COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI ICADE CIHS

Máster en Big Data. Tecnología y Analítica Avanzada (MBD).

Introducción al Análisis Estadístico con Lenguajes de Programación para Machine Learning (IAELPML). 2023-2024.

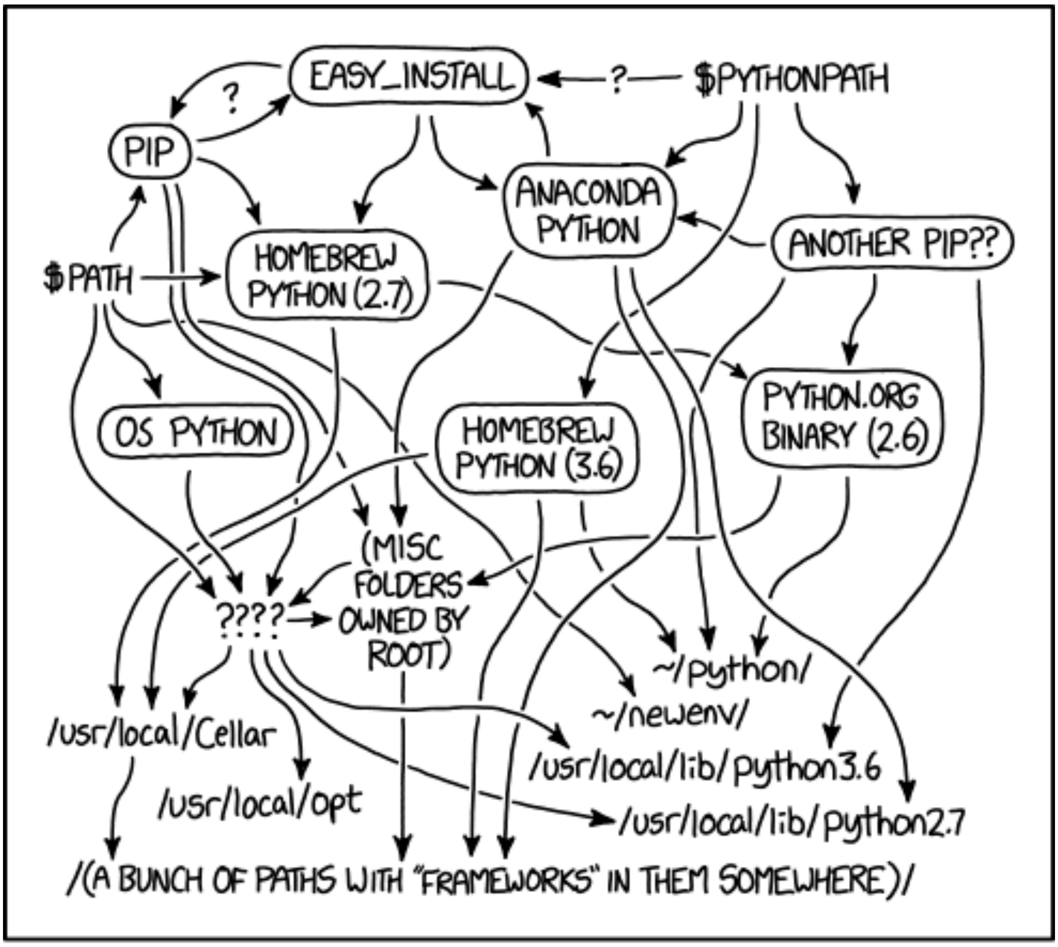
Software Setup, MacOS Version

IMPORTANT NOTE

IF YOU WILL BE USING A RECENT MAC WITH AN APPLE SILICON PROCESSOR FOR THIS COURSE (M1, M2 and similar processors) PLEASE TALK TO US BEFORE INSTALLING THE SOFTWARE BELOW. AND IF YOU ARE NOT SURE, LOOK IN THE APPLE MENU FOR THE "ABOUT THIS MAC" INFORMATION.

Before you proceed

- Please, read this document completely before starting your setup, so as to get a general idea of what is coming. In particular, please read the final *Alternative Setups* section. If you have any doubts, we are happy to help.
- **Be careful!** Another reason to read this document in advance is that we have seen too often users ruining their Python setups because of a rushed decision. Wrecking your system can be a matter of minutes, getting it back can take you hours or even days. Don't let the cartoon below be you!! When in doubt, please talk to us first.



