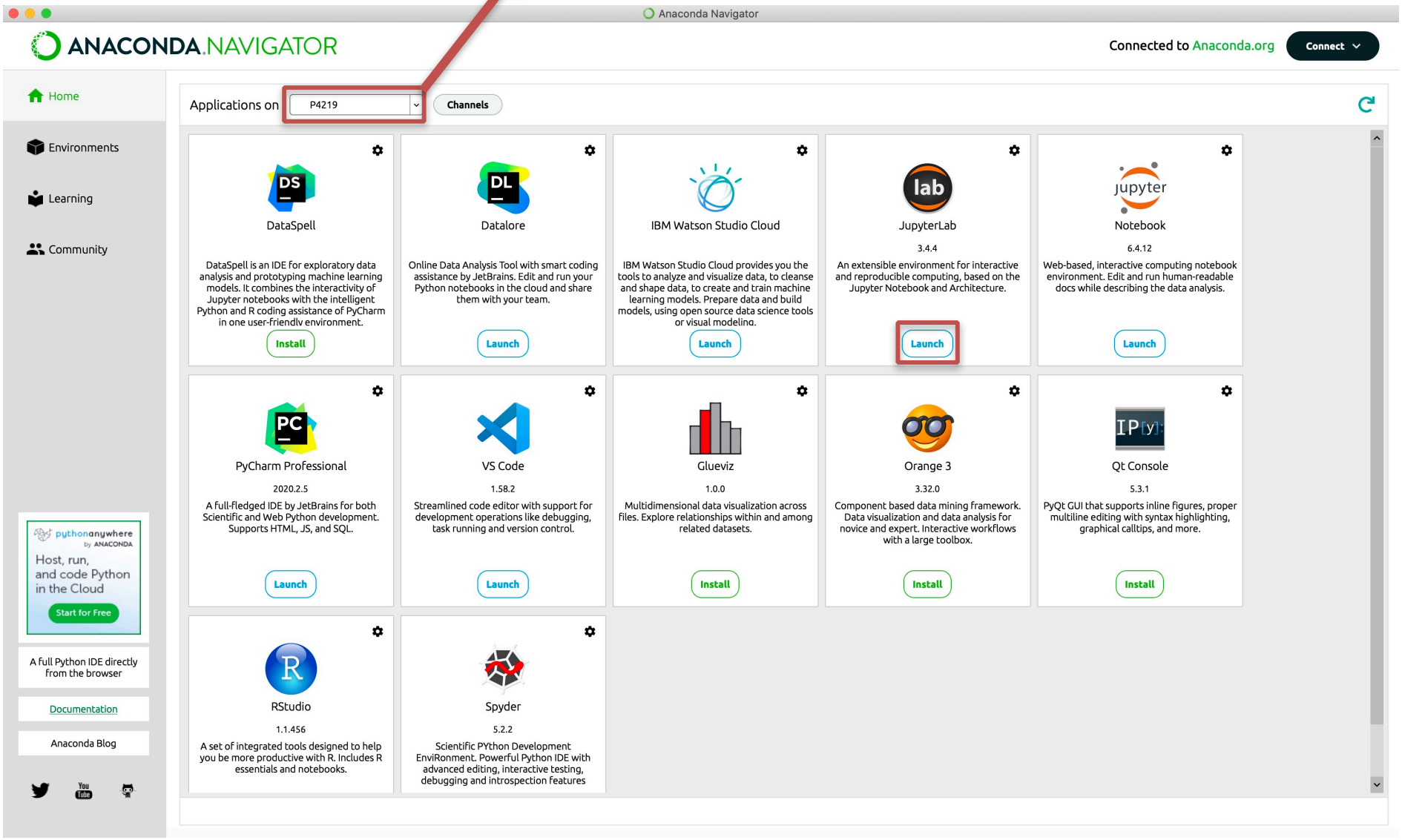
how do you use the virtual environment?

you'll need to "Install" (really adding more packages to your new environment)



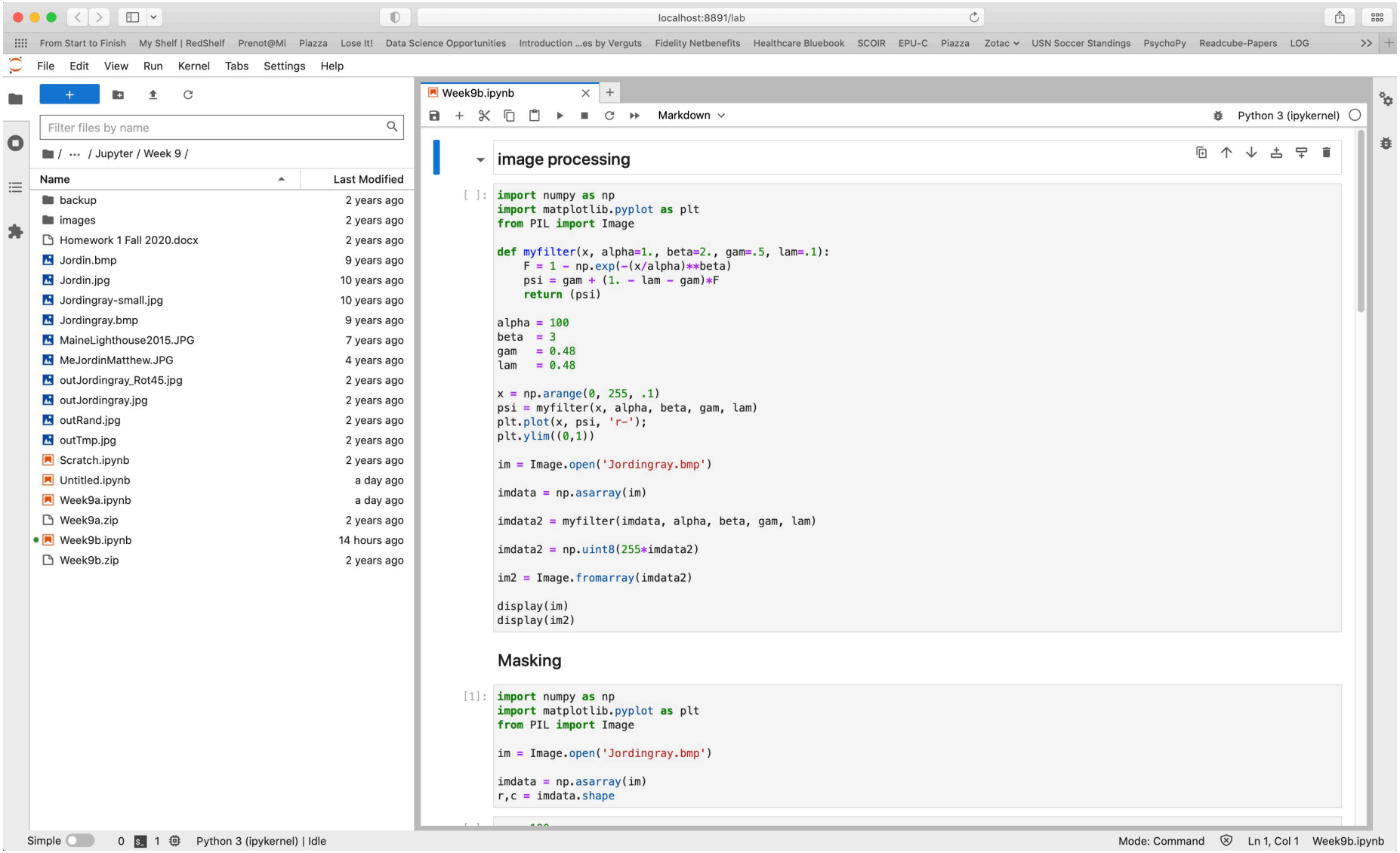
launch JupyterLab (or Jupyter Notebook)

using this environment

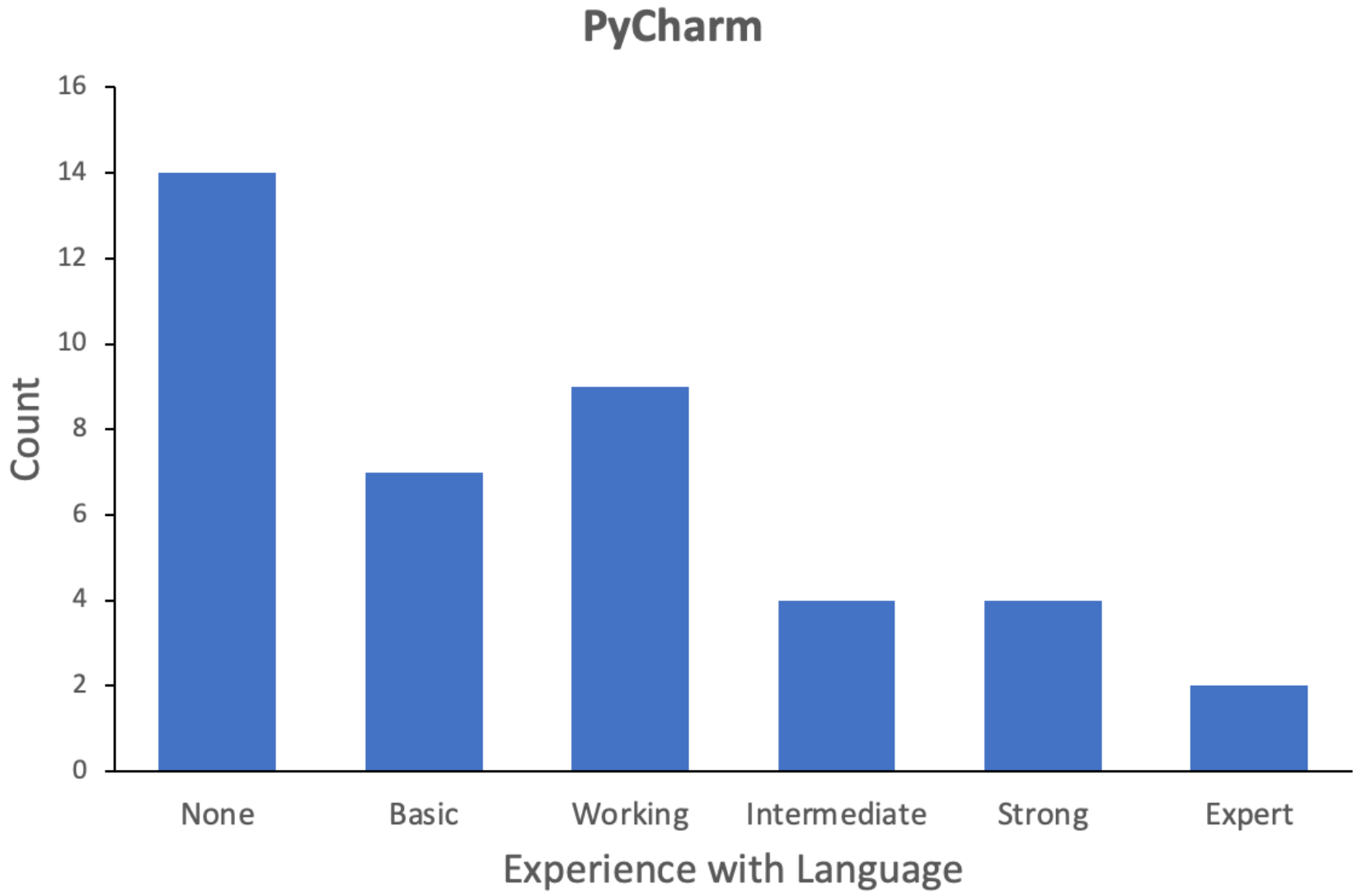


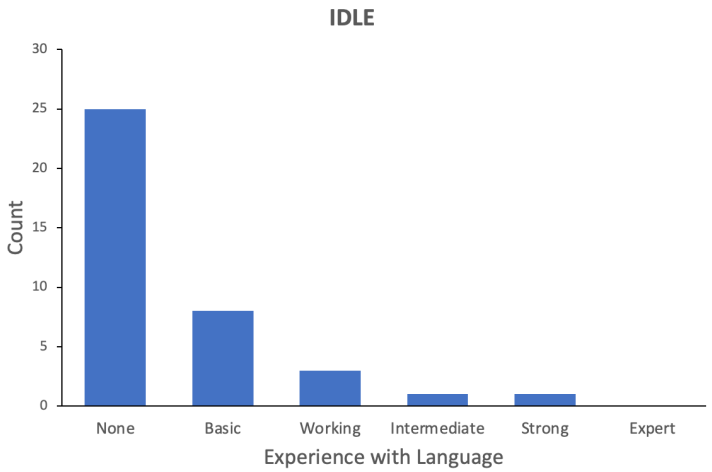
Jupyter (Lab and Notebook) runs in a browser

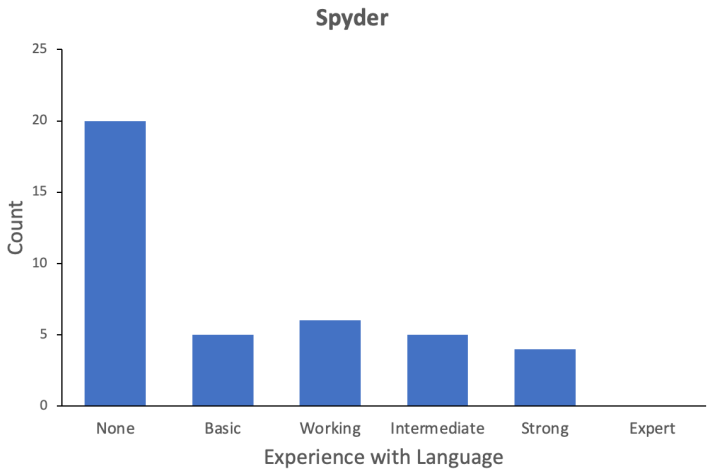
we'll talk about using Jupyter in a bit

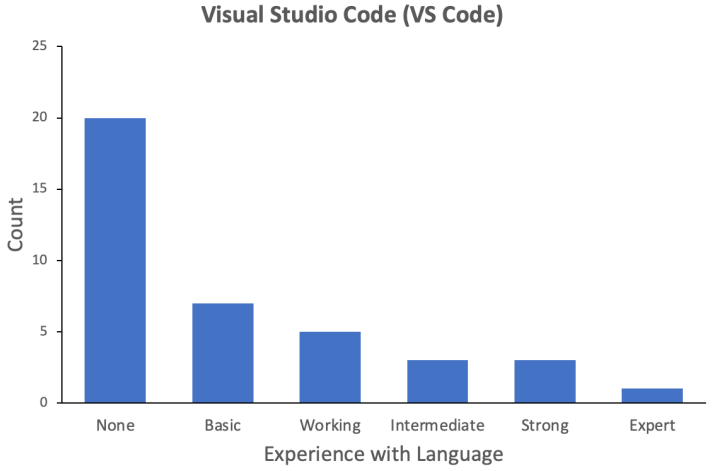


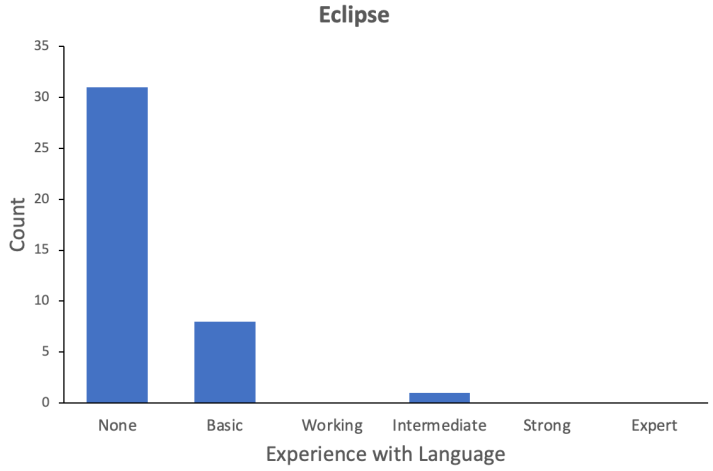
Integrated Development Environments

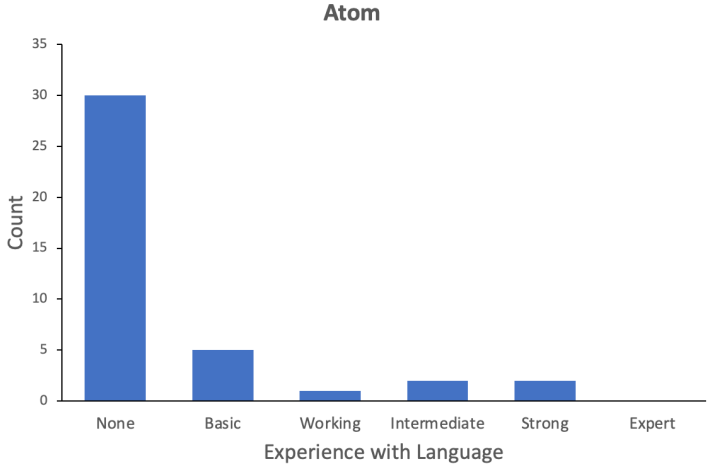


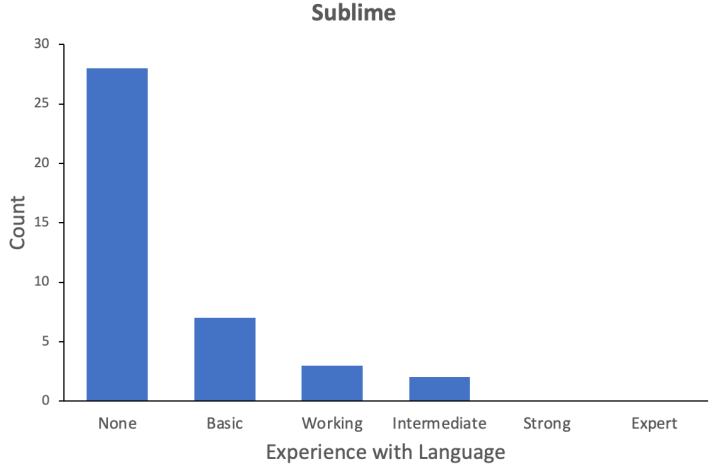

IDLE
Count
30
25
20
15
10
5
0
None
Basic
Working
Intermediate
Strong
Expert
Experience with Language


