Anaconda Setup

Anaconda - Individual Edition

“The open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.”

“Individual Edition is an open source, flexible solution that provides the utilities to build, distribute, install, update, and manage software in a cross-platform manner. Conda makes it easy to manage multiple data environments that can be maintained and run separately without interference from each other.”

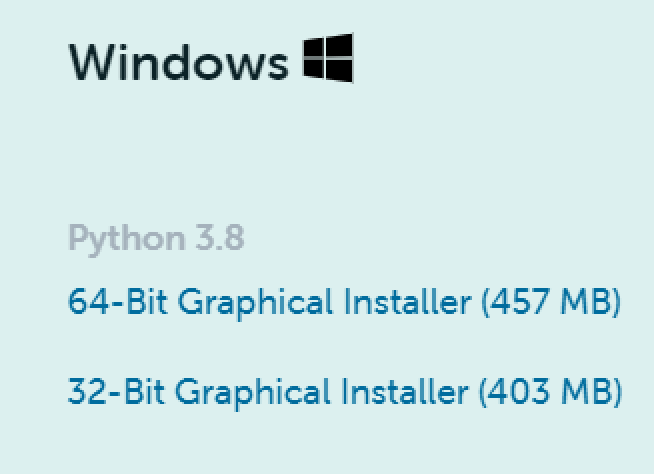
- Anaconda presentation page

Anaconda setup

You can follow the guides found at https://docs.anaconda.com/anaconda/install/ for all platforms.
We’ll cover the steps from the Anaconda documentations for Windows and Linux users here. We will only be working with Python 3.

Installing on Windows

1. Download the Anaconda installer



Windows

Python 3.8

64-Bit Graphical Installer (457 MB)

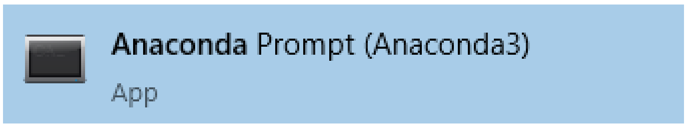
32-Bit Graphical Installer (403 MB)

Choose your Operating System (most likely 64-Bit)

2. Install

> Double click the installer to launch
> Click Next
> Agree to licensing terms
> Install for “Just Me”
> Select destination folder and click Next
> Recommended - Do not add Anaconda3 to PATH environment variable (we will use the Anaconda Prompt each time when working with Anaconda)
> Recommended - Register Anaconda3 as default Python <python_version>
> Click Install
> Next

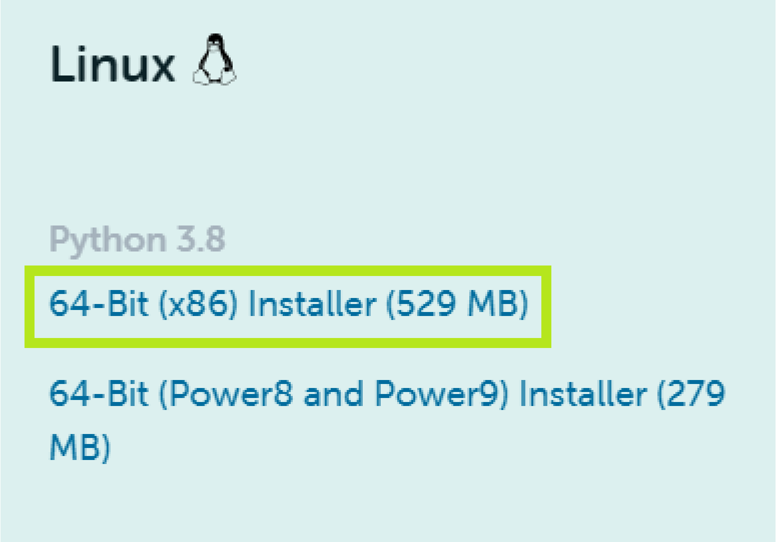
3. Open Anaconda Prompt



You will be using this instead of cmd (or you can use PyCharm virtual environments)

