1. Neural Networks

We implemented a simple 2-layer regularized neural network, trained it using gradient descent, and used it on the provided toy datasets and MNIST handwritten digits datasets.

We choose to use the softmax formulation as described by Bishop for our loss function. Please note that although this is a different loss function than we were asked to use in our assignment description, there was a follow-up discussion on Piazza (see note @504 if not familiar) which clarified that softmax is actually the correct formulation for 1-of-K classification.

The likelihood according to softmax is:

$$p(\mathbf{t} \mid \mathbf{X}, \mathbf{w}) = \prod_{k=1}^{K} \prod_{n=1}^{N} y_k(\mathbf{x}_n, \mathbf{w})^{t_{nk}} \tag{1}$$

And taking the negative log likelihood we have our unregularized loss function:

$$l(\mathbf{w}) = -\sum_{k=1}^{K} \sum_{n=1}^{N} t_{nk} \ln y_k(\mathbf{x}_n, \mathbf{w}) \tag{2}$$

We note that $\mathbf{w}$ can be considered a vector that represents all of the weights of the neural network, but it is preferable to think of the weights as being organized into two matrices, which we denote $W^{(1)}$ and $W^{(2)}$. With the weights represented as matrices, we can vectorize the computation of the unit activations, for example for the first layer:

$$a^{(1)} = W^{(1)} x_{aug} \tag{3}$$

Gradient Calculation

Implementing 2-Layer Neural Network

Stochastic Gradient Descent

Testing the Neural Network Code

Testing the Neural Network Code

MNIST Data (Parts 5 and 6)