CSE 455/555 Spring 2010 Homework 2: Parametric Learning and Dimensionality

Jason J. Corso
Computer Science and Engineering
SUNY at Buffalo
jcorso@buffalo.edu
Date Assigned 9 Feb 2011
Date Due 4 March 2011
Homework must be submitted in class. No late work will be accepted.

Remember, you are permitted to discuss this assignment with other students in the class (and not in the class), but you must write up your own work from scratch.
*I am sure the answers to some or all of these questions can be found on the internet. Copying from **any** another source is indeed cheating.*
This class has a zero tolerance policy toward cheaters and cheating. Don't do it.

This homework contains both written and computer questions. You must turn in the written questions (which may be parts of the computer questions) in class and you must submit the computer code via the CSE submit script. For the computer parts, on this homework in particular, it is highly recommended that you use Matlab (available on all University machines) or Python/NumPy/SciPy. However, you are free to choose your poison (C/C++ or Java). If you do so, I recommend you acquaint yourself with the CLAPACK (C Linear Algebra Package) or JAMA (Java Numerics `http://math.nist.gov/javanumerics/jama/doc`*).*

Problem 1: MLE Learning (20%)

Suppose we have a set of independent samples $\mathcal{D} = \{x_1, x_2, \ldots, x_N\}$ which are labeled in c classes $\omega = 1, 2, \ldots, c$. Each class ω_i has n_i samples with $n_1 + n_2 + \cdots + n_c = N$. Our objective is to learn the prior model $p(\omega)$ from $\mathcal{D}$. Suppose we represent the prior model $p(\omega)$ by c parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_c)$ with the constrains that $\theta_i \geq 0$ and $\sum \theta_i = 1$.

1. Formulate the log-likelihood function $l(\boldsymbol{\theta})$.
2. Show that the ML-estimator that maximizes the log-likelihood above under the constraint that $\sum \theta_i = 1$ is

$$\theta_i = \frac{n_i}{n}, \quad i = 1, 2, \ldots, c \tag{1}$$

Problem 2: Multivariate Gaussian MLE (15%)

Derive the equations for the maximum likelihood solution to the mean and covariance matrix of a multivariate Normal distribution.

Problem 3: Dimensionality (20%)

As we discussed in the class nearest neighbor algorithms are very intuitive for data in low-dimensions. They assume we should be able to find a good number of examples close to a given point x and use the examples to estimate the class of x (these are its neighbors). However, in high-dimensions, this assumption breaks down; this makes learning hard and is an instance of the curse of dimensionality. Consider inputs distributed in a d-dimensional unit hypercube $[0, 1]^d$. Suppose we define a hypercubical neighborhood around a point x to capture a fraction r of the unit volume. The point and the hypercubical neighborhood are assumed to be fully inside of the unit hypercube without loss of generality. Denote the length of each side of the hypercubic neighborhood as l.

1. Derive l for $d = 1$ and $d = 2$.
2. Show that $l = r^{1/d}$. What is l with $d = 100$ for $r = 0.1$? In other words, to capture 10 percent of the data in the 100-dimensional space, what is that length of a side of the hypercubic neighborhood?
3. For $d = 10100$, how big on average do you need to make l in order to capture 1% of the data? How big for 10%?

Problem 4: Component Analysis (45%) *Now, let's get our hands dirty!*

As we saw in the case of faces, principal component analysis provides a way of creating an optimal low-dimensional representation of a dataset. Now, let's do such a PCA analysis on handwritten digits.

Download the dataset from the website:
`http://www.cse.buffalo.edu/~jcorso/t/555pdf/homework2-data.tar.gz`. The datafiles are also available on the CSE network at `/projects/jcorso/CSE555/homework2-data`. This is a subset of the LeCun's MNIST dataset containing just the digits 0,1, and 2. The full dataset is available at `http://yann.lecun.com/exdb/mnist/` or in a more convenient Matlab format `http://www.cs.toronto.edu/~roweis/data/mnist_all.mat`. The dataset is split into training and testing pictures. Do all PCA analysis on the training pictures, and reserve the testing pictures until the last part (nearest neighbor classification).

The images are all in the PGM format (Matlab can read these directly). The format is simple:

```
// Line1 P5
// Line2 WIDTH HEIGHT
// Line3 255
// Line4 ROW-WISE-BYTE-CHUNK-OF-PIXELS
```

1. Write a function to perform PCA on a group of images. This will require you to vectorize the images (i.e., do not do IMPCA). Input the number of dimensions k you want to estimate and output the set of eigenvectors and their corresponding eigenvalues (for the largest k).
2. Use the PCA function from question 1 to compute the Digit-0-Space. Plot the mean image and then the first 20 eigenvectors (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Describe what you find in both plots.
3. Use the PCA function from question 1 to compute the Digit-2-Space. Plot the mean image and then the first 20 eigenvectors (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Describe what you find in both plots.
4. Use the PCA function from question 1 to compute the Digit-Space. Plot the mean image and then the first 20 eigenvectors (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Compare and constrast what you find in this plots to the ones you created in questions 2 and 3.
5. Implement a nearest-neighbor (NN) classifier to input a training dataset, compute the PCA space, and then take a query image and assign it the class of its nearest neighbor in the PCA space.
6. Use the NN classifier to classify the testing images? Prepare a figure that shows 5 correctly classified images of each class and 5 incorrectly classified images of each class. Prepare a table giving the quantitative results over all of the testing data. Explain your findings.

How and what to submit: explains, figures and tables should be part of the "written" assignment due in class. The code (whether it is Matlab, Python, C or Java) should be tar'd and submitted via the CSE submit script:

1. Login to a department student Unix machine, hadar, metallica, nickelback, pollux, styx, timberlake (the code MUST work on the CSE machines).
2. Use "tar -cvf homework2.tar list-of-files-or-directories".
3. Then, type "submit_cse555 homework2.tar" if you are in 555 or "submit_cse455 homework2.tar" if you are in 455.

Include a README file to explain what is there, but only submit the code. If it is C/C++ code, you need to include a makefile that will compile on the CSE machines (it is thus encouraged to link to any libraries available there rather than submitting them). If it is Java code, you need to include an ant file that will compile on the CSE machines. In all cases, you need to include a script named "go" that will (once compiled) do everything requested in the assignment (in the case of C or Java, it is sufficient to generate image files of the plots and save them to reasonably named files). The script can be bash/tcsh for C/Java and should be a simple Matlab/Python script for Matlab/Python.