# Linear Classification Homework 1

CIS 620

January 23, 2005

This homework is due 11pm, Sunday, January 30. It should be submitted by email to me (`pereira@cis.upenn.edu`) as a `tar` or `zip` file. The homework relies on data files in this site containing feature vectors and labels for three tasks. They are provided in two formats as follows, where $n$ is the number of instances and $d$ the number of features (dimension of instance vector):

- space-separated rows of numbers. Each row represents an instance, with the first element of the row the label ($-1$ or $+1$) and the rest of the elements the feature values.
- MATLAB MAT files with two matrices: `x` ($d \times n$) of instances, and `y` ($1 \times n$) of labels.

1. Synthetic data `synthtrain.dat`, `synthtest.dat`, `synthtrain.mat`, `synthtest.mat`.
2. Handwritten digit classification `digitstrain.dat`, `digitstest.dat`, `digitstrain.mat`, `digitstest.mat`.
3. Text relation extraction `relntrain.dat`, `relntest.dat`, `relntrain.mat`, `renltest.mat`.

In all of the experimental problems that follow, you are asked to implement and use an online learning algorithm as a batch algorithm. What that means is that you run the learning algorithm over the training data for several iterations, each time starting with the weight vector produced by the previous iteration. At the end of each iteration, you evaluate the learned classifier on the test data and report the error.

Since the optimal separator for these data sets may not run through the origin, you may be able to get better results by including a constant feature as discussed in class.

1. Implement the perceptron algorithm (this is easy to do in MATLAB using the built-in matrix operations) and run it on the three datasets. **Deliverables**: code, and the test error results for five iterations for the three datasets.

2. Implement the dual perceptron algorithm and run it on the three datasets with polynomial kernels of orders 2, 3, and 5. In the dual form or the algorithm presented in class, for the online case, the update sets $\alpha_i = 1$ if the current classifier misclassifies instance $i$. It is easy to see that in the batch case we should use the update $\alpha_i \leftarrow \alpha_i + 1$. **Deliverables**: code, and the test error results for five iterations for the three datasets with the three different kernels.

3. The perceptron update does not guarantee that the instance being considered achieves a reasonable margin after the update. Determine what update for the weight $\alpha_i$ of an instance would ensure that if instance $i$ has margin $< 1$ before the update, it achieves margin 1 after the update. Of course, the margin of instance $i$ might change later as other instances are considered. **Deliverable**: your update and a brief explanation.

4. Implement the variant above and do the same tests as in question 2. **Deliverables**: same as for question 2.