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

Image from XKCD (https://xkcd.com/1987/)

Folder structure for the course

- You should create a **specific folder** for this course under your user folder. Under MacOS a typical location could be something like (the ellipsis represents possible intermediate folders):

```
/Users/your_user_folder/.../MasterBD/IAE/
```

- **The folder that you have created will always be referred to in these notes as the Course Folder**. In the above example we named it `IAE` but you can give it any name you want. As usual, it is a good idea to keep it short and meaningful (and free of spaces, unusual characters, etc.)
- We will be using Git and GitHub (https://github.com) repositories for the Course Notes and Assignments. In particular, the following repository will be used to share all the common course material:
  https://github.com/mbdfmad/IAELPML (https://github.com/mbdfmad/IAELPML)
  In the next section we will tell you how to get a local copy of this repository to your computer. For the time being you can browse the contents of the repository, but do not download it yet! This repository **will always be referred to in these notes as the Common Repository**. Soon you will be also using your *personal repository* and it is important to get the distinction and name conventions right from the beginning.

Software Setup and First Steps

Installing Homebrew

- Homebrew (https://brew.sh) is a *package manager* for macOS (or Linux), widely used to install software which is harder to install or unavalaible in other channels, such as the Apple App Store for MacOS. Many programming languages and utilities are easy to install using Homebrew.
- To install Homebrew itself you just need to foloow the simple instructions in the *Install Homebrew* right at the top of the HomeBrew web page. But **before doing that** we highly recommend reading through the following section, called *What Does Homebrew Do?*. And in case you have any doubts please ask us before proceeding.

Installing Git

Important Note: If you already have a working and recent setup of Git (current version is 2.37), skip to the *Installing Homebrew Python* subsection. If you do not know, open a terminal and try to run this command:

```
git --version
```

If you receive an error message then run:

```
brew install git
```

This will take some time. After the installation, try running the `git --version` command again.

Create a GitHub account

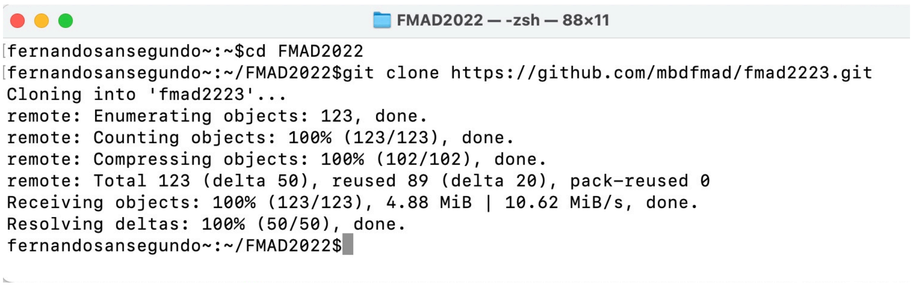
- **Important Note:** even if you already have a GitHub account you need to have an account that is linked with your university email (the one ending in `@alu.comillas.edu`). You can create that account using this link (https://github.com). Make sure to log out first if you were already logged with a *non-university-associated* account. And we recommend reading this advice first (https://happygitwithr.com/github-acct.html#username-advice).

Clone the *common repository*

- Using the Terminal navigate to the *Course Folder*. You can get there using the `cd` command. **Make sure that you have changed into the right folder before going further!!** You can use the `pwd` command to check your current location. Ask for help if you run into trouble.
- In that folder run the following command:

  ```
  git clone https://github.com/mbdfmad/IAELPML.git
  ```

  Git will begin cloning (downloading) the contents of the repo to a folder named `IAELPML` in your computer. The process looks like this (the prompt and number of cloned objects will surely be greater in your case):

  ```
  fernandosansegundo~:~$cd FMAD2022
  fernandosansegundo~:~/FMAD2022$git clone https://github.com/mbdfmad/fmad2223.git
  Cloning into 'fmad2223'...
  remote: Enumerating objects: 123, done.
  remote: Counting objects: 100% (123/123), done.
  remote: Compressing objects: 100% (102/102), done.
  remote: Total 123 (delta 50), reused 89 (delta 20), pack-reused 0
  Receiving objects: 100% (123/123), 4.88 MiB | 10.62 MiB/s, done.
  Resolving deltas: 100% (50/50), done.
  fernandosansegundo~:~/FMAD2022$
  ```

