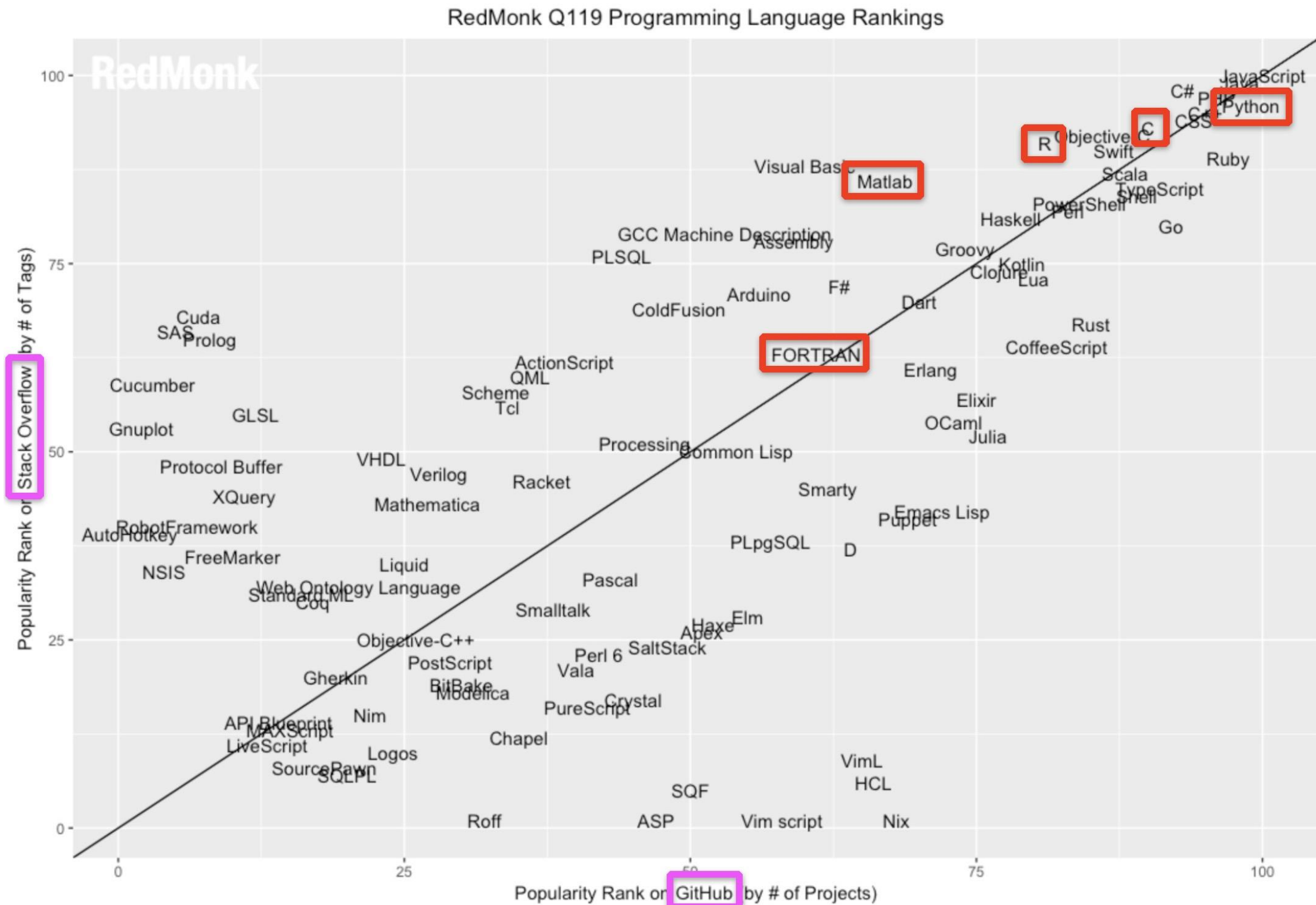


**Python**

# Python

- popular high-level language



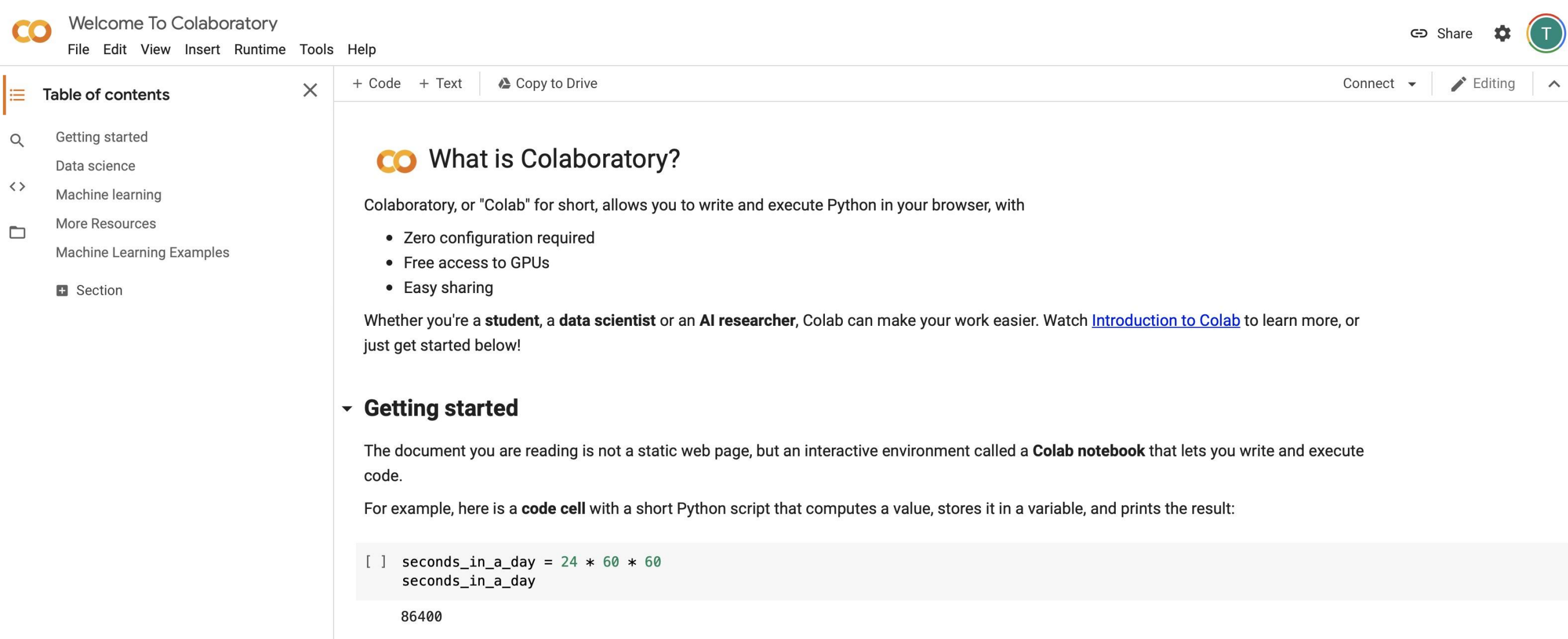
# Python

- popular high-level language
- interpreted (rather than compiled) language
- allows for full programming (.py)
- as well as scripts and notebooks (Jupyter or Colab)
- free, open source
- many powerful IDEs available
- thousands of available modules and packages
- scipy, numpy, matplotlib
- keras and tensorflow for neural networks
- scikit-learn (and others) for computational analyses

**Google Colab**

# Google Colab

<https://colab.research.google.com/>



The screenshot displays the Google Colaboratory web interface. At the top, the Google Colab logo is on the left, and a 'Welcome To Colaboratory' message is in the center. On the right, there are links for 'Share', 'Settings', and a user profile icon. Below the welcome message, a menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. A sidebar on the left contains a 'Table of contents' section with links to 'Getting started', 'Data science', 'Machine learning', 'More Resources', and 'Machine Learning Examples'. The main content area features the 'What is Colaboratory?' heading, followed by a paragraph explaining that Colaboratory allows writing and executing Python in the browser. A bulleted list highlights three features: 'Zero configuration required', 'Free access to GPUs', and 'Easy sharing'. Below this, a paragraph states that Colab can make work easier for students, data scientists, and AI researchers, with a link to 'Introduction to Colab'. A 'Getting started' section follows, explaining that the document is an interactive 'Colab notebook' and providing an example of a code cell with a Python script that calculates the number of seconds in a day, resulting in 86400.

CO Welcome To Colaboratory

File Edit View Insert Runtime Tools Help

Share Settings T

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Machine Learning Examples
- Section