Spyder
Count
25
20
15
10
5
0
None
Basic
Working
Intermediate
Strong
Expert
Experience with Language


Visual Studio Code (VS Code)
Count
25
20
15
10
5
0
None
Basic
Working
Intermediate
Strong
Expert
Experience with Language


Eclipse
Count
35
30
25
20
15
10
5
0
None
Basic
Working
Intermediate
Strong
Expert
Experience with Language


Atom
Count
35
30
25
20
15
10
5
0
None
Basic
Working
Intermediate
Strong
Expert
Experience with Language


Sublime
Count
30
25
20
15
10
5
0
None
Basic
Working
Intermediate
Strong
Expert
Experience with Language

Integrated Development Environments

- edit code (often with hints, autocompletion, instant documentation)
- run code
- debug code (break points, stepping through code, watching variable)
- Python terminal
- more advanced tools

PyCharm

if you want to be able to follow along in PyCharm (using PyCharm is not required)

while we will not use PyCharm for a while, I would like you to get PyCharm (or another IDE) working now at the start of the semester (to address issues)

the main thing we're trying to do now is to make sure PyCharm uses the virtual environment we just created

(if you are using another IDE, you will need to make sure you set up that IDE to use that environment - Jason knows VS Code, but for other IDEs you will need to play around on your own or use Piazza for help)

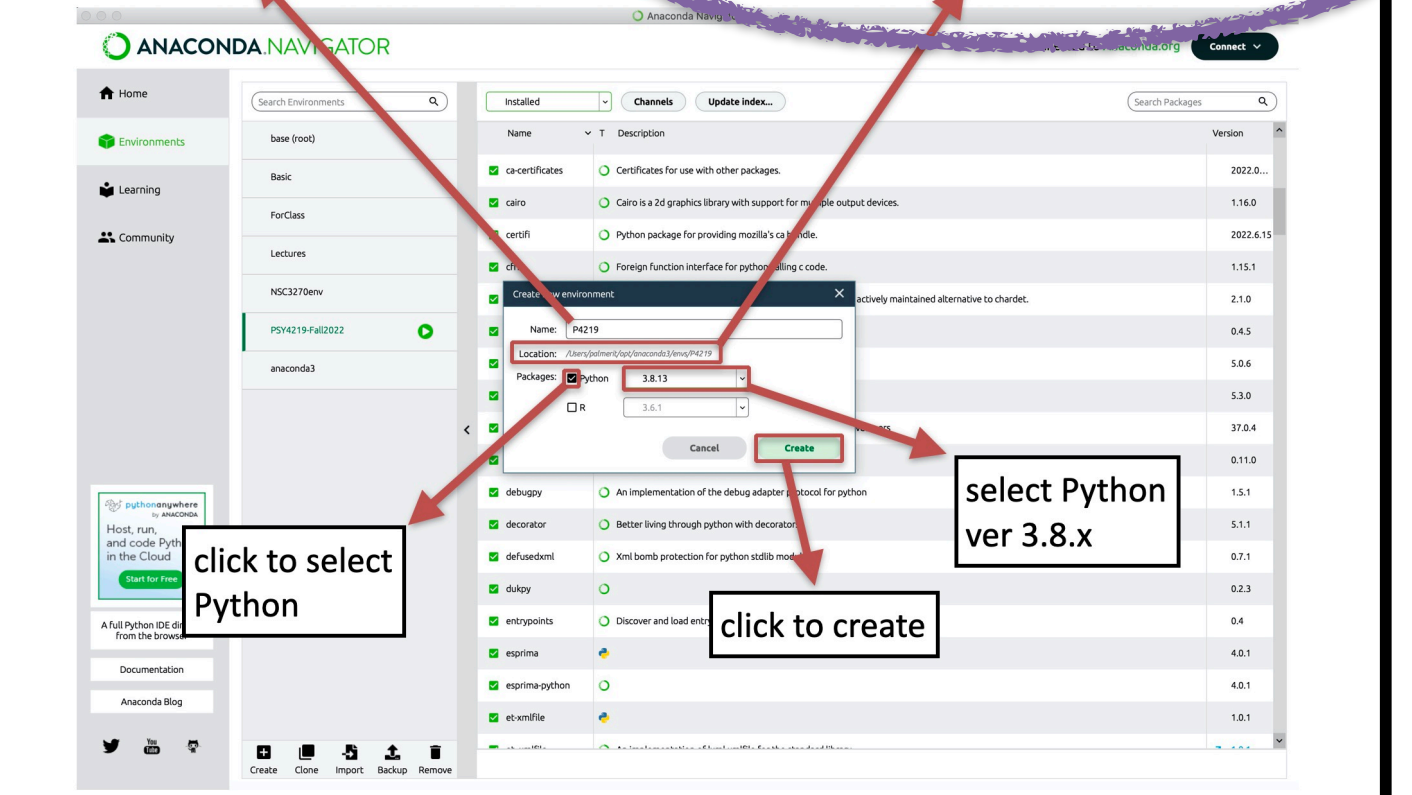
Installing PyCharm

- Do not install PyCharm from Anaconda (to ensure that you install the full free educational version)
- https://www.jetbrains.com/shop/eform/students
- make sure you register using your vanderbilt.edu email address to get the full featured free educational version
- we will talk about using PyCharm later (feel free to explore on your own)

Setting up PyCharm

remember when I said you need to write down the location?
--- you'll need it now ---

virtual environments in Anaconda Navigator



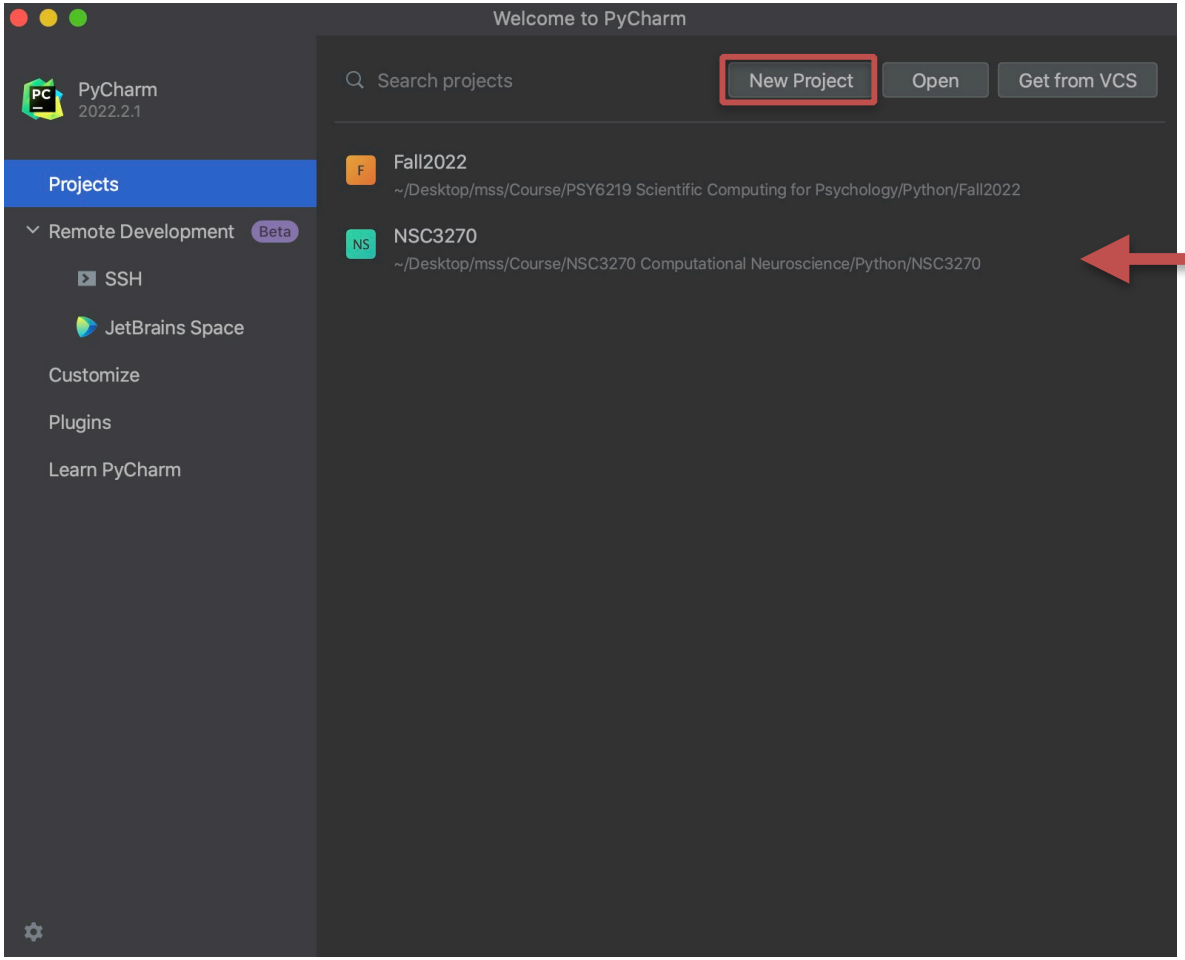
https://docs.anaconda.com/anaconda/navigator/

Setting up PyCharm

launch Python ... click "New Project"

note that after you create a project, PyCharm will open with that project (until you close it)

you can create a new project for every assignment
or use the same project for every assignment

