Create and Work with Virtual Environment

For creating a virtual environment we can use a .yml file with dependencies or we can create a new environment from scratch.
We will first try using the provided .yml. If this does not work, skip to the next section.

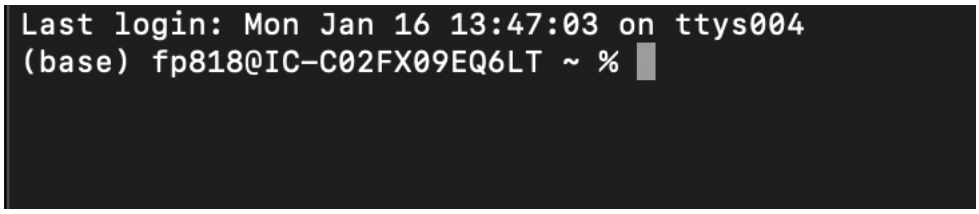
Create Virtual Environment from .yml file.

1. Download the ML4Neuroscience repository from Github.
   https://github.com/PBarnaghi/ML4NS

2. Extract the zipped file in your preferred location

3. Open terminal

If you have properly installed Anaconda, you will get a (base) before your username. This means that you are currently in your base environment (your OS).



4. Move to the folder containing your ML4Neuroscience unzipped folder and locate the environment .yml file

   cd FOLDER_PATH

   Example:

   cd /Users/fp818/Downloads /ML4NS/Virtual\ Environment\ Settings

   Be careful if you have spaces in your FOLDER_PATH, you will need to overwite them with "\ ". E.g Virtual Environment Settings → Virtual\ Environment\ Settings

   You should now have this in your terminal

```
(base) fp818@IC-C02FX09EQ6LT Virtual Environment Settings %
```

5. We can now create the Anaconda virtual environment using the **virtual_env_mac.yml** in the folder.
   To create a new virtual environment from a file, we need to write:

   conda env create --name ENV_NAME --file FILENAME

   In our case, if we want to create a virtual environment called “machine_learning” from the virtual_env_mac.yml file, we will need to write:
   conda env create –name machine_learning –file virtual_env_mac.yml

```
(base) fp818@IC-C02FX09EQ6LT Virtual Environment Settings % conda env create --name machine_learning --file virtual_env_mac.yml
```

6. If everything is working correctly, Anaconda will download and install all the packages and their corresponding versions written in the virtual_env_mac.yml file and you should visualise the following final message

```
done
#
# To activate this environment, use
#
#     $ conda activate machine_learning
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

7. At this point you can activate the new virtual environment and you will have all the requested dependencies available (no need to pip install anything else or running cells with !pip install)

   conda activate ENV_NAME

   And, instead of (base), you will visualise the name of the new virtual environment next to your username, (machine_learning) in our example

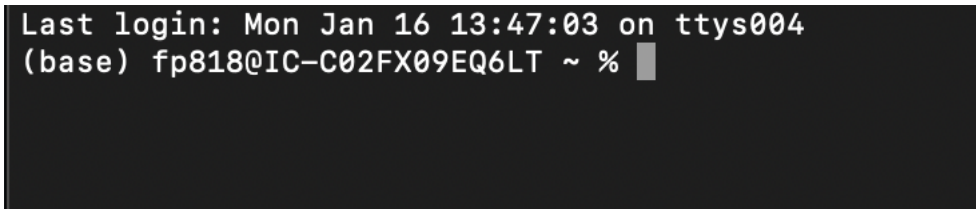
```
(base) fp818@IC-C02FX09EQ6LT Virtual Environment Settings % conda activate machine_learning
(machine_learning) fp818@IC-C02FX09EQ6LT Virtual Environment Settings %
```

Create Virtual Environment from scratch.

If the previous installation failed, you can try creating a new virtual environment from scratch.

1. Open terminal

If you have properly installed Anaconda, you will get a (base) before your username. This means that you are currently in your base environment (your OS).



2. Create the new environment specifying the version of Python to use (3.9 in our case). The location of the terminal does not influence the virtual environment, you can run this command from whichever folder you prefer.

   This can be done:

   conda create --name ENV_NAME python=3.9

   E.g
   conda create --name machine_learning python=3.9

   You will then receive this message

```
done
#
# To activate this environment, use
#
#     $ conda activate machine_learning
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