+ Code + Text Copy to Drive

Connect Editing ^

## CO What is Colaboratory?

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser, with

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

### ▼ Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day

86400
```

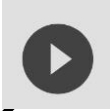
Julia, Python, R

Jupyter Notebook that runs on Google's cloud services

*we will talk more about how to use Jupyter Notebooks*

# Google Colab

<https://colab.research.google.com/>

- **as part of Homework 1 :**
  - download `autoencoder.ipynb` from Brightspace
  - File -> Upload Notebook
  - click on the Python code cell
  - hit shift+return (or shift-enter) to run
  - or click 
  - it will take a few minutes to train the network
  - capture a screen shot of the output and submit with other screen shots (as a ZIP file) on Brightspace

# Google Colab

<https://colab.research.google.com/>

Safari File Edit View History Bookmarks Develop Window Help

colab.research.google.com/drive/1i11eCbkVUSBRVwz8rJVcTfVpABRa4t#scrollTo=nXUxFOaqFPAk

Piazza Zotac USN Soccer Standings PsychoPy USN Plan Piazza Readcube-Papers LOG Interfolio PSY Faculty Search DSI Leadership Meeting Fidelity Netbenefits Assist Login GitHub CatLab Amazon Prime Music My stuff Poll Everywhere MCMC Interactive Gallery

02.02-The-Basics-Of-NumPy-Arrays.ipynb - Colaboratory autoencoder.ipynb - Colaboratory

autoencoder.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

235/235 [=====] - 3s 12ms/step - loss: 0.0928 - val\_loss: 0.0916  
