L3 MIASHS - Computer Science S6 - March31, 2022

General Instructions

- You will recover the data from the EPI (letter.data.zip file). If your system does not support the zip format, you can download Keka.
- You will hand in your program on the EPI (Not by mail!)
- The answers to the different questions will be clearly marked
- The different functions should not cause any error. Otherwise, I will not make any kind of error correction: you will be noted on a simple glance at your program. I therefore advise you to be clear in the design of these.

The objective of this exersise is to program a classifier to recognize handwritten letters.

Data

The data are in the file **letter.data**. The first part concerns the recovery of the data (reading the file), the formatting of the data (what is the input, what is the output, how to represent them), the visualization.

- Write a function named **read_file(filename)** that reads the file whose name is specified, and returns it in a format that can be used in python (e.g. a line can be represented by a list, andthe file can be represented by a list of lists).
  You can use **with open() as f**.
- Write a function **data_entry_output()** that takes as input the file in the format you chose in the previous question, and returns two matrices, one of the X inputs and one of the Y outputs, so that you can use these formats directly in the logistic regression program you wrotelast week.
- Write a function **visualization(X,Y,i)** which allows to visualize the image of index i, as well as its label. Check that you have interpreted the different fields of the data file correctly.
- Write a function **split(X,Y,p,q)** that splits the data into a part **X_l, Y_l** that will be used for the learning task and a part **X_t**, **Y_t** that will be used to test the solution. An element of the sample (a line) will be sent with probability **p** to the learning sample, and probability **q** to the test sample.

Learning

- Adapt last week's program to write a gradient descent leading to a classifier on the training data (we will use a split for p = 0.7, q = 0.3).
- Write a function to count the number of errors made by the learned classifier on a data set (training or test)

Curves

- Write a program that computes the number of errors made by the classifier on a dataset, as a function of the number of iterations of the gradient descent. (e.g. 10, 20, 30, 40, ..., 1000). We must reach a point where the number of errors has reached its minimum.
- Plot the learning curve (i.e. the number of errors versus the number of iterations) for the training data and the test data (on the same graph, of course)