3. We can now activate the environment

   conda activate ENV_NAME

   And, instead of (base), you will visualise the name of the new virtual environment next to your username, (machine_learning) in our example

```
(base) fp818@IC-C02FX09EQ6LT Virtual Environment Settings % conda activate machine_learning
(machine_learning) fp818@IC-C02FX09EQ6LT Virtual Environment Settings %
```

4. Once we are in the correct environment we can then pip install the requested libraries whenever we need them or we can use the cells in the Jupyter Notebook to install them

   a. If you want to install the libraries from the terminal, you can just run the following command:

      pip install LIBRARY_NAME

      E.g if you want to install Numpy, you will:
      pip install numpy

      You may need to accept the installation by pressing ENTER on your keyboard

```
(machine_learning) fp818@IC-C02FX09EQ6LT Virtual Environment Settings % pip install numpy
Collecting numpy
  Downloading numpy-1.24.1-cp39-cp39-macosx_11_0_arm64.whl (13.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.9/13.9 MB 16.3 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-1.24.1
```

      IMPORTANT: If you are unsure about the name of a library, please check on Google!
      - Jupyter notebook library name is Jupyterlab
      - sklearn library name is scikit-learn
      - Tensorflow and Pytorch may be complicated to install, check on Google how

to properly install them

b. If you want to install a library from the cell of a Jupyter Notebook (e.g. like in the tutorial), run jupyter-notebook or jupyter lab:

```
(machine_learning) fp818@IC-C02FX09EQ6LT Virtual Environment Settings % jupyter-notebook
```

From the notebook, you can run the cell including

!pip install numpy

If numpy is not installed in your virtual environment, it will be downloaded and installed.

Open Jupyter Notebooks from Virtual Environment

1. From the terminal, if you are not in your correct virtual environment, activate the correct environment

   conda activate ENV_NAME

   Example
   conda activate machine_learning

```
(base) fp818@IC-C02FX09EQ6LT Virtual Environment Settings % conda activate machine_learning
(machine_learning) fp818@IC-C02FX09EQ6LT Virtual Environment Settings %
```

2. Navigate to the folder where your notebook is located

   cd FOLDER_PATH

   Example:

   cd /Users/fp818/Downloads /ML4NS/

   Be careful if you have spaces in your FOLDER_PATH, you will need to overwite them with "\ ". E.g Virtual Environment Settings → Virtual\ Environment\ Settings

```
(machine_learning) fp818@IC-C02FX09EQ6LT ML4NS
```

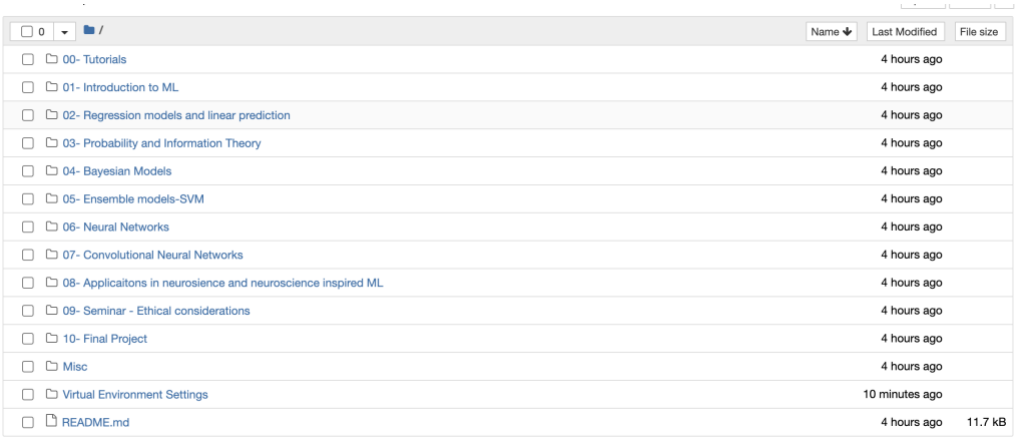
3. Depending on you, if you want to use jupyter-notebook or jupyterlab, you will need to write:

a. For Jupyter notebook:

jupyter-notebook

```
(machine_learning) fp818@IC-C02FX09EQ6LT ML4NS % jupyter-notebook
```

It will then open a new browser page with all the folders that you have in the path that you run the command



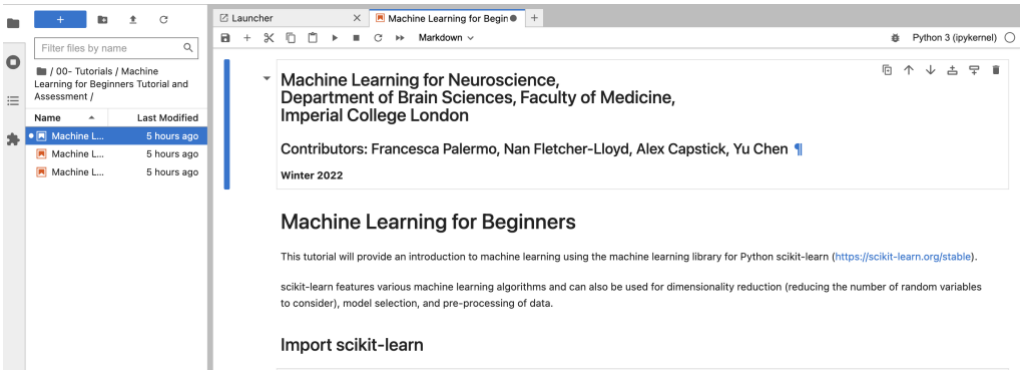
If you want to open the Python for Beginner notebook, just click on the correct folder and open the file

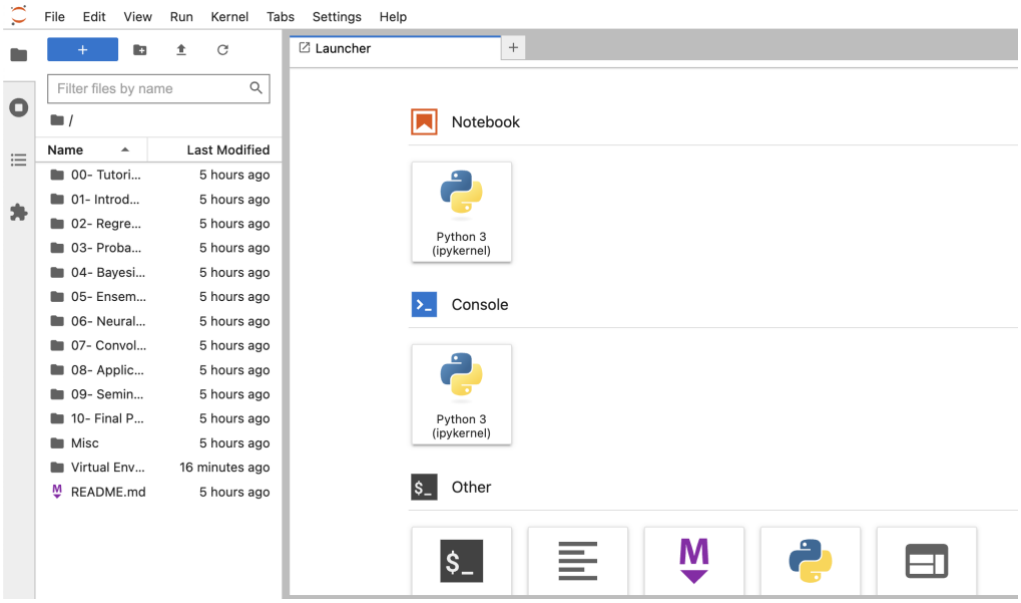
b. For jupyter lab, just write:

jupyter lab

```
(machine_learning) fp818@IC-C02FX09EQ6LT ML4NS % jupyter lab
```

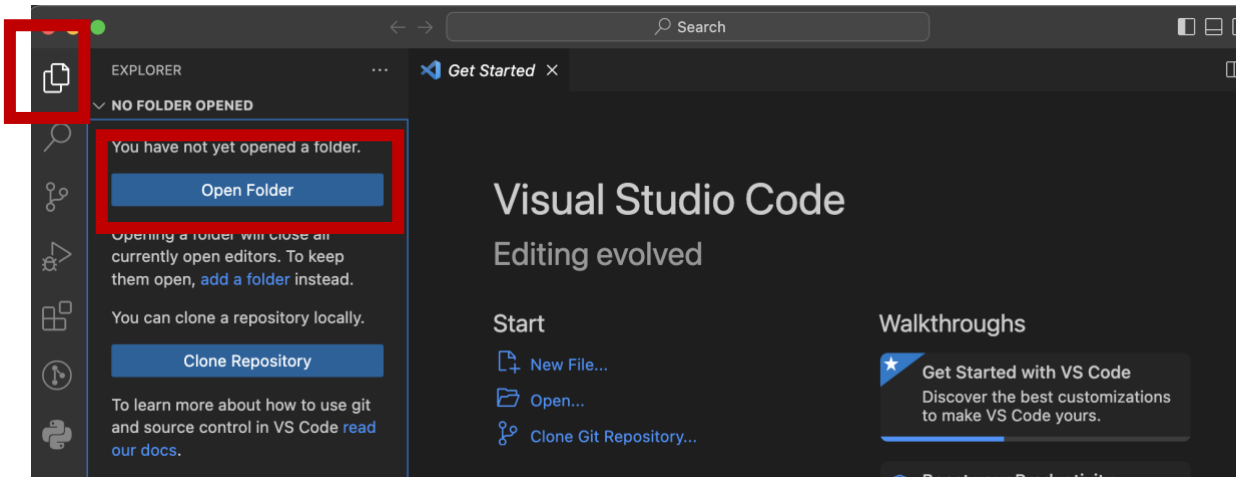
It will then open a new browser page with all the folders that you have in the path that you run the command



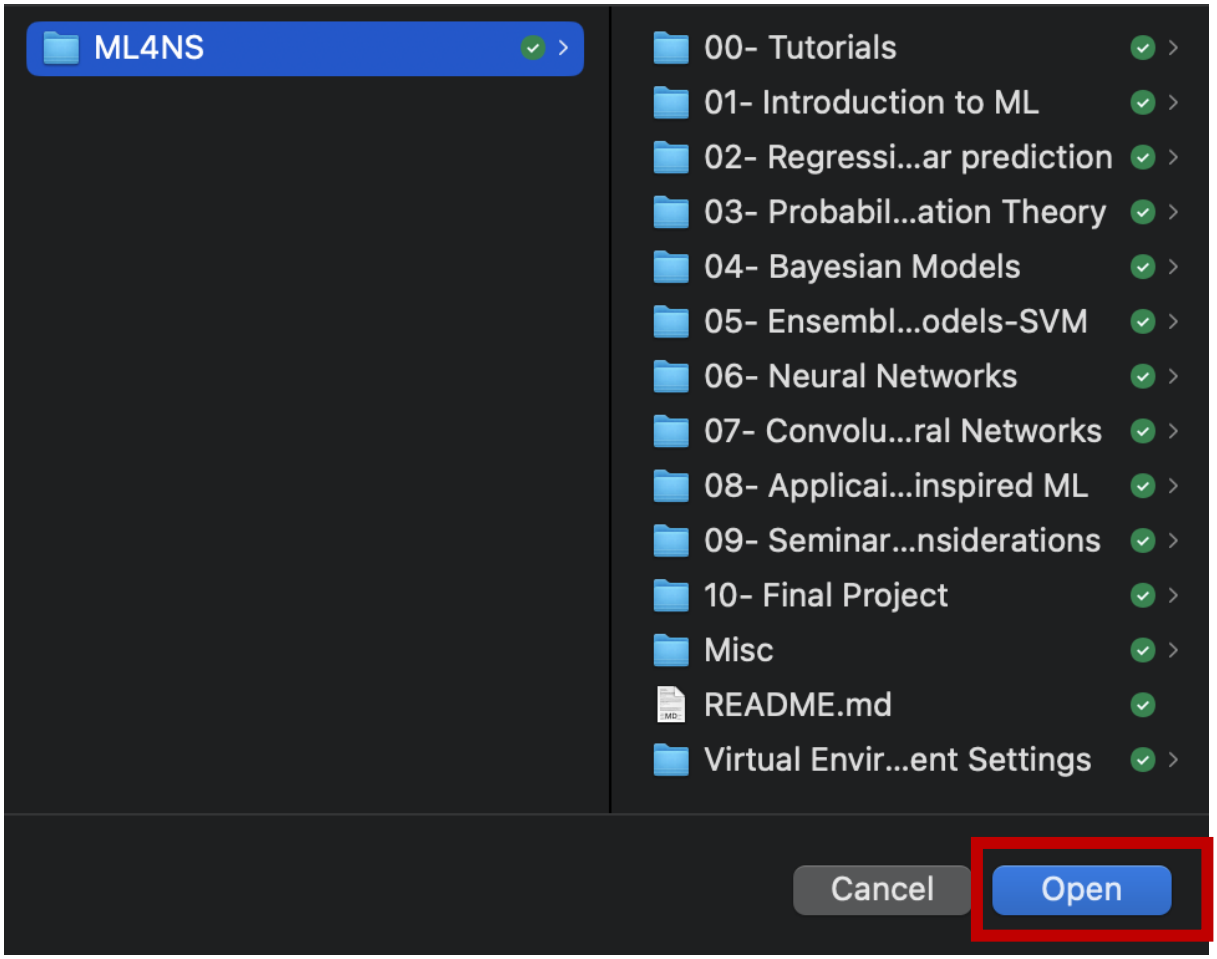


Open Jupyter Notebooks on Visual Studio Code (Suggested)

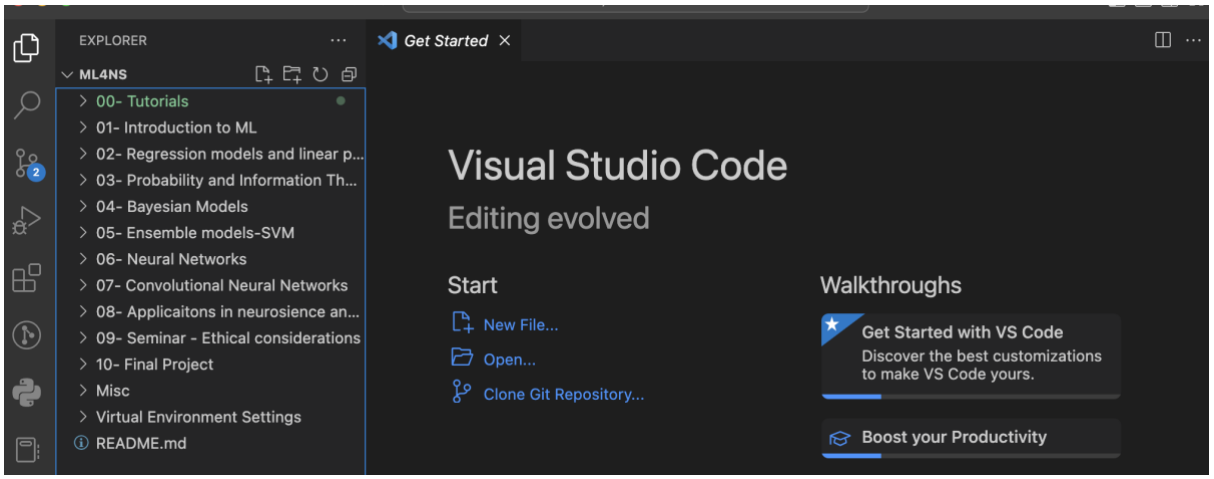
1. Open Visual Studio Code and Select Open Folder



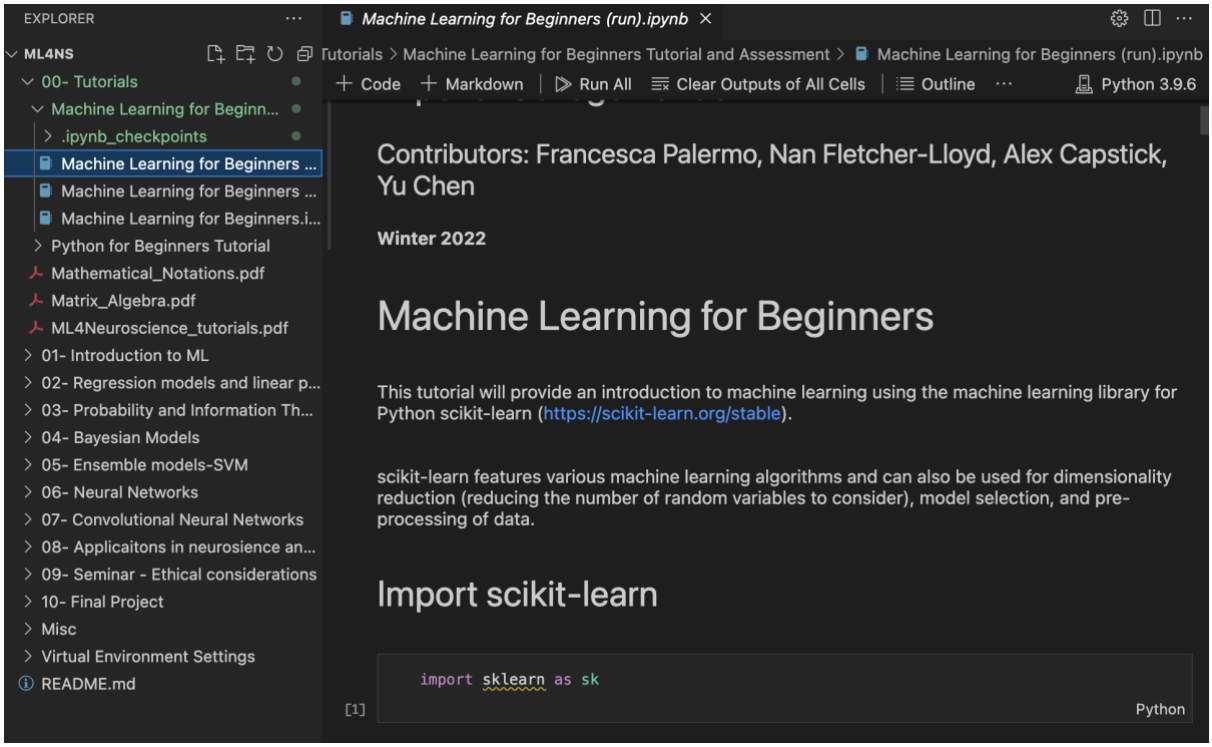
2. Locate the folder that you want to open, ML4NS in this case, and press Open



3. You will have the full list of folders and file on the left of your workspace

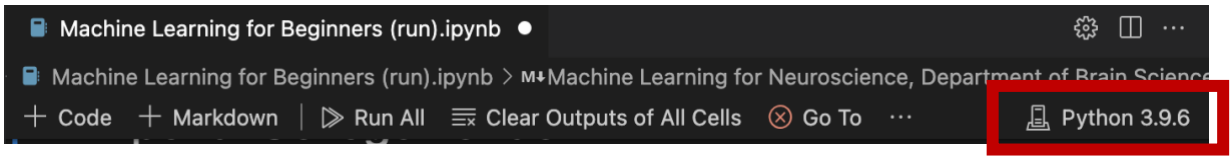


4. Open the file you are interested in. Machine Learning for beginners in this case.

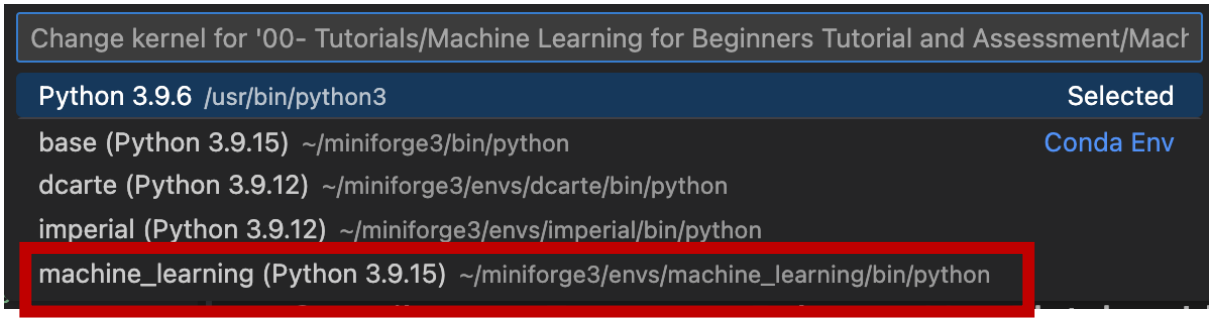


5. Before running the notebook, be careful on using the right kernel and virtual environment.

   Click on Python 3.9.6 or whichever version is showing on your visual studio code



   And choose the right virtual environment, machine_learning in this example



You will now have the correct environment and can run the experiments