Creating a Machine Learning Environment on your local machine

The following instructions will enable you to install a Machine Learning Environment suitable for this course on your local machine.

- We strongly suggest that you don't deviate from the instructions
- The more you stray from the suggestions, the more likely it will be that you encounter problems

Install the basic Machine Learning Environment

We will be installing Anaconda Individual Edition, a complete environment for Machine Learning.

It works on a variety of operating systems.

The instructions that follow assume that you **do not** already have a prior version of Anaconda installed.

If you do have a prior version, see the section Multiple versions of Anaconda on a single machine (Setup_ML_Environment_NYU.ipynb#Multiple-versions-of-Anaconda-on-a-single-machine)

Click on the link to begin the installation on your local machine.

https://www.anaconda.com/products/individual#Downloads
(https://www.anaconda.com/products/individual#Downloads)

- Easiest choice: Choose the "64 bit graphical installer"
- If prompted for any choices during installation
    - Accept the default choice, by simply pressing the Enter key
    - Allow the `.bashrc` file to be modified

Potential issue installing on a Mac with the ARM chip (M1/M2)

When installing via the graphical installer: the user should be given the option to create a link on their desktop to start up the Anaconda Navigator.

- Navigator is a convenience tool that presents buttons for starting common tools, e.g., Jupyter
- It is a convenience, not a necessity

At the time of writing, the "M1" graphical installer for the Mac ARM architecture is reported to **not** install Navigator.

The description and solution to the issue is given here (https://docs.anaconda.com/anaconda/install/mac-os/).

Multiple versions of Anaconda on a single machine

You might already have a version of Anaconda installed; if that is not the case please **skip this section**.

Anaconda makes it easy for multiple versions to co-exist.

- Each is called an *environment* in Anaconda terminology
- Multiple environments can co-exist
- You can easily switch between environments

So, if you want to install the latest Anaconda without destroying an existing installation

- create a new enviroment

It is important to note:

- all Anaconda commands that you issue (without explicitly giving an environment as an argument)
    - e.g., `conda install`
- will apply to the **current** environment
- all other environments will be un-affected

If you think that you have installed a package but can't find it

- you probably installed it in some other enviroment !

The "default" environment is called `base` ; this is the environment you will be in *until* you switch.

You can create a new environment (e.g., called `ml_course`) with the command

```
conda create --name ml_course
```

Each time you want to use an environment (e.g., `ml_course`), you need to **first activate it**

```
conda activate ml_course
```

Your command line prompt will change: it displays the name of the active environment.

You deactivate the enviroment with the command

```
conda deactivate
```

Thus, if you want to install the latest Anaconda to be compatible with this course

- but don't want to destroy an existing installation
- simply create a new environment for this course
    - and be sure to always activate it when using it for the course

It is **not recommended** to try to install a new version "on top of" an existing version.

- It is best to uninstall the prior version
- to uninstall it, following [these instructions (https://docs.anaconda.com/anaconda/install/uninstall/)](https://docs.anaconda.com/anaconda/install/uninstall/).

Install Jupyter extensions (optional)

Although this step is not strictly necessary, installing the following extensions to Jupyter will make you more productive.

Enter the following commands from a terminal:

```
# Install contributed extensions
conda install -y -c conda-forge jupyter_contrib_nbextensions

# Install Javascript and CSS files
jupyter contrib nbextension install --user

# Activate extensions
jupyter nbextension enable toc2/main
jupyter nbextension enable collapsible_headings/main
jupyter nbextension enable livemdpreview/livemdpreview

# Extension for slides
conda install -y -c conda-forge rise
jupyter nbextension enable rise
```

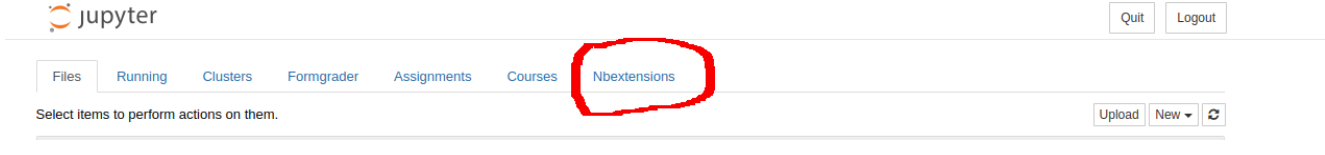
Potential issues in installing Jupyter extensions

Some users have reported difficulties in installing the extensions in new releases of Anaconda.

