**Running a recognition experiment**

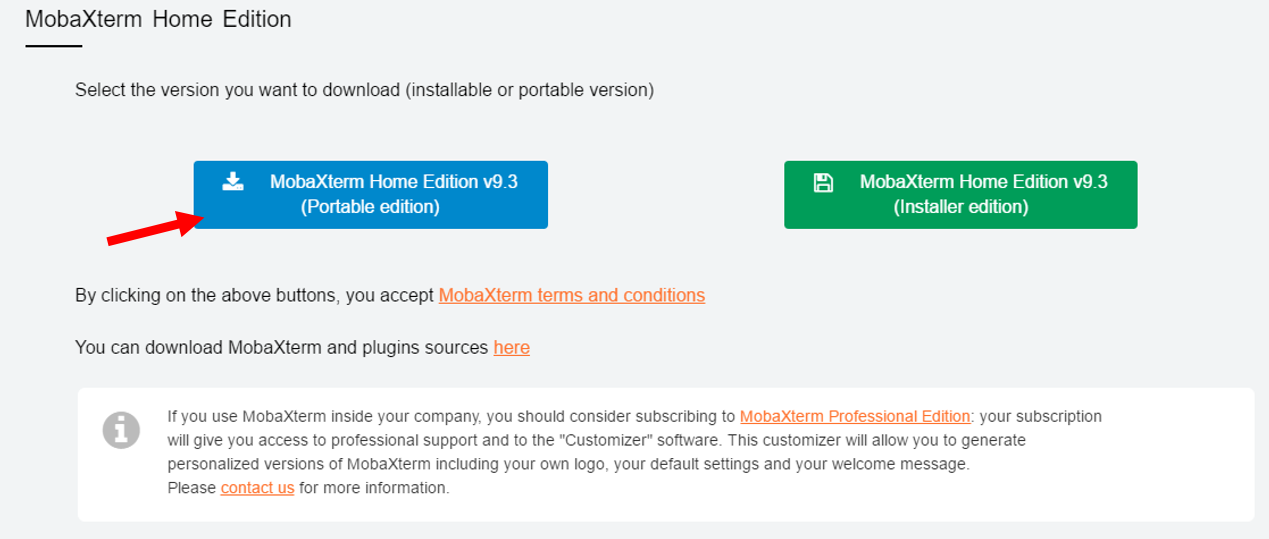
**Phase I: Software Installation and Remotely Connecting to the Cluster**

This phase will walk you through the steps for the installation and setup of the software tools that you will need in order to complete this section of the course.

One of the things that you will need to do in this part of the course is to login remotely to [Neuronix](https://www.isip.piconepress.com/projects/neuronix/html/neuronix_overview.shtml), a High Performance Computer (HPC) cluster. Once you log into this HPC, you will be able to run our demo code, write and execute your own scripts and complete the activities for the class. In order to successfully log into Neuronix, you need to have access to an [SSH client](https://en.wikipedia.org/wiki/Comparison_of_SSH_clients). The SSH client that we will be using is called MobaXterm. Mac users **do not** need to install the MobaXterm. The steps to install this program are presented as follows:

**Steps for the installation of MobaXterm (Windows Users):**

1. Follow this link: <http://mobaxterm.mobatek.net/download-home-edition.html>
2. Click on the Portable Edition, as indicated in , and select a location to save the files.

Figure 1. Selection of the MobaXterm version to install

1. Open the folder that you just downloaded and extract all the content by clicking “Extract All”, as shown in .
2. Open the recently extracted folder and double-click on MobaXterm\_Personal\_9.3. This will start the MobaXterm program and conclude the installation process.

Once the MobaXterm is installed, you have to follow some steps in order to setup your session and log into Neuronix. These are the steps:

**Steps to connect to Neuronix (Windows Users):**

1. Click on “Session”, as shown in .
2. Click on “SSH” and fill in your information. You must enter the following information:

Remote Host: neuronix.nedcdata.org

Username: Your TU username (e.g.: tuf30000)