Epoch 40/50  
235/235 [=====] - 3s 12ms/step - loss: 0.0928 - val\_loss: 0.0916  
Epoch 41/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0928 - val\_loss: 0.0917  
Epoch 42/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0928 - val\_loss: 0.0916  
Epoch 43/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0928 - val\_loss: 0.0915  
Epoch 44/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0927 - val\_loss: 0.0916  
Epoch 45/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0927 - val\_loss: 0.0915  
Epoch 46/50  
235/235 [=====] - 3s 12ms/step - loss: 0.0927 - val\_loss: 0.0915  
Epoch 47/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0927 - val\_loss: 0.0916  
Epoch 48/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0927 - val\_loss: 0.0915  
Epoch 49/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0927 - val\_loss: 0.0916  
Epoch 50/50  
235/235 [=====] - 3s 11ms/step - loss: 0.0927 - val\_loss: 0.0916

7 2 1 0 4 1 4 9 5 9

7 2 1 0 4 1 4 9 5 9

[ ]

2m 28s completed at 7:26 PM

RAM Disk Editing

# Google Colab

<https://colab.research.google.com/>

- a backup in case your local Python does not work
- or in case your computer stops working
- something to use if you need to learn (relearn) Python until you get Python installed and running on your computer
- Colab is free for some amount of usage but then you need to start paying for usage
- may be useful to use Google Colab for later assignments that might take too long to run on your computer (Colab has access to powerful GPUs & TPUs on Google cloud service)