  The `IAELPML` folder will then be a subfolder of the *Course Folder*. You should consider this `IAELPML` folder as a *read-only* folder. You will soon see how to update its contents as the course progresses, but **you are not expected to directly modify the contents of the `IAELPML` folder** (and you shouldn't). You can *copy* any object from this folder (e.g. the code examples that we will provide) to other locations in your computer if you want to make changes. Finally, if you are a seasoned Git user you can opt for a fork of the repo instead of cloning it.

Keeping your local copy of the *common repository* updated.

- If you are used to auto syncing tools such as Dropbox, OneDrive, ICloud Drive and the like, you must keep in mind that **Git repos are not auto-syncing**. To keep the information in your repos updated you have to manually update them (both ways, from your computer to GitHub and viceversa).
- **This will be very important for our workflow in the lectures of the course.** So make sure to get it right, and ask for help soon if you need it. Every work session for this course will begin with the following ritual:
    - Open a *Terminal*.
    - Navigate to your local `IAELPML` folder. Do not confuse this with its parent folder, the *Course Folder*; in my case the parent course folder is called `IAE` , but in your case the name could be different. The repository folder name should however be always `IAELPML` .
    - Run `git status` to check that everything is ok.
    - Run `git pull origin main` . Depending on the update status of your local copy of the repo you will either download the new contents from the repo at GitHub or you will receive a message saying that you are `Already up to date` .
    - If you are going to modify any of the files in `IAELPML` and you want to keep the changes, **make sure to copy the files first to a different folder outside of the common repository.** Just create as many additional subfolders of the course folder as you see fit, but make so outside of the common repository.

  Don't worry too much, however. You cannot break anything, since you won't have write permissions for this GitHub repo. Worst case scenario, you can (backup your local files and) delete the `IAELPML` folder and clone it again to get the latest contents.

Setting up a local Git configuration for the Common Repository

- Using still the terminal, change into the `IAELPML` folder and set your local Git configuration with these commands (the first one is the folder change from the *Course Folder* to `IAELPML`):

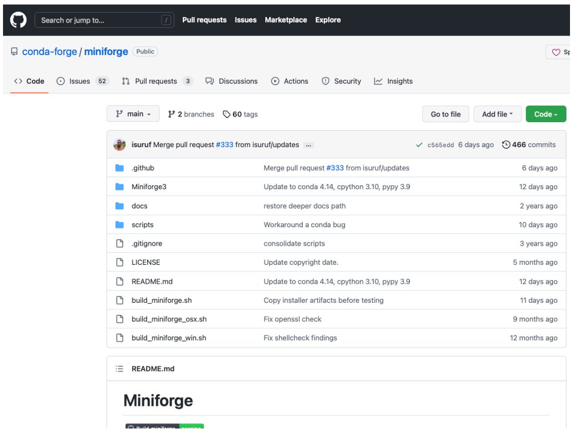
```
cd IAELPML
git config --local user.name
'replace_this_with_your_GitHub_username'
git config --local user.email
'replace_this_with_your_university_email'
git config --global init.defaultBranch main
```

- By the way, if you are doing this setup for the first time, you can now close the *Git Bash* terminal window.

Installing Python with HomeBrew

Important Note: If you already have a working setup of Python using conda for package and environment management (with Python 3.8 or later) then skip to the *Create a Conda Environment for the Course* subsection. If your Python version is older: get updated, people!

