CMSC 25010 Assignment 4

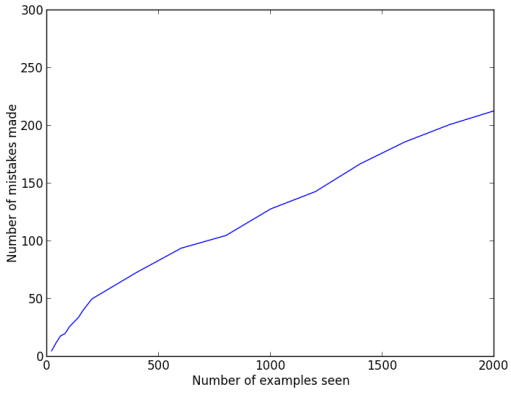
Andrew Beinstein

May 3, 2013

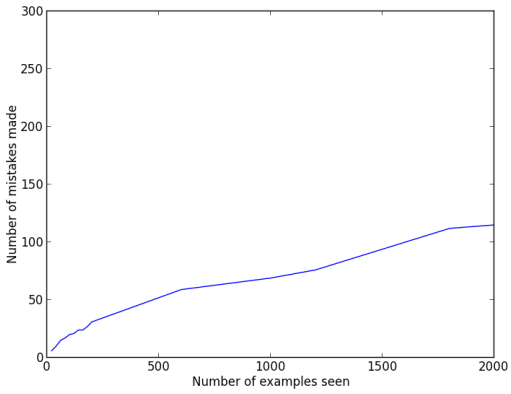
1 Number of Mistakes Made

The following plot shows the number of mistakes made by the linear and kernel perceptrons, as a function of the number of digits they were exposed to.

Linear:



Gaussian:



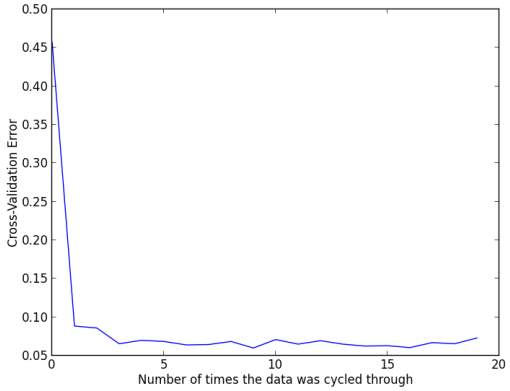
To generate these plots, I calculated the number of mistakes the perceptron made after seeing a set number of examples. We see that the data resembles a log curve. As the number of examples is increased, the number of mistakes starts to flatten, as the algorithm improves. The Gaussian curve shows that the number of mistakes grows more slowly than in the linear perceptron curve. This suggests that the Gaussian perceptron is a more accurate algorithm for recognizing digits.

2 What is the optimal σ value?

The optimal value of σ will fit the data closely enough to provide a low testing error, but will also avoid over-fitting. To achieve this goal, I used 5-fold cross validation. I split the training examples into 5 equal pieces, and ran the perceptron 5 times, using 4 of these pieces as the training set, and the fifth piece as the test set. The cross-validation error decreases until it hits the optimal value at 5, and then increases again. At σ values less than 5, the perceptron is over-fitting so the perceptron cannot accurately recognize new input. However, at σ values greater than 5, the perceptron does not fit the data closely enough. So, I will use a σ value of 5 throughout my data.

3 How many times to cycle through the data?

In batch mode, the perceptron cycles through the data multiple times. In order to calculate the optimal number of times to cycle through the data, I used cross-validation again. Below is the graph for the linear perceptron:



As you can see, after 3 cycles, there is no further gain in accuracy for cycling through the data. So, I posit that three cycles through the data is the optimal number of cycles.

4 Other observations

It appears that the best value I can get from either perceptron is about a 6 percent error rate. With $\sigma = 5$, the kernel perceptron outperforms the linear perceptron, but they are very similar. In a problem as simple as recognizing 3's and 5's, it may be more time-efficient to use the linear perceptron, as it is much faster than the Gaussian perceptron. I'd be interested in exploring other kernel functions that are faster than computing the Gaussian.