Installing on Linux

1. In your browser, download the Anaconda installer for Linux.


Linux
Python 3.8
64-Bit (x86) Installer (529 MB)
64-Bit (Power8 and Power9) Installer (279 MB)

Choose the first 64-Bit Installer

2. Install

> bash ~/Downloads/Anaconda3-2020.02-Linux-x86_64.sh (replace ~/Downloads/ with the path to where you downloaded/moved your .sh file)

> Press Enter to view license terms
> Agree to license terms

> Recommended - Accept default install location
> Recommended - Choose "yes" when asked if installer should run conda init
> Close and open your terminal or just type `source ~/.bashrc`

Conda environments

Anaconda vs Conda

`conda` is both a command line tool, and a python package.
Miniconda installer = Python + `conda`
Anaconda installer = Python + `conda` + meta package `anaconda`

Conda environments

Anaconda environments help maintain separate projects with the help of different separate virtual environments.
Different projects/applications likely have different package dependencies. Some dependencies from one project may be incompatible with other dependencies from a different project. This is why the ideal solution is to create one environment for each project.
The packages inside virtual environments can be installed either with pip or conda. (running pip install while inside a conda environment will install the package only inside that environment).

Managing environments

Conda environments documentation - https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

In order to work with conda environments you will always be using the *conda* command line tool.

Create conda environments

1. Empty environment creation

```
conda create --name myenv
```

press y when asked "proceed ([y]/n)?"

replace myenv with whatever name you want to give your environment (could be the name of the project for which you'll be using the environment)

2. Environment with specific python version

   ```
   conda create -n myenv python=3.6
   ```

3. Environment with specific python version and multiple packages (not necessary, you can just create your environment and install the packages later)

   ```
   conda create -n myenv python=3.6 scipy=0.15.0 astroid babel
   ```

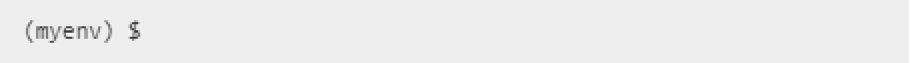
```
(-n and --name mean the same thing, specify the name of your environment)
```

Activate conda environments

Simply creating the environment will not be sufficient, you will need to also activate it in order to change your current environment to the newly created one.

```
conda activate myenv
```

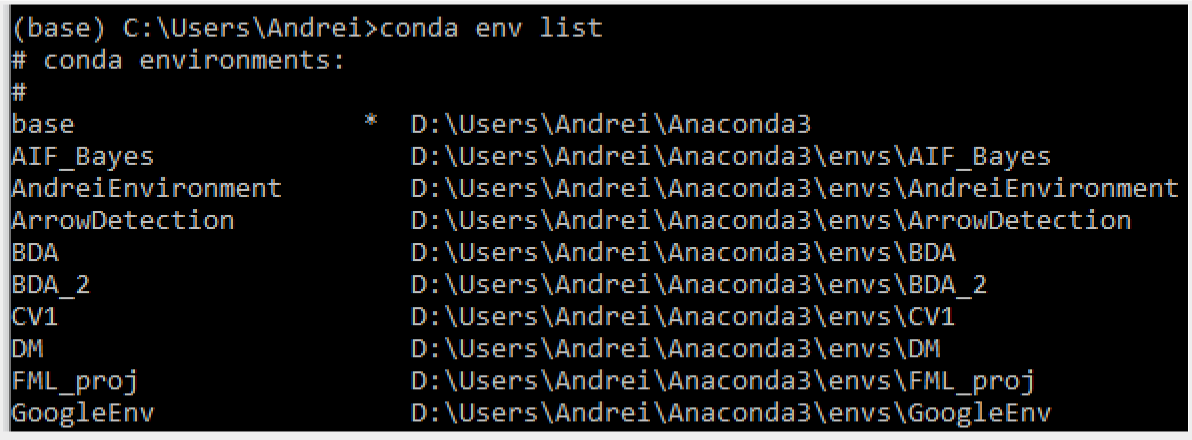
After running the command, you should be able to see to the left of your command line that you are indeed in the desired environment. By default, the active environment---the one you are currently using---is shown in parentheses () or brackets [] at the beginning of your command prompt.



```
(myenv) $
```

