Practical work 02 – 25th of September 2018
Classification Systems - KNN

Summary for the organisation :

- Submit the solutions of the practical work before Monday 12h00 next week in Moodle.
- Preferred modality : archive with iPython notebook(s).
- Alternative modality : pdf report with annotated code and outputs.
- The file name must contain the number of the practical work, followed by the names of the team members by alphabetical order, for example `02_dupont_muller_smith.zip`.
- Put also the name of the team members in the body of the notebook (or report).
- Only one submission per team.

Exercice 1 `Numpy` tutorial

This exercise is to get you more familiar with `numpy`. Read the content of the ipython notebook `numpy-tutorial-stud.ipynb` that you will find on Moodle. Pay a special attention to the *broadcasting* section that allows to gain significant speedup when processing large numpy arrays. Regarding this, it is usually more efficient to use *broadcasting* instead of for loops in your code.

At the end of the tutorial, you have to complete some manipulations of images stored by arrays.

Exercice 2 Classification system with KNN - Student dataset

The objective of this exercise is to build classification systems to predict whether a student gets admitted into a university or not based on their results on two exams[1]. You have historical data from previous applicants that you can use as a training set. For each training example n, you have the applicant's scores on two exams $(x_{n,1}, x_{n,2})$ and the admissions decision y_n. Your task is to build a classification model that estimates an applicant's probability of admission based on the scores from those two exams.

1. Data source : Andrew Ng - Machine Learning class Stanford

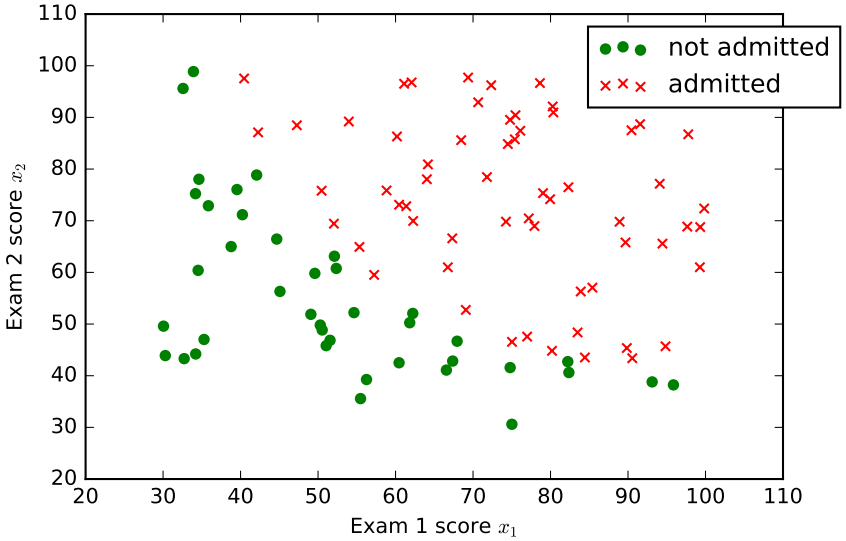

FIGURE 1 – Training data

a. Getting started

a) Read the training data from file `ex1-data-train.csv`. The first two columns are x_1 and x_2. The last column holds the class label y.

b) Plot the training data using a scatter plot. You should get something similar to what is displayed in Figure 1.

c) Build a *dummy* recognition system that takes decisions randomly.

d) Compute the performance $N_{correct}/N$ of this system on the test set `ex1-data-test.csv`, with N the number of test samples and $N_{correct}$ the number of correct decision in comparison to the ground truth.

b. KNN classifier

Build a k-nn classifier on the data using an Euclidian distance computation and a simple majority voting criterion, i.e. decide C_0 when there is a majority of points in class 0 in the k nearest neighbours. Compute the performance of the system as a function of $k = 1 \dots 7$. What value of k gives you the best performances ? Comment your result.

Remark : How is your system taking decisions when you have an equal number of votes for both classes with values of $k = 2, 4, 6$?

Exercice 3 Classification system with knn - MNIST dataset

It is now time to move to larger datasets and more intensive tasks. We will use the MNIST database that contains images of handwritten digits. This page offers a description of the dataset : http://yann.lecun.com/exdb/mnist/. It has a training set of 60,000 examples, and a test set of 10,000 examples. It is actually a subset of a larger set available from NIST [2]. In MNIST, the digits have been size-normalized and centered in a fixed-size image, as depicted in Fig 2.

a) Download the dataset `mnist.zip` from Moodle and expand the archive.

b) Download the notebook file `knn-mnist-stud.ipynb` from Moodle.

c) Follow the steps explained in the notebook.

You need to hand in the modified Python notebook with inline answers to questions and code completed wherever there is a `TODO` indication.

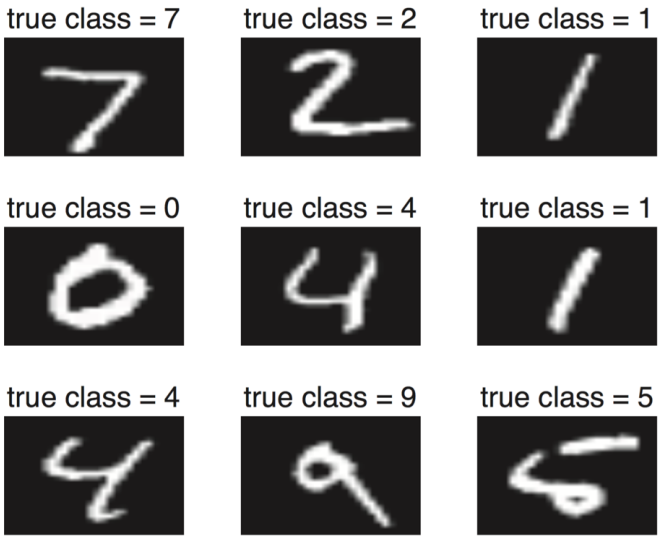

FIGURE 2 – MNIST dataset examples.

2. National Institute for Standards and Technology - USA