Anaconda + Tensorflow (Keras) setup guide

The following guide describe easy steps to setup Anaconda and Tensorflow, in order to create and train neural networks in Keras. In the following you will need

- your computer and internet connection

- a web browser

- testfiles `test.ipynb` and `plotmatrix.py`. These files will be used at the end to test if your installation was successful.

Important! The following guide is a simplified setup instructive that has worked fine for me. In this steps doesn't work for you, I recommend to see the installation guide suggested in the Anaconda documentation websites:

- https://docs.anaconda.com/anaconda/install/windows/

- https://docs.anaconda.com/anaconda/install/linux/

- https://docs.anaconda.com/anaconda/install/mac-os/

The actual Anaconda installer corresponds to Python 3.8.

In the case of the Windows installation, this has worked well in Windows 7 Professional, or later versions. You must be sure if your computer is 32bit or 64bit, before download the correct version of the installer (recent computers are probably 64bit).

In the case of Linux, this steps have worked successfully in Ubuntu distributions version 16.04, 18.04 and Ubuntu 20. If you have other Linux distribution, (*e.g.* Debian, RedHat, Fedora, Gnome, CentOs, ...). I suggest you to read the official documentation of Anaconda before any installation.

For those Mac-Os users! I must say that I have never used Mac-Os, so I have not guarantee that this guide will work in your the case of Mac-OS systems. Fortunately, they work pretty similar to Linux. Anyway, I encourage you to read the official Anaconda documentation before any installation.

Contents

1 Installing Anaconda

Download the anaconda installer at

- https://www.anaconda.com/products/individual

(at the end of the webpage, in the section Anaconda Installers). Once you have downloaded the installer file, proceed with the following steps. If you already have installed Anaconda in your computer, you can skip this and go direct to Step 2.

Windows

Go to the Downloads folder, and run the downloaded `.exe` file. For example
```
Anaconda3-2020.07-Windows-x86_84.exe
```

Ubuntu

Open your terminal shell. Go to the Downloads folder, *e.g.*
```
$ cd Downloads
```

and run the downloaded `.sh` file. for example
```
$ ./Anaconda3-2020.07-Linux-x86_84.sh
```

Another option to run the file is as follows:
```
$ sh Anaconda3-2020.07-Linux-x86_84.sh
```

Or, if you prefer, you can run the executable file in the graphic interface.

MacOS

Open your terminal shell. Go to the Downloads folder,
```
$ cd Downloads
```

and run the downloaded `.pkg` file. for example
```
$ installer -pkg Anaconda3-2020.07-MacOSX-x86_84.pkg
```

or
```
$ sudo installer -pkg Anaconda3-2020.07-MacOSX-x86_84.pkg
```

2 Working with Anaconda

Once you have completed the Anaconda installation, you must be able to open the Anaconda prompt (for those Windows users), or to call conda from shell (for those Linux and MacOs).
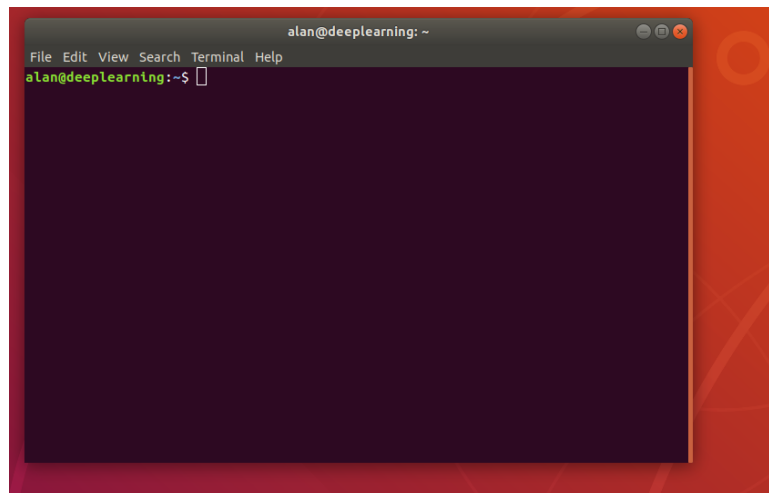
Windows

Go to the Windows menu (in the low left corner of your screen), and search for the **Anaconda Prompt**. When you click on it, it will display a command prompt window that looks similar to the usual Windows command prompt (the difference is that you are now inside Anaconda).
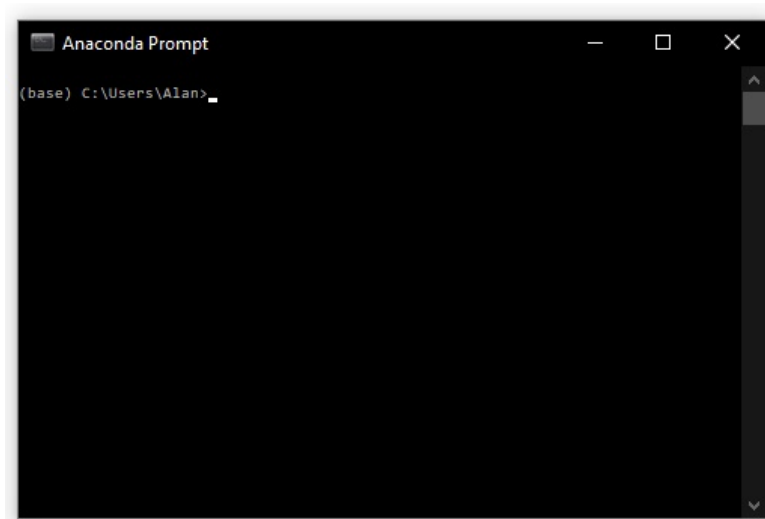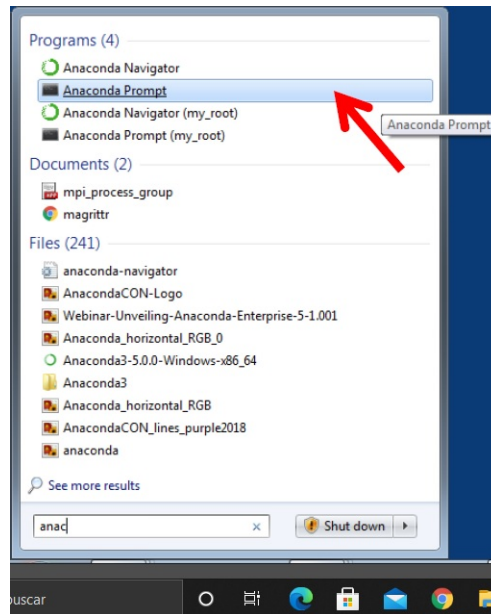
Ubuntu and MacOS
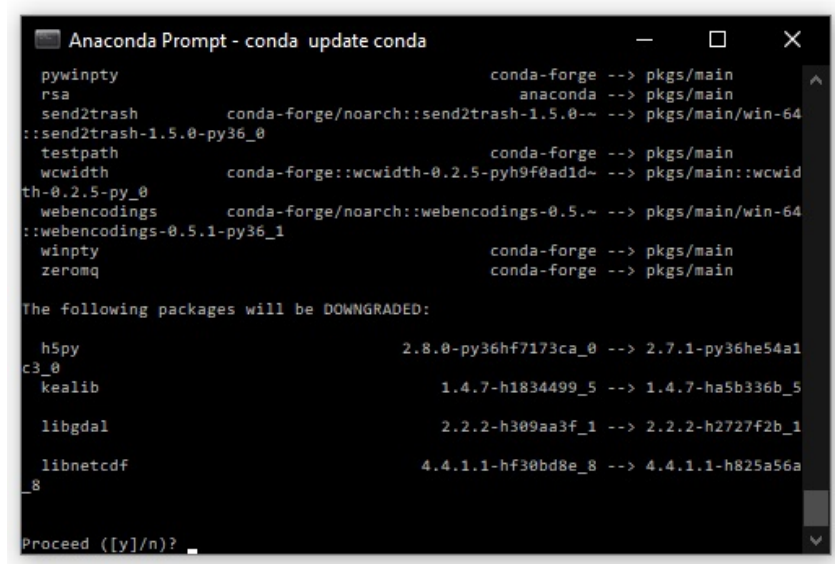
Open your terminal shell.

2.1 Updating Anaconda

For all of you that previously had installed Anaconda (I mean, this is not the fist time you install Anaconda), I recommend to update conda. Updating conda, will update conda packages to the latest compatible version. In other words, it will update the information from the official Anaconda package repository, and when install new packages, it will install the latest versions that are compatible with all other packages in the environment.

For those first installers, you can skip this (but you can also do it to be sure that conda has all its packages updated).

All systems: Windows, Ubuntu or MacOS

In your Anaconda prompt or terminal, write the following command
```
conda update conda
```

This will take some minutes. When the command requires you to proceed

simple type y and press Enter. Again, this will take some minutes since conda is updating a list of basic packages to its latest version.

2.2 Create a new environment

When you install Tensorflow for the first time, it is usually recommended to do it in a new conda environment. The reasons for this are the following:

- Sometimes when you install a new package, it brokes some older previous packages. If you use a new environment, it maintains everything you already have in your actual Anaconda intact. The new installation doesn't change what you already had there before.

- Conda manages its environments pretty easy. You can update the version of the installed packages inside an environment, and keep all other installations that already work well in other environments without changes.

- Probably the most important: if your installation fails, it saves the base environment, and it is easy to remove the new (failed) environment, and start again the installation,
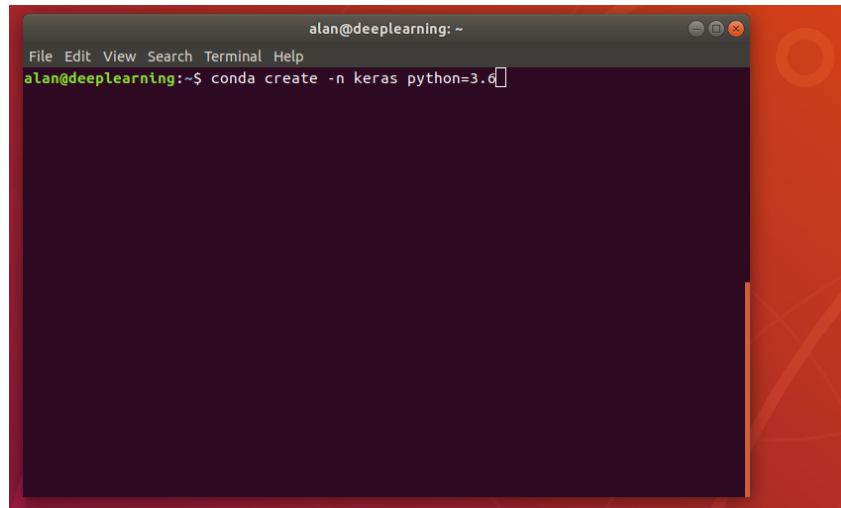
After updating, we will now create a new conda environment. You will need to name your new environment, (so take some time to think well the name, since conda has not a rename option). I suggest to use a name related to the main package you will install in this new environment. For example, in this case we can use one of the names:

- `tensorflow` (since we will install Tensorflow as our main package),

- or `tf2.0` (since we will install Tensorflow 2.0),

- or `keras` (since we will use Keras inside Tensorflow 2.0).

From now on, I will illustrate the following steps using the name `keras`. We will create a new environment with Python 3.6 version on it.

All systems: Windows, Ubuntu or MacOS

In your Anaconda prompt or terminal, write the following command
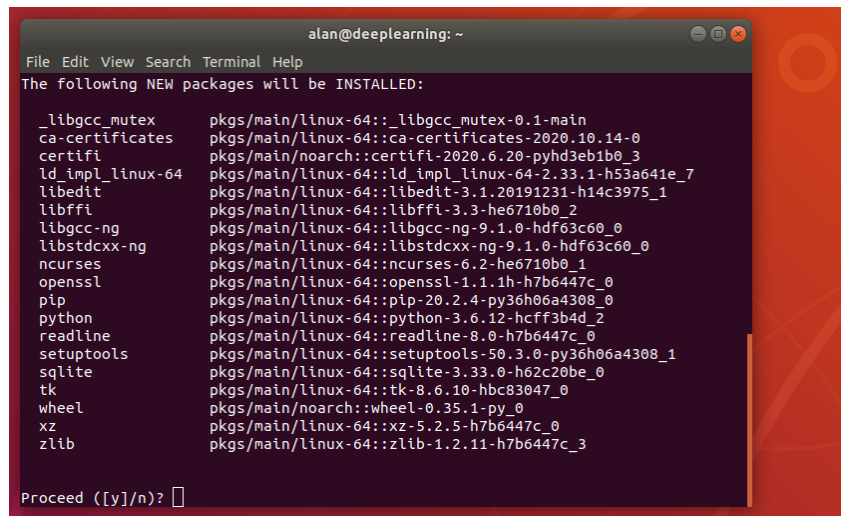`conda create -n keras python=3.6`

When the command requires you to proceed

type y and press Enter. This will take a few minutes. When conda completes the work, it will display a couple of messages as follows

Now you can check if the new environment was created successfully, by displaying a list of all your conda environments. Just write in your Anaconda prompt or terminal the command
`conda env list`

Now you will see the name `keras` (or whatever name you have chosen for your environment) below the `base` environment. Observe that the base environment has a star on it. This means that now we are working over the base (the default) environment of Anaconda.

3   Activate your new environment

We will need to activate our new created environment. To do this simply

Windows

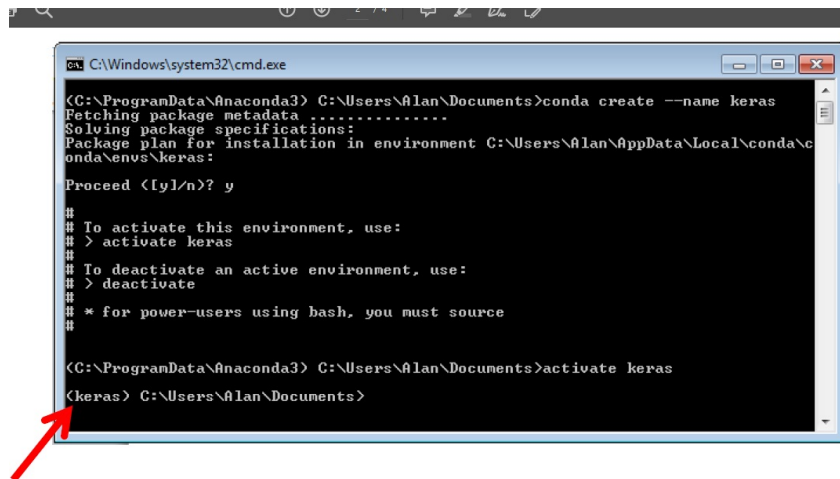write the following command in your Anaconda prompt
`activate keras`

or
`conda activate keras`
will also work.

Ubuntu and MacOS

write the following command in your terminal
`source activate keras`

(In the new versions of conda, usually they suggest to use instead the command)
`conda activate keras`
If it doesn't work, back to the previous alternative.

Now you will see that the root name in your prompt or terminal will change. For example, instead of (base) (or nothing) at the beginning of your terminal symbol, you must see the name (keras) (in parenthesis), as follows:

3.1 Deactivate your environment

If for some reason you want to deactivate your working environment and back to the base environment. Just

Windows

write the following command in your Anaconda prompt
`deactivate`

`conda deactivate`
will also work.

Ubuntu and MacOS

write the following command in your terminal
`conda deactivate`

To move back to your created environment, just re-write the corresponding command as is indicated in Step 3:
`conda activate keras`
In the following we will work inside our new environmnent (so be sure that the name (keras) is at the beginning of your terminal symbol), to be sure that you are working on yout new created env.

For more information on how to manage conda environments, visit the official conda documentation

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

4   Installing Tensorflow

Now we will install conda packages in our `keras` environment. We will install the Anaconda full set of packages. This will include a bunch of packages, some familiar to you (*e.g.* numpy, matplotlib, scipy, pandas, scikit-learn, the jupyter-lab environment, and so on...).

We will also install the Tensorflow 2.0, our main tool for working with neural networks. Also, we will install the Tensorflow-probability package (to work with bayesian neural networks).

Disclaimer! We will install Tensorflow 2.0 to work only with the CPU. To those of you that want to install Tensorflow to be available to work on GPU's, we don't do it here. Unfortunately, if you want to work with GPU's, there is a long path to work. For example, you need to install a lot of resources:

- Nvidia/AMD drivers compatible with your system, and GPU manufacturer, model and version,

- `docker` drivers also compatible,

- the `Cuda` and `CuDNN` packages compatible with your drivers,

- a `tensorflow-gpu` version compatible with all previous, ...
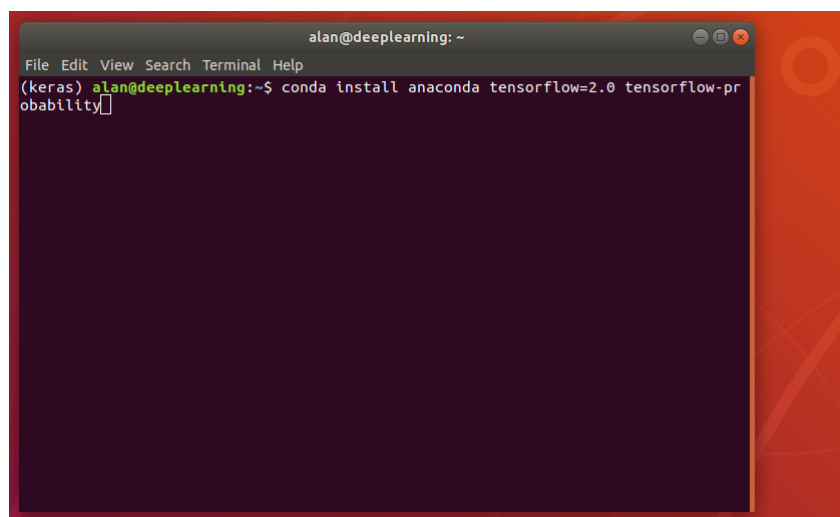
This is another chapter for another book.

4.1   Installing Anaconda and Tensorflow 2.0

Ok, let's go to the work. To install Anaconda packages and Tensorflow, do the following.

All systems: Windows, Ubuntu and MacOS

write the following command in your terminal
```
conda install anaconda tensorflow=2.0 tensorflow-probability
```


You will see a long list of packages to be installed (other will be updated, others downgraded or suppressed). When the command requires you to proceed type y and press Enter. This will take between 10 to 20 minutes or more, since conda is installing a lot of packages. (You can take a break or prepare some coffee if you want).

At the end of the process, conda will display some messages that the installation has been completed.

5 Jupyter-lab

We have installed Tensorflow 2.0. Now we will test our installation to be sure it works well. We will use the Jupyter-lab environment. To enter Jupyter-lab, just

All systems: Windows, Ubuntu and MacOS

write the following command in your terminal or Anaconda prompt
`jupyter-lab`

It will display a new tab in your default web browser. I suggest to setup Chrome as your default browser if you are working on Windows (Jupyter usually do not perform well on other browsers). In the case of Ubuntu, the Firefox is fine, and for MacOS users I think Safari or Opera will be fine (but I am not sure). You will see a window as follows:
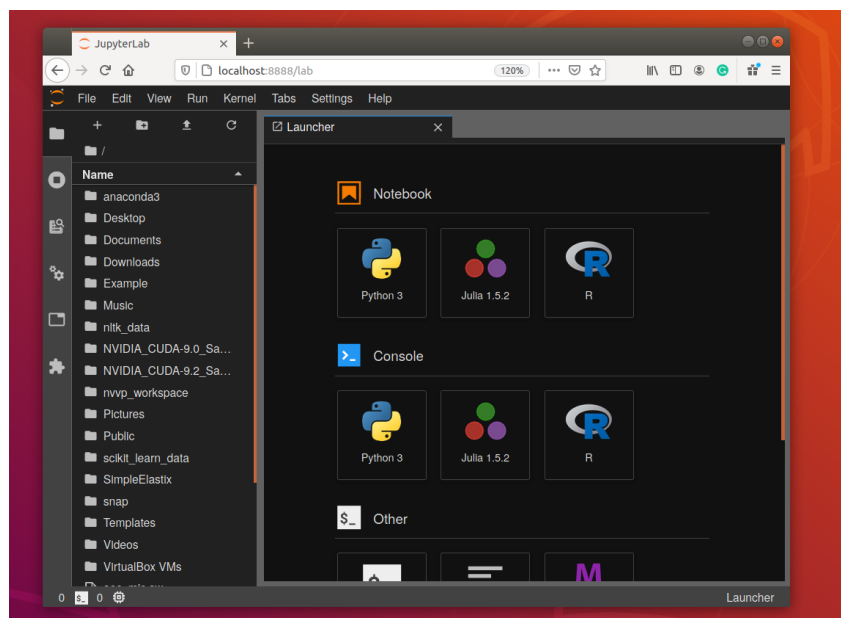
Your will probably see it in black font over white background (I have changed mine), and probably you will see only a Python 3 kernel box, which is the default for Jupyter (I have installed here other kernels).

6   Testing our installation

Now in Jupyter-lab, move to the directory path where the test files `test.ipynb` and `plotmatrix.py` are located (both must be in the same path). To do this, on the left panel of your Jupyter environment (see previous Figure), observe the folders on the directory structure on your computer. You can navigate along these folders, and move back using the folder icon or path just above the "Name" subtitle.

Open the `test.ipynb` notebook in Jupyter-lab, and in the menu Run, click on **Run all**. This will execute all cells of code contained in the notebook. In particular, here it downloads the Mnist dataset (it probably will take some seconds), then it creates a convolutional neural network that will classify the digits. The training part probably will take some minutes (3 to 5 minutes usually if you are on a laptop), and finally you are testing the model, and display some confusion matrices. You must see two matrices at the end (surely with different values):
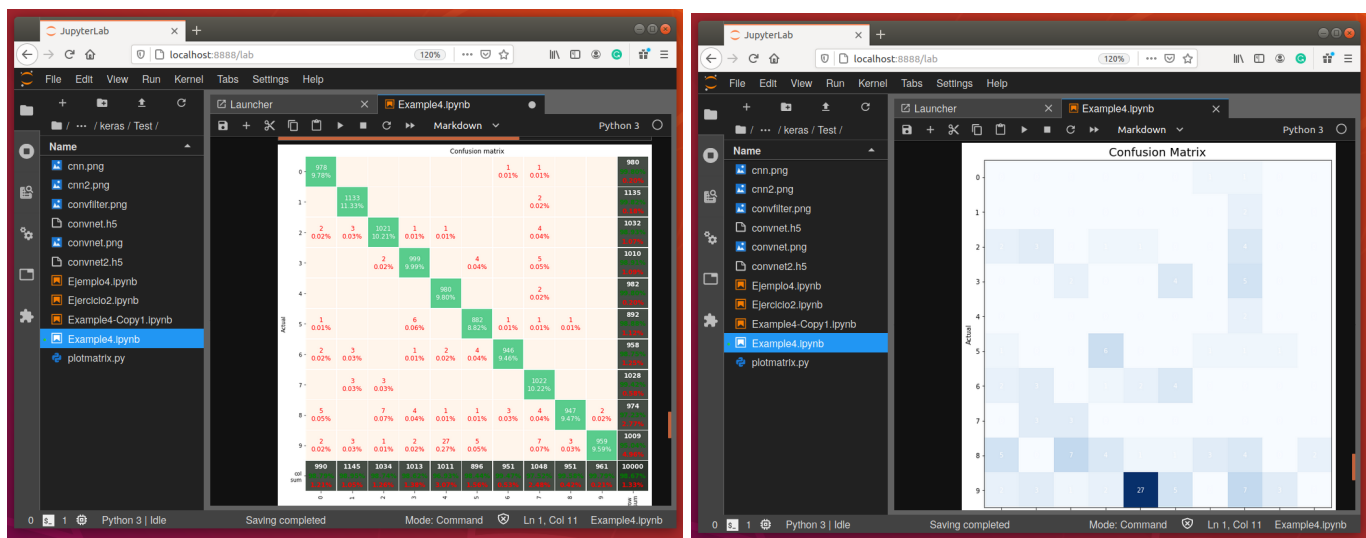
If this works with no error, our Tensorflow installation was successful.

Now you can close your test file (In the File menu, use **Close and Shutdown Notebook**), and exit Jupyter-lab (in the File menu, use **Shutdown**) and close the browser window. You will return to the terminal or command prompt. If you see that the command prompt process has not finished after some seconds, you can kill the process by pressing CTRL+C. Now you can close the terminal or prompt by writing `exit`.

7   Accessing the keras and Jupyter-lab environment again

If you want to return to work in the Jupyter-lab on your `keras` environment, just do the following:

1. Open your Anaconda prompt (Windows) or terminal (Ubuntu and MacOS).

2. write `activate keras` (Windows), `source activate keras` (Ubuntu) or `conda activate keras` (MacOS), as in Step 3.

3. write `jupyter-lab`, as in Step 5.

Now you can work.