Chapter 1 Neural Network: Learning

1.1 Cost Function and Backpropagation

1.1.1 Cost Function

Let's define symbols for n-class classification:

- the input feature and its class: $(x^{(1)}), y^{(1)}, (x^{(2)}), y^{(2)}, \cdots, (x^{(n)}), y^{(n)}$
- L is total number of layers
- s_l is the number of units in layer l

For logistic regression, the cost function is

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{i=1}^{n}\theta_i^2\right]$$

For a neural network, it's:

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log\left(h_\Theta\left(x^{(i)}\right)\right)_k + \left(1 - y_k^{(i)}\right)\log\left(1 - \left(h_\Theta\left(x^{(i)}\right)\right)_k\right)\right]$$
$$+ \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(\Theta_{ji}^{(l)}\right)^2$$

where: $h_\Theta(x) \in \mathbb{R}^K, \quad (h_\Theta(x))_i = i^{th}$ output

The first term is for all K dimension output, and the last is the regular term of all weight in the neural network.

1.1.2 Backpropagation Algorithm

Backpropagation algorithm is a way to minimize the cost.

To use gradient descent, we need $J(\theta)$ and $\dfrac{\partial J(\theta)}{\partial \Theta_{i,j}^{(l)}}$.

So we have to compute the partical terms. We define the error of the L layer's node j:

$$\delta_j^{(l)} = a_j^{(l)} - y_j$$

Then, for earlier layers:

$$\delta^{(l-1)} = (\Theta^{(l-1)})^T\delta^{(l)} .* g'(z^{(l-1)})$$

Where:

$$g'(z^{(l)}) = a^{(l)} .* (1 - a^{(l)})$$

Finally:

$$\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta) = a_j^{(l)} \delta_i^{(l+1)}, \text{when } \lambda = 0$$

For a training set $(x^{(1)}), y^{(1)}, (x^{(2)}), y^{(2)}, \cdots, (x^{(m)}), y^{(m)}$:

```
Delta(l)(i,j) = 0
for i = 1 : m
   set a(1) = x(i)
   compute for a(l) for l = 2,3,..,L
   with y(i), compute delta(L)
   then compute delta(L-1), ...,delta(1)
   Delta(l)(i,j) = Delta(l)(i,j) + a(l)(j) * delta(l+1)(i)
endfor
```

Or in vector form:

$$\Delta^{(l)} := \Delta^{(l)} + \delta^{(l+1)}(a^{(l)})^T$$

Then:

$$D_{i,j}^{(l)} = \frac{1}{m}\Delta_{i,j}^{(l)} + \lambda\Theta_{i,j}^{(i)}, \text{ if } j \neq 0$$
$$D_{i,j}^{(l)} = \frac{1}{m}\Delta_{i,j}^{(l)} + \lambda\Theta_{i,j}^{(i)}, \text{ if } j = 0$$

And:

$$\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta) 1 = D_{i,j}^{(l)}$$