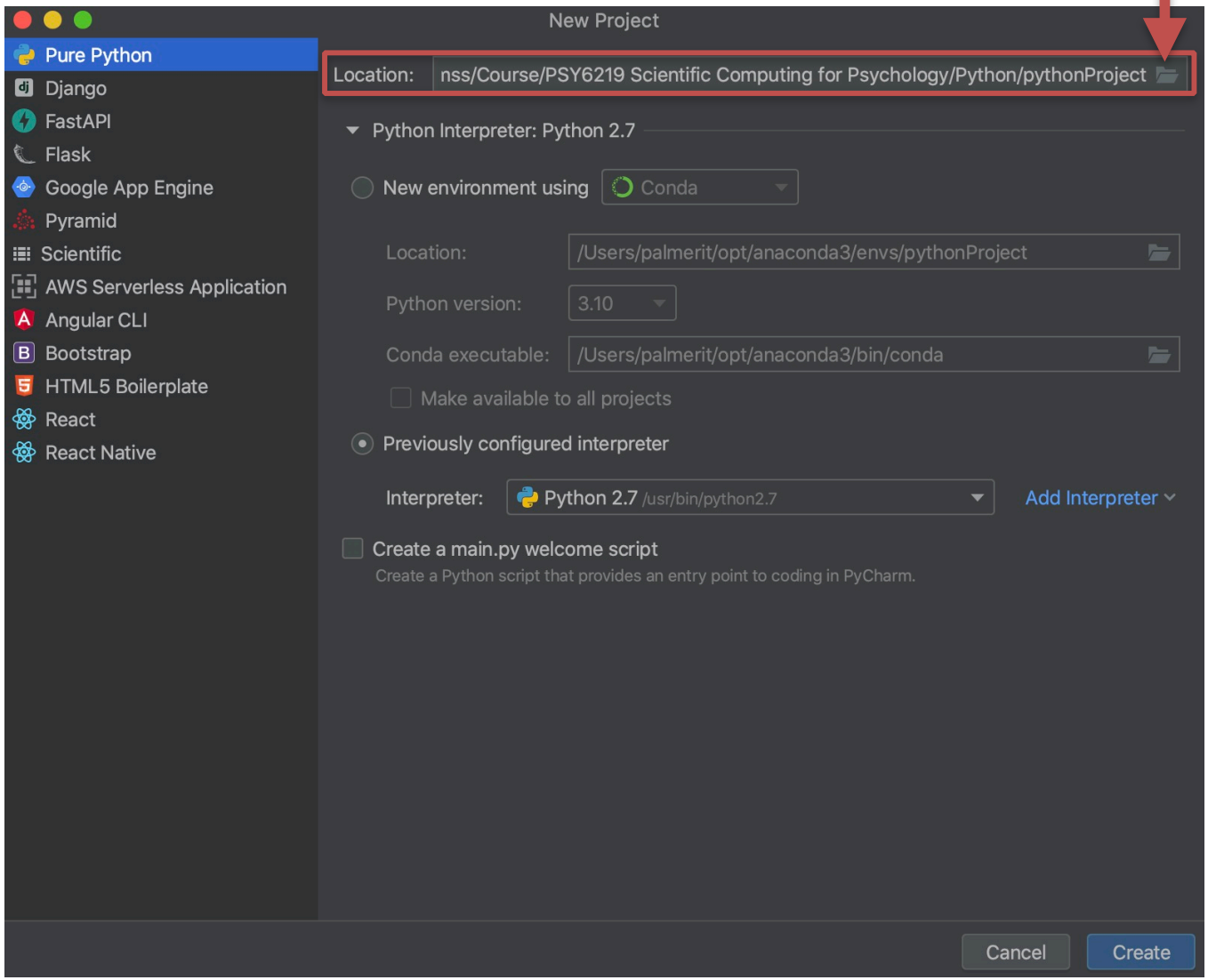


existing project will be listed here

Setting up PyCharm

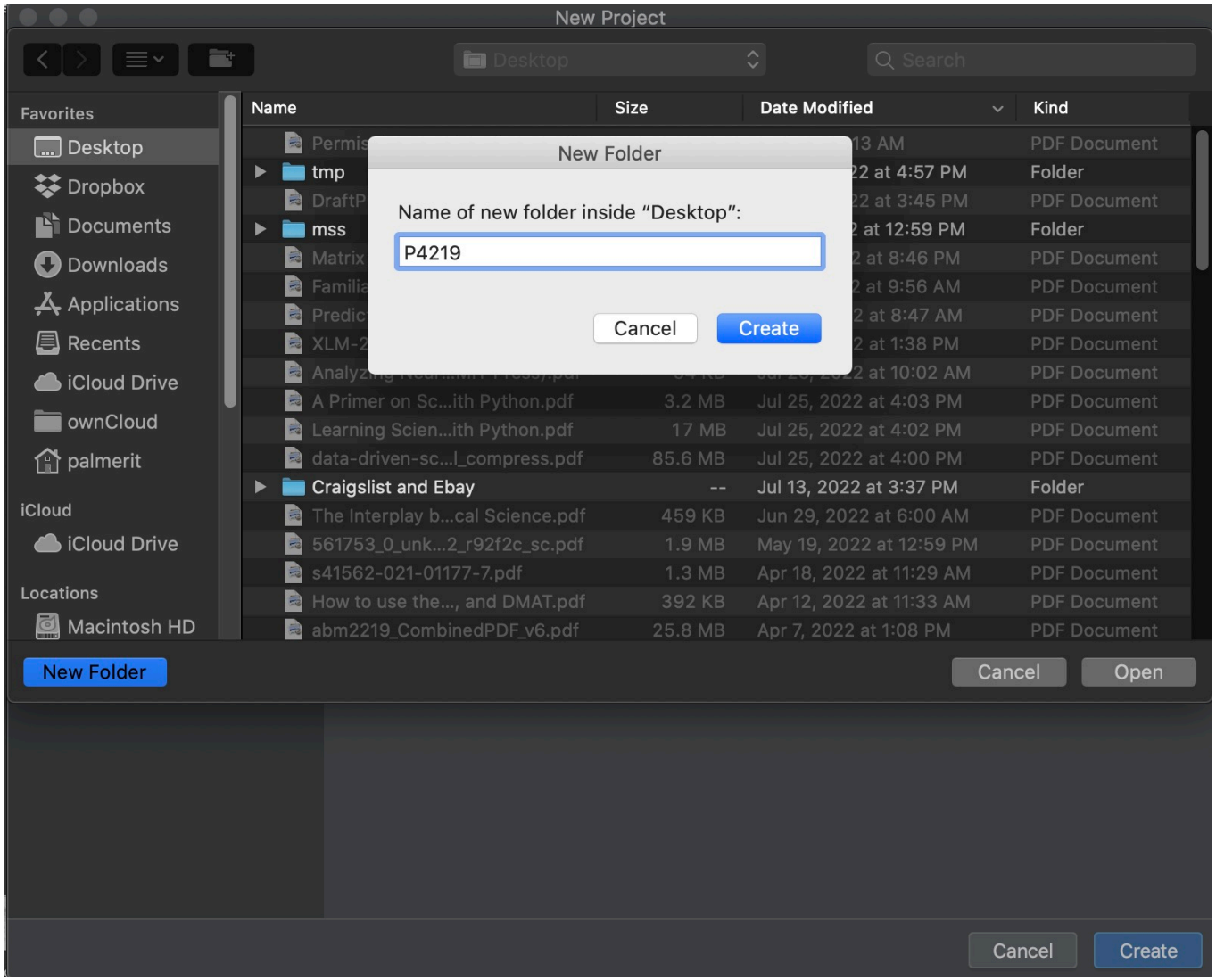
select a location for your project
(the folder/directory where your code will be stored)

click here



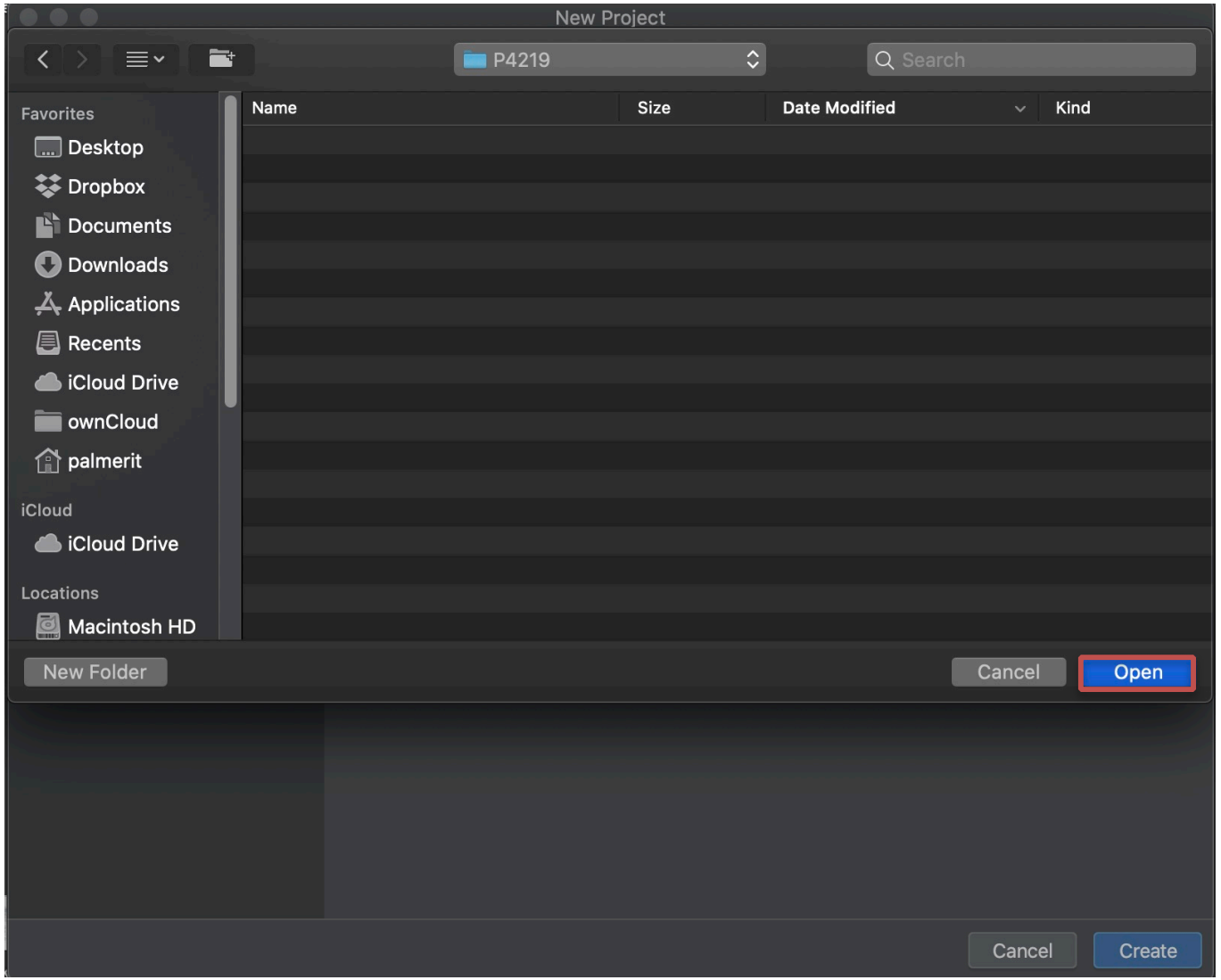
Setting up PyCharm

here, I'm creating a new folder
you can also select an existing folder



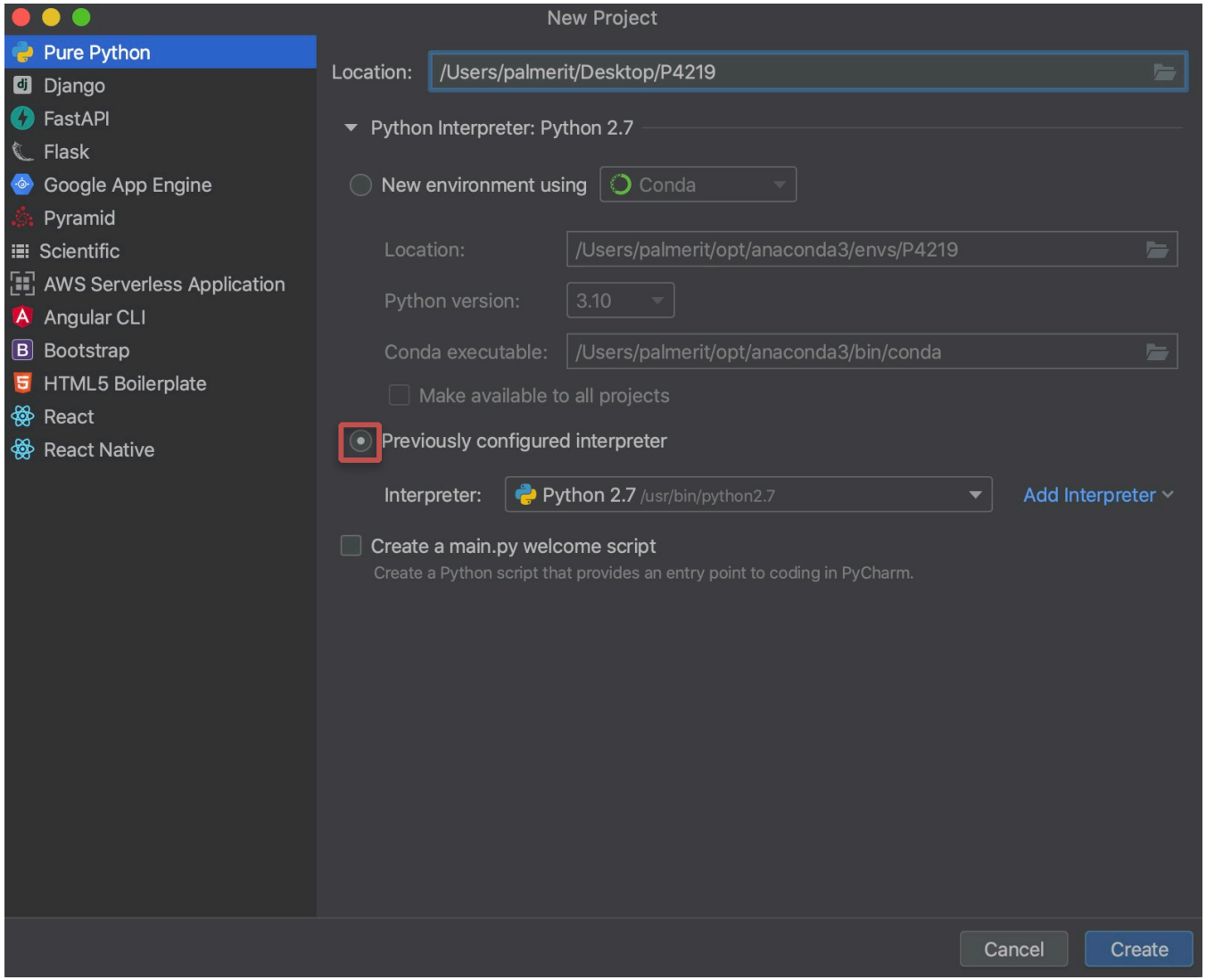
Setting up PyCharm

click "Open"



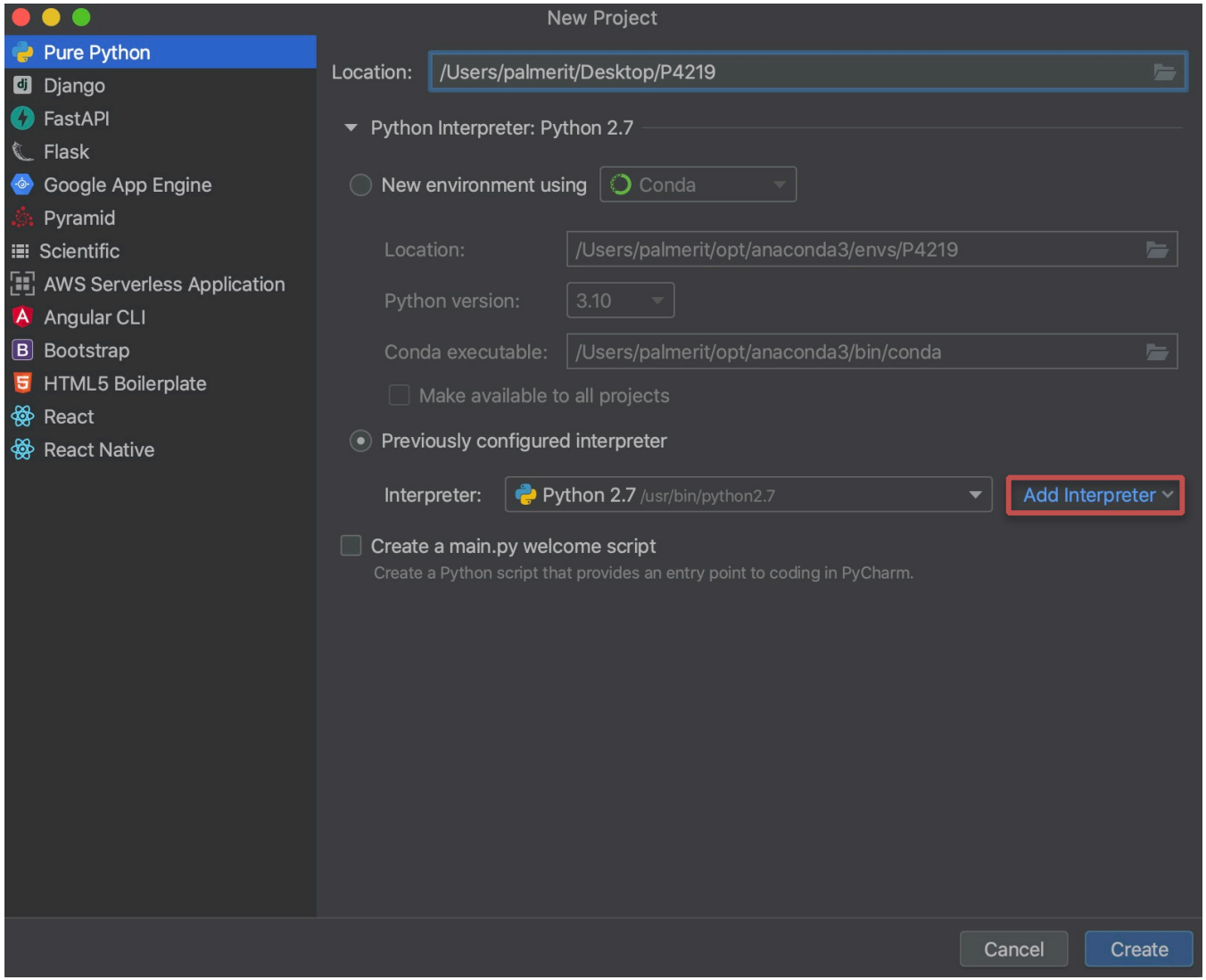
Setting up PyCharm

select "Previously configured interpreter"
(environment)



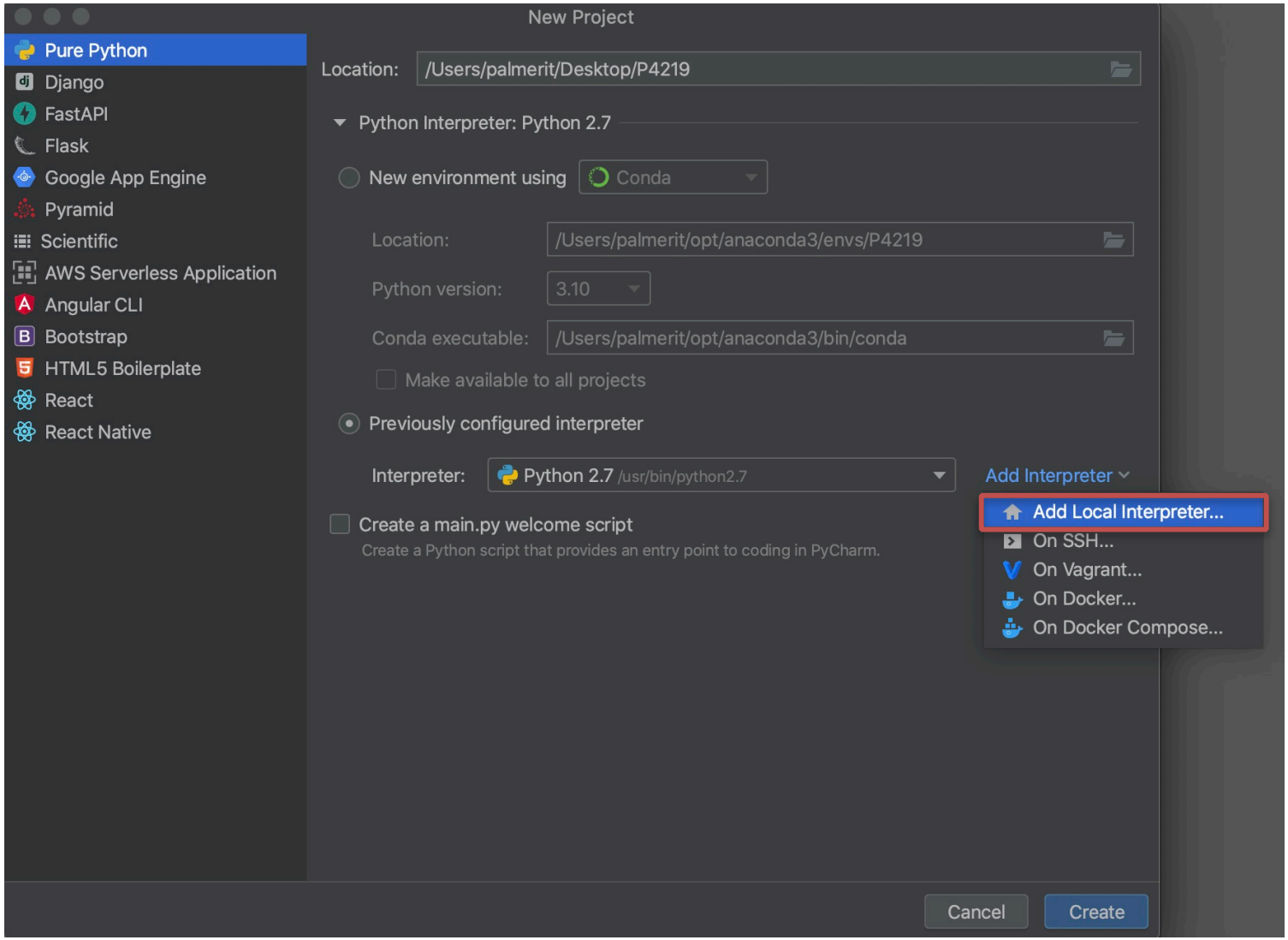
Setting up PyCharm

click "Add Interpreter"



