Linear classification

Rémi Flamary

During the practical session you will need to use the Python `numpy` `pylab` and `scipy` toolboxes. It is recommended that you import them at the beginning of all your scripts with the following code :

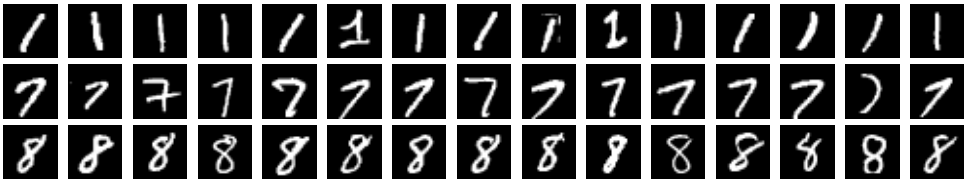
```
import numpy as np
import pylab as pl
import scipy as sp
```

You will then be able to access most of their functions using their short name followed by a dot (for instance `np.zeros(10)` for the function `zeros` of numpy). In the following, for each question, the list of necessary `numpy/pylab` functions is given between parenthesis .

1 Data loading and pre-processing

- Download data file "digits.npz".
- Load the file in memory using function `np.load`. The file contains the following matrices:
  - **x** and **xt**: data matrix containing respectively $n = 3000$ and $nt = 1500$ trining example of manuscript digits. Each line in those matrices is a 28×28 image stored as a transposed vector (a line).

    Some examples of the images in the training sets:

  - **y** and **yt**: the labels of the images described above. they are vectors containing the classes $(1, 7, 8)$ of each images in **x** and **xt**.
- Use function `np.reshape` to reconstruct 28×28 images. Visualize a few examples from each class using function `pl.imshow`.
- Normalize the data and use the normalize data in the following (`np.mean,np.std`). Be careful not to introduce `NaN` values.
- We want to create a binary classification problem from those 3 classes. You can first try and discriminante class 8 VS 1 and 7. Compute a vector `yb` containing labels $(-1, 1)$ for training and `ytb` for testing (operator `==`) .

2 Binary Linear Discriminant Analysis

- Estimate probabilities p_+ and p_- of each class from the training data.
- Estimate means μ_- et μ_+ of each class from the training data. Visualize them as images and interpret them.
- Estimate the covariance matrix Σ (function `np.cov`) as the average of the covariance matrix of each class.
- Compute the parameters $\mathbf{w}$ and b for the LDA. What happens when the LDA is not regularized ($\lambda = 0$)?
- Compute the accuracy of the classifier on the training and test data (`np.mean,==`).

3 Logisctic regression

- Code the gradient descent algorithm for the logistic regression.
- Compute the cost along the iteration and plot it to check convergence (`pl.plot`).
- Update the code to implement the regularized logistic regression with ridge regularization.
- What is the impact of each parameter of the gradient descent (step, regularization, number of iterations)
- Compute the accuracy of the classifier on the training and test data (`np.mean,==`).
- Visualize some of the missclassified examples as images, conclusions (`np.reshape,pl.imshow`).
- Visualize the classifier $\mathbf{w}$ as an image and interpret it (`np.reshape,pl.imshow`).

4 Multiclass problem

- We will use the "One-Against-All" strategy to train a multiclass classifier.
- To this end you need to estimate one binary classifier per class.
- Compute the prediction score for each class and each examples (`np.dot`).
- Final prediction consists in choosing for each example the class with the highest score (`np.argmax`).
- Compute the accuracy on the training and testing sets.