Practical work 01 – 20th of septembre 2018
Let's get started

The main objective of this Practical Work (PW) is to set up your computer environment and to bring you on track for this class on **Machine Learning and Deep Learning**.

We are going to use Python as programming language to perform some of the PW. This weeks' PW will include setting up Python on your computer and helping you to get familiar with this language.

For Python, many good libraries are available to work with data, to visualise data and for performing machine learning tasks. That's why we have chosen to use Python in this class.

For this class, we will work with iPython Notebooks which is also a good format to hand in your practical works. You can create, edit and run these notebooks from a web browser and write Python code or text (using Markdown) in so-called cells and also create nice output (e.g. tables, plots).

Warning : There are two "concurrent" versions of Python, versions 2 and 3 with some minor incompatible differences in language syntax. As the Python world has now been migrating strongly towards Python 3, **we will use Python 3**.

Moodle Subscription

As mentioned in the lecture, you will hand in the solutions to assignments by using Moodle.

Register on moodle at https://moodle.msengineering.ch. To find the class navigate to Home → Zürich → Technical scientific specialization modules → S1 (HS) - 2018-2019 → TSM_DeLearn. Use subscription key : deep1819.

Hand-In of Practical Work

Most of the time, you'll need to submit your PW reports **by Wednesday, 10am of the following week**. However, we may override this rule. In any cases, the dates indicated in Moodle are the ones you should follow.

For this first PW, we expect you to **submit a solution for exercise 3** in form of a **iPython Notebook and a small report as pdf** (follow the instructions in the iPython Notebook).

Exercice 1   Python Installation on Your Computer

Skip this part if you are already set up with Python, Jupyter notebooks and your favorite IDE for Python. Otherwise, we recommend the following installations :

— Anaconda platform - a distribution of Python with popular data science packages that are pre-installed or easily installable. `https://www.continuum.io`. Select the 3.6 version of Python, as illustrated on Figure 1 below.
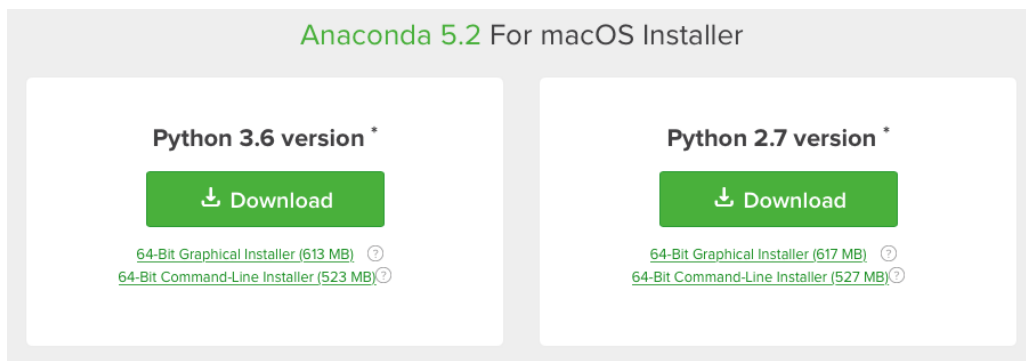
FIGURE 1 – Installing the Anaconda platform

— PyCharm IDE - an integrated development environment for Python. `https://www.jetbrains.com/pycharm`. You may select Professional or Community edition - the Community edition should be enough for this class. Free licenses are typically given to students, use your school email address for the registration by JetBrains.

Exercice 2   Python Language in a Nutshell

We assume here that students are knowledgeable in other programming languages such as Java or C and that basic data structure concepts are known. If you know already Python and the concept of notebooks, then you can skip this exercise.

— Open Jupyter from the Anaconda Navigator. Jupyter Python notebooks run in your browser so, after launching Jupyter from Anaconda, a browser should show up. Open a new notebook from menu File and follow the User Interface Tour as illustrated in Figure 2.
— Download the file `intro-python-3.ipynb` from moodle and open it from Jupyter in Anaconda. You need to navigate where you stored the ipynb file. See Figure 3 below.
— Go through the content of the `intro-python-3.ipynb` notebook and play with the cells. This document should give you a quick introduction to Python assuming that you are fluent with other programming languages. You may also want to get familiar with Markdown syntax if not known already.
— If you want a more fully fledged introduction to Python, read the official tutorial from `https://docs.python.org/2.7/tutorial/`.
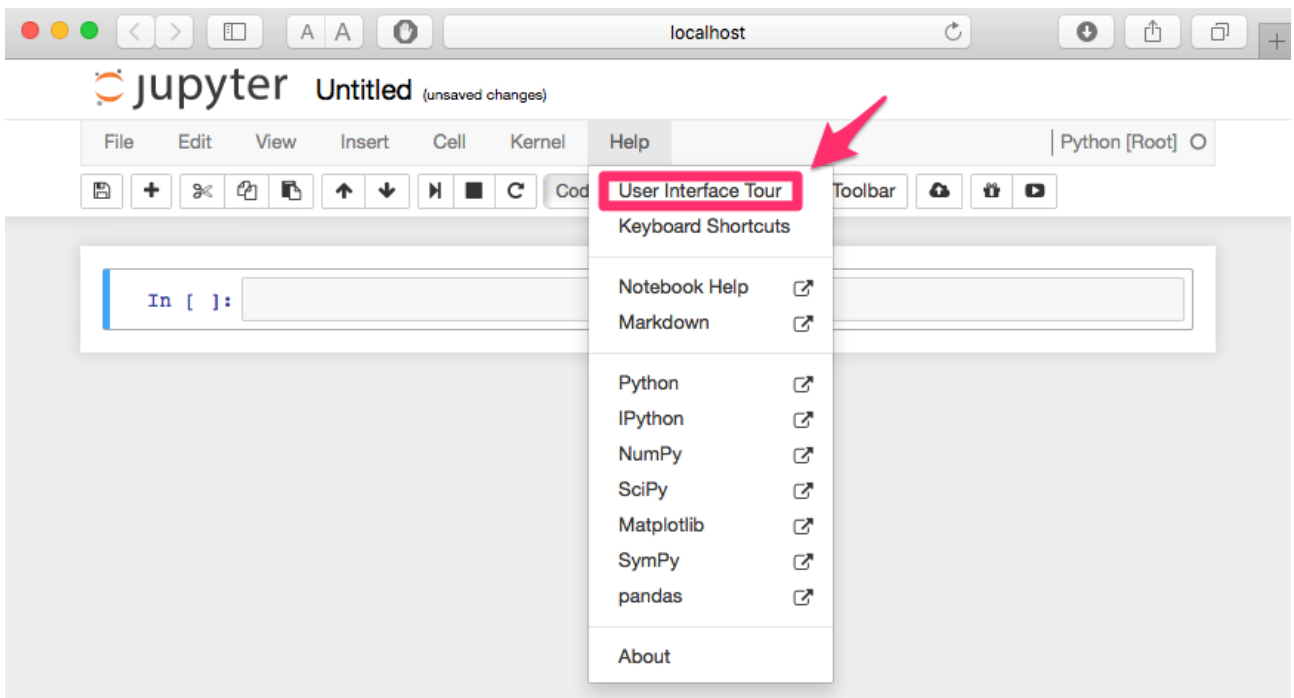
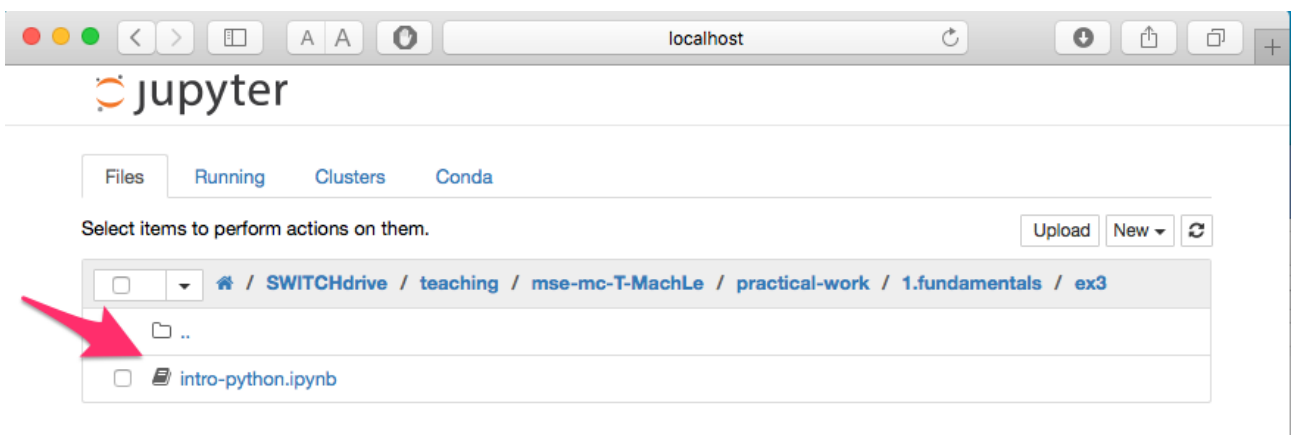FIGURE 2 – First steps with Python notebooks.

