Logistic Regression

In this homework you will implement logistic regression, and then add a term to the objective function to prevent overfitting and evaluate it's impact.

The MNIST dataset is a well-studied collection of handwritten digits. It is often used to test multi-class classification algorithms, where there is one class for each of the 10 digits (0 - 9). I've made two files available for you:

- The raw MNIST data, which is a text file containing 10,000 rows. Each row contains 28 * 28 = 784 integers in the range 0 to 255. Each integer is the pixel value from a 28 x 28 image of a handwritten digit. Every row corresponds to a vector in the dataset that is to be clustered.
- The labels for the raw data are in a file with 10,000 rows. The first row contains the correct digit label for the first row in the raw data. The second row is the label for the second instance, and so on.

Implement 2-class logistic regression using gradient descent as outlined in the lecture notes. You can do either batch or stochastic versions of the algorithm. You will only use your algorithm for this dataset, so you can hard-wire in the number of instances and the size of each instance. The goal is not to write a generic version of the algorithm (though you can if you wish). The goal is to understand how it works on real data.

Use your algorithm to learn a classifier that determines whether an input image is an 8 or one of the other digits and record the classification accuracy on the training set (the full dataset I provided). Note that you'll have to come up with some stopping criterion, which could be to simply run for a fixed number of iterations and then quit.

After training is complete, create a 28x28 image of the learned weights. The largest weight (most positive) should map to black, the smallest weight (most negative) to white, and the other weights should linearly interpolate between those extremes.

Recall from class that one form of regularization (a way to prevent overfitting) is to encourage the weight vector to be sparse. That is commonly done in logistic regression by adding a term to the objective function that is the sum of the squares of the individual weights. We'd like to minimize that while maximizing the likelihood of the data.

Modify your code from above to include this term. Note that you'll have to consider the derivative of this term when computing gradients, and the formulation in class is maximizing likelihood whereas we want to minimize the sume of squared weights. Introduce a parameter, lambda, that is a constant used to multiply the sum of squares of weights. If lambda = 0, then you are running logistic regression without regularization. If you increase lambda, then you are putting more emphasis on small weights.

Generate a plot of training set accuracy as a function of lambda for several values of lambda. Briefly explain what is happening.