- Begin by navigating to the Miniforge GitHub repository (https://github.com/conda-forge/miniforge):



isuruf Merge pull request #333 from isuruf/updates ... ✓ c565edd 6 days ago 466 commits

.github	Merge pull request #333 from isuruf/updates	6 days ago
Miniforge3	Update to conda 4.14, cpython 3.10, pypy 3.9	12 days ago
docs	restore deeper docs path	2 years ago
scripts	Workaround a conda bug	10 days ago
.gitignore	consolidate scripts	3 years ago
LICENSE	Update copyright date.	5 months ago
README.md	Update to conda 4.14, cpython 3.10, pypy 3.9	12 days ago
build_miniforge.sh	Copy installer artifacts before testing	11 days ago
build_miniforge_osx.sh	Fix openssl check	9 months ago
build_miniforge_win.sh	Fix shellcheck findings	12 months ago

README.md

Miniforge

I suggest you at least take a look at the first sections. After that, if you skip the Downloads and scroll down to the `Install` section, there is a subsection called `Homebrew` (https://github.com/conda-forge/miniforge#homebrew). There you will see that all you need to do is open a terminal and run:

```
brew install miniforge
```

Wait for `HomeBrew` to finish setting the miniforge flavor of Python for you. Then in the same terminal execute

```
python
```

and you should see output like this:

```
Python 3.10.5 | packaged by conda-forge | (main, Jun 14 202
2, 07:07:06) [Clang 13.0.1 ] on darwin
Type "help", "copyright", "credits" or "license" for more i
nformation
>>>
```

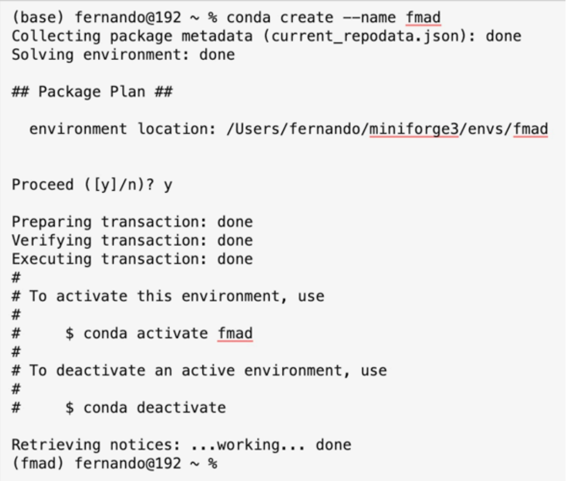
The `>>>` line is the Python prompt waiting for you. But we will not be using Python this way by now. Therefore simply type `exit()` in the Python prompt and execute it to return to the regular prompt of the MacOS terminal.

Create a Conda Environment for the Course

- A conda environment allows us to organize our work in projects, keeping each project dependencies isolated from the rest of your work. It is also the best way to avoid running into the potentially catastrophic situation in where you have several Python versions and they conflict to create a mess (https://xkcd.com/1987/). We urge you to **always use environments** to organize your work in Python. You can learn more about the use of Conda environments in this link (https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#). And to learn more about environments in Python in general, you can check these two links:
  - A Guide to Python's Virtual Environments (https://towardsdatascience.com/virtual-environments-104c62d48c54)
  - The Definitive Guide to Conda Environments (https://towardsdatascience.com/a-guide-to-conda-environments-bc6180fc533)
- We will be using a Conda environment named `IAELPML` for this course. To create the environment we will keep working in the terminal (make sure that you exited Python in the previous step, you must be in a regular terminal prompt for this step to work).
- In that prompt execute the command

  ```
  conda create --name IAELPML
  ```

  and after some moments something like this should happen (with IAELPML instead of fmad in the following screenshots):

  ```
  (base) fernando@192 ~ % conda create --name fmad
  Collecting package metadata (current_repodata.json): done
  Solving environment: done

  ## Package Plan ##

    environment location: /Users/fernando/miniforge3/envs/fmad


  Proceed ([y]/n)? y

  Preparing transaction: done
  Verifying transaction: done
  Executing transaction: done
  #
  # To activate this environment, use
  #
  #     $ conda activate fmad
  #
  # To deactivate an active environment, use
  #
  #     $ conda deactivate

  Retrieving notices: ...working... done
  (fmad) fernando@192 ~ %
  ```

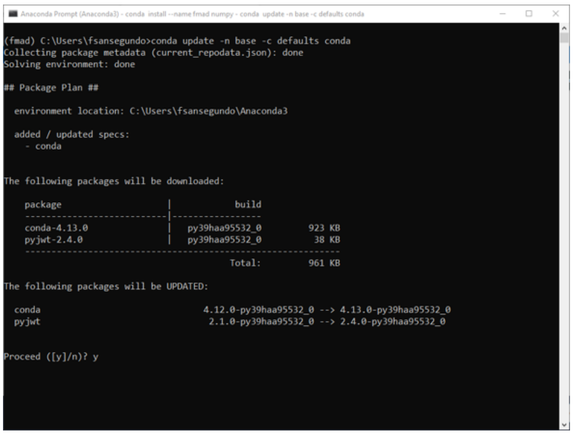
  For the time being ignore the warning about updating Conda if it appears (we will deal with it in the next step) and just answer *y* and hit *Enter*.

