Problem 1

$$t = h(x) + \varepsilon$$

Here x is K-dimensional vector, $h(x)$ is a deterministic function of x, where x includes x_0 bias & ε is random noise that has Gaussian probability with 0-mean & variance σ^2.

Linear function approximation with vector w, we model the data as

$$y = \sum_{i=0}^{K} w_i x_i$$

Prove, $w^* = \arg\min \sum_{n=1}^{N} (t^n - y^n)^2$ that minimizes the SSE.

$\rightarrow$, $P(t^*|x^*) = \prod_{k=1}^{N} P(t^k|x^k)$ —

Let there be N ~~samples~~ categories, so the combined probability of our data is $P(t|x)$ & ~~if all samples are independent~~ if all distributions are independent their probabilities can be multiplied.

Now, since, $t = h(x) + \varepsilon$, where ε is Gaussian noise

$$P(\varepsilon_k) = \frac{1}{(2\pi\sigma^2)^{1/2}} e^{(-\varepsilon_k^2/2\sigma^2)} \quad — 1)$$

Now, we are modelling function $h_k(x)$ by neural network with outputs $y_k(x;w)$, using $t_k = h_k(x) + \varepsilon_k$ — 2)

& 1) we get

$$P(t_k|x) = \frac{1}{(2\pi\sigma^2)^{1/2}} e^{\left(-\frac{(y_k(x,w) - t_k)^2}{2\sigma^2}\right)}$$

$$= \frac{1}{(2\pi\sigma^2)^{1/2}} e^{\left(-\frac{(y_k - t_k)^2}{2\sigma^2}\right)}$$

{If we replace $h_k(x)$ by $y_k(w,x)$ the value our model predicts}

Now, Likelihood of data

$$\mathcal{L} = \prod_n p(x^n, t^n)$$

& $E = -\ln \mathcal{L} = -\sum_n \ln p(t^n|x^n) = -\sum_n \ln p(t^n|x^n) - \sum_n \ln p(x^n)$

Since second term is independent of network parameters we can eliminate it

$$E = \frac{1}{2\sigma^2} \sum_n^N \sum_k^c \{ y_k^n(x^n; w) - t_k^n \}^2 + Nc \ln \sigma + \frac{Nc}{2} \ln(2\pi)$$

→ we can remove the last 2 terms for purpose of minimization as they are independent of w & also factor σ^2

$$E = \frac{1}{2} \sum_n \sum_k y_k(x^n, w) - t_k)^2$$

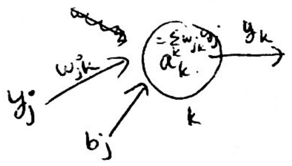
$$E = \frac{1}{2} \sum_n (y^n - t^n)^2 \quad — 3)$$

Finding minimum set of w^* which minimizes the error function E in 3). is same as

$$w^* = \arg\min_w \sum_{n=1}^N (t^n - y^n)^2 .$$

2) δ for output layer δ_k, output layer is represent as k, hidden layer as j.

$$\delta_k = -\frac{\partial E}{\partial a_k}, \quad a_k = \sum_k w_{jk} y_j$$



Now, By chain rule

$$\delta_k = -\frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial a_k} \quad — 1)$$

Since, a_k is the input to a unit k after applying the ~~act~~ summation. & y_k is the output of a unit in k.

So $$\frac{\partial y_k}{\partial a_k} = y_k' \quad — 2)$$

Now, $$\cancel{\frac{\partial E}{\partial y_k} = \frac{\partial E}{\partial y_k} \frac{\partial y_w}{\partial w}} = \frac{\partial E}{\partial w_{jk}} \cancel{\frac{\partial w_{jk}}{\partial y_k}}$$

For logistic & softmax units

$$\frac{\partial E}{\partial y_k} = \frac{y_k - t_k}{y_k'}, \quad — 3)$$

Using 2) & 3) $$-\frac{\partial E}{\partial a_k} = t_k - y_k .$$

~~For this~~ Derivation of equation ~~of~~ $\frac{\partial E}{\partial a_k}$
For softmax function

$$\frac{\partial y_l}{\partial a_m} = \frac{\dfrac{\sum e^{a_l}}{\sum_x e^{a_x}}}{\partial a_m} \quad \text{if} \quad m = l, \text{ then}$$

$$m = l, \frac{\partial y_l}{\partial a_l} = \frac{e^{a_l} \sum_x e^{a_x} - e^{a_l} e^{a_l}}{\left(\sum_x e^{a_x}\right)^2} \quad \left\{ \text{by } \frac{d}{dx}\frac{u(x)}{v(x)} = \frac{u'v - vu'}{v^2} \right\}$$

$$= \frac{e^{a_l}}{\sum_x e^{a_x}} \left[1 - \frac{e^{a_l}}{\sum_x e^{a_x}} \right] = (y_l)(1 - y_l)$$

If $m \neq l$

$$\frac{\partial y_l}{\partial a_m} = \frac{0 - e^{a_l} e^{a_m}}{\left(\sum_x e^{a_x}\right)^2} = -y_l y_m$$