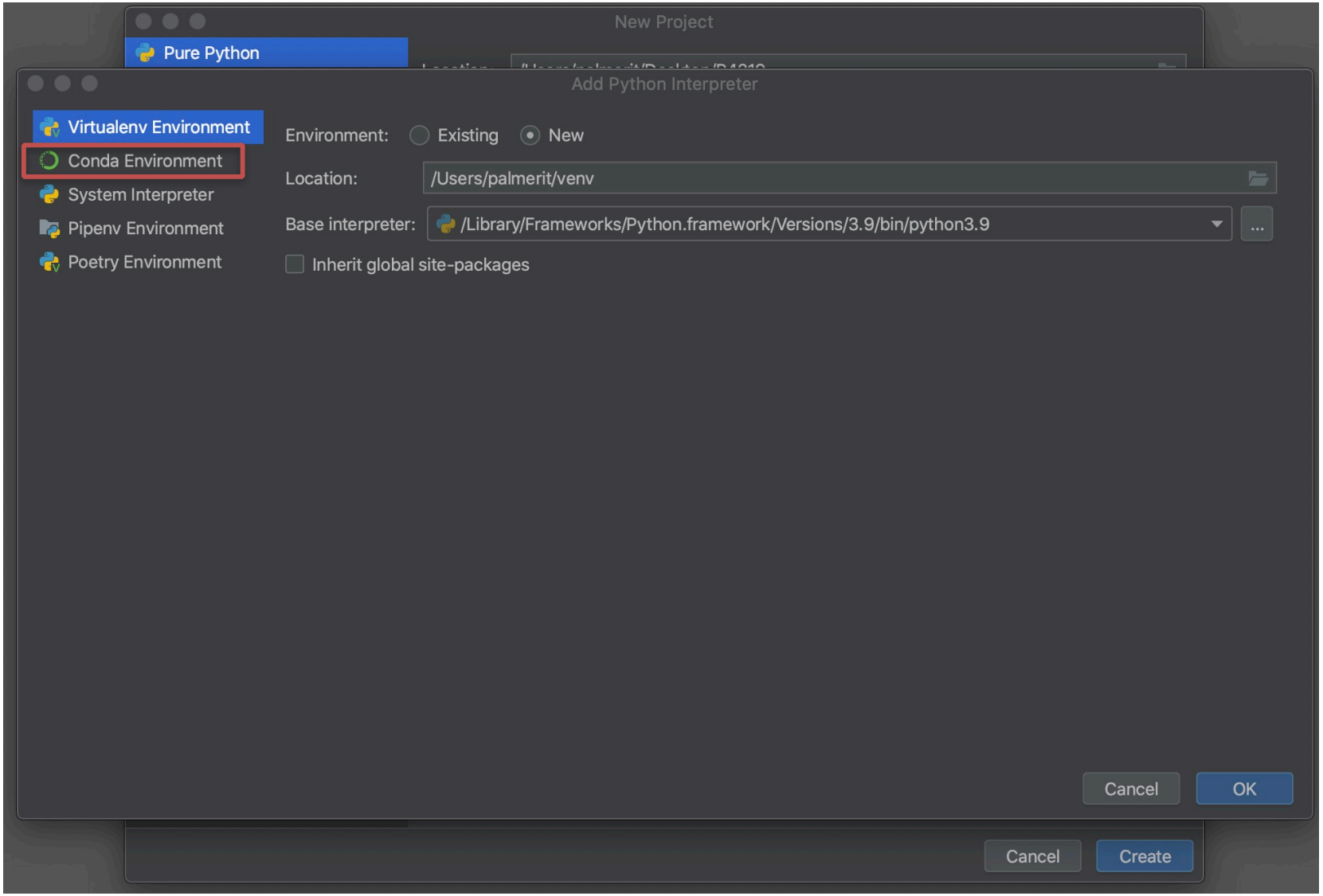
Setting up PyCharm

click "Add Local Interpreter..."



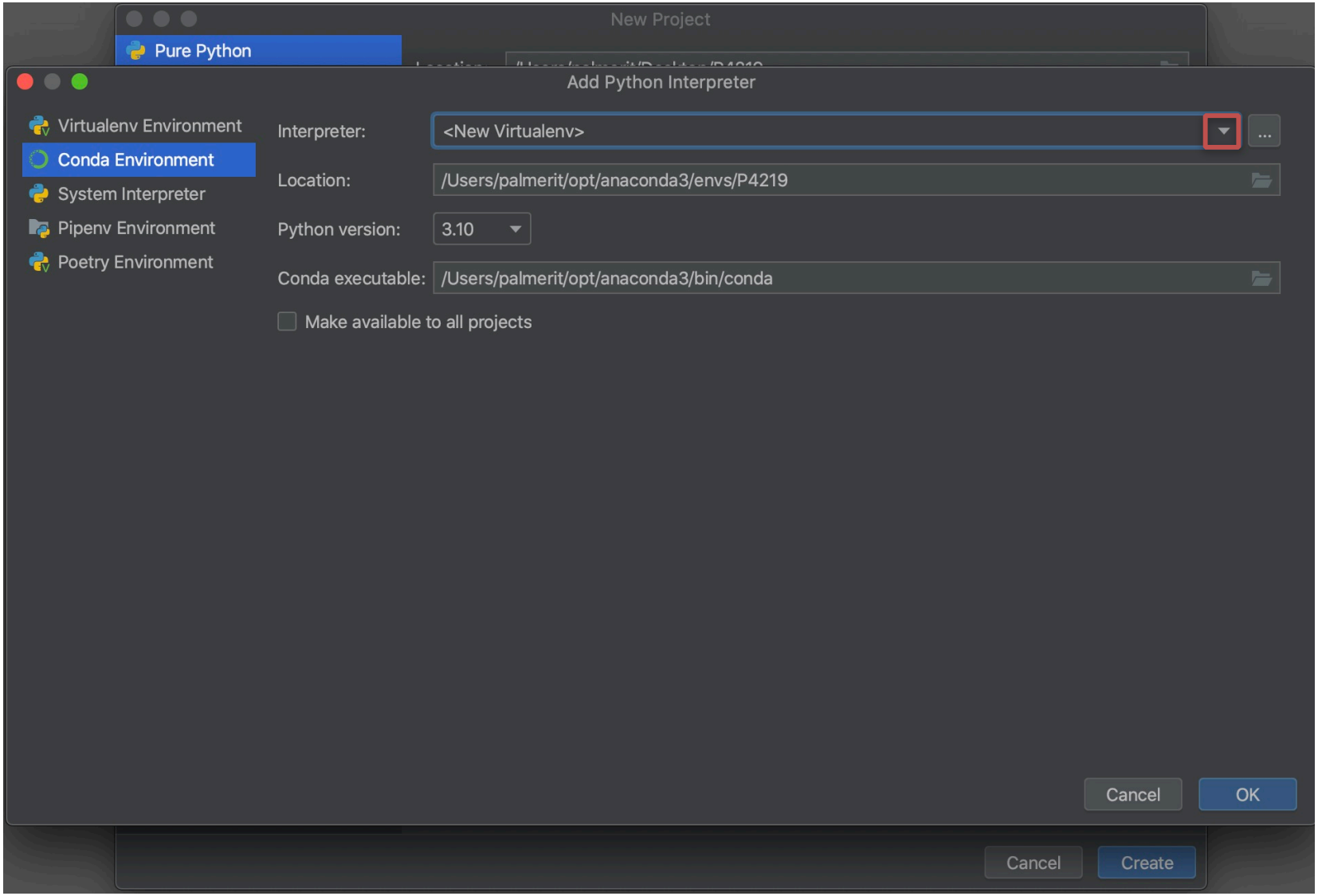
Setting up PyCharm

select "Conda Environment"



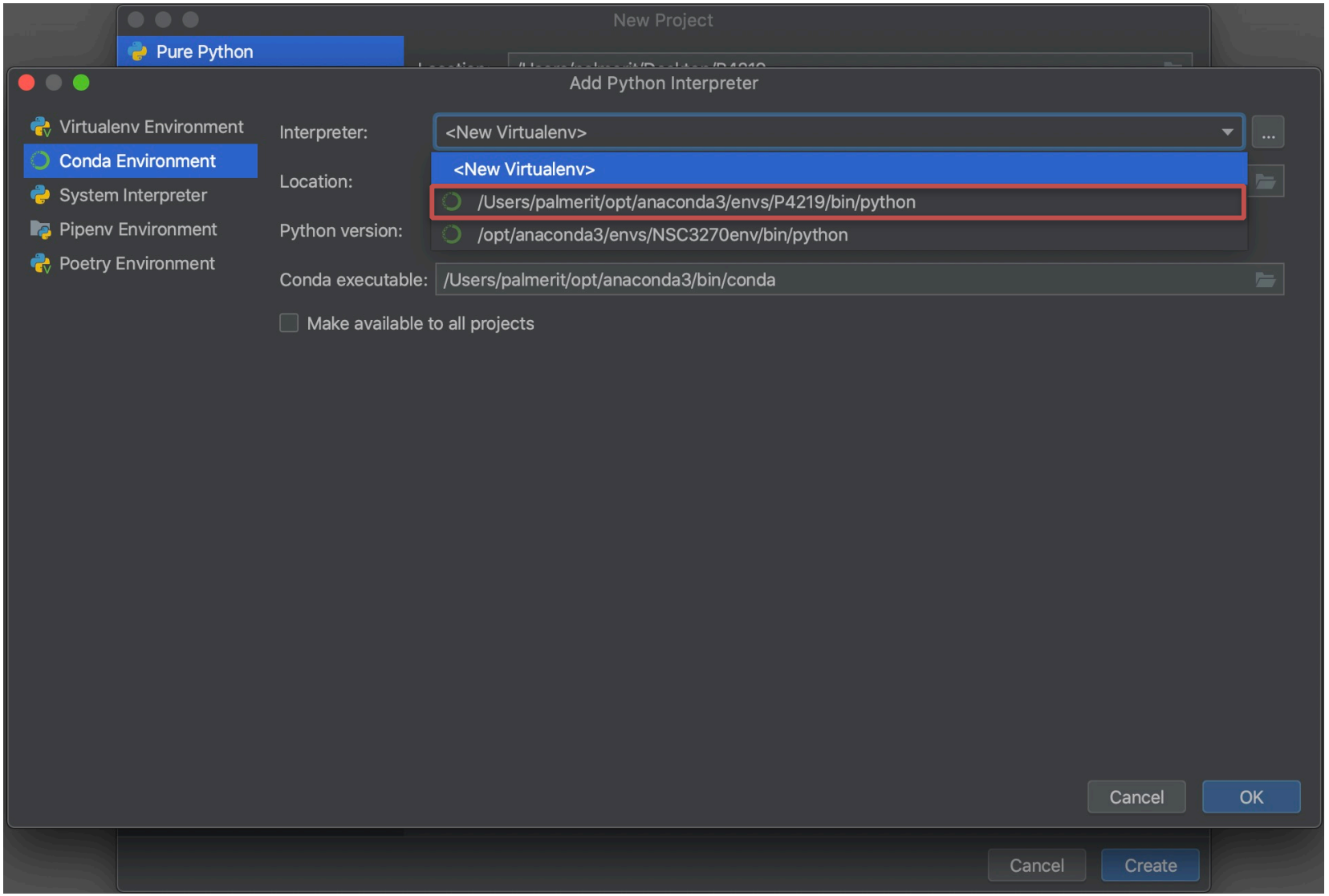
Setting up PyCharm

select Interpreter



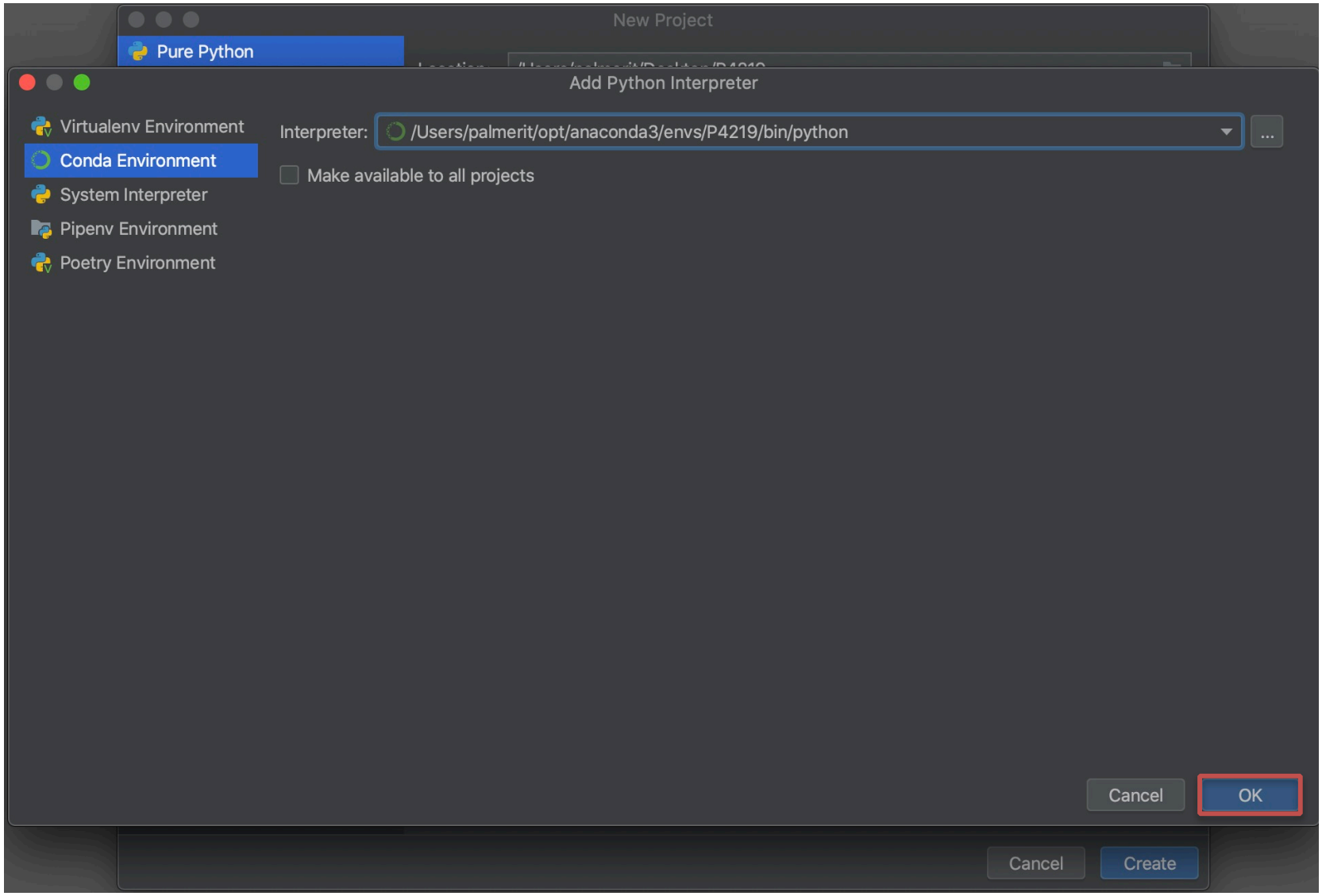
Setting up PyCharm

select the Python in the folder for your virtual environment



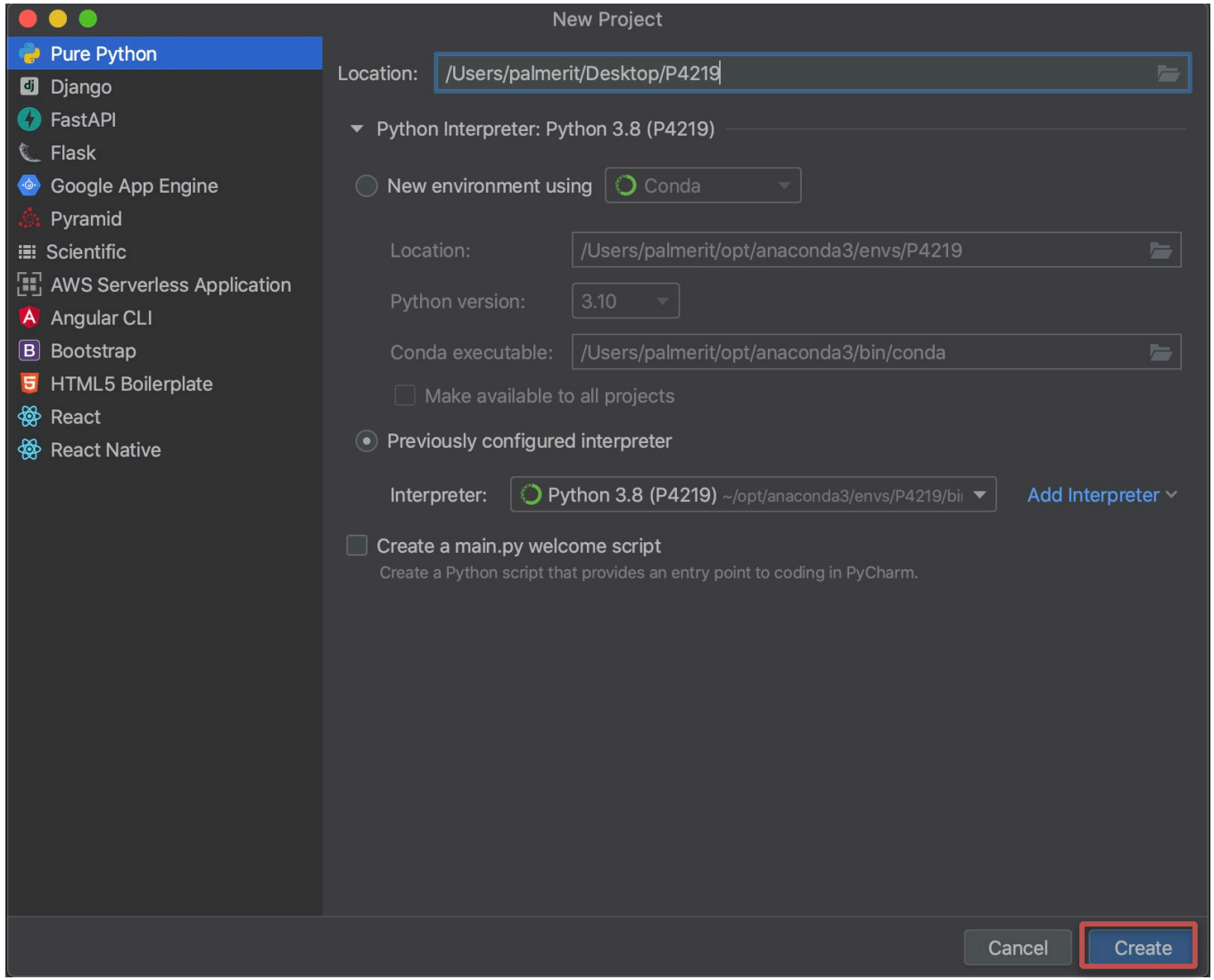
Setting up PyCharm

click "OK"



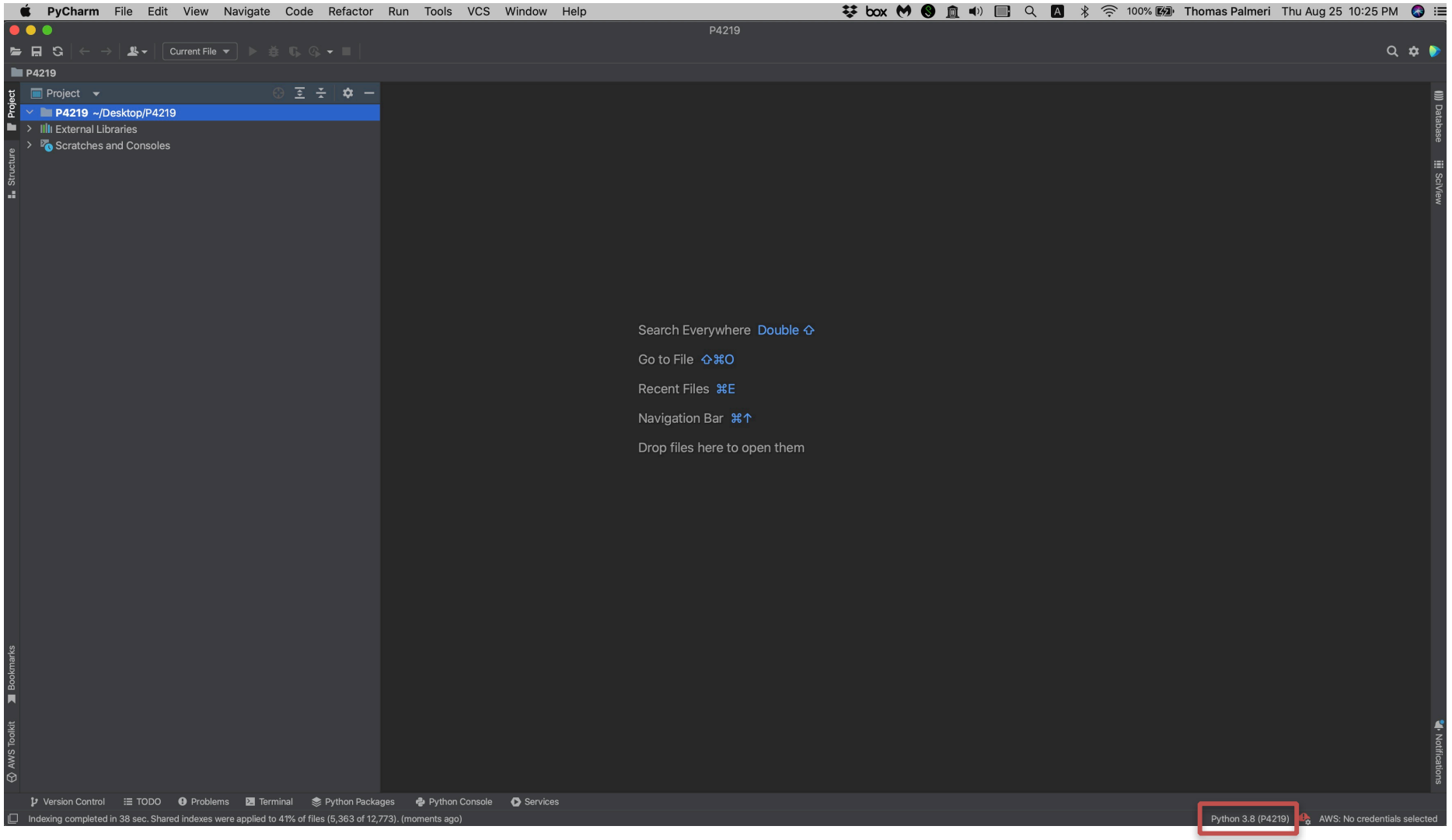
Setting up PyCharm

click "Create"



