T-61.231 Principles of Pattern Recognition

Answers to exercise 7: 11.11.2002

1. The objective is to find the minimum regarding ω for the joint distribution expectation value,

$$\min_{\underline{\omega}} E_{x,d}\left[(y(x)-d)^2\right] \ .$$

It helps to recall that $E_{x,d}[\cdots] = E_x[E_{d|x}[\cdots]]$ and $E_{d|x}[E_{d|x}[\cdots]] = E_{d|x}[\cdots]$.

Now to the task at hand:

$$\begin{aligned}
E_{x,d}\left[(y(x)-d)^2\right] &= E_{x,d}\left[((y(x)-E_{d|x}[d]) + (E_{d|x}[d]-d))^2\right] \\
&= E_{x,d}\left[(y(x)-E_{d|x}[d])^2 + 2(y(x)-E_{d|x}[d])(E_{d|x}[d]-d) + (E_{d|x}[d]-d)^2\right] \\
&= E_x\left[E_{d|x}[(y(x)-E_{d|x}[d])^2] + 2(E_{d|x}[y(x)]-E_{d|x}[d])(E_{d|x}[d]-E_{d|x}[d])\right. \\
&\quad \left. +(E_{d|x}[d]-E_{d|x}[d])^2\right] \\
&= E_x\left[E_{d|x}[(y(x)-E_{d|x}[d])^2]\right]
\end{aligned}$$

So,

$$\min_{\underline{\omega}} E_{x,d}\left[(y(x)-d)^2\right] = \min_{\underline{\omega}} E_x\left[E_{d|x}[(y(x)-E_{d|x}[d])^2]\right] \ .$$

Since $\left(y(x)-E_{d|x}[d]\right)^2 \geq 0$, the expectation is minimized when

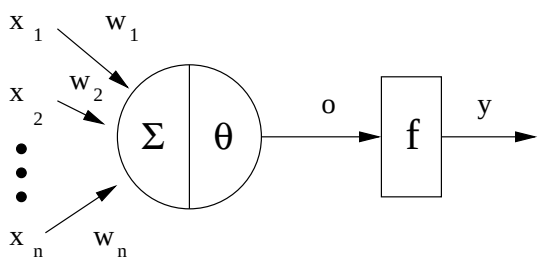
$$(y(x)-E_{d|x}[d])^2 = 0 \Rightarrow y(x) = E_{d|x}[d]$$

and

$$E_{d|x}[d] = P(d=1|x)\cdot 1 + P(d=0|x)\cdot 0 = P(d=1|x)$$

which is the optimal discriminant function for a Bayesian classifier.

2. $y = f(o) = f(\sum_{i=1}^{n} w_i x_i - \theta)$ is the actual output of the perceptron and d is the desired output.



The squared error $E = (y-d)^2$ is minimized using the gradient descent method. The gradient descent method moves the parameter vector $\underline{w} = (w_1, w_2, \ldots, w_n, \theta)^T$ to the opposite direction of the gradient $\nabla_{\underline{w}} E = (\frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial \theta})^T$:

$$w_i^{new} = w_i^{old} + \Delta w = w_i^{old} - \eta \frac{\partial E}{\partial w_i}$$

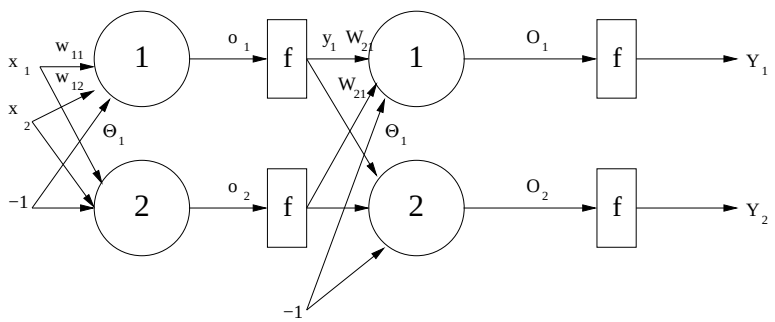
According to the chain rule

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial o}\frac{\partial o}{\partial w_i} \ .$$

Now we have $\frac{\partial E}{\partial y} = 2(y-d)$, $\frac{\partial y}{\partial o} = 1-y^2$ (since $\frac{d\tanh(x)}{dx} = 1-\tanh(x)^2, \tanh(x) = \frac{e^x-e^{-x}}{e^x+e^{-x}}$) and $\frac{\partial o}{\partial w_i} = x_i$.

Thus we can write $\Delta w_i = f(y,d)x_i$, where $f(y,d) = -\eta(y-d)(1-y^2)$.