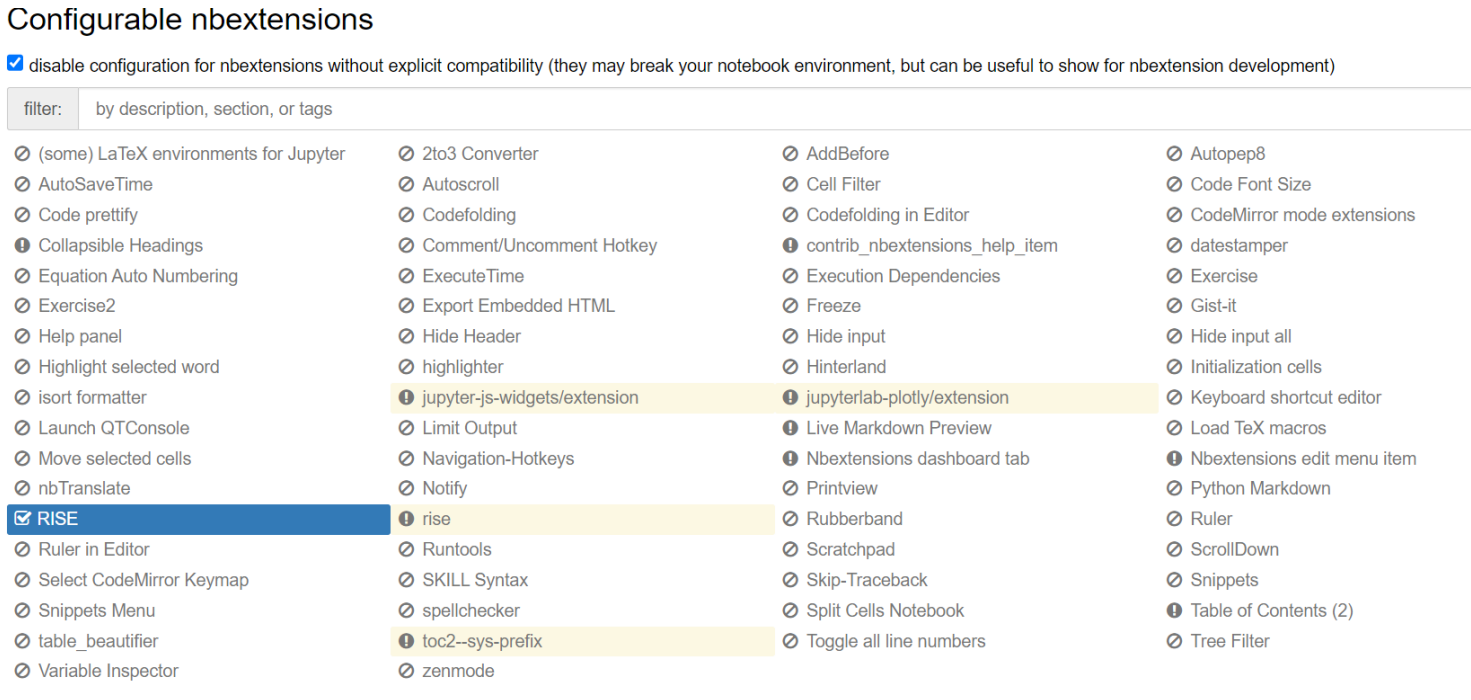
- Symptom: some of the above extensions are not active

The following steps have been reported as partial solutions.

- When you start Jupyter, you will see the control panel
  - the "Nbextensions" tab should be visible along the top banner



- Clicking on the Nbextensions tab will take you to a screen where you can manually activate/deactivate extensions



- Make sure to uncheck the checkbox: `disable configuration for nbextensions without explicit compatibility ..`
- You will then be able to manually activate/deactive the extensions we tried to install
    - `Collapsible headings`
    - `RISE`
    - `Table of Contents`

Jupyter security (optional)

If you run the Jupyter server on a machine that is connected to a network

- It is possible to allow remote users to connect to your server
- This **will not** happen by default

Should you decide to let remote users connect, it is strongly advised that you protect the notebook server with a password.

See [Public Jupyter Server (https://jupyter-notebook.readthedocs.io/en/stable/public_server.html)](https://jupyter-notebook.readthedocs.io/en/stable/public_server.html) for details.

Create a directory for your notebooks

We will refer to this directory as the *notebooks directory*.

- It can be anywhere on your machine
- My convention is to have a directory called "Notebooks" in my home directory.
    - The instructions below will use `Notebooks` as the notebooks directory; modify the instructions if you choose a different location

This is the location to which the course materials will be downloaded.

Enter the following commands from a terminal:

```
# Create directory for notebooks
cd
mkdir Notebooks
cd Notebooks
```

Download the course material

We are using a tool called `git` to manage our code.

`git` defines the notion of a *repository* as a collection of material related to a project.

The following will

- Install `git`
- Download the course material from a repository we have created
    - Into the current directory
    - Which is the `Notebooks` directory if you followed the previous step
- Download material from the recommend text books

There is a repository on Github for this course.

- The name changes every semester but follows a pattern like

  `{course}_{term}_{year}`

  - for example

    `ML_Spring_2023` or `ML_Advanced_Spring_2023`

Check with the instructor or GA to make sure you are using the correct repo !

In the instructions that follow

- either replace references to $REPO with the name of the repo
- OR set an environment variable REPO equal to that name, for example

  ```
  REPO="ML_Spring_2023"
  ```

  - this will cause references to $REPO to evaluate to the name that you set
  - so you don't need to explicitly replace references to $REPO

Enter the following commands from a terminal:

```
# Install git
conda install -y git

# Clone git repositories
git clone https://github.com/kenperry-public/$REPO.git

# Change to the course directory and clone the repos for the external reference
s
cd $REPO
mkdir external
cd external

git clone https://github.com/jakevdp/PythonDataScienceHandbook.git
git clone https://github.com/ageron/handson-ml2.git
```

You should verify that your notebooks directory has the following subdirectories

- `$REPO`
    - where $REPO represents the string name of the repository
    - The course material
    - This is where you will find the lectures
- `external` subdirectory containing
    - `handson-ml`
        - Notebooks associated with the recommended textbook by Geron.
    - `PythonDataScienceHandbook`
        - The *complete text* and associated notebooks of the recommended textbook by VanderPlas

Refreshing your local copy of the repo

In advance of each week's lecture, new material will be published to the repo.

Additionally: corrections might be made for previously published material.

You can always synchronize your local copy with the main repo via the command:

```
git pull
```

If you have made changes to your local repo that conflict with the main repo

- `git pull` will result in an error

Should this occur, you should use

```
git stash
git pull
```

The `git stash` will "hide" the changes you made and restore your local copy to be consistent with the main repo.

Fear not ! Your changes are hidden, not lost, and you will be able to restore them.

- Please see the `git stash` documentation.

Deep Learning Course: Install additional packages for Deep Learning

You have installed the "basic" Machine Learning Environment, which suffices for the Classical Machine Learning part of the course.

When you are ready for the Deep Learning part of the course (or want to play with Neural Networks/Deep Learning) you will need to install additional packages.

The installation method varies depending on whether or not you are using an Apple machine with the new ARM based architecture (e.g., M1 or M2 chip).

Instructions if you are ARE NOT using an Apple ARM based machine

Enter the following commands from a terminal:

```
# Install Tensorflow
conda install -y tensorflow
```

Instructions if you ARE using an Apple ARM based machine

See the instructions here (Installing_TensorFlow_on_MAC_M1_or_M2.ipynb)

Tools for visualization of graphs (optional)

Some of our notebooks uses a particular package (GraphViz) to visualize graphs

- Decision trees
- Deep Learning models (using the Keras `plot_model` command)

If you want to use this, you must install additional packages as follows

```
conda install pydot
conda install python-graphviz
```

As always: if you have multiple Conda environments, be sure you have activated the desired environment before issuing any `conda install` command.

Potential issue: visualization tools based on pydot not working

Students (mainly using Mac M1/M2 machines) have reported issues using

- the Keras `plot_model` function
- probably also in module for Decision Trees

when running in Jupyter.

A message arises, suggesting that

```
"You must install pydot ...".
```

You should have previously done so (first changing to the correct Conda environment, e.g., "tensorflow") via the commands

```
conda install pydot
conda install python-graphviz
```

Yet, the same error message is returned for some.

The resolution is to insert the following command in Jupyter before invoking any function that complains about not being able to find `pydot`

```
%run plot_model_fix.ipynb
```

It's not necessary for you to understand the solution, but for those who are interested, keep reading.

The code is open source so we can examine the source of the failure.

`plot_model` is calling a command line program (dot).

In order for this to work, the path to this program must be included in your PATH environment variable.

The error message occurs because

- the program can't be found in the PATH existing when the notebook starts.
- Our solution remedies this by modifying the environment variable.

It is less than desirable to have functions call command line programs

- e.g., because of the difficult to diagnose type of error we encounter.

It would be far better for the command line program

- to be using an API that could also be called by `plot_model`.

But it may be the case that there is no such API

- in which case the authors of the function have no choice but to resort to this problematic type of code.