Now, cross entropy function is given by

$$E = -\sum_{l=1}^{C} t_l \log y_l$$

$l = k$

$$\frac{\partial E}{\partial a_k} = \frac{-\partial}{\partial a_k} \sum_{l=1}^{C} t_l \log y_l = -\sum_{l=1}^{C} \frac{t_l}{y_l} \frac{\partial y_l}{\partial a_k} = -\frac{t_k}{y_k}(y_k)(1-y_k) - \sum_{l=1}^{C} \frac{t_l}{y_l}(-y_l)(y_k)$$

$$= -t_k + \underset{l=k}{t_k y_k} + \sum_{\substack{l=1 \\ l \neq k}}^{C} t_l y_k$$

$$= -t_k + \sum_{l=1}^{C} t_l y_k$$

$$= -t_k + y_k \quad \left\{ \sum_{l=1}^{C} t_l = 1 \right\}$$

$$= y_k - t_k$$

Now, $\boxed{\frac{\partial E}{\partial a_k} = \cancel{\frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial a_k}}\ y_k - t_k}$

for output layer.

For hidden layer

$$\delta_j = -\frac{\partial E}{\partial a_j} = -\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial a_j} \quad \{\text{By chain rule}\}$$

$$= \sum_k -\frac{\partial E}{\partial y_j}\frac{\partial a_k}{\partial a_k}\frac{\partial y_j}{\partial a_j}$$

$$= -\sum_k \frac{\partial E}{\partial a_k}\frac{\partial a_k}{\partial y_j}\frac{\partial y_j}{\partial a_j} \qquad -1)$$

Since, y_j is the output of a unit in hidden layer & a_j is its input

$$\therefore \frac{\partial y_j}{\partial a_j} = y'_j \qquad -2)$$

Also, $\frac{\partial a_k}{\partial y_j} = \frac{\partial \sum_x w_{xk} y_x}{\partial y_j} = w_{jk}$ when $x = j$ $-3)$

Since, $a_k = \sum_j w_{jk} y_j$ {~~where y_j is the output of previous layer~~}.

using 2) & 3) in 1)

we get

$$\delta_j = y'_j \sum_k w_{jk}\delta_k \qquad \left\{\because -\frac{\partial E}{\partial a_k} = \delta_k\right\}$$

So, $\delta_j^n = y'_j \sum_k w_{jk}\delta_k^n$.

b) Update Rule :-

$$w_{ij} = w_{ij} - \eta\frac{\partial E}{\partial w_{ij}}$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial a_j}\frac{\partial a_j}{\partial w_{ij}}$$

$$= \delta_j \frac{\partial a_j}{\partial w_{ij}} \qquad -1) \quad \{\text{By part a}\}$$

$$a_j^o = \sum w_{xj}^{oo} z_x^o$$

$$\frac{\partial a_j^o}{\partial w_{ij}^o} = z_i^o \quad \text{when } x = i \qquad -2)$$

Using this 1)

$$\frac{\partial E}{\partial w_{ij}^o} = -\delta_j^o z_i^o$$

So, $w_{ij}^o = w_{ij}^{o} + \eta \delta_j^o z_i^o$

$$= w_{ij}^o + \eta y_{jk}^{'} \sum \delta_k z_i w_{jk}$$

$$= w_{ij}^o + \eta \sum_k (t_k - y_k) z_i w_{jk}^o y_j^{'o} \qquad \text{(Part a)}$$

$$= w_{ij}^o + \eta \sum_{n=1}^{N} y_j^n (1 - y_j^n) z_i^n \sum_{k=1}^{C} (t_k^n - y_k^n) w_{jk}.$$

considering the hidden unit as sigmoid

c) Vectorization

* In vectorized form, we can represent our input ~~to~~ be matrix of $N \times I$ where N is the total no. of examples & I is the no. of units in each example. In our case of MNIST data, $I = 784 + 1$. 784 (28x28) & 1 more for bias.
* The weights for hidden layer can be represented as w_{ij} as a matrix of size $I \times H$ where H is the no. of ~~weights~~ hidden units.
→ The weights for output layer can also be represented (as w_{jk}) as a matrix of size $H \times C$ where C is the no. of classifications (10) in case of MNIST (0-9)

* For each layer input is represented matrix Z
for input layer Z has dimensions $N \times I$.

for hidden layer Z is $N \times H$

for output layer Z is $N \times C$

* For each layer output is represented as a matrix Y
for input layer Y's dimensions $N \times I$
hidden Y's dimension $N \times H$
output layer Y's dimension $N \times C$

* For each layer, the corresponding matrix is denoted by Z^l, Y^l, δ^l, W^l.

Now, for weight update

$$W^l = W^l + \eta(\delta^l \cdot (Z^l)^T)$$ { ~~dot product or elementwise multiplication~~

For hidden layer, δ

$$\delta^l = (\sigma'(Z^l)) * (\delta^{l+1} \cdot (W^{l+1})^T)$$ { $\cdot$: matrix multiplication
$*$: dot product or elementwise multiplication }

For output layer

$$\delta^k = T - Y^k$$ { T : Target matrix containing hot ones representation of labels }