FIGURE 3 – Intro to Python language and Python notebooks.

Exercice 3 Perceptron Learning Algorithm

The goal of this exercise is to implement the perceptron learning algorithm and apply it to some example data. To be specific, you will program the learning algorithm supposed to find a decision boundary that separates between the red and the blue crosses (similar the green dashed line in the Figure below) - by starting from a first trivial guess (the magenta dashed line in the Figure below).
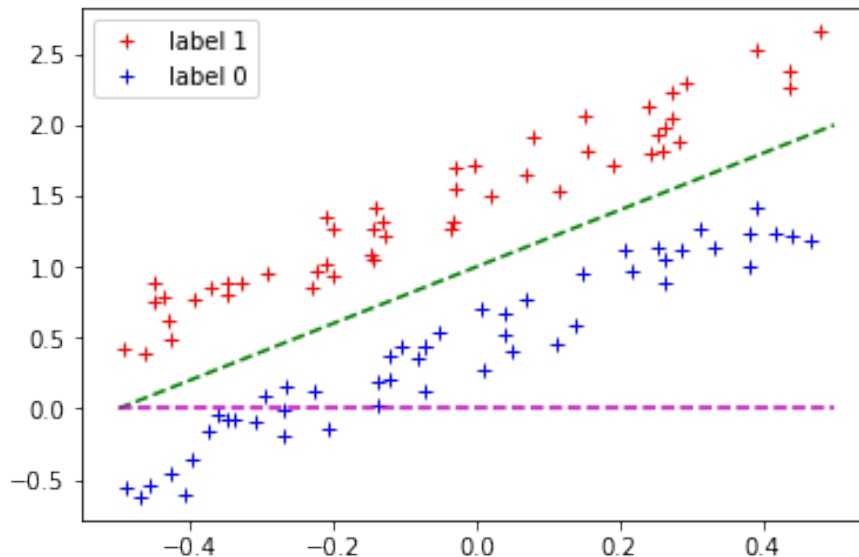

FIGURE 4 – The linearly separable example data used to train a classifier with the perceptron learning rule. The ideal decision boundary would be the green dashed line (the max margin classifier) - the magenta dashed line will be the first trivial guess which is bad since the number of mis-classifications is high.

— Open the notebook named `perceptron_learning_stud.ipynb` and inspect the functions defined therein. For better readability, you can use $help(function\_name)$ to inspect the documentation of what the functions should do.
— Some of the functions contain missing blocks of code that you should properly complete. These sections are marked as follows.

START YOUR CODE

END YOUR CODE

— If you manage to complete all the missing code properly the code in section *3/ Test Your Implementation* should find the correct (but not ideal) decision boundary in a finite number of steps. Prepare a small report (pdf) with the findings as described in `perceptron_learning_stud.ipynb`.