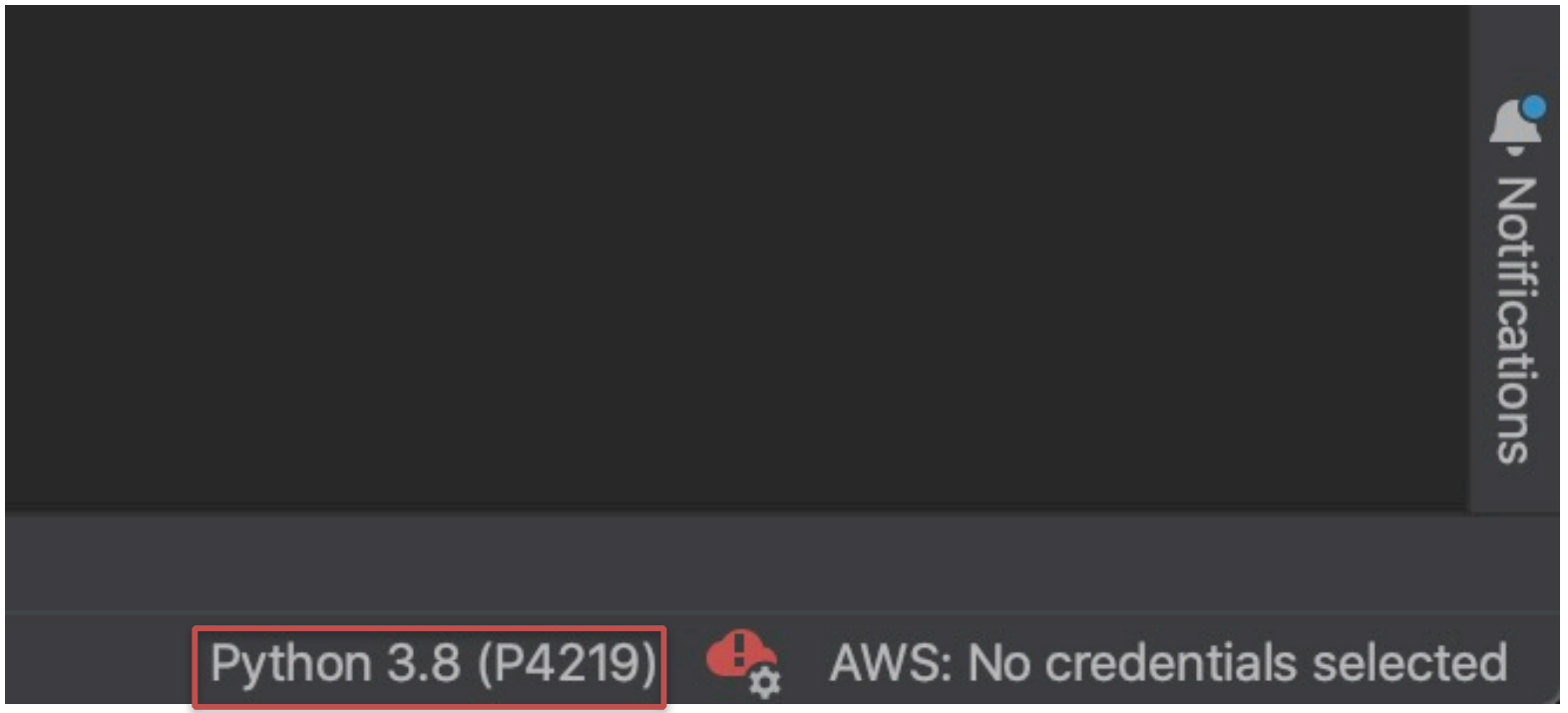
Setting up PyCharm

confirm that your environment is selected in lower right corner



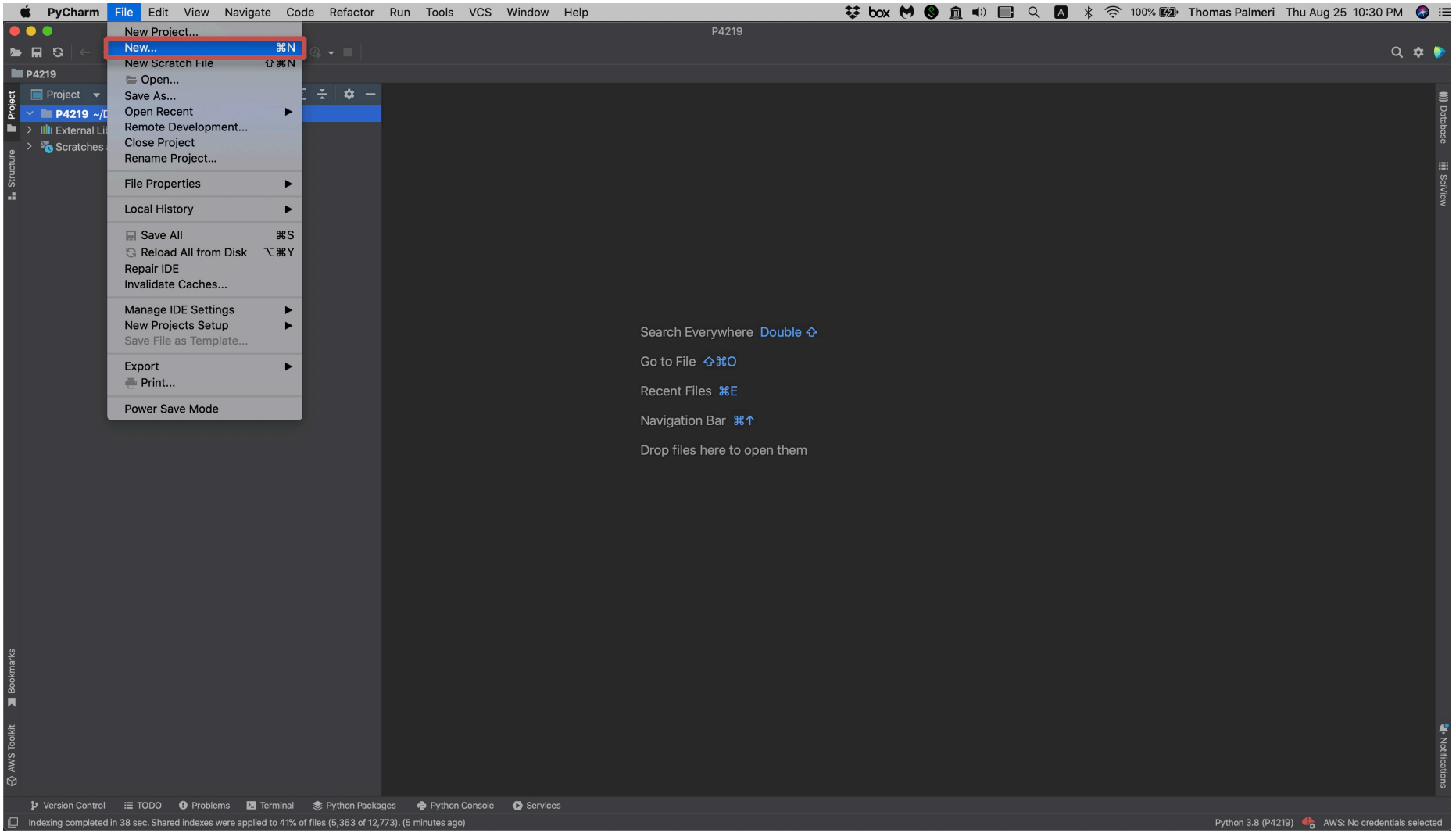
Setting up PyCharm

confirm that your environment is selected in lower right corner



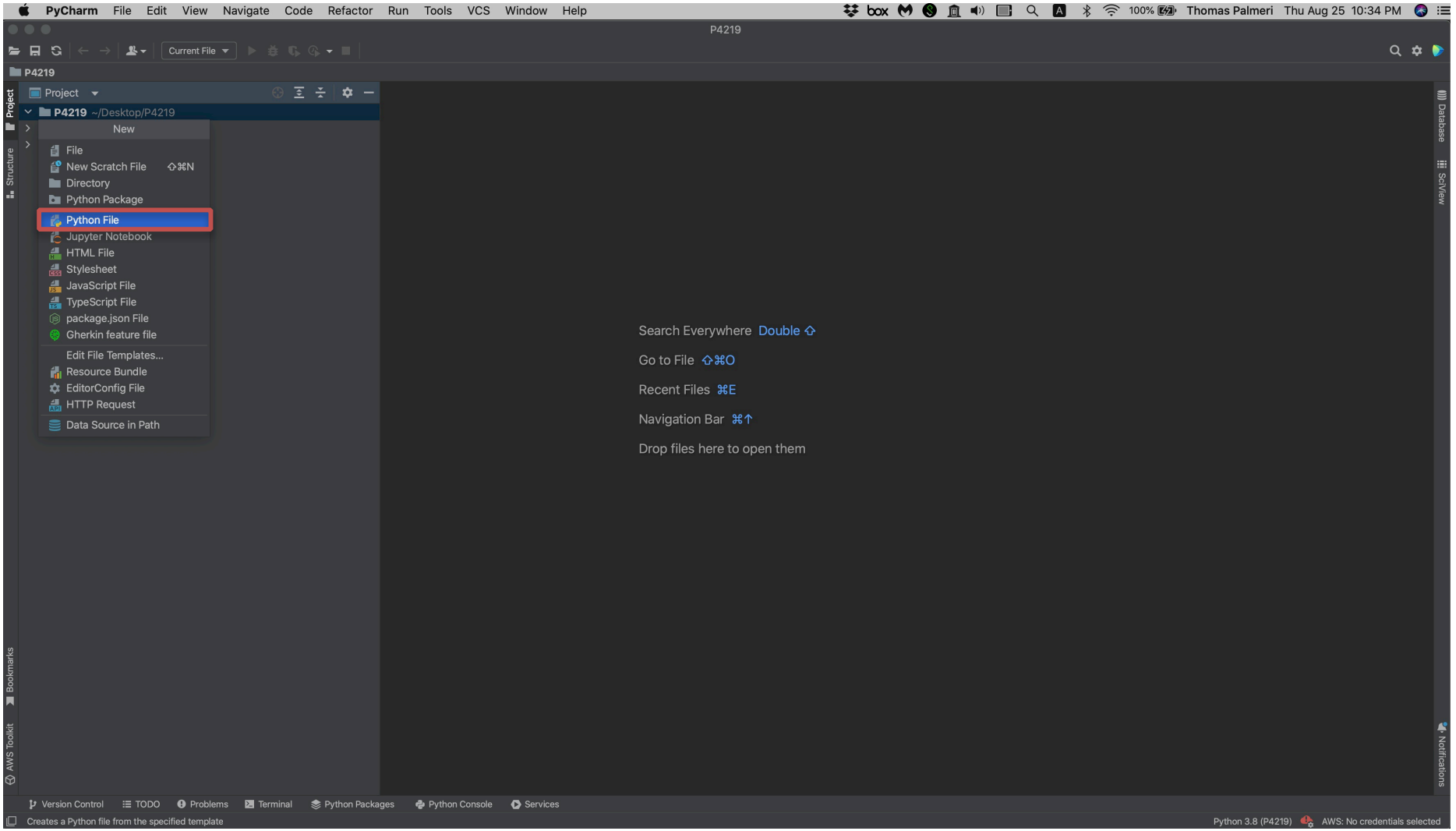
Setting up PyCharm

select "File --> New"



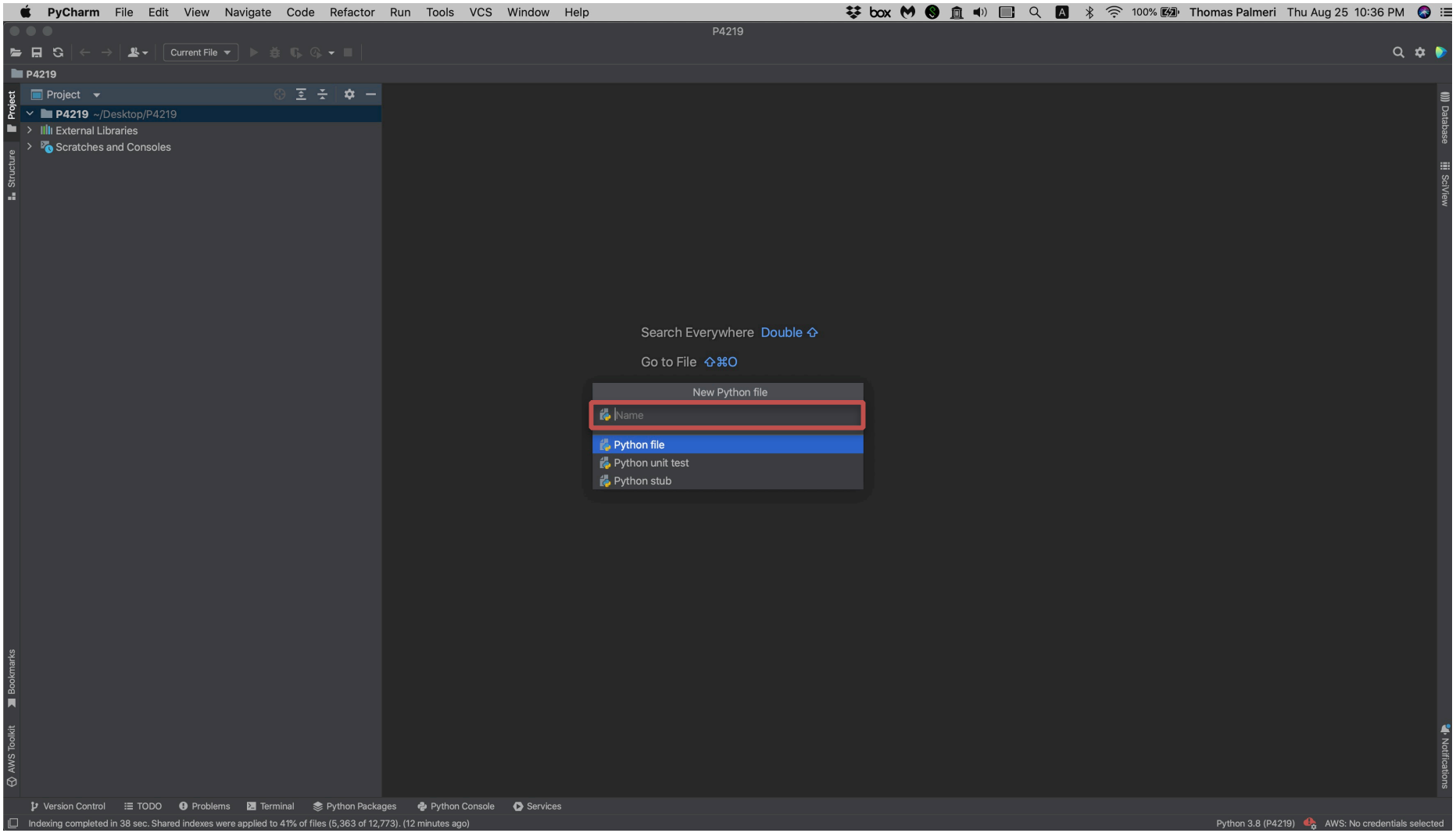
Setting up PyCharm

select "Python File"



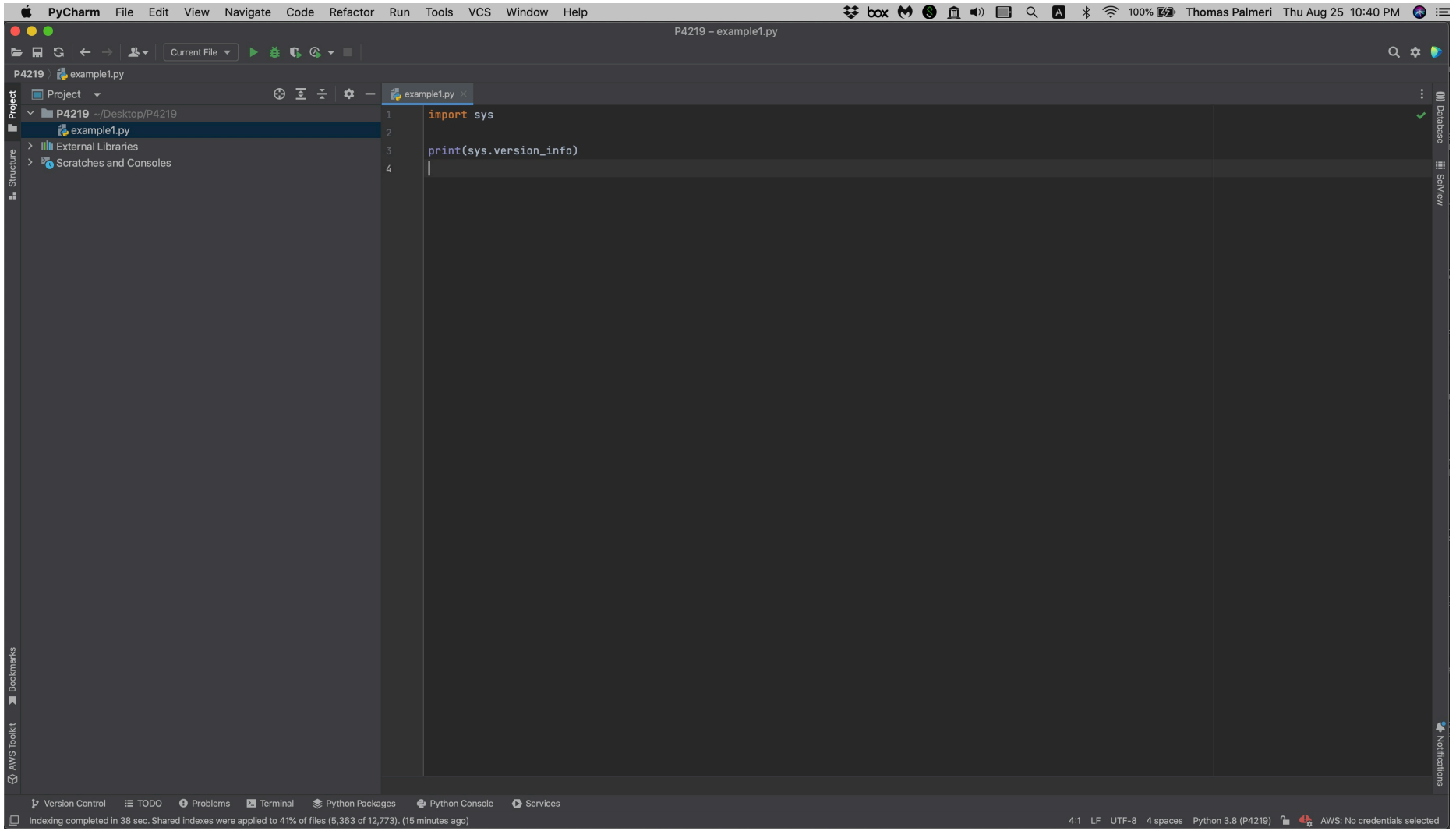
Setting up PyCharm

enter file name (e.g., "example1")