3. With a hidden layer and an output layer



$$o_l = \sum_{k=1}^{2} w_{kl}x_k - \theta_l, \ y_l = f(o_l) \text{ (hidden layer)}$$

$$O_l = \sum_{i=1}^{2} W_{ij}y_i - \Theta_l, \ Y_i = f(O_j) \text{ (output layer)}$$

The back-propagation algorithm updates the parameters by minimizing the squared error $E = ||\underline{Y} - \underline{D}||^2$ using the gradient descent method. $\underline{Y}$ is the output of the network and $\underline{D}$ is the desired output.

$$E = ||\underline{Y} - \underline{D}||^2 = \sum_{j=1}^{2}(Y_j - D_j)^2 = \sum_{j=1}^{2}(f(O_j) - D_j)^2 = \sum_{j=1}^{2}(f(\sum_{i=1}^{2} W_{ij}y_i - \Theta_j) - D_j)^2$$

Now we have the *logsig* function $f(x) = \frac{1}{1+e^{-x}}$, so $\frac{dlogsig(x)}{dx} = logsig(x)(1-logsig(x))$.

For the output neurons, with the chain rule,

$$\frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial Y_j}\frac{\partial Y_j}{\partial O_j}\frac{\partial O_j}{\partial W_{ij}}$$

Here $\frac{\partial E}{\partial Y_j} = 2(Y_j - D_j)$, $\frac{\partial Y_j}{\partial O_j} = Y_j(1-Y_j)$ (derivative of *logsig*) and $\frac{\partial O_j}{\partial W_{ij}} = y_i$. Also $\frac{\partial O_j}{\partial \Theta_j} = -1$. Now let $\delta_j^{(2)} = (Y_j - D_j)Y_j(1-Y_j)$.

Thus

$$\frac{\partial E}{\partial W_{ij}} = 2(Y_j - D_j)Y_j(1-Y_j)y_i = 2\delta_j^{(2)}y_i$$
$$\frac{\partial E}{\partial \Theta_j} = -2(Y_j - D_j)Y_j(1-Y_j) = -2\delta_j^{(2)}$$

And the update rules become, with the learning parameter η,

$$W_{ij}^{(t+1)} = W_{ij}^{(t)} - \eta\delta_j^{(2)}(t)y_i(t)$$
$$\Theta_j^{(t+1)} = \Theta_j^{(t)} + \eta\delta_j^{(2)}(t)$$

And for the hidden layer neurons, again using the chain rule,

$$\begin{aligned}\frac{\partial E}{\partial w_{kl}} &= \sum_{j=1}^{2} \frac{\partial E}{\partial Y_j}\frac{\partial Y_j}{\partial O_j}\frac{\partial O_j}{\partial y_l}\frac{\partial y_l}{\partial o_l}\frac{\partial o_l}{\partial w_{kl}} \\ &= \sum_{j=1}^{2} 2(Y_j - D_j)Y_j(1-Y_j)W_{ij}y_l(1-y_l)x_k \\ &= 2y_l(1-y_l)x_k \sum_{j=1}^{2}(Y_j - D_j)Y_j(1-Y_j)W_{ij} \\ &= 2y_l(1-y_l)x_k \sum_{j=1}^{2}\delta_j^{(2)}W_{ij}\end{aligned}$$

Now let $\delta_l^{(1)} = y_l(1-y_l)\sum_{j=1}^{2}\delta_j^{(2)}W_{ij}$. Now

$$\begin{cases} \frac{\partial E}{\partial w_{kl}} = 2\delta_l^{(1)}x_k \\ \frac{\partial E}{\partial \theta_l} = -2\delta_l^{(1)} \end{cases} \Rightarrow \begin{cases} w_{kl}^{(t+1)} = w_{kl}^{(t)} - \eta\delta_j^{(1)}(t)x_k(t) \\ \theta_l^{(t+1)} = \theta_l^{(t)} + \eta\delta_j^{(1)}(t) \end{cases}$$

4. Let us use the notation of Theodorodis in problems 4 and 5, since the problems originate from the book. Thus, $y_k(i)$ and $\hat{y}_k(i)$ denote here the desired and actual outputs of kth output neuron, respectively.

   The cross-entropy function to be optimized is $J = -\sum_{i=1}^{N}\sum_{k=1}^{k_L} y_k(i)\ln(\frac{\hat{y}_k(i)}{y_k(i)})$ and the activation function is the sigmoid $f(x) = \frac{1}{1+\exp(-ax)}$. The gradient $\delta_k^L(i)$ can be calculated from the cost function, and since

$$J = \sum_{k=1}^{k'} \mathcal{E}(i) \Rightarrow \mathcal{E}(i) = -y_k(i)\ln(\frac{\hat{y}_k(i)}{y_k(i)})$$

   Thus

$$\begin{aligned}\delta_k^L(i) &= \frac{\partial \mathcal{E}(i)}{\partial v_k^L(i)} \\ &= -\frac{\partial y_k(i)\ln(\hat{y}_k(i)/y_k(i))}{\partial v_k^L(i)} \\ &= -y_k(i)\left(\frac{\partial}{\partial v_k^L(i)}\ln\hat{y}_k(i) - \frac{\partial}{\partial v_k^L(i)}\ln y_k(i)\right) \\ &= -y_k(i)\frac{\partial}{\partial v_k^L(i)}\ln\hat{y}_k(i) \\ &= -y_k(i)\frac{\partial}{\partial v_k^L(i)}\ln(\frac{1}{1+e^{-av_k^L(i)}}) \\ &= -y_k(i)\frac{\partial}{\partial v_k^L(i)}\{\ln 1 - \ln(1+e^{-av_k^L(i)})\} \\ &= -y_k(i)\frac{-ae^{-av_k^L(i)}}{1+e^{-av_k^L(i)}} \\ &= -ay_k(i)\left(-1 + \frac{1}{1+e^{-av_k^L(i)}}\right) \\ &= ay_k(i)(1-\hat{y}_k(i))\end{aligned}$$