Exercice 4   Optional : Data Visualization

To train a bit yourself, we propose you to do a visualisation workout with the Iris dataset
https://en.wikipedia.org/wiki/Iris_flower_data_set. Download the file `iris.txt` from
moodle. Put it in a Python data structure and attempt to reproduce a plot close to the following
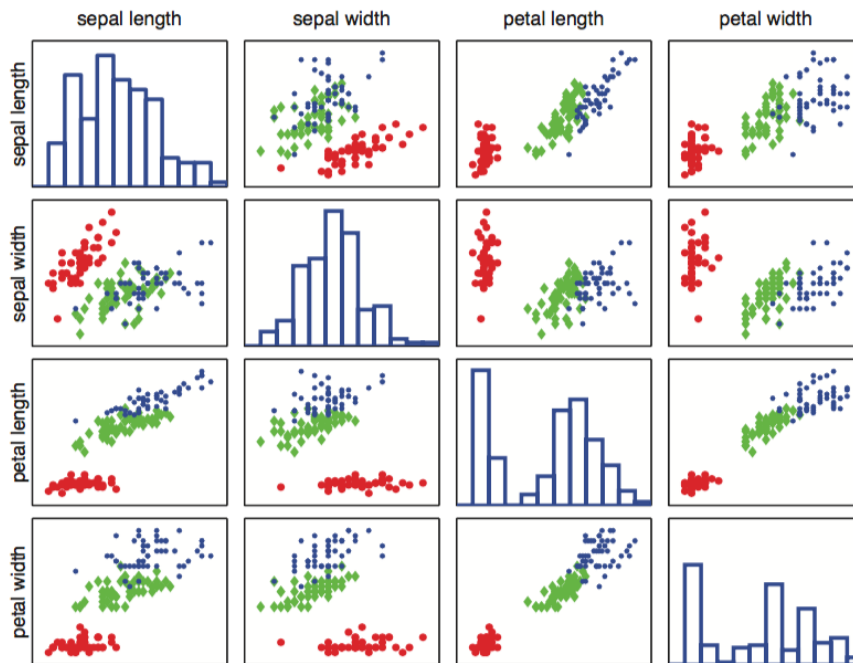one.

FIGURE 5 – Visualization of the Iris data as a pairwise scatter plot. The diagonal plots the
marginal histograms of the 4 features. The o diagonals contain scatterplots of all possible pairs
of features. Red circle = setosa, green diamond = versicolor, blue star = virginica.

The iris species classification task is a classical one. The goal is to distinguish three kinds of
iris : setosa, versicolor and virginica. In the data set, a botanist has extracted 4 *features* : sepal
length, sepal width, petal length and petal width.

What can you say regarding class separation ? One class seems easier to distinguish from the
others, which one ? How would you separate the other two ?

Exercice 5   Optional : Review Questions

1) Supervised vs. unsupervised systems

Of the following examples, which one would you address using a supervised or an unsupervised
learning algorithm ? Give some explanations for your answers.

a) Given email labeled as spam/not spam, learn a **spam filter**.

b) Given a set of news articles found on the web, group them into sets of **related articles**.

c) Given a database of customer data, automatically discover **market segments** and group customers into different market segments.

d) Given a dataset of patients diagnosed as either having **glaucoma** or not, learn to classify new patients as having glaucoma or not.

2) Classification vs. regression systems

Can we transform a regression problem into a classification problem? What would be the benefits of doing so?

3) Perceptron

— For what kind of problems are Perceptrons suited?
— For what kind of problems will the Perceptron Learning Algorithm converge?
— Give an example for which the Perceptron Learning Algorithm will not converge. Explain why not.

Exercice 6 Optional - Reading Assignments

— Read the first chapter of Murphy's book "Machine Learning".

Build your own summary of these chapters by doing a taxonomy of machine learning problems. You can find the pdfs on Moodle.