1. Click on “Advanced SSH Settings” and uncheck the box for X11-forwarding. This is shown in .

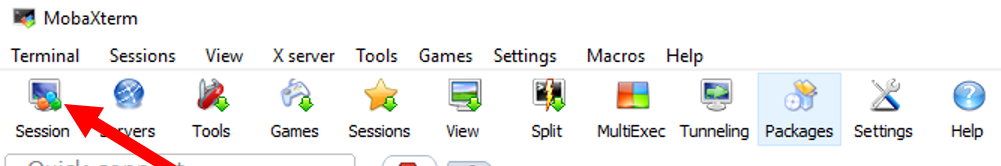


Figure 2. Editing session

1. Click “OK” and your Neuronix session should start.

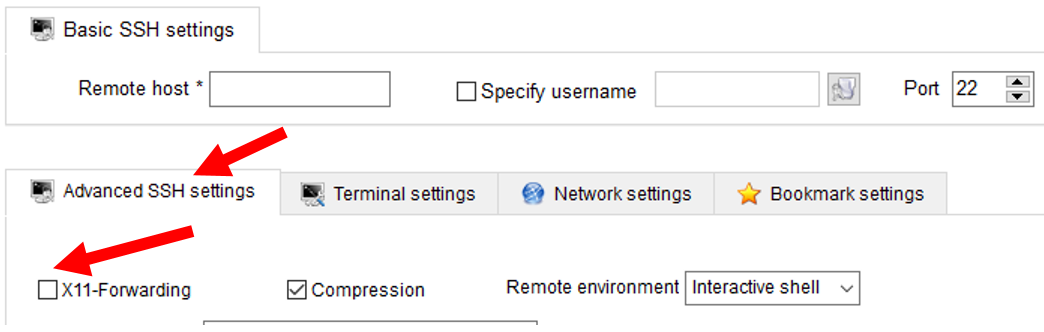


Figure 3. Uncheck the X11-Forwarding box

**Steps to Connect to Neuronix (Mac Users):**

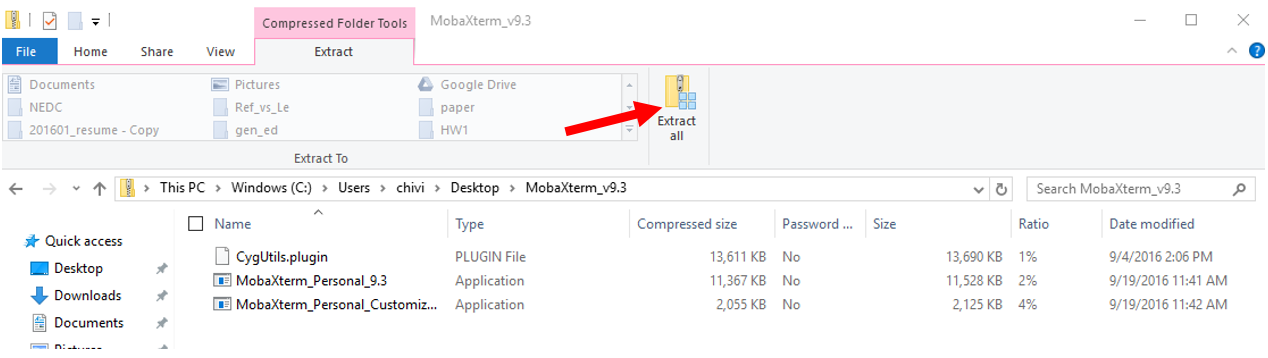


Figure 4. Extracting the contents

1. Click on *Finder.*
2. On the left side panel of the *Finder* window, click on *Applications.*
3. From the *Applications*, select *Utilities.*
4. Click on *Terminal.* The icon looks like the one shown in Figure 5.
5. Once the terminal opens, connect to Neuronix by entering the following command (see Figure 6):

