Solution 1

For the convenience of proof, here we denote $[w, b]^T$ by w^T and assume the learning rate be 1 first. Here we will prove that w^T will converge after k_{max} iterations.

Since we assume data points are separable, it means that there exists $w^\star$ which separates the points correctly.

Without loss of generality, suppose initial value of $\mathbf{w}_1 = 0$.

After k correction,

$$w_{k+1} = e(1)p(1) + e(2)p(2) + ... + e(k)p(k)$$

By multiplying each side of this equation with $w^{\star T}$, we get

$$w^{\star T}w(k+1) = w^{\star T}e(1)p(1) + w^{\star T}e(2)p(2) + ... + w^{\star T}e(k)p(k)$$

Here we define

$$\alpha = min w^{\star T}e(k)p(k)$$

Applying this equation, we get

$$w^{\star T}w(k+1) >= k\alpha$$

It means that

$$||w(k+1)||^2 >= \frac{k^2\alpha^2}{||w^\star||^2}$$

This proves the increasing trend. In the following part, we will prove the upper bound.

Since

$$w(k+1) = w(k) + e(k)p(k)$$

By applying L_2 norm, we get

$$||w(k+1)||^2 = ||w_k^2|| + 2w^T(k)e(k)p(k) + ||p(k)||^2$$

Since $w^T(k)e(k)p(k) >= 0$, we get

$$||w(k+1)||^2 <= ||w_k||^2||p(k)||^2$$

Here we define

$$\beta = max||p(k)||^2$$

Because w_{k+1} will increase at most k iterations. So $||w(k+1)||^2 <= k\beta$

Combine those two parts, we know $\frac{k^2\alpha^2}{||w^\star||^2} <= k\beta$. So $k < k_{max} = \frac{||w^\star||^2\beta}{\alpha^2}$.

Therefore, the solution is ensured to be found after at most k_{max} iterations.
The proof process is similar when the learning rate does not equal zero. The convergence is ensured and related to the choice of learning rate.

Solution 2

The relation of output gradient between two adjacent layer is

$$\delta(i) = (\sum_j u_{ji}x_i + v_{ji})\delta(j)f'(i)$$

$f'(i)$ is the derivative of activation function and here we do not contain the layer number k for clearance.
Naturally,

$$\delta(u_{ji}) = x_i^2\delta(i)$$

$$\delta(v_{ji}) = x_i\delta(i)$$

In online learning case,

$$Error = \sum_{j=1}^{N_k} e_j^2$$

So the $\delta(j)$ in output layer can be computed through

$$\delta(j) = -f'(j)e_j$$

But in batch learning case

$$ErrorBatch = \frac{1}{N_{BatchSize}}Error$$

So the $\delta(j)$ in output layer should be computed by

$$\delta(j) = -f'(j)\frac{1}{N_{BatchSize}}\sum_i^{N_{BatchSize}} e_{ji}$$

Solution 3

The code written in Torch is on my github.
[https://github.com/sunyasheng/Neural-Network-Assignment.git]
It is really wired that the test accuracy seems to stay at 83 percent precision unrelated to the hyperparameters I tuned.