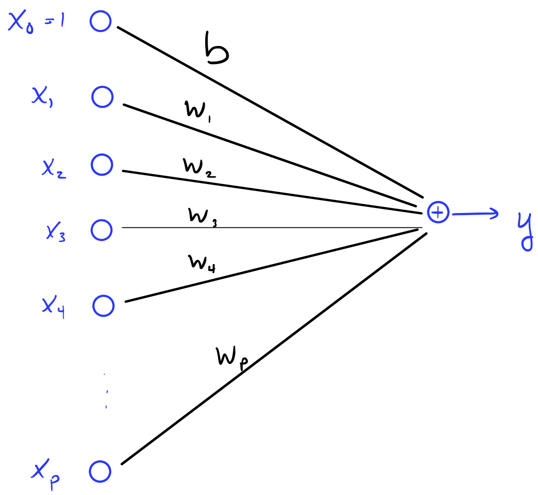
Lecture 14

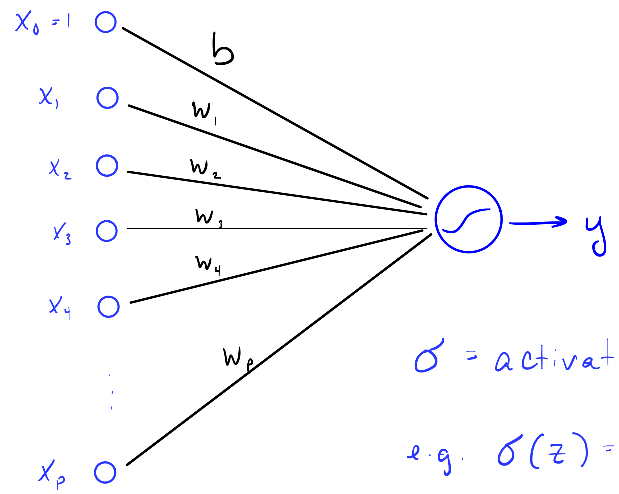
Backpropagation

A Simple Neural Network

$$y = \underline{x}^T \underline{w} + b$$



$$y = \sigma(\underline{x}^T \underline{w} + b)$$



σ = activation function

e.g. $\sigma(z) = \max(0, z)$ = ReLU

$\sigma(z) = \frac{1}{1+e^{-z}}$ = logistic

$\sigma(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} = \tanh(z)$

for $\hat{y} = \sigma(\underline{x}^T \underline{w} + b)$ and $\sigma(z) = \frac{1}{1+e^{-z}}$, how do we learn weights?

$$\text{loss} \quad f(\underline{w}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$= \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - \sigma\big(\underbrace{\langle \underline{x}_i, \underline{w}\rangle + b}_{z_i}\big)\Big)^2$$

$$\hat{y}_i = \sigma(\underline{x}_i^T \underline{w} + b) = \sigma(z_i)$$

$$z_i = \underline{x}_i^T \underline{w} + b$$

aside: $\frac{d\sigma}{dz} = \sigma'(z)$

$$= \sigma(z)(1 - \sigma(z))$$

$$\frac{d}{dz}(1+e^{-z})^{-1} = -(1+e^{-z})^{-2}(-e^{-z})$$

$$= \underbrace{\frac{1}{1+e^{-z}}}_{\sigma(z)} \; \underbrace{\frac{e^{-z}}{1+e^{-z}}}_{1-\sigma(z)}$$

learn $\underline{w}$ via Stochastic Gradient Descent!

@ iteration t

- choose i_t uniformly at random
- set $\underline{w}^{(t+1)} = \underline{w}^{(t)} - \tau \nabla f_{i_t}(\underline{w}^{(t)})$

$$b^{(t+1)} = b^{(t)} - \tau \nabla f_{i_t}(b^{(t)})$$

What is $\nabla f_i(\underline{w}^{(t)})$?

$$\left.\frac{df_i}{dw_j}\right|_{\underline{w}^{(t)}} = \frac{df_i}{d\hat{y}_i}\cdot\frac{d\hat{y}_i}{dz_i}\cdot\left.\frac{dz_i}{dw_j}\right|_{\underline{w}^{(t)}}$$

$$= 2(\hat{y}_i - y_i)\cdot\sigma'(z_i)\cdot \left. x_{ij}\right|_{\underline{w}^{(t)}}$$

$$= 2(\hat{y}_i - y_i)\,\sigma(z_i)(1-\sigma(z_i))\left. x_{ij}\right|_{\underline{w}^{(t)}}$$

$$= \underbrace{2(\hat{y}_i - y_i)\,\hat{y}_i(1-\hat{y}_i)}_{\text{scalar, independent of } j}\, x_{ij}$$

call $\delta_i = \delta_i(\underline{w}^{(t)})$

$$\Rightarrow \nabla f_i(\underline{w}^{(t)}) = \delta_i \underline{x}_i$$

$$\Rightarrow \text{SGD: } \underline{w}^{(t+1)} = \underline{w}^{(t)} - \tau\delta_{i_t}\underline{x}_{i_t}, \quad b^{(t+1)} = b^{(t)} - \tau\delta_{i_t}$$