View list of conda environments

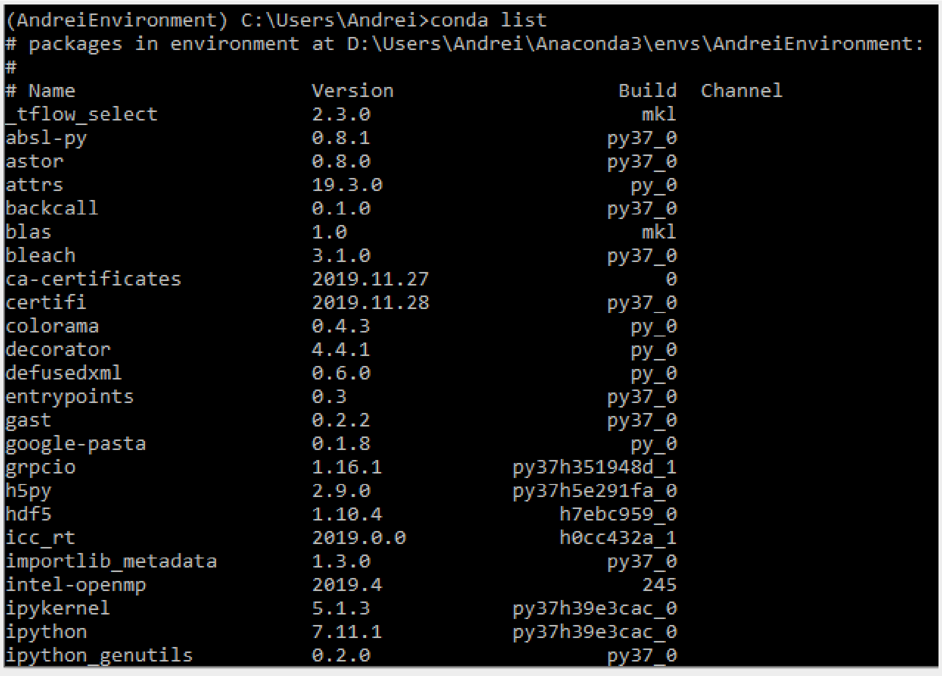
```
conda env list
```



```
(base) C:\Users\Andrei>conda env list
# conda environments:
#
base                  *  D:\Users\Andrei\Anaconda3
AIF_Bayes                D:\Users\Andrei\Anaconda3\envs\AIF_Bayes
AndreiEnvironment        D:\Users\Andrei\Anaconda3\envs\AndreiEnvironment
ArrowDetection           D:\Users\Andrei\Anaconda3\envs\ArrowDetection
BDA                      D:\Users\Andrei\Anaconda3\envs\BDA
BDA_2                    D:\Users\Andrei\Anaconda3\envs\BDA_2
CV1                      D:\Users\Andrei\Anaconda3\envs\CV1
DM                       D:\Users\Andrei\Anaconda3\envs\DM
FML_proj                 D:\Users\Andrei\Anaconda3\envs\FML_proj
GoogleEnv                D:\Users\Andrei\Anaconda3\envs\GoogleEnv
```

View environment packages list

```
conda list
```



```
(AndreiEnvironment) C:\Users\Andrei>conda list
# packages in environment at D:\Users\Andrei\Anaconda3\envs\AndreiEnvironment:
#
# Name                    Version                   Build  Channel
_tflow_select             2.3.0                       mkl
absl-py                   0.8.1                    py37_0
astor                     0.8.0                    py37_0
attrs                     19.3.0                     py_0
backcall                  0.1.0                    py37_0
blas                      1.0                         mkl
bleach                    3.1.0                    py37_0
ca-certificates           2019.11.27                    0
certifi                   2019.11.28               py37_0
colorama                  0.4.3                      py_0
decorator                 4.4.1                      py_0
defusedxml                0.6.0                      py_0
entrypoints               0.3                      py37_0
gast                      0.2.2                    py37_0
google-pasta              0.1.8                      py_0
grpcio                    1.16.1           py37h351948d_1
h5py                      2.9.0            py37h5e291fa_0
hdf5                      1.10.4               h7ebc959_0
icc_rt                    2019.0.0             h0cc432a_1
importlib_metadata        1.3.0                    py37_0
intel-openmp              2019.4                      245
ipykernel                 5.1.3            py37h39e3cac_0
ipython                   7.11.1           py37h39e3cac_0
ipython_genutils          0.2.0                    py37_0
```

Installing packages inside environments

There are 2 ways to install packages in conda environments:
1st way is by using the conda install command

```
conda install package_name
```

The 2nd (and perhaps more frequent) method is by using the pip package manager

```
pip install package_name
```

You may have to install pip using conda first if it is not already preinstalled

```
conda install pip
```

You should be all set with Anaconda and conda environments now. In short, in order to get a conda environment up and running you will have to run:

1. conda create --name my_environment_name
2. conda activate my_environment_name
3. pip install <my_package_name> - for all the packages that you want to install

Jupyter Notebook

"Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document."

from https://www.nature.com/articles/d41586-018-07196-1

Install Jupyter Notebook

`conda install -c conda-forge notebook` - with conda

`pip install notebook` - or with pip

or

Install JupyterLab

```
conda install -c conda-forge jupyterlab
```

```
pip install jupyterlab
```

Start Jupyter Notebook

`jupyter notebook`

or

Start JupyterLab

```
jupyter-lab
```

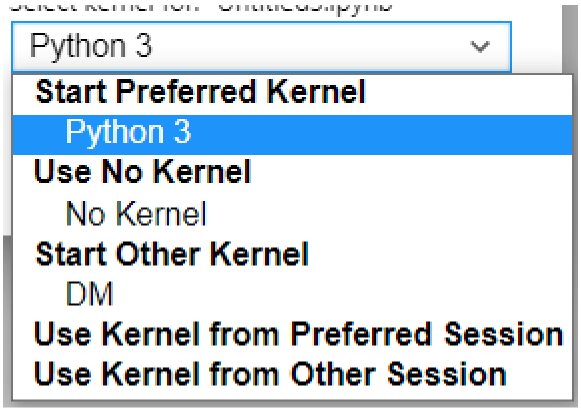
Add a conda environment to Jupyter Notebook or JupyterLab

```
conda activate cenv
```

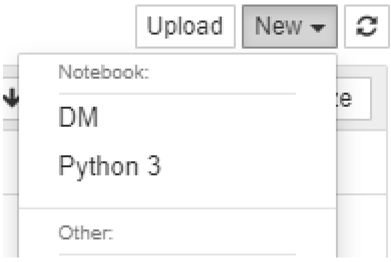
```
conda install ipykernel
```

```
ipython kernel install --user --name=<any_name_for_kernel>
```

All set, now you can start/restart your J Notebook/Lab and should be able to see your environment.



In JupyterLab when creating a new Notebook you can choose what environment (Kernel) to use.



The same thing can be done in Jupyter Notebook (choose between default Python 3 - base, or DM environment).

Experiment creation of conda environments:

1. Create a conda environment called Env1_3.7 that uses python 3.7 and install the jupyter notebook package inside of it.

   Run the following code in a newly created notebook inside jupyter notebook

   ```
   import sys
   print("Python version")
   print (sys.version)
   ```

2. Create a conda environment called Env1_3.8 that uses python 3.8.8 and install the jupyter notebook, numpy and pandas packages inside of it.

   Use this environment and **proceed with the DM1.ipynb file** for this laboratory.