6.891 Fall 1999: Problem Set #4

Due: Thursday, 11/4/99
***** Final Version *****

Problem 1: Preparing for the final project.

For the final project of 6.891 you should pick out a machine learning problem and attempt to solve it using one (or more) of the approaches you have seen in class. I expect you to do roughly 20 hours of work on the project... no more.

You can proceed in one of several ways:

- Look around for recent papers on machine learning that interest you. Study the approaches used. Acquire the data (or some similar data). Try out your own variants of the algorithms.
- Take a problem from your work or life and try to solve it using machine learning. You might try to predict whether the email you just received is junk or important. Or you could try to predict stock prices. There are thousands of problems out there.
- You might have a problem from your research (more likely if you are a Ph.D student) that might be solved using machine learning.

In any case you should spend some time *now* thinking about the data you might use. Let us know if you need any help or advice on this project.

Problem 2: Radial Basis Functions

Consider a data set $\{X, T\}$ which we have fit using a kernel function network with Gaussian basis functions of the form

$$G_i(x) = e^{-\frac{1}{2}(x-\mu_i)^T \Sigma^{-1}(x-\mu_i)} \tag{1}$$

(Note: these functions are not radially symmetric.)

Find the linear transformation of the input vectors X such that the new transformed problem, $\{\tilde{X}, T\}$ can be solved by a Radial Basis Function (RBF) network with the same number of basis functions each of which now have the form

$$G_i(x) = e^{-\frac{1}{2}(x-\tilde{\mu}_i)^T(x-\tilde{\mu}_i)} \tag{2}$$

and for which this new RBF gives the same results as the first RBF.

Write down the linear transformation relating $\tilde{X}$ to X, and a formula relating $\tilde{\mu}_i$ to μ_i.

Problem 3: Pixel Features

Consider using a dataset of images. Each component of the input vectors corresponds to one pixel of the image. Now consider using a feature space of polynomial kernels in your favorite machine learning algorithm. If the image is binary what do these features correspond to in the image? Please describe what these features look like.

If your machine learning algorithm returns a function which is a thresholded sum (non-weighted) of features, what is it computing?

Problem 4: Let's start reading some papers.

Read the paper titled "Simple Learning Algorithms for Training Support Vector Machines" by Campbell and Cristianini. It is available from
`http://www.ai.mit.edu/courses/6.893/related-info.html`.

Implement their learning algorithm and try it out on the some of the data sets from the earlier psets. Turn in the algorithm operating on the artificial data set.

The adatron algorithm (as it stands) does not allow for multiple output classes. Plot classifiers for each possible dichotomy: 1 v.s. 2&3 , 2 v.s. 1&3 , and 3 v.s. 2&1.

A: Propose an approach for combining the results of these classifiers to build a multi-way classifier. Implement this technique and show your results on the three-way classification.

B: How might you modify the Adatron algorithm so that it could learn a mult-way classifier directly? (Optional: try implementing the approach.)

In order to cement your understanding of Support Vectors, you should take a look at "A Tutorial on Support Vector Machines for Patter Recognition" by Burges. Is also available on
`http://www.ai.mit.edu/courses/6.893/related-info.html`.

Problem 5: Try out the Support Vector Code

We will be supplying two implementation of SVM code. One works in MATLAB and has a nice user interface. The other works in C and is much more efficient (though there are other more efficient implementation as well).

Try out both these implementation on the data sets... be warned, that the MATLAB code is very unlikely to work on the two larger datasets.

We have transformed `artificial.mat, pen.mat, and mystery.mat` into a format appropriate for the support vector machine matlab code. These files are located in the data files link on the problem set web page. The files include the same input, target, and test matrices except now target is a column of vector where each entry is -1 or 1. So for the three classes in artificial.mat the data has been split in class 1 vs classes 2 and 3, class 2 vs classes 1 and 3, etc. The same has been done to the other data sets. The files are:

```
artificial_1v23.mat
artificial_2v13.mat
artificial_3v12.mat
pen_1vrest.mat
pen_2vrest.mat
pen_3vrest.mat
mystery.mat
```

Try to build up your intuition about this learning algorithm.

Problem 6: Hacking RBF's

For this problem, you are going to try fitting RBF's to the data in `ps4a.mat` using a number of techniques for picking the centers and for a couple of different basis functions. For each method, you will train the network on the matrices 'trainx' and 'trainy' (the input and output for training respectively) and then report the squared error when you run this trained network on the matrices 'testx' and 'testy' (the input and output for testing respectively).

Run the experiments listed below using the basis function listed and the method of picking centers listed. Write a brief paragraph describing the difference you notice amoung the experiments. Plot the estimated functions using plotim and report the testing error for each method.

Note that we have supplied a number of functions which might prove helpful: plotfn (plots a function as a surface), plotim (plots a function as a grayscale image), and kmeans (performs K-means clustering).

Note as well that solving the first two trials with all of the data points as centers could take a while on the full data set (it took me about 15 minutes to do the pseudo-inverse necessary). The others shouldn't take as long.

- use **every training data point** as a center and **Gaussian** basis functions with $\sigma^2 = 10$
- use **every training data point** as a center and the **absolute value** as the basis functions
- **randomly pick 50 data points** as centers and **Gaussian** basis functions with $\sigma^2 = 10$
- **randomly pick 50 data points** as centers and the **absolute value** as the basis functions
- **randomly pick 200 data points** as centers and **Gaussian** basis functions with $\sigma^2 = 10$
- **randomly pick 200 data points** as centers and the **absolute value** as the basis functions
- use **K-means to pick 50 centers** and **Gaussian** basis functions with $\sigma^2 = 10$
- use **K-means to pick 50 centers** and the **absolute value** as the basis functions
- use **K-means to pick 200 centers** and **Gaussian** basis functions with $\sigma^2 = 10$
- use **K-means to pick 200 centers** and the **absolute value** as the basis functions

A: Optional: By the time you are finished with the problem you will have noticed that we have asked you to approximate an image using a few samples. This is an important problem from image compression. Where one may wish to compress an image by sending only a few of the pixels and reconstructing the rest. In this case it is very important for the compressor to send the best possible samples to the decompressor (rather than sending a random subset). Can you propose a better technique for picking a minimal number of samples than any listed above?