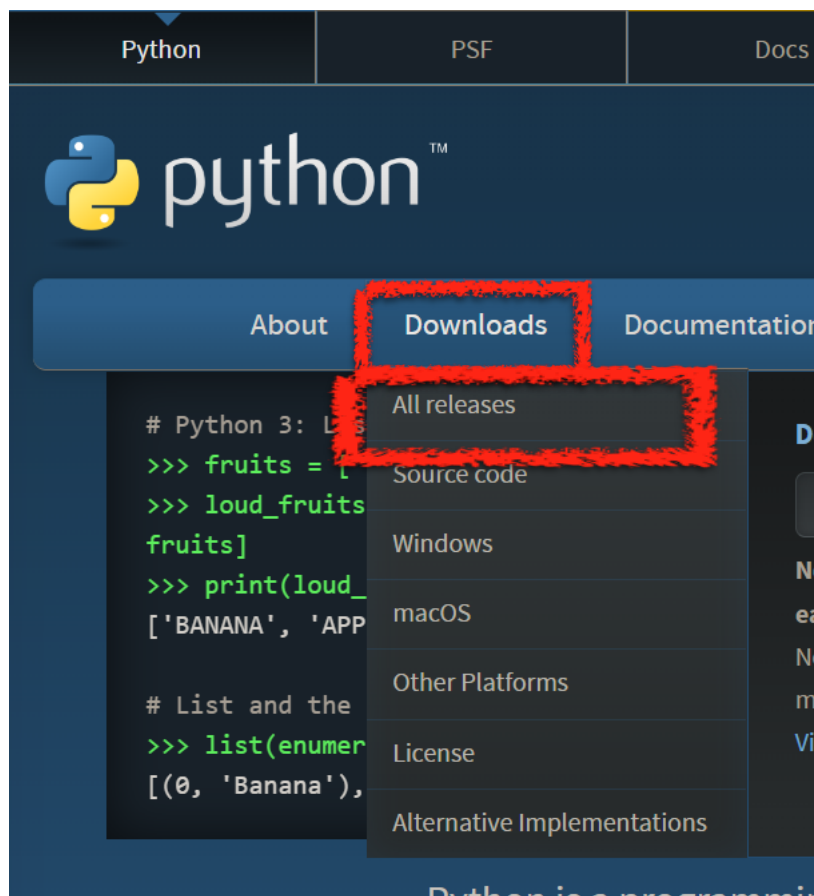


# **Fresh Python installation and setup**

even if you already have another version  
of Python on your computer

# install Python 3.9

- go to [python.org](https://python.org)
- click on Downloads -> All releases -> Python 3.9.13
- selection the version under “Files” that matches your OS



Looking for a specific release?  
Python releases by version number:

Release version	Release date
<a href="#">Python 3.10.6</a>	Aug. 2, 2022
<a href="#">Python 3.10.5</a>	June 6, 2022
<a href="#">Python 3.9.13</a>	May 17, 2022
<a href="#">Python 3.10.4</a>	March 24, 2022
<a href="#">Python 3.9.12</a>	March 23, 2022
<a href="#">Python 3.10.3</a>	March 16, 2022
<a href="#">Python 3.9.11</a>	March 16, 2022

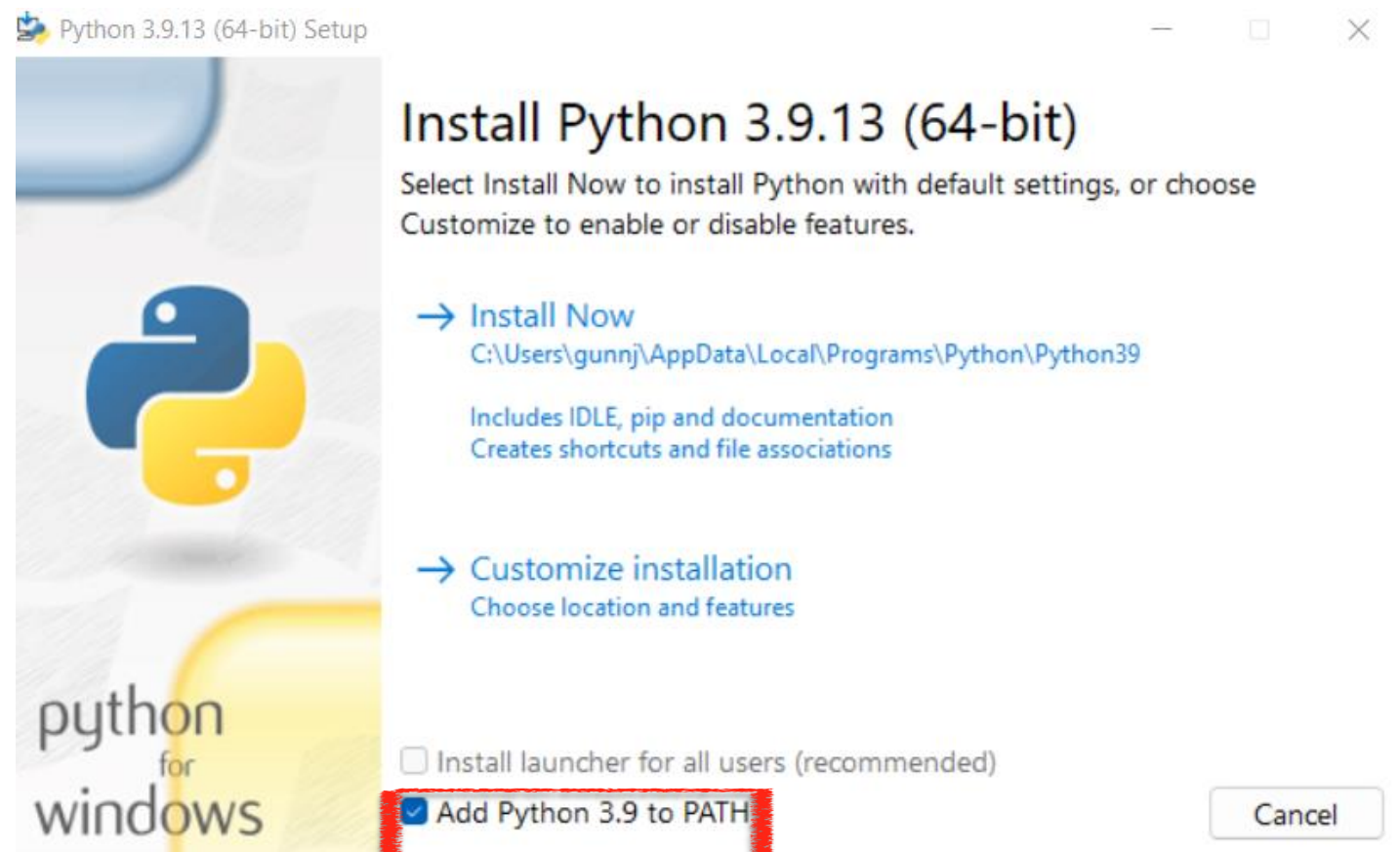
[View older releases](#)

### Files

Version	Operating System
<a href="#">Gzipped source tarball</a>	Source release
<a href="#">XZ compressed source tarball</a>	Source release
<a href="#">macOS 64-bit Intel-only installer</a>	macOS
<a href="#">macOS 64-bit universal2 installer</a>	macOS
<a href="#">Windows embeddable package (32-bit)</a>	Windows
<a href="#">Windows embeddable package (64-bit)</a>	Windows
<a href="#">Windows help file</a>	Windows
<a href="#">Windows installer (32-bit)</a>	Windows
<a href="#">Windows installer (64-bit)</a>	Windows

# install Python (on a PC)

- On Windows, when installing Python, check the box at the bottom of the installer window that says “Add Python 3.9 to PATH”
- This will let you use the “python” command in your powershell or command prompt. Otherwise you’ll need to use the full path of your python.exe installation as the command instead

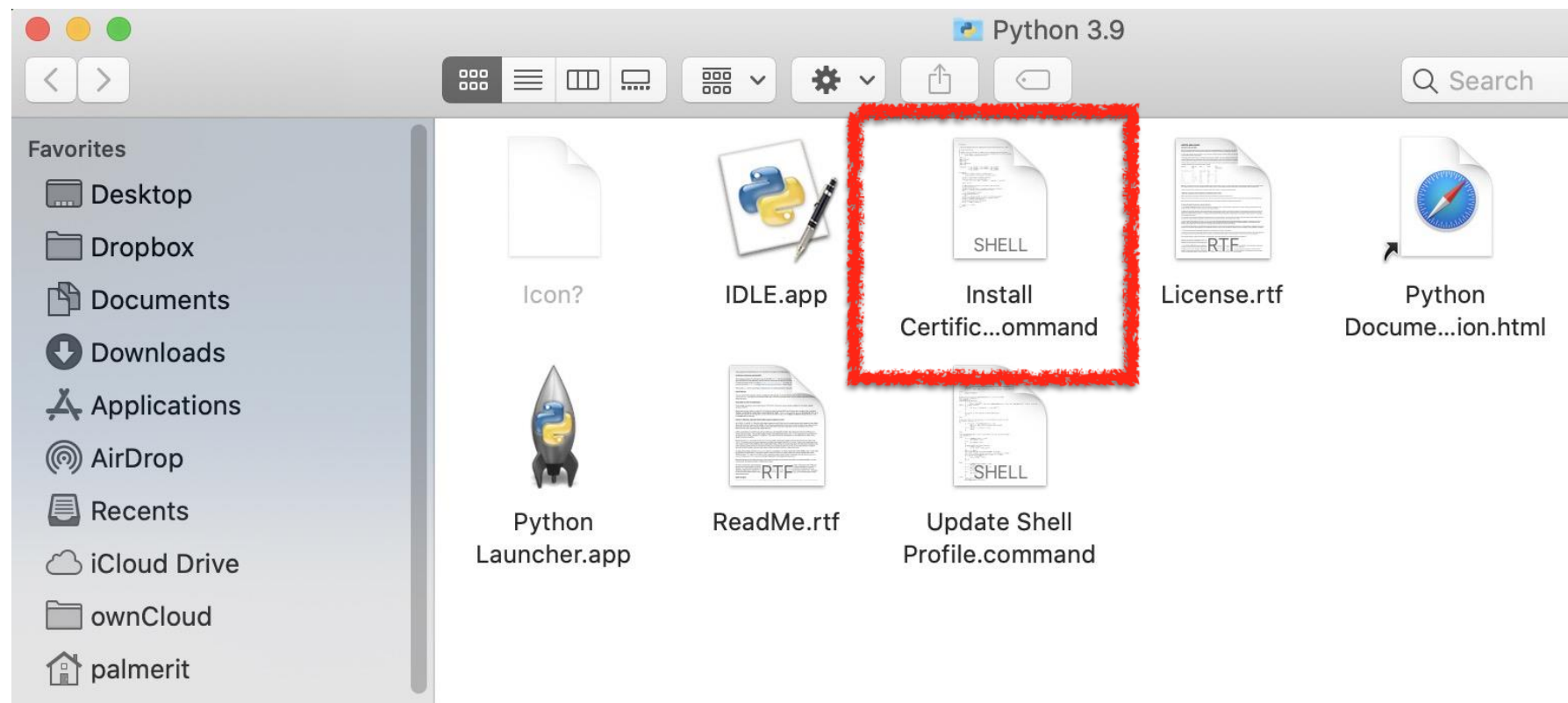


# install Python (on a Mac)

- on a Mac, after installing Python, you will see a note like this:

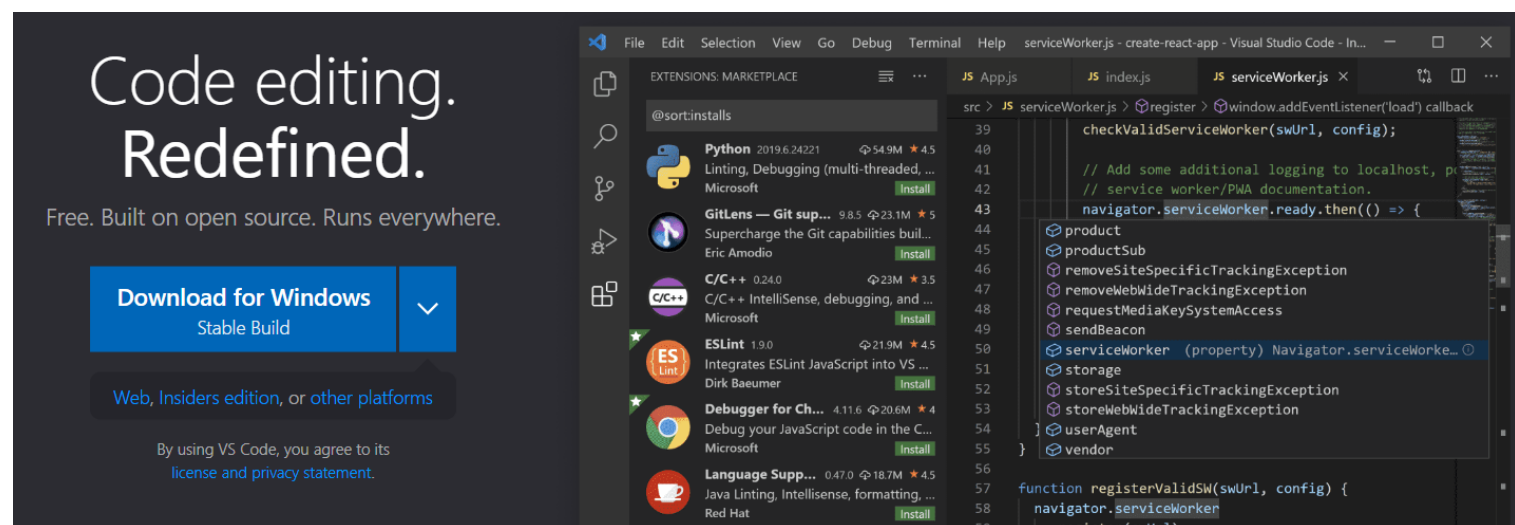
Congratulations! **Python 3.9.13 for macOS 10.9 or later** was successfully installed.

One more thing: to verify the identity of secure network connections, this Python may need a set of SSL root certificates. You can download and install a current curated set from the Certifi project by double-clicking on the Install Certificates icon in the Finder window. See the ReadMe file for more information.



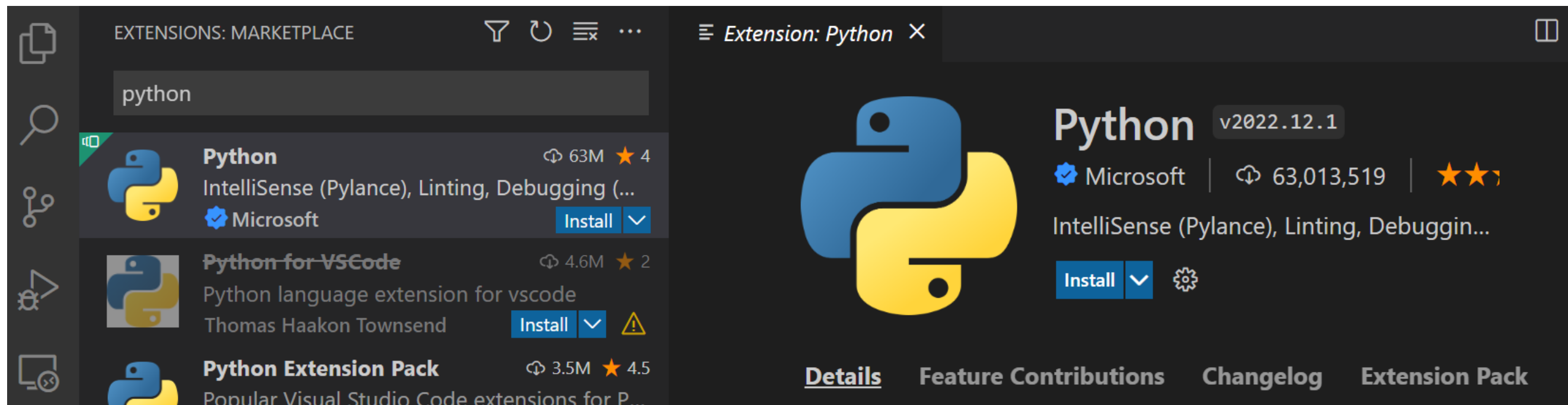
# install VSCode IDE

- You can use whatever IDE or code editor you want: PyCharm, VS Code, Eclipse, Spyder, Wing IDE etc.
- I will use VSCode (and Jupyter Notebooks) in class and these notes will assume VSCode for setting up Python
- you can download VSCode from here and install: <https://code.visualstudio.com/>
- we will talk about how to use VSCode a bit in class, but you are free to learn more from whatever online documentation, videos, and tutorials you can find



# Install Python Extension for VSCode

- For full Python language support, you will want to install the Python Extension for VSCode
- Run VSCode. On the opening screen, click **View** then select **Extensions**. Type in **Python** and install the top result (published by Microsoft).



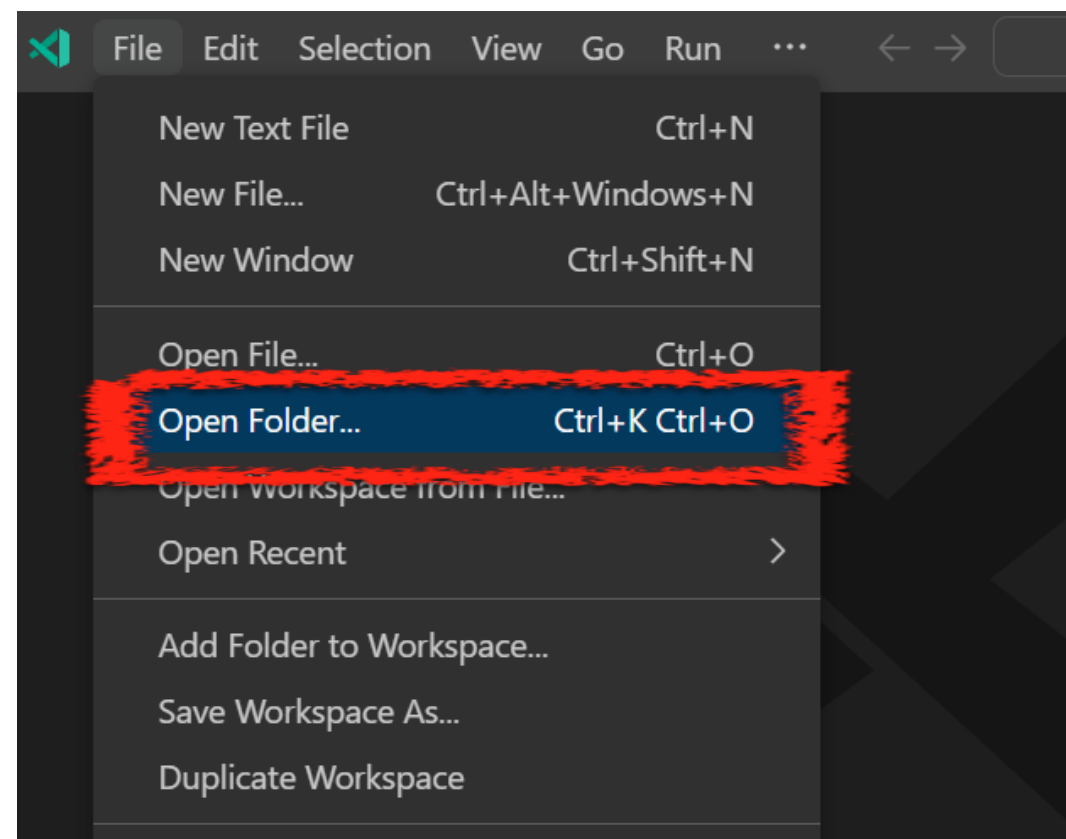
# setting up Python environment within VSCode

- the easiest way to configure Python is to keep everything for this course (Python .py and .ipynb files and the environment folders) in a single folder (and subfolders) - you can organize things differently, but you will need to know how to configure things yourself
- **let's assume you create a folder called** NSC3270 (of course, you can name it whatever you want)



# setting up Python environment within VSCode

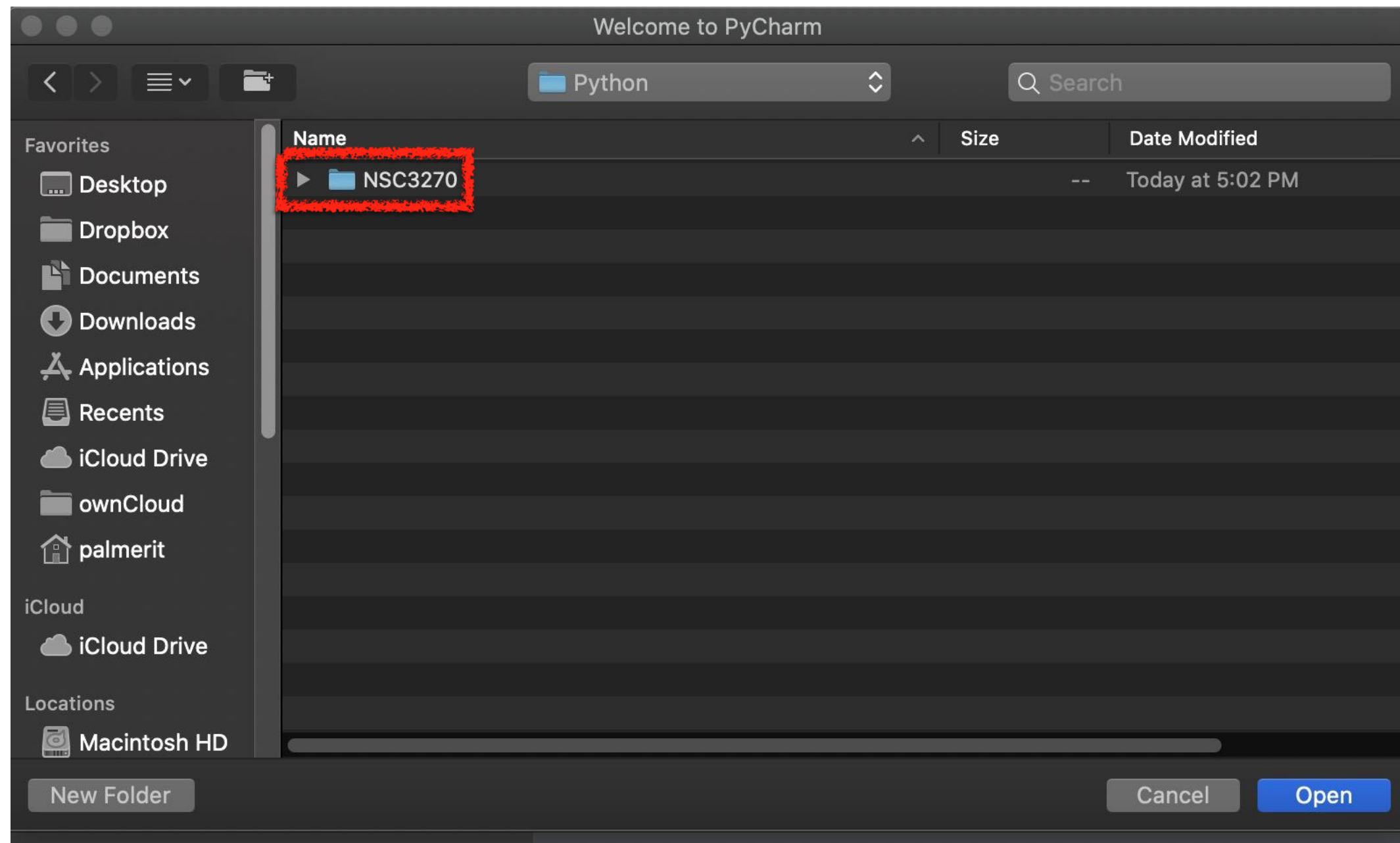
- run VSCode
- on the opening screen, click File, select Open Folder, and then finally select NSC3270





# setting up Python environment within PyCharm

- run PyCharm
- on the opening screen, click Open and select NSC3270



# setting up Python environment within VSCode

- now we create a virtual environment in VSCode
- virtual environments let you have different configurations of Python for different purposes on the same computer (e.g., a configuration needed for NSC3270, maybe a completely different configuration for a project that's part of your honors thesis), defaulting to perhaps different versions of Python and different collections of modules and packages

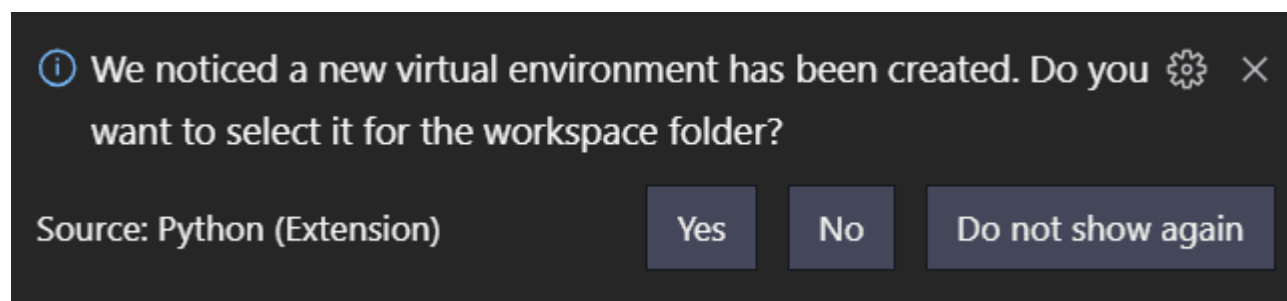
# setting up Python environment within VSCode

- With your workspace folder opened in VSCode, to the **View** menu and select **Terminal**
- Execute one of the following commands depending on your operating system:

```
# macOS/Linux
python3 -m venv .venv

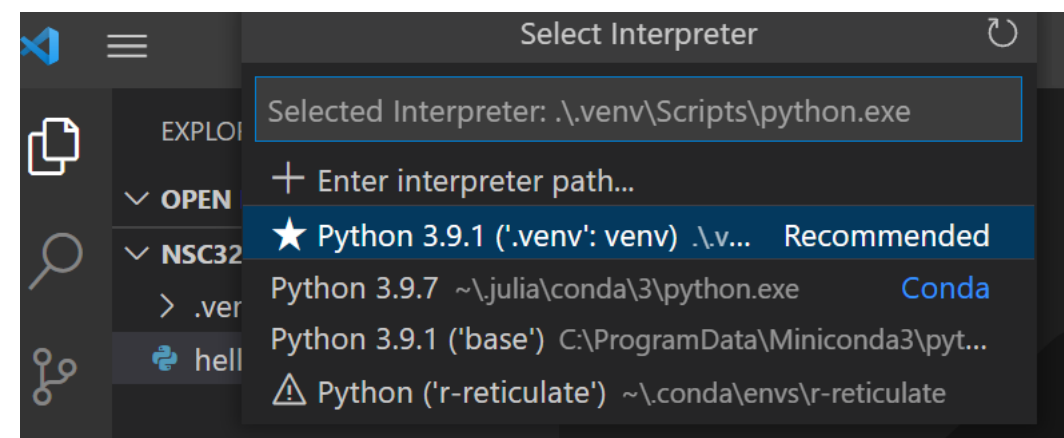
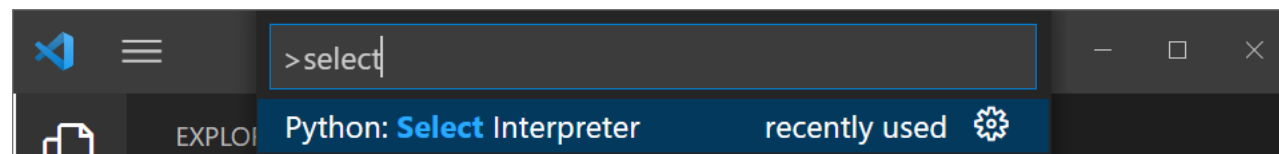
# Windows
python -m venv .venv
```

- When you create a new virtual environment, a prompt is usually displayed to allow you to select it for the workspace. Select Yes.



# If Python extension does not detect an interpreter..

- VSCode may fail to detect your new virtual environment
- To do it manually, go to the **View** menu and select **Command Palette**
- type “python select” to see and select the option for **Python: Select Interpreter**.
- If things are configured properly, your virtual environment (probably labeled something like **Python 3.9.1 (.venv': venv)** should be one of the available interpreters.



# Installing packages within VSCode

- You can install packages to easily use code others have written
- Reopen your **Terminal** from the **View** menu in VSCode
- Activate your virtual environment with one of these two commands depending on your OS:
- Terminal (MacOS): `source .venv/bin/activate`
- Powershell (Windows), two commands at first:
  - `set-executionpolicy RemoteSigned -Scope CurrentUser` (Only needed once! Respond “Yes” (Y) to give Powershell permission to run scripts)
  - `env\scripts\activate.bat` (to activate)
- Execute the command `pip install tensorflow`
- You can follow the same process to install each of the following packages all at once with one line:  
`pip install matplotlib seaborn pandas notebook jupyterlab scikit-learn nipy`

**now test that Python installation works in VSCode**

# test that everything works in VSCode

- download `autoencoder.py` from Brightspace and load into VSCode (or another IDE you are using)
- run `autoencoder.py`
- ignore any errors related to CUDA, GPU, or optimization
- you should see it produce something like this:

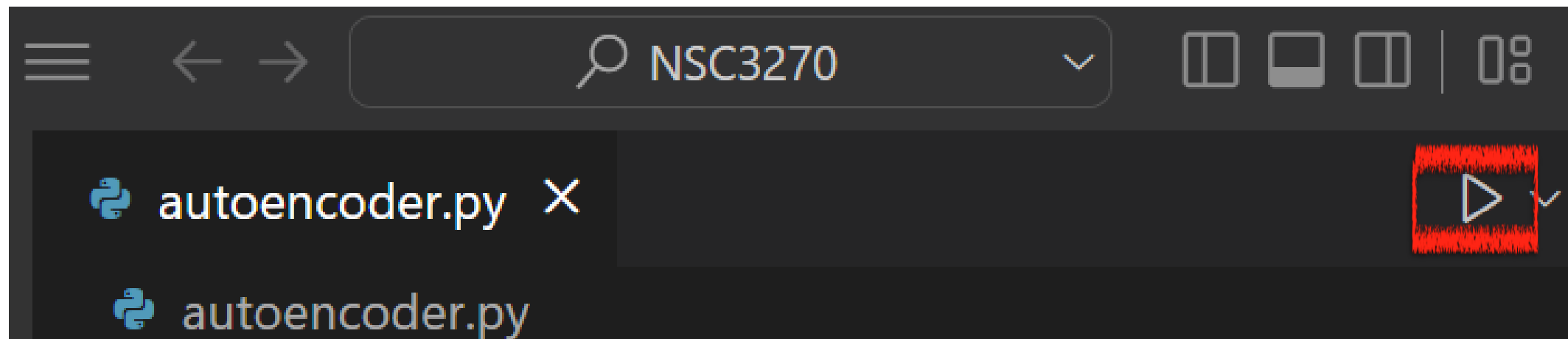


## as part of Homework 1 :

- take a screen shot (make sure it includes the digits)
- submit with other screen shots (as a ZIP file) on Brightspace

# How to Run Jupyter Notebooks in VSCode

- to run a Python script in VSCode, open the file in your editor

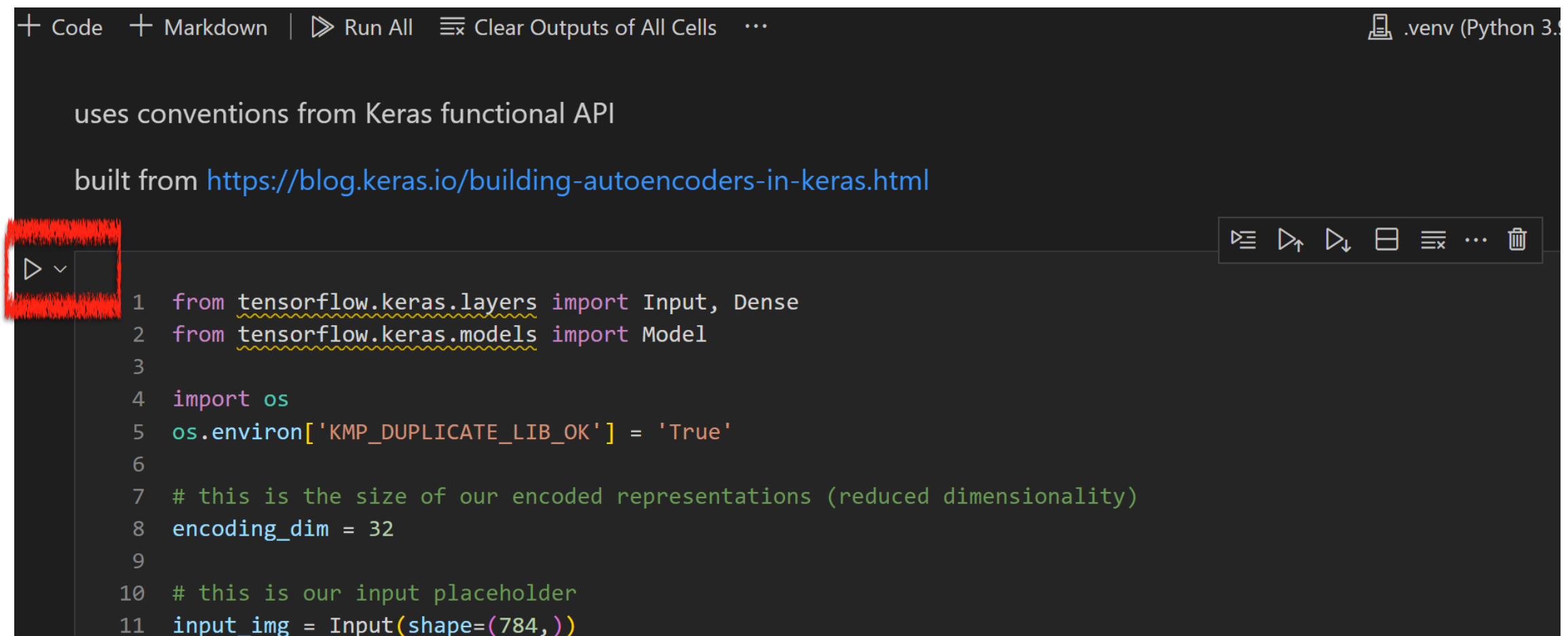


- This should open your terminal and generate some commands and messages within the window activating your virtual environment and executing the script



# How to Run Jupyter Notebooks in VSCode

- to run a Python script in VSCode, open the file in your editor
- click on Python code cells
- hit shift-return (or shift-enter) to run
- or click “Run” arrow or “Run all” at top of Editor window.



The screenshot shows the VS Code editor interface. At the top, there's a toolbar with buttons for '+ Code', '+ Markdown', 'Run All' (a play icon), and 'Clear Outputs of All Cells'. On the right, it says '.venv (Python 3.9)'. The main editor area contains a code cell with the following text:

```
uses conventions from Keras functional API

built from https://blog.keras.io/building-autoencoders-in-keras.html
```

Below this, there's a Python code cell. The first line of code is highlighted with a red box, indicating it's the active cell. The code is:

```
1 from tensorflow.keras.layers import Input, Dense
2 from tensorflow.keras.models import Model
3
4 import os
5 os.environ['KMP_DUPLICATE_LIB_OK'] = 'True'
6
7 # this is the size of our encoded representations (reduced dimensionality)
8 encoding_dim = 32
9
10 # this is our input placeholder
11 input_img = Input(shape=(784,))
```

# test Jupyter notebooks

- **as part of Homework 1 :**
  - download `autoencoder.ipynb` from Brightspace
  - save to your NSC3270 course folder
  - open (double-click) within Jupyter notebook
  - click on the Python code cell
  - hit shift-return (or shift-enter) to run
  - it will take a few minutes to train the network
  - capture a screen shot of the output and submit with other screen shots (as a ZIP file) on Brightspace



**what if you encounter an error?**

## **if you encounter an error in VSCode or Jupyter**

- check Piazza to see if someone had a similar problem
- post a question to Piazza
- make sure you include as much detail as possible (including the entire error message) and note how you installed Python and what IDE you're using
- you can also search online using the error message as a search term - that's what Jordan or I will do if we do not recognize the source of the error
- help out by answering questions posed by others on Piazza

# if you encounter an error in VSCode or Jupyter

- if you cannot get an answer on Piazza, then email Jordan [jordan.gunn@vanderbilt.edu](mailto:jordan.gunn@vanderbilt.edu)
- you will need to bring your computer in to meet with Jordan
- if anyone finds an error or omission in the slides, please email both me ([sean.polyn@vanderbilt.edu](mailto:sean.polyn@vanderbilt.edu)) and Jordan
- remember that you can use Google Colab until you get your local Python working on your computer