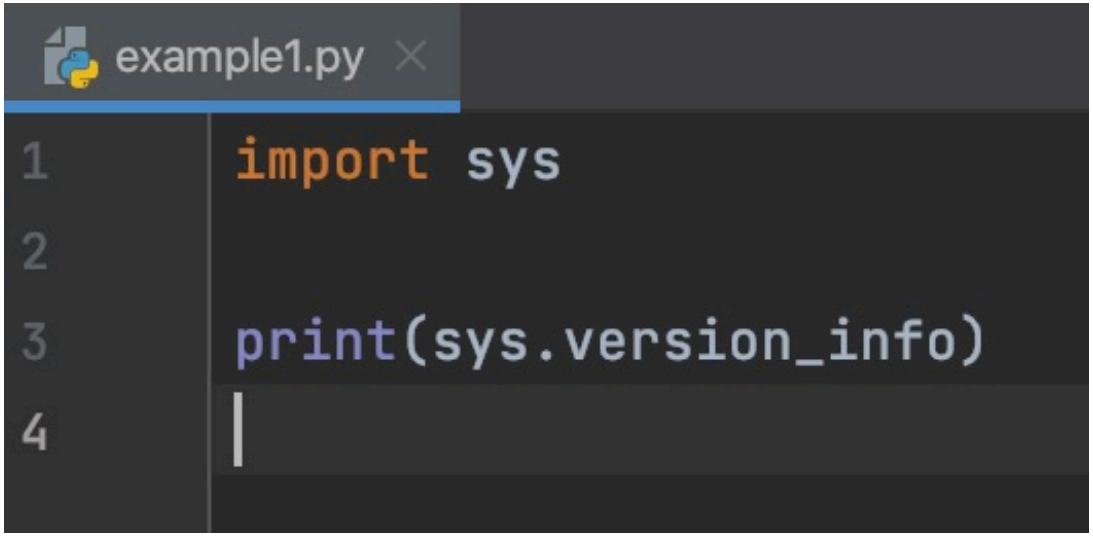
Setting up PyCharm

type in some Python code



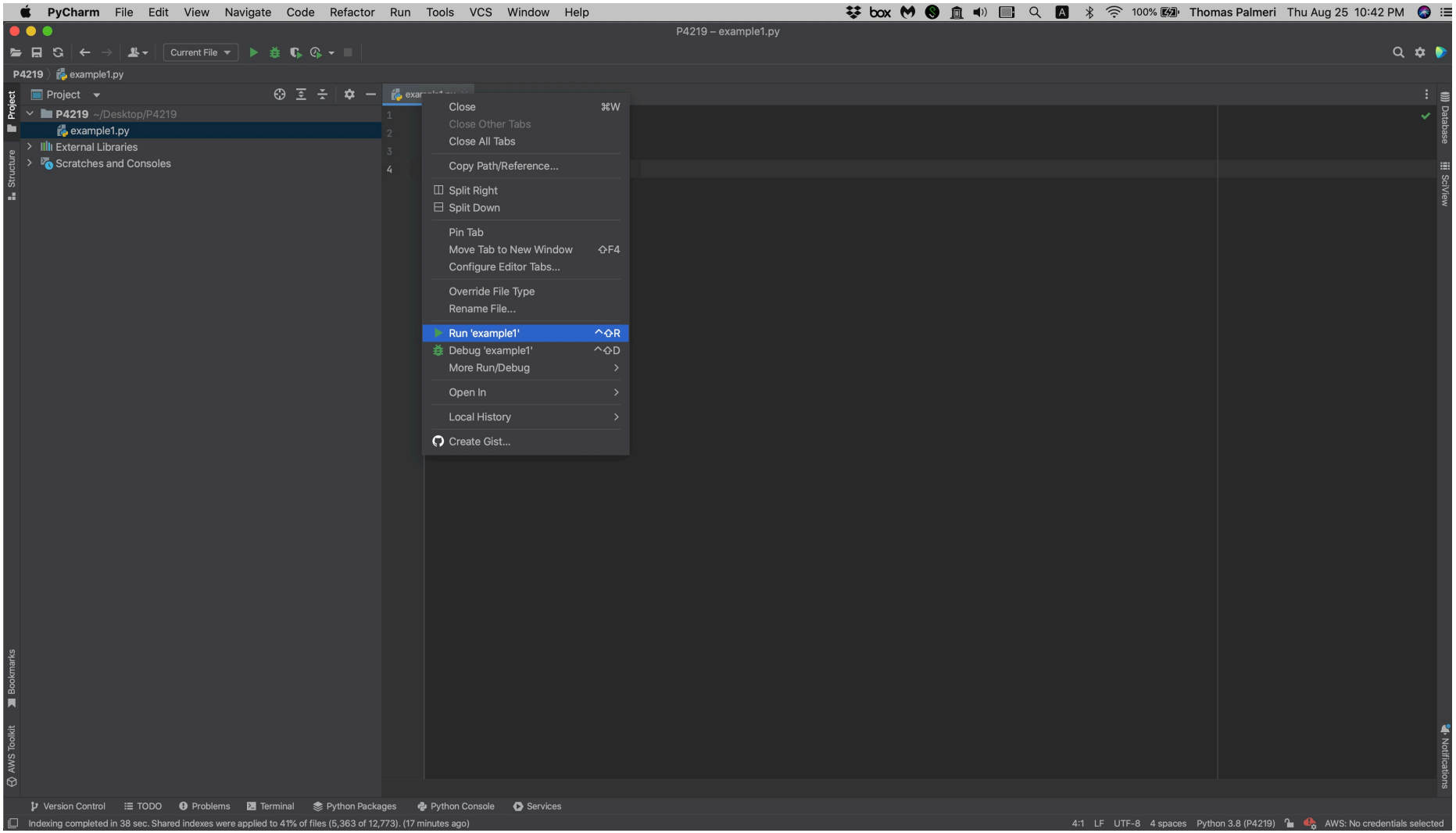
Setting up PyCharm

type in some Python code



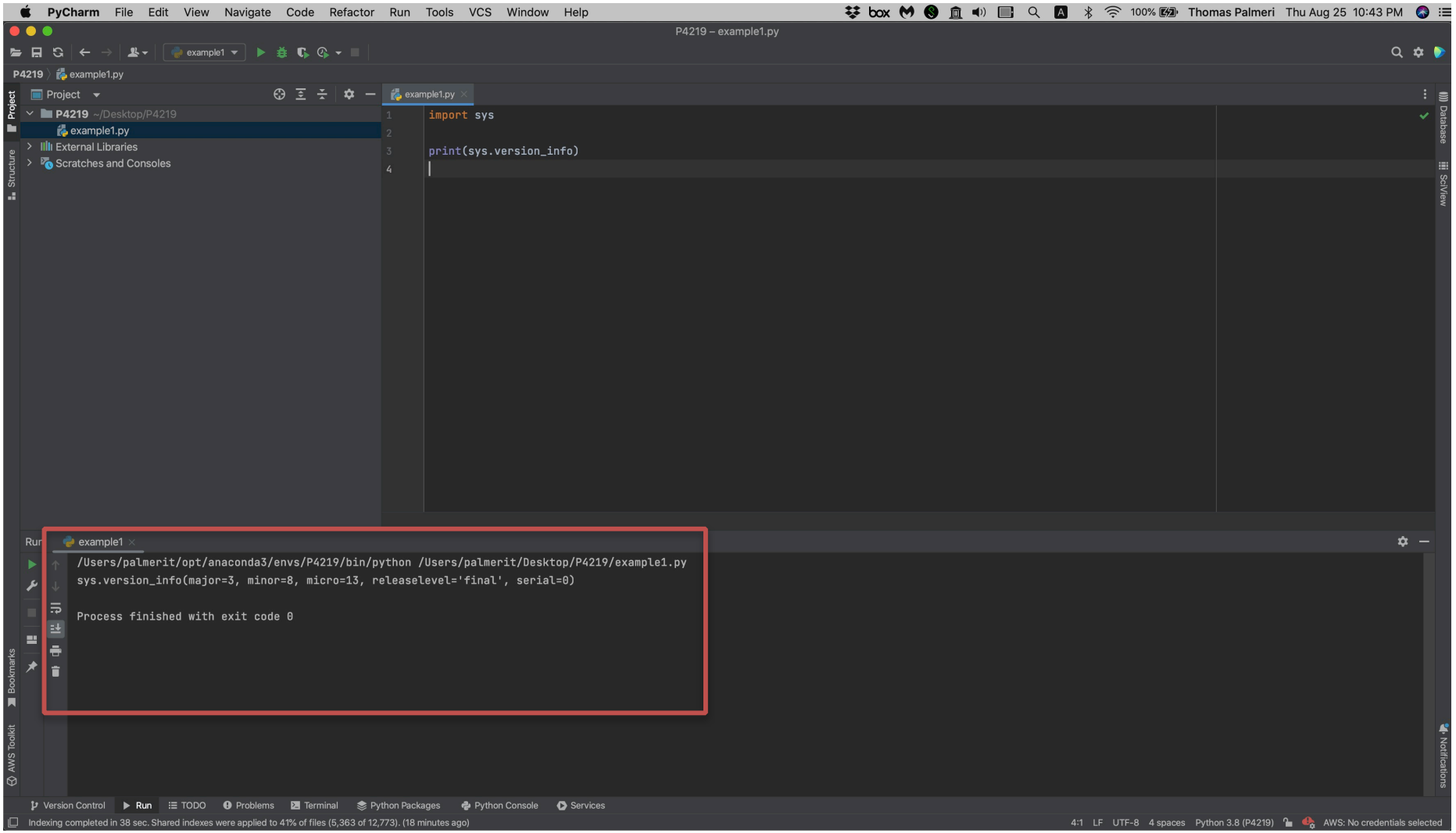
Setting up PyCharm

right click on the tab with the file name, select "Run"



Setting up PyCharm

see the printout from running the program



Run PyCharm

(or another IDE)

(part of) HOMEWORK 1:

load `TestPsychoPy.py` into PyCharm (or another IDE) (on Brightspace) and run it

grab a screen shot (after it runs) and submit (with other parts)

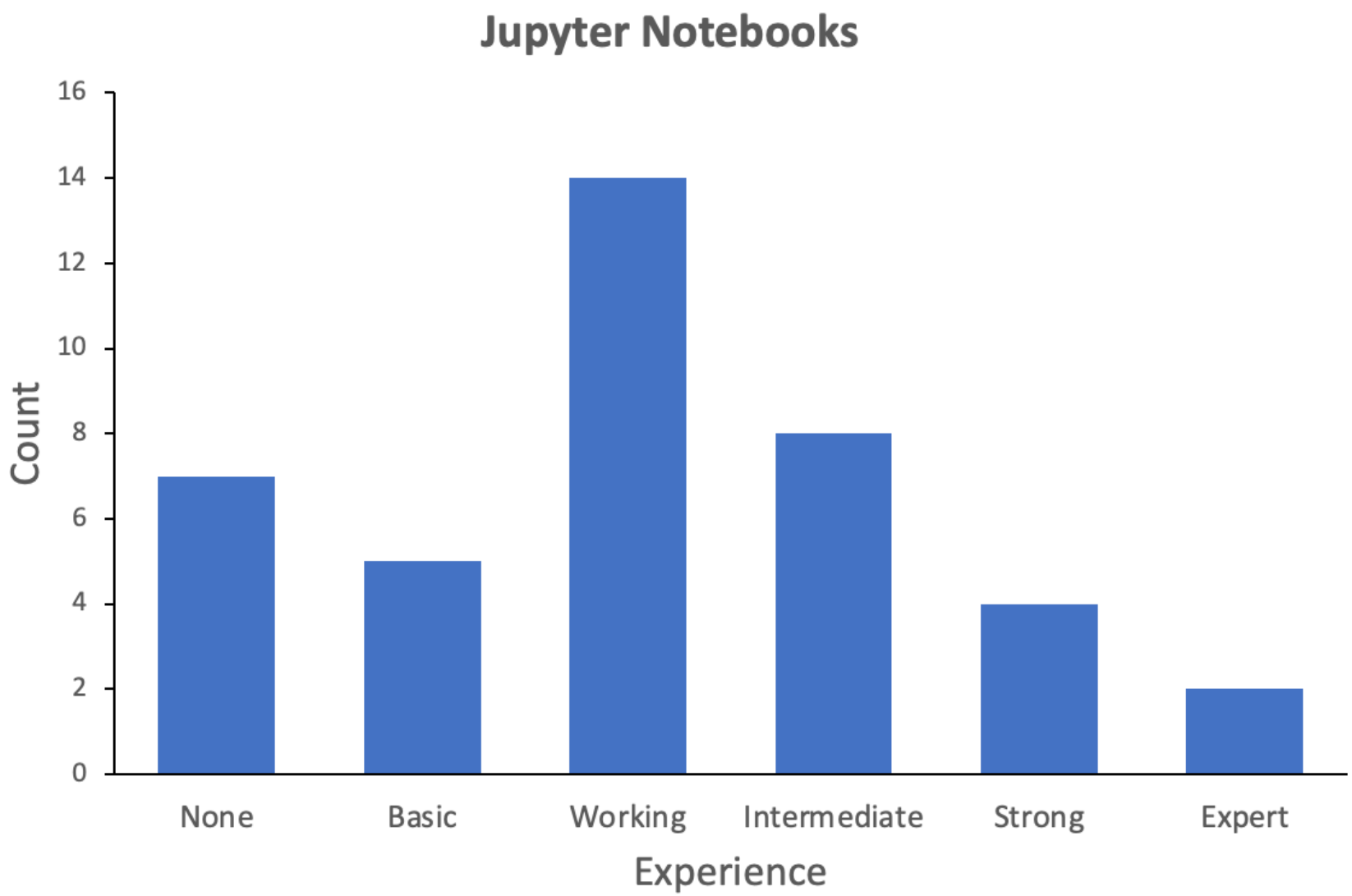
Run `TestPsychoPy.py`

Jupyter Notebooks

we'll use PyCharm later - for now, we'll use Notebooks

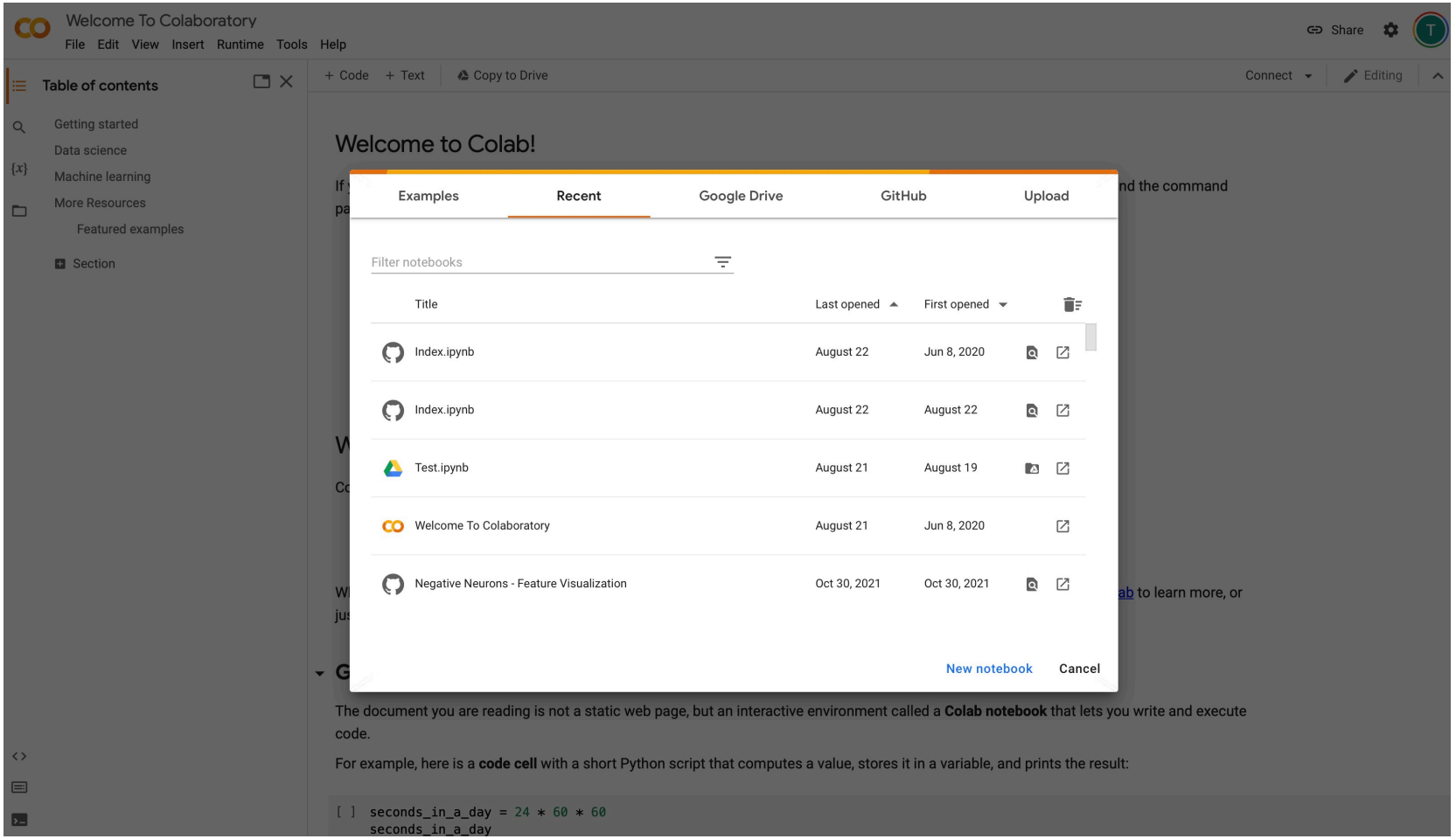
Jupyter Notebooks

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.


Jupyter Notebooks
16
14
12
10
8
6
4
2
0
Count
None
Basic
Working
Intermediate
Strong
Expert
Experience

Google Colab

https://colab.research.google.com



can upload notebooks (.ipynb files), load notebooks from GitHub,
or create a new notebook, and obviously you can then download it

Google Colab

https://colab.research.google.com

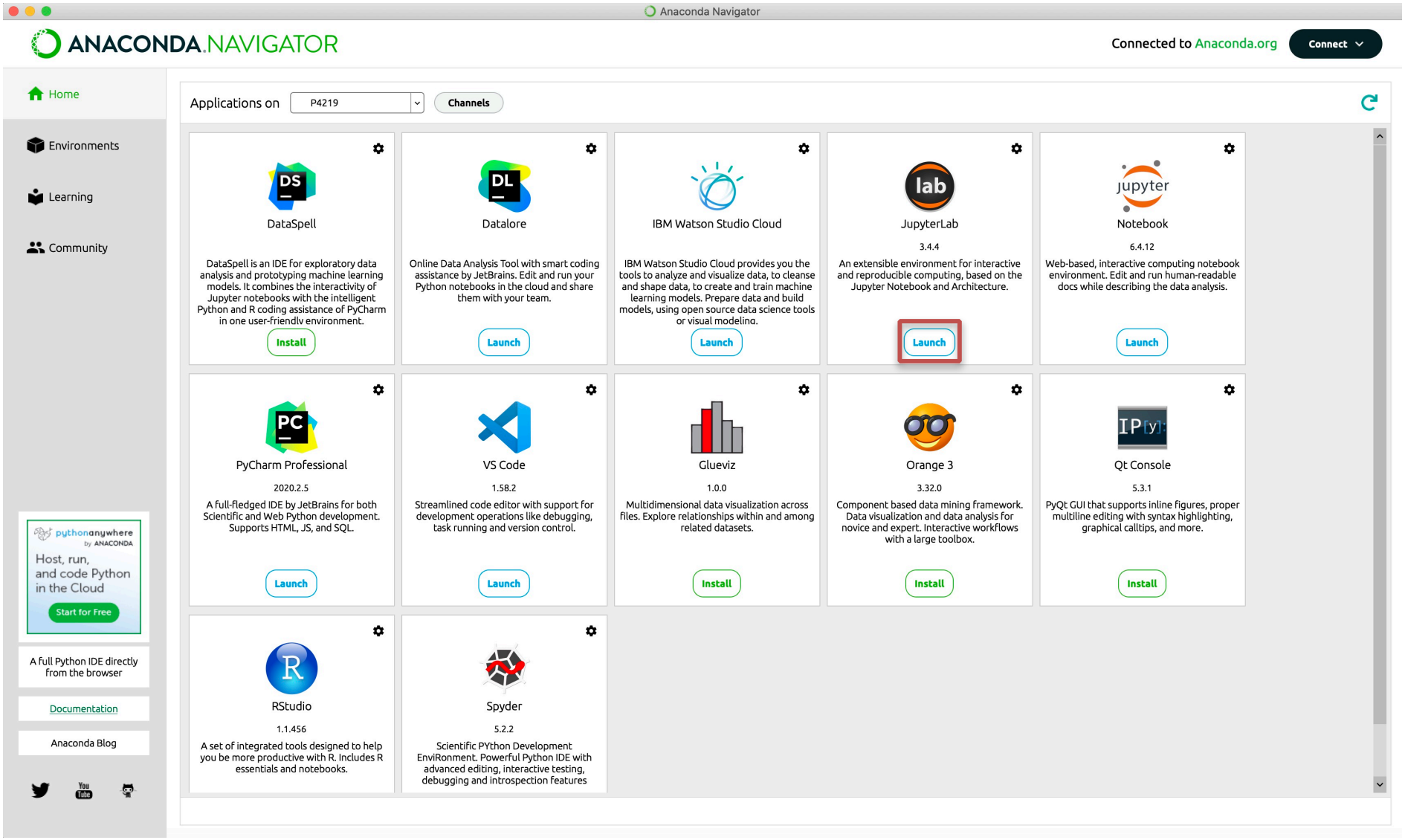
(part of) HOMEWORK 1:

upload `JupyterNotebooks.ipynb` to Google CoLab (on Brightspace) and run it

grab a screen shot (after it runs) and submit (with other parts)

we'll talk about how to run in a bit

launch JupyterLab (or Jupyter Notebook)



you can also run from command line:

