Machine Learning 6.867 - Pset 3

November 6, 2015

1 Multi-Class SVM

2 Neural Networks

Neural networks are used in machine learning to make predictions, similar to logistic regression, SVM, or regression. We can represent neural networks using a graph with nodes and edges (see Bishop figure 5.1). Assume that we observe data $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}), i = 1, \ldots, N$, where $\mathbf{x}^{(i)} \in \mathbb{R}^D$ and $\mathbf{y}^{(i)} \in \{0,1\}^K$. Let $(\mathbf{x}, \mathbf{y}) = ([x_0, x_1, \ldots, x_D], [y_1, \ldots, y_K])$ be a general observation. We create nodes for each of the features x_i, referred to as *inputs*, and nodes for each of the class labels y_i, referred to as *outputs*. Next, we introduce a series of nodes in the middle of the graph, called *hidden units*, and we draw edges connecting the **inputs $\rightarrow$hidden units$\rightarrow$ outputs**. The key idea in neural networks is that we can model the hidden units as functions of the inputs, and model the outputs as functions of the hidden units.

For 2-layer neural networks, we have one layer of hidden units denoted by $[z_0, z_1, \ldots, z_m]$. There are weights $\mathbf{w}^{(1)}_{ji}$ for each edge $x_i \rightarrow z_j$ and $\mathbf{w}^{(2)}_{kj}$ for each edge $z_j \rightarrow y_k$, which are unknown and will be learned through training the model. Let $\sigma(\cdot)$ denote the logistic function, which we will use as the *activation function* for both the hidden and output layers of our neural network. The predicted value for output k given parameters $\mathbf{w} = \{\mathbf{w}^{(1)}, \mathbf{w}^{(2)}\}$ and input $\mathbf{x}$ will be:

$$h_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=0}^{M} w^{(2)}_{kj} \sigma\left(\sum_{i=0}^{D} w^{(1)}_{ji} x_i\right)\right). \tag{1}$$

2.1 Implementation

2.1.1 Gradient Descent

We implemented a 2-layer regularized neural network using gradient descent. The loss function, which is the negative log-likelihood, is equal to:

$$l(\mathbf{w}) := \sum_{i=1}^{N}\sum_{k=1}^{K}\left[-y^{(i)}_k \log(h_k(\mathbf{x}^{(i)}, \mathbf{w}) - (1 - y^{(i)}_k)\log(1 - h_k(\mathbf{x}^{(i)}, \mathbf{w})\right] \tag{2}$$

We add a regularizer term, and the final cost function becomes:

$$J(\mathbf{w}) := l(\mathbf{w}) + \lambda(\|\mathbf{w}^{(1)}\|_F^2 + \|\mathbf{w}^{(2)}\|_F^2), \tag{3}$$

where $\|A\|_F = \sqrt{\sum_{i,j} A_{ij}^2}$ is the matrix Frobenius norm. Taking the gradient of J with respect to the different sets of parameters $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$, we find:

$$\nabla_{\mathbf{w}^{(1)}} J(\mathbf{w}) = \frac{1}{h(\mathbf{x}, \mathbf{w})} \sum_k \sigma(a_2)(1 - \sigma(a_2)) \tag{4}$$

$$\nabla_{\mathbf{w}^{(2)}} J(\mathbf{w}) = \tag{5}$$

2.1.2 Stochastic Gradient Descent

2.2 Computational Results

2.2.1 Toy Problem

2.2.2 MNIST Data