Update conda

- To update conda (if the warning appeared in the previous step) run this command:

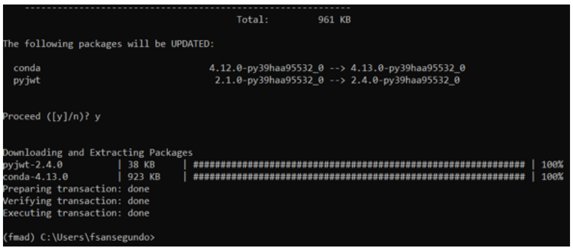
  ```
  conda update −n base −c defaults conda
  ```

  As the following picture illustrates conda will present you with a summary of the update operation. The picture is from Windows but the MacOS version is almost identical. Answer *y* and hit *Enter*.


Anaconda Prompt (Anaconda3) - conda install --name fmad numpy - conda update -n base -c defaults conda

```
(fmad) C:\Users\fsansegundo>conda update -n base -c defaults conda
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\fsansegundo\Anaconda3

  added / updated specs:
    - conda


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    conda-4.13.0               |   py39haa95532_0         923 KB
    pyjwt-2.4.0                |   py39haa95532_0          38 KB
    ------------------------------------------------------------
                                           Total:         961 KB

The following packages will be UPDATED:

  conda                               4.12.0-py39haa95532_0 --> 4.13.0-py39haa95532_0
  pyjwt                                2.1.0-py39haa95532_0 --> 2.4.0-py39haa95532_0


Proceed ([y]/n)? y
```

  Then conda will download the required software to autoupdate. The expected result looks like this:



```
    ------------------------------------------------------------
                                           Total:         961 KB

The following packages will be UPDATED:

  conda                               4.12.0-py39haa95532_0 --> 4.13.0-py39haa95532_0
  pyjwt                                2.1.0-py39haa95532_0 --> 2.4.0-py39haa95532_0


Proceed ([y]/n)? y


Downloading and Extracting Packages
pyjwt-2.4.0          | 38 KB     | ############################################################ | 100%
conda-4.13.0         | 923 KB    | ############################################################ | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(fmad) C:\Users\fsansegundo>
```

Activate the environment

- When we created the `IAELPML` environment in a previous step, Conda showed us the activation command:

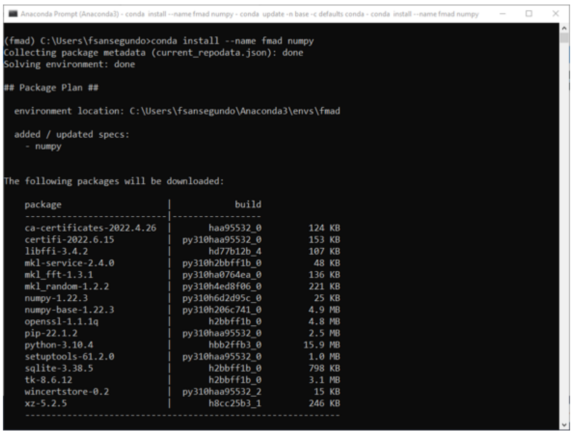
  ```
  `conda activate IAELPML`

  Run this command now. This will activate the newly created
  environment `IAELPML`. Note that the name of the active env
  ironment appears in parenthesis at the beginning of the pro
  mpt line (again, here you will see `IAELPML` instead of `fm
  ad`):

  <img src="fig/004-AnacondaEnvironmentCreate03.png" width=30
  %/>

  **Make sure that the environment is correctly activated bef
  ore proceeding!!** If you have any problem ask for help.
  ```

Install the first library

- The first library that we will install is [NumPy (https://numpy.org)](https://numpy.org), which provides the foundation for much of the work we will doing in this course. We will discuss NumPy in coming sessions of the course. But to use it we first need to install it in our `IAELPML` environment. Execute this command now:

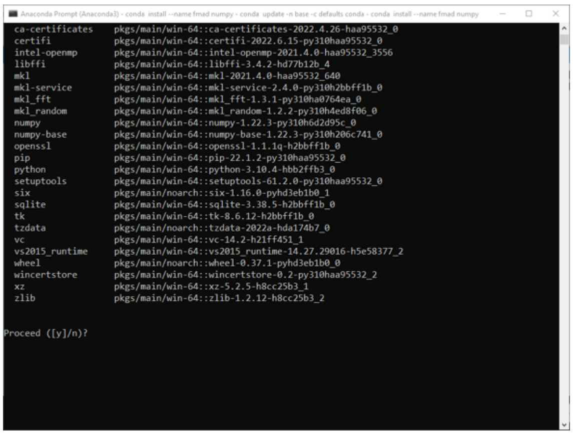
  ```
  conda install --name IAELPML numpy
  ```

  A summary of the libraries that need to be downloaded for the NumPy setup will appear

  
  Anaconda Prompt (Anaconda3) - conda install --name fmad numpy - conda update -n base -c defaults conda - conda install --name fmad numpy

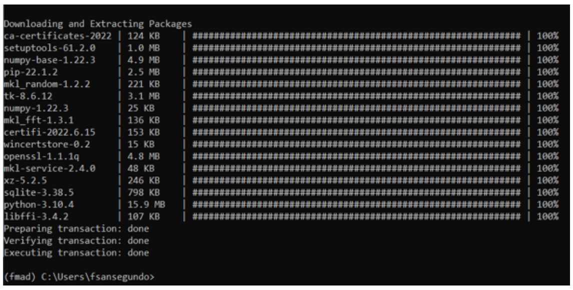
  ```
  (fmad) C:\Users\fsansegundo>conda install --name fmad numpy
  Collecting package metadata (current_repodata.json): done
  Solving environment: done

  ## Package Plan ##

    environment location: C:\Users\fsansegundo\Anaconda3\envs\fmad

    added / updated specs:
      - numpy


  The following packages will be downloaded:

      package                    |            build
      ---------------------------|-----------------
      ca-certificates-2022.4.26  |       haa95532_0         124 KB
      certifi-2022.6.15          |  py310haa95532_0         153 KB
      libffi-3.4.2               |       hd77b12b_4         107 KB
      mkl-service-2.4.0          |  py310h2bbff1b_0          48 KB
      mkl_fft-1.3.1              |  py310ha0764ea_0         136 KB
      mkl_random-1.2.2           |  py310h4ed8f06_0         221 KB
      numpy-1.22.3               |  py310h6d2d95c_0          25 KB
      numpy-base-1.22.3          |  py310h206c741_0         4.9 MB
      openssl-1.1.1q             |       h2bbff1b_0         4.8 MB
      pip-22.1.2                 |  py310haa95532_0         2.5 MB
      python-3.10.4              |       hbb2ffb3_0        15.9 MB
      setuptools-61.2.0          |  py310haa95532_0         1.0 MB
      sqlite-3.38.5              |       h2bbff1b_0         798 KB
      tk-8.6.12                  |       h2bbff1b_0         3.1 MB
      wincertstore-0.2           |  py310haa95532_2          15 KB
      xz-5.2.5                   |       h8cc25b3_1         246 KB
      ------------------------------------------------------------
  ```

  followed by a rather lengthy list of the libraries about to be installed (in most cases these two lists are almost equal)

  
  Anaconda Prompt (Anaconda3) - conda install --name fmad numpy - conda update -n base -c defaults conda - conda install --name fmad numpy

  ```
    ca-certificates    pkgs/main/win-64::ca-certificates-2022.4.26-haa95532_0
    certifi            pkgs/main/win-64::certifi-2022.6.15-py310haa95532_0
    intel-openmp       pkgs/main/win-64::intel-openmp-2021.4.0-haa95532_3556
    libffi             pkgs/main/win-64::libffi-3.4.2-hd77b12b_4
    mkl                pkgs/main/win-64::mkl-2021.4.0-haa95532_640
    mkl-service        pkgs/main/win-64::mkl-service-2.4.0-py310h2bbff1b_0
    mkl_fft            pkgs/main/win-64::mkl_fft-1.3.1-py310ha0764ea_0
    mkl_random         pkgs/main/win-64::mkl_random-1.2.2-py310h4ed8f06_0
    numpy              pkgs/main/win-64::numpy-1.22.3-py310h6d2d95c_0
    numpy-base         pkgs/main/win-64::numpy-base-1.22.3-py310h206c741_0
    openssl            pkgs/main/win-64::openssl-1.1.1q-h2bbff1b_0
    pip                pkgs/main/win-64::pip-22.1.2-py310haa95532_0
    python             pkgs/main/win-64::python-3.10.4-hbb2ffb3_0
    setuptools         pkgs/main/win-64::setuptools-61.2.0-py310haa95532_0
    six                pkgs/main/noarch::six-1.16.0-pyhd3eb1b0_1
    sqlite             pkgs/main/win-64::sqlite-3.38.5-h2bbff1b_0
    tk                 pkgs/main/win-64::tk-8.6.12-h2bbff1b_0
    tzdata             pkgs/main/noarch::tzdata-2022a-hda174b7_0
    vc                 pkgs/main/win-64::vc-14.2-h21ff451_1
    vs2015_runtime     pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
    wheel              pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0
    wincertstore       pkgs/main/win-64::wincertstore-0.2-py310haa95532_2
    xz                 pkgs/main/win-64::xz-5.2.5-h8cc25b3_1
    zlib               pkgs/main/win-64::zlib-1.2.12-h8cc25b3_2


  Proceed ([y]/n)?
  ```

  Answer yes to make the setup begin and wait patiently while NumPy and all of its requirements are downloaded and installed. The proccess should end like this:

  ```
  Downloading and Extracting Packages
  ca-certificates-2022 | 124 KB    | ############################################################ | 100%
  setuptools-61.2.0    | 1.0 MB    | ############################################################ | 100%
  numpy-base-1.22.3    | 4.9 MB    | ############################################################ | 100%
  pip-22.1.2           | 2.5 MB    | ############################################################ | 100%
  mkl_random-1.2.2     | 221 KB    | ############################################################ | 100%
  tk-8.6.12            | 3.1 MB    | ############################################################ | 100%
  numpy-1.22.3         | 25 KB     | ############################################################ | 100%
  mkl_fft-1.3.1        | 136 KB    | ############################################################ | 100%
  certifi-2022.6.15    | 153 KB    | ############################################################ | 100%
  wincertstore-0.2     | 15 KB     | ############################################################ | 100%
  openssl-1.1.1q       | 4.8 MB    | ############################################################ | 100%
  mkl-service-2.4.0    | 48 KB     | ############################################################ | 100%
  xz-5.2.5             | 246 KB    | ############################################################ | 100%
  sqlite-3.38.5        | 798 KB    | ############################################################ | 100%
  python-3.10.4        | 15.9 MB   | ############################################################ | 100%
  libffi-3.4.2         | 107 KB    | ############################################################ | 100%
  Preparing transaction: done
  Verifying transaction: done
  Executing transaction: done

  (fmad) C:\Users\fsansegundo>
  ```

Install aditional libraries

- NumPy is just the first of the libraries that we will use in the course, but many others will also come into play. To name a few of the most relevant:
    - NumPy (https://numpy.org)
    - Pandas (https://pandas.pydata.org)
    - MatPlotLib (https://matplotlib.org)
    - Scikit-Learn (https://scikit-learn.org/stable/)

  Take a few moments to visit the homepage for each of these libraries.

Exercise: install all these libraries to the `IAELPML` environment. To install several libraries you don't need to repeat `conda install` for each library; you can use a single install command and separate the names of the libraries with a space.