5. The cross-entropy function to be optimized is $J = -\sum_{i=1}^{N}\sum_{k=1}^{k_L} y_k(i)\ln(\frac{\hat{y}_k(i)}{y_k(i)})$. Using the softmax activation function $\hat{y}_k(i) = \frac{e^{v_k^L}}{\sum_{k'=1}^{L} e^{v_{k'}^L}}$, we get

$$\begin{aligned}
\delta_k^L(i) =& \frac{\partial \mathcal{E}(i)}{\partial v_k^L(i)} \\
=& -\frac{\partial y_k(i)\ln(\hat{y}_k(i)/y_k(i))}{\partial v_k^L(i)} \\
=& -y_k(i)\left(\frac{\partial}{\partial v_k^L(i)}\ln\hat{y}_k(i) - \frac{\partial}{\partial v_k^L(i)}\ln y_k(i)\right) \\
=& -y_k(i)\frac{\partial}{\partial v_k^L(i)}\ln\hat{y}_k(i) \\
=& -y_k(i)\frac{1}{\hat{y}_k(i)}\frac{\partial\hat{y}_k(i)}{\partial e^{v_k^L}}\frac{\partial e^{v_k^L}}{\partial v_k^L(i)} \\
=& -y_k(i)\frac{1}{\hat{y}_k(i)}\left(\frac{1}{\sum_{k'} e^{v_{k'}^L}} + e^{v_k^L}(-1)(\textstyle\sum_{k'} e^{v_{k'}^L})^{-2}\right)e^{v_k^L} \\
=& -y_k(i)\frac{1}{\hat{y}_k(i)}(\hat{y}_k(i) - \hat{y}_k(i)^2) \\
=& -y_k(i)(1-\hat{y}_k(i)) \\
=& \hat{y}_k(i)y_k(i) - y_k(i)
\end{aligned}$$

Note: the result $\delta_j^L(u) = \hat{y}_j(i) - y_j(i)$ shown in the assignment in the book (*Theodorodis 4.7, p. 130*) would appear to be incorrect and is probably just an error in the book.

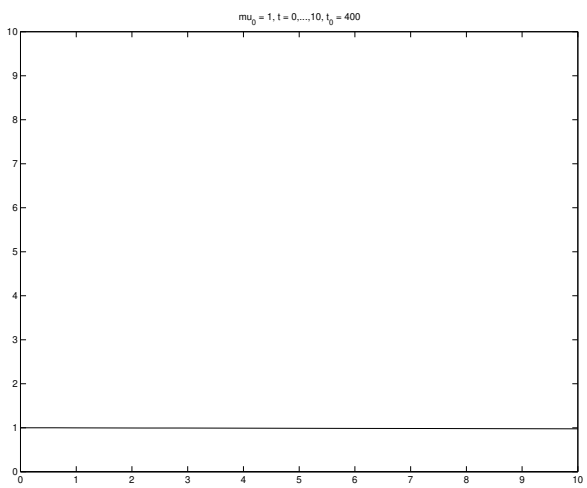
6. The behavior of the function can easily be seen, as

$$\mu = \mu_o\frac{1}{1+\frac{t}{t_0}} = \mu_o\frac{t_0}{t_0+t}$$

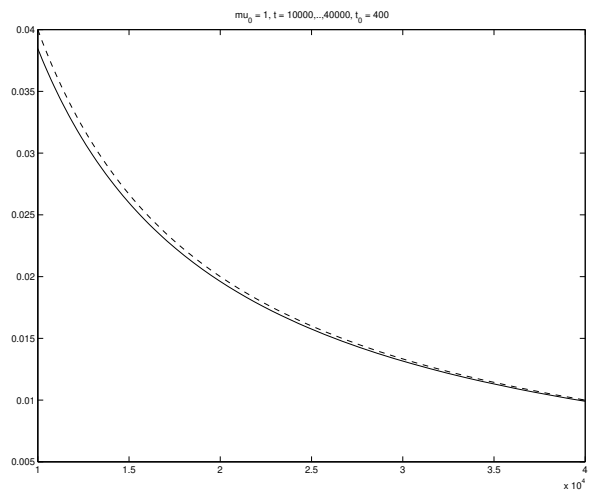
It is rather obvious, that when $t << t_0$ $\mu \approx \frac{t_0}{t_0}\mu_0 = \mu_0$. For example, if $t_0 = 400$ and $t = 1$, $\mu \approx 0.998\mu_0$.

Also, when $t >> t_0$, $\mu \approx \frac{t_0}{t}\mu_0$ which is inversely proportional to t.

The behavior in both situations has been illustrated in the figures below.



$t << t_0$



$t >> t_0$