Windows 10 - from Command Prompt

```
C:\> jupyter notebook
```

MAC OSX - from Terminal

```
$ jupyter notebook
```

if you know how to activate environments
from the command line

you can also run from command line:

Windows 10 - from Command Prompt

```
C:\> jupyter lab
```

MAC OSX - from Terminal

```
$ jupyter lab
```

if you know how to activate environments
from the command line

you can also run from command line:

Windows 10 - from Command Prompt

```
C:\> jupyter lab
```

MAC OSX - from Terminal

```
$ jupyter lab
```

if you know how to activate environments
from the command line

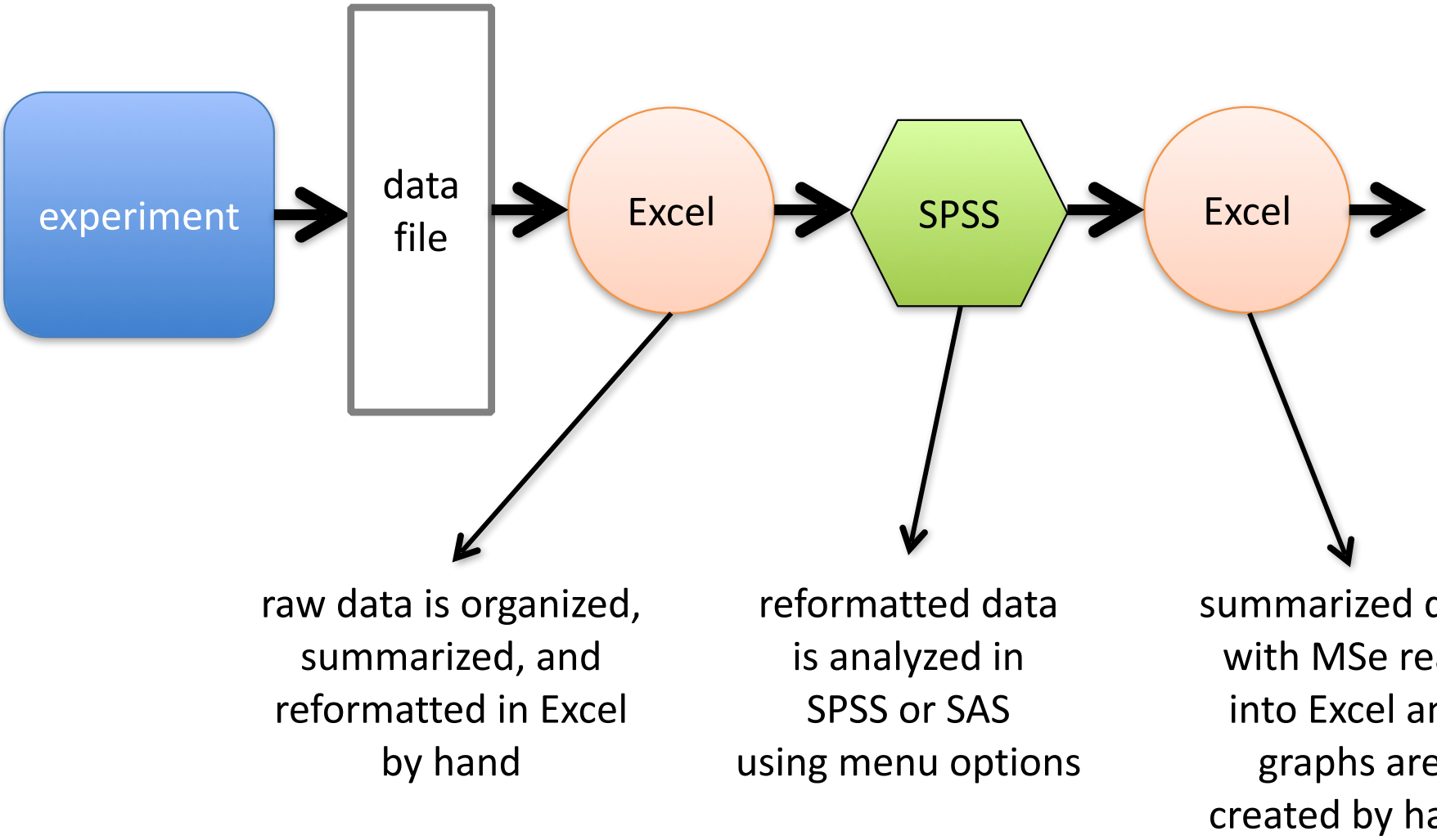
you can also run from the terminal
within an IDE like PyCharm

akin to lab digital notebooks

Jupyter Notebooks (are great) as Scripts

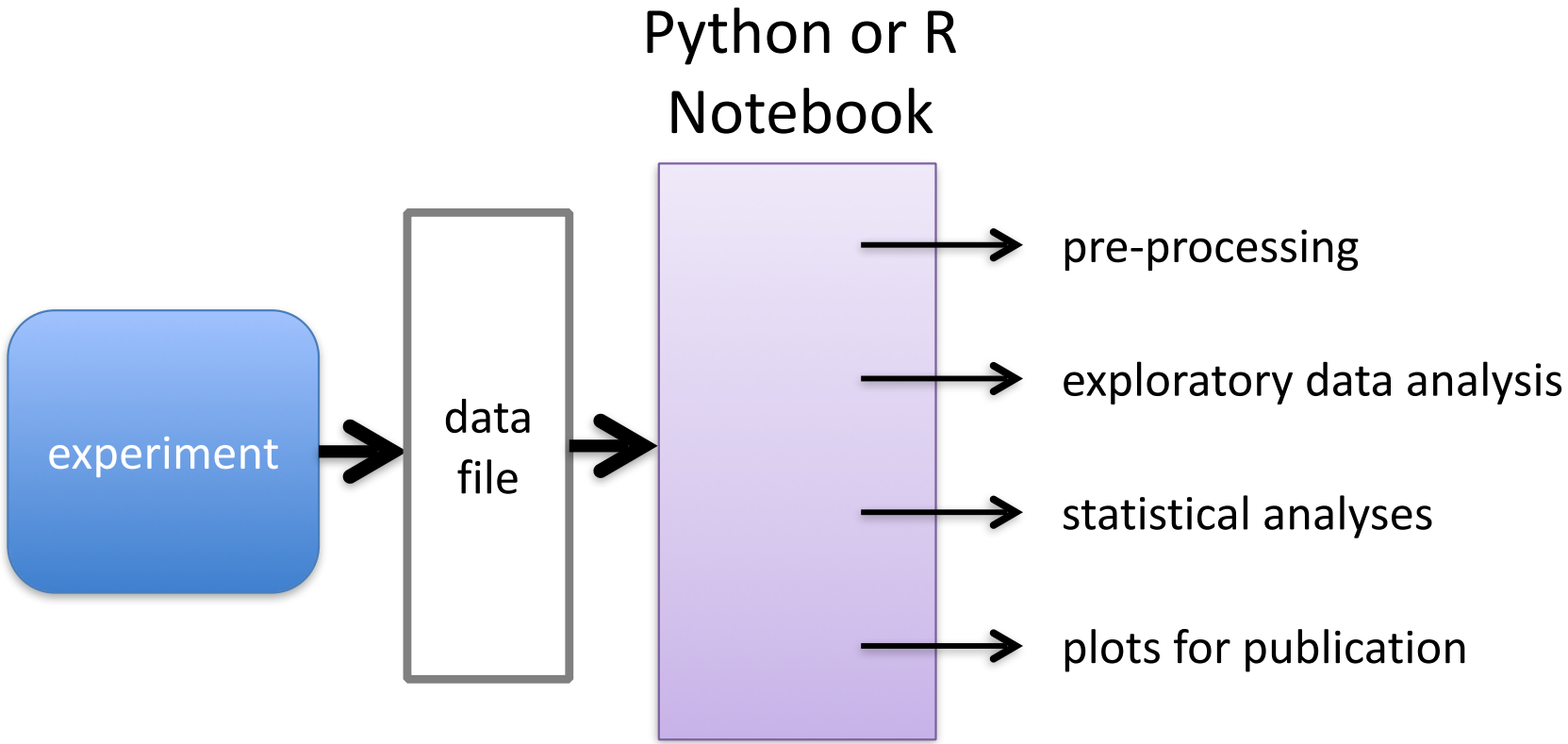
- Jupyter Notebooks are great for trying out snippets of Python code (as is the Python Console, which we will use within the PyCharm IDE)
- "Scripts" are a series of commands (though you can define functions within them). Scripts are not stand-alone programs nor are they modules/ packages that can be used by other programs.
- Jupyter Notebooks are fantastic for fully documenting a data analysis / modeling pipeline. They are a computational equivalent of a paper lab notebook. Preferred over more ad hoc analysis pipelines commonly used.

common (old) data analysis pipeline



maybe (hopefully) notes are kept
(paper or electronic, often separate from analyses)

Best Practice : Jupyter Notebooks



promises more reliable, robust, reproducible research

share data and Jupyter Notebook publicly (lab, GitHub, OSF)

using Jupyter

`JupyterNotebooks.ipynb`

using JupyterLab (or Jupyter Notebook)

make sure you reset the kernel and clear all outputs and rerun your code before turning in

cells can be rerun in any order (note numbers next to cells), but we will run your code from top to bottom

both `.py` and `.ipynb` files are text files

`TestPsychoPy.py`

Fall2022 — -bash — 87×41

```
Tom-MacBook-Pro-2020:Fall2022 palmerit$
Tom-MacBook-Pro-2020:Fall2022 palmerit$
Tom-MacBook-Pro-2020:Fall2022 palmerit$
Tom-MacBook-Pro-2020:Fall2022 palmerit$ cat TestPsychoPy.py
#
# TestPsychoPy.py
#
# drawing example
#

from psychopy import visual, core, event
import numpy as np


def main(mywin):

    # https://www.psychopy.org/api/visual/line.html#psychopy.visual.Line
    el1 = visual.Line(win=mywin, start=[+.3,-.2], end=[-.4,-.3], lineWidth=20, lineColo
r='blue')
    el1.autoDraw = True
    print('el1 : ', dir(el1))
    #el1.draw()
    mywin.flip(); core.wait(0.5)

    # https://www.psychopy.org/api/visual/circle.html#psychopy.visual.Circle
    el2 = visual.Circle(win=mywin, radius=.1, pos=[-.2,+.1], edges=128, fillColor='gree
n')
    el2.autoDraw = True
    print('el2 : ', dir(el2))
    #el2.draw()
    mywin.flip(); core.wait(0.5)

    # https://www.psychopy.org/api/visual/rect.html#psychopy.visual.Rect
    el3 = visual.Rect(win=mywin, size=[.2,.1], pos=[+.3,+.2], lineWidth=10, lineColor='
#F016A3')
    print('el3 : ', dir(el3))
    el3.autoDraw = True
    #el3.draw()
    mywin.flip(); core.wait(0.5)

    # https://www.psychopy.org/api/visual/shapestim.html#psychopy.visual.ShapeStim
```

`JupyterNotebooks.ipynb`

Jupyter — -bash — 87×41

```
Tom-MacBook-Pro-2020:Jupyter palmerit$
Tom-MacBook-Pro-2020:Jupyter palmerit$ cat JupyterNotebooks.ipynb
{
 "cells": [
  {
   "cell_type": "markdown",
   "id": "ecf4a5b0-d17f-4391-a06c-7b34f3593f01",
   "metadata": {
    "tags": []
   },
   "source": [
    "## Introduction to Jupyter Notebooks\n",
    "- two main types of cells - Markdown and Code\n",
    "- click to highlight a cell\n",
    "- double-click to edit a formatted Markdown cell\n",
    "- click the \"Run\" buttom above to execute highlighted cell\n",
    "- or do Shift-Return (Mac) or Shift-Enter (PC)\n",
    "- Cell -> Run All to run all cells in the notebook\n",
    "- Kernel -> Restart and then Cell -> Run All or Kernel -> Restart & Run All (reset
 everything and rerun)\n",
    "\n",
    "before finalizing any Jupyter notebook and turning it in for homework, please make
 sure you do a Kernel -> Restart to make sure your code runs completely (since when I r
un your code to grade it, everything will be similarly reset when I load it on my compu
ter)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "dbf639a1-062f-4931-864c-6d9bd3fffa30",
   "metadata": {},
   "outputs": [],
   "source": [
    "x = 1\n",
    "print(\"x = \", x)"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "12f6a269-a5c0-433d-997b-7a016586537e",
```

(JSON format)

swap between lecture slides and Python

slides

Numeric Types

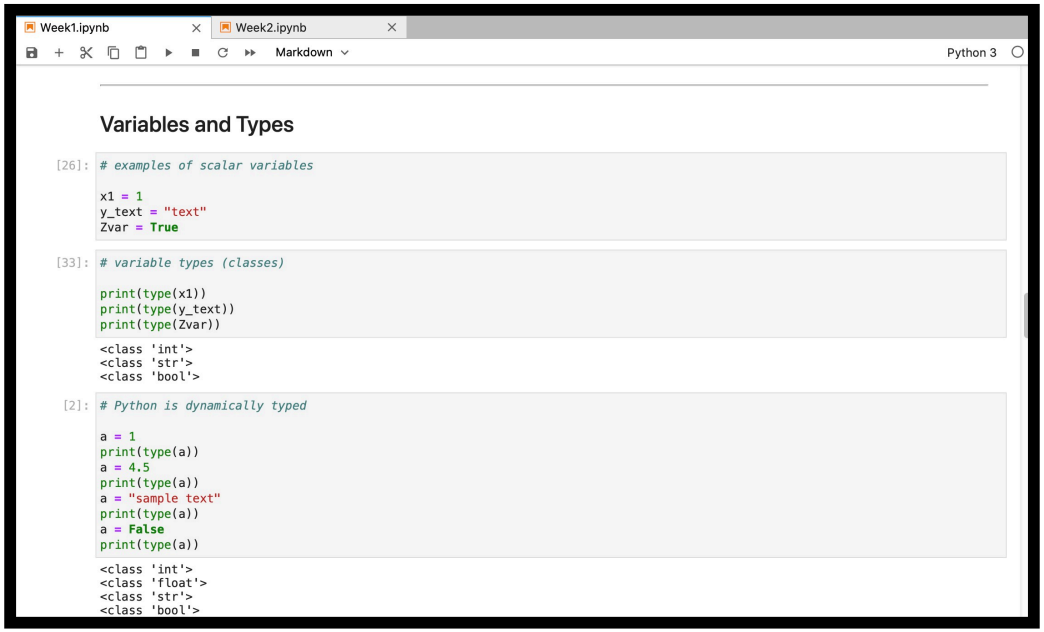
- int types

```
x = 1
y = 3
```

- float types

```
x = 1.
y = 3.4
```

Jupyter or PyCharm



sometimes I will talk about things on the slides,
sometimes things in the code

you will need to review both the slides and the code
(I will sometimes skim over details in one or the other)