Start up a Jupyter Notebook server

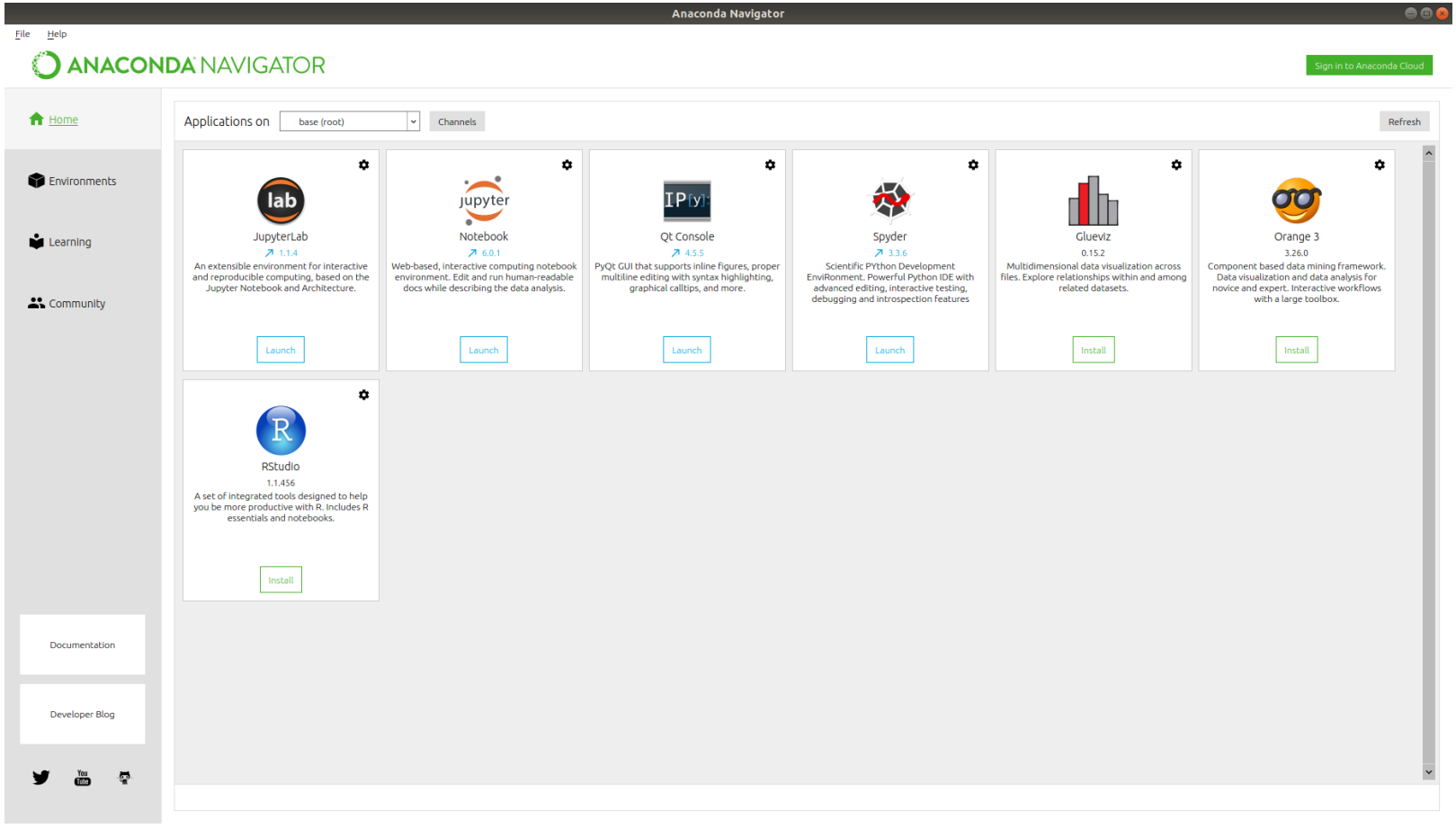
As part of the installation of Anaconda, a tool called the Anaconda Navigator was installed.

Find and start it

- An icon may have been created on your desktop
- Otherwise, enter the following commands from a terminal:

cd ~/anaconda3/bin/anaconda-navigator &

You will hopefully see something like this



- Click on the icon for `Jupyter Notebook` to start the Jupyter server.
- Navigate to the `$REPO` directory to find the course material directory
- Find the `Index.ipynb` notebook and open it
    - This will take you to the "start page" for the lectures

In [2]:
```
print("Done")
```

```
Done
```