Install the Jupyter Notebook & Jupyter Lab

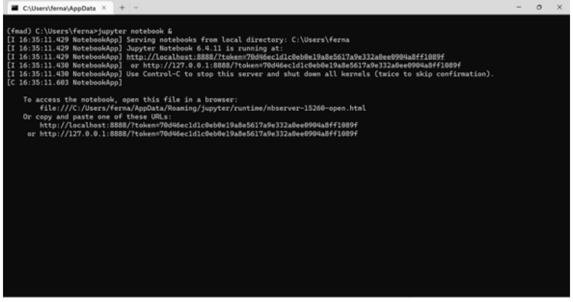
- In this course we will talk to Python using the onterface provided by *Jupyter Notebooks*. The [Jupyter Project (https://jupyter.org)](https://jupyter.org) provides a general purpose development environment which is well suited for Python (but not limited to it). It will play a central role in our work with Python, but it installs (to the `IAELPML` environment) just like any other library.

  ```
  conda install --name IAELPML jupyter
  ```

- When the setup ends execute the following two commands in order. The first one changes your current directory to your user folder in Windows. The second command opens the Jupyter Notebook in your *default Web Browser*:

  ```
  cd %HOMEPATH%
  jupyter notebook &
  ```

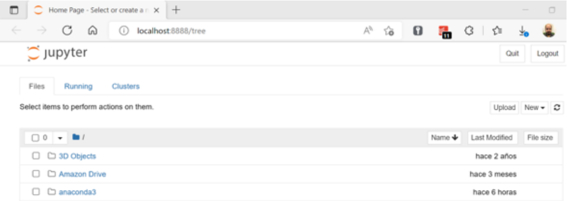
  Like this:



```
(fmad) C:\Users\ferna>jupyter notebook &
[I 16:35:11.429 NotebookApp] Serving notebooks from local directory: C:\Users\ferna
[I 16:35:11.429 NotebookApp] Jupyter Notebook 6.4.11 is running at:
[I 16:35:11.429 NotebookApp] http://localhost:8888/?token=70d46ec1d1c0eb0e19a8e5617a9e332a0ee0904a8ff1089f
[I 16:35:11.430 NotebookApp]  or http://127.0.0.1:8888/?token=70d46ec1d1c0eb0e19a8e5617a9e332a0ee0904a8ff1089f
[I 16:35:11.430 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 16:35:11.603 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/ferna/AppData/Roaming/jupyter/runtime/nbserver-15260-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=70d46ec1d1c0eb0e19a8e5617a9e332a0ee0904a8ff1089f
     or http://127.0.0.1:8888/?token=70d46ec1d1c0eb0e19a8e5617a9e332a0ee0904a8ff1089f
```

  Note: We recommend to always start Jupyter from a folder high enough in your folder structure: you won't be able to easily move up from the starting point later. Make sure, however, that you are at the very least, at the *Course folder* level.

  In a few moments your default *Web Browser* will open to a new page that shows the entry point for the running Jupyter server that we have just started:



Select items to perform actions on them.

Name	Last Modified	File size
3D Objects	hace 2 años	
Amazon Drive	hace 3 meses	
anaconda3	hace 6 horas	

You can use that page to navigate your folder structure until you get to the *Course Folder*. Once you are seeing the contents of that folder, open the `01a_test_your_setup.ipynb` file and follow the instructions there. If errors appear try to go over the setup procedure again and if you are still having trouble ask for our help.

Final Remarks

- The terminal window that you used to start Jupyter should remain open during all the work session. **Be careful:** Closing it can make the server crash and you may lose part of your work.
- Remember to make external copies of the files in `IAELPML` before modifying them.
- When you are done with the setup checks use the `File` menu and then `Close and Halt` the notebook. If your browser asks for confirmation that you really want to close the page: don't worry, you can safely close it.
- Every time you finish working with Jupyter you should go back to the `Home Page` of the Jupyter server (the one which shows your folder structure) and use the `Quit` button on the top right side of that window to shutdown Jupyter. A popup window will appear to inform you that you should close that browser tab.
- Return to the terminal that you used to initiate Jupyter. You should now see some messsage confirming that the server is shut down. If you are done working in this Python environment, issue the command:

  ```
  conda deactivate
  ```

  The prompt will change to indicate that you are back to the `base` conda environment. You can now close the terminal window as well if you wish.

Jupyter extensions

- If we need to install the extensions, please run this command:

  ```
  conda install -n IAELPML jupyter_contrib_nbextensions
  ```