ssh tuf?????@neuronix.nedcdata.org

1. Enter your password and your connection will be established

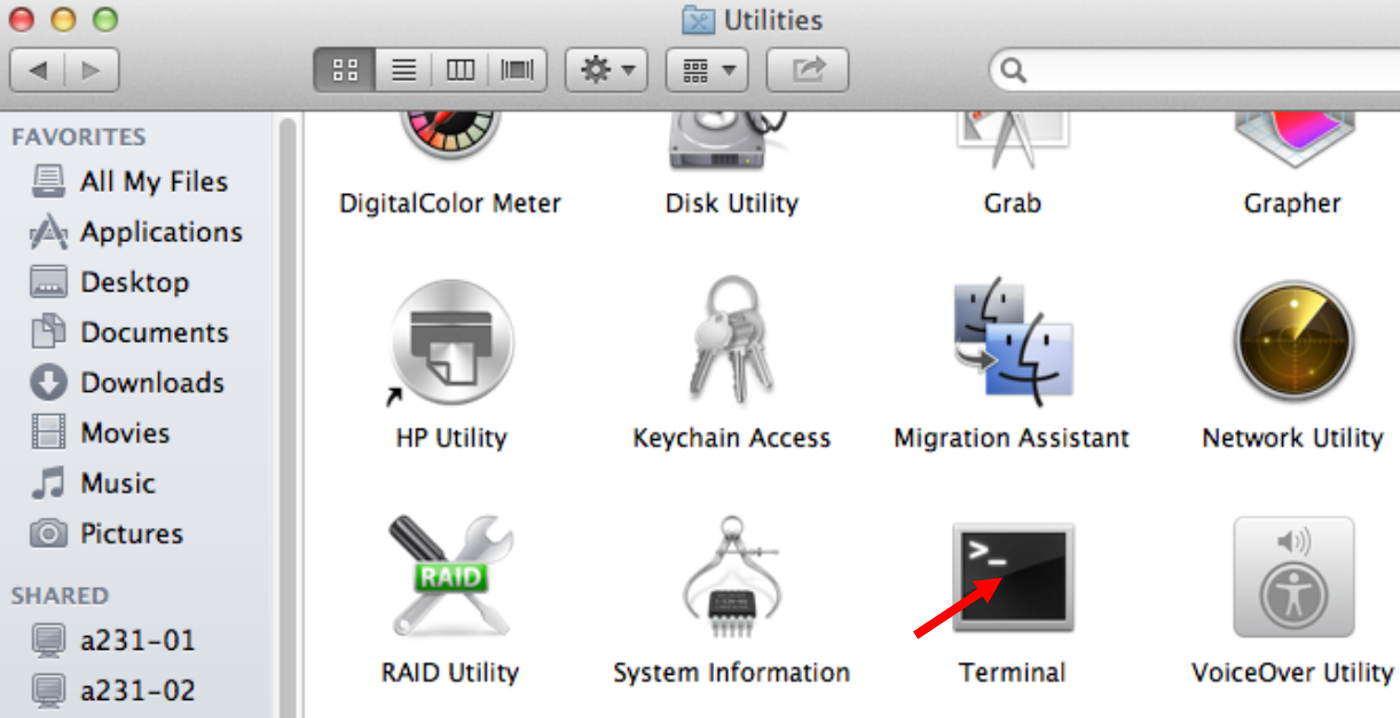


Figure 5. Open the Terminal

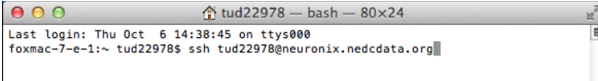


Figure 6. Enter your credentials

**Phase II: Computer Programming**

[Python](http://en.wikipedia.org/wiki/Python_(programming_language)) is a widely used general-purpose, high-level programming language. Its main feature is that it is a scripting language. This means you can type instructions as plain text, store them in a file, and run them directly using a Python interpreter. Though you sacrifice some efficiency, you gain much in terms of flexibility and reduced software development time. There is a wealth of information available on [how to learn Python](https://www.python.org). It is easy to use if you are familiar with basic programming concepts. In this phase of the lab, we will focus on learning the basics of Python.

The most important thing to remember is that any time you need to figure out how to do something in Python, just do a Google search. You will undoubtedly find that someone has written an entire web page devoted to solving your problem.

These are excellent resources to get familiar with Python:

* [*The Python Tutorial*](https://docs.python.org/2/tutorial/): walks you through the basics of Python programming;
* [*Python Tutorial for Beginners*](https://www.youtube.com/watch?v=29mq1Bn52GY): a short YouTube video demonstrating how to write a short program;
* *Learn Python*: a nice interactive web site that teaches you the basics.

Python is a popular choice for a language today because so many people are using it. There are millions of web pages devoted to its use and you can generally find a package that was developed for about any task. You can write many different types of code using Python. In this course, we will focus on running some Python deep learning and machine learning demos, which will allow you to classify a series of handwritten images.

**Activity 1: Hello World**

Write a simple program to display the text “Hello World” on your computer screen. This can be done by following these steps:

1. Open the emacs text editor by typing emacs and the name of the file you want to create:

emacs hello\_world.py

Make sure that you do not use blank spaces in the name of your file and that you add “.py” as the extension. This is the extension for a Python script.

1. Follow [these instructions](http://www.learnpython.org/en/Hello,_World!) to write your Hello World script.
2. Save the script that you just wrote by typing Ctrl-x Ctrl-s
3. Close the script by typing Ctrl-x Ctrl-c
4. To execute your script, just type “python” followed by the name of your script:

python hello\_world.py

Upon execution of the script, “Hello, World!” should be printed in your terminal.

This demonstrates how we run python from the command line.

**Phase III: Running the Demo Code**

The following steps show you how to use our code to create and implement a logistic regression classifier. In this case, we are going to automatically classify a series of handwritten digits that are part of the [MNIST database](http://yann.lecun.com/exdb/mnist/). Basically, we are going to build a mathematical model that will be able to recognize the digits in the handwritten photos. The following section explains the steps that must be followed and shows the commands that you need to use in order to complete each step (it is important that you entered the commands exactly as they are shown).

1. Create a directory (folder) for your experiment:

mkdir experiment

1. Go to the directory that you created:

cd experiment/

1. Copy the files that you need to your experiment:

cp -r /data/isip/exp/theano/exp\_0018/ .

1. Go to the scripts directory:

cd scripts/cpu\_scripts

This directory contains the scripts that you will need to run the code that will train our classifier.

1. Run the code that trains the classifier. There are two ways to run this code: (a) with its default options for the output of the reports and (b) with options that allow the users to decide where the output will be stored.
2. python mnist\_logistic.py

Running this script will start the training of the model. The output reports will be generated and stored in your current directory.

1. mkdir reports; python mnist\_logistic.py -t reports -v reports

This will do two things: create a directory called “reports”, and then train the model. The output reports are saved inside the “reports” directory.

There are two output reports for this. The first one is the test\_report.txt, which shows the performance of the model when it is evaluated on the test data. The second one is the validation\_report.txt, which is the performance of our model, but in the validation data. The reports show the error rate for each training epoch.

After you follow these steps, you will have a trained classifier that is ready to make predictions on some data. This model is stored in a file called best\_model.pkl for future usage.

**Phase IV: Running the Demo Code**

After you have trained a model, you can utilize it in order to make predictions or recognize patterns in new data. In this case, we are going to use the model in order to determine which digit was written in some of the test images for MNIST. These steps will help you with the utilization of your model:

1. Close your current terminal
2. Connect to the terminal setting the display variable:
   1. Windows Users:
      1. Follow steps 1 and 2 from the Connecting to Neuronix section.
      2. This time, check the X11-forwarding box that you can see in Figure 3.
      3. Continue the Neuronix login process as normal
   2. Mac users:
      1. Open a new terminal (See steps in the Connecting to Neuronix section)
      2. Enter the following command:

ssh -X tuf?????@neuronix.nedcdata.org

1. Go to the script directory for your experiment:

cd experiment/scripts/cpu\_scripts/

1. Run the display\_images.py code. This script uses the model that we train before in order to decode the number that was written in every image:

python display\_images.py

The code will generate a prediction for the first N images in the test set and output the recognized characters. You will also be able to see the original images, so you can observe what the model is predicting. Shows 5 correct predictions. If you look at more samples, you will see that the model makes a few prediction mistakes.

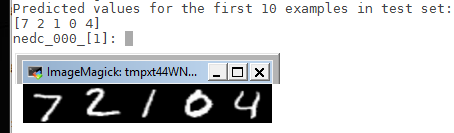


Figure 7. The model recognizes the first 5 digits as 7, 2, 1, 0, 4. The original images show that, in this case, there was no error

**Phase V: Creating your Own Testing Data**

This section will show you how you can create your own test data, upload it to Neuronix and use the trained model that you saved after running the logistic regression experiment in order to recognize your own handwriting.

1. Make a new directory for your experiment

mkdir experiment\_test

1. Change directory to the created experiment

cd experiment\_test

1. Copy the necessary information to your experiment

cp -r /data/isip/exp/theano/exp\_0023/ .

1. Create a set of test files. You just have to write a number from 0 to 9 in a piece of white paper (with a black pen or marker), take one picture of each individual number (just one number in the picture), and upload the files to the subdirectories located in Neuronix under:

/home/tu??????/experiment\_test/exp\_0023/images

You must place each of the images under their corresponding directories. For example, if the image contains a number 1, you must locate it under the subdirectory called “one”:

/home/tu??????/experiment\_test/exp\_0023/images/1

Figure 8 shows some example images.

1. After you upload the images, change directory to the scripts directory and run the logistic regression experiment to train the model once more:

cd ~/experiment/exp\_0023/scripts/

python mnist\_logistic.py

1. From the scripts directory, run the evaluation script, which will use the model that you trained to guess what number you have written.

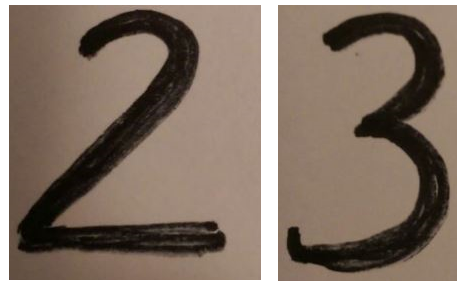


Figure 8. Sample test images

sh mnist\_eval.sh

1. When the evaluation script is done running, you can change directory to the directory that contains the output report.

cd ../output/reports/

1. Open the report called eval\_report.txt and you should be able to see the output of your classification, as shown in Figure 9.

emacs eval\_report.txt

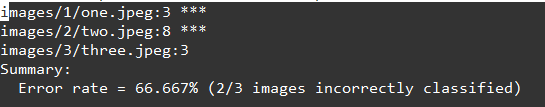


Figure 9. Sample output report

The output seen in Figure 9 shows that number 1 was incorrectly classified as a 3, number 2 was misclassified as 8 and number 3 was correctly classified. The error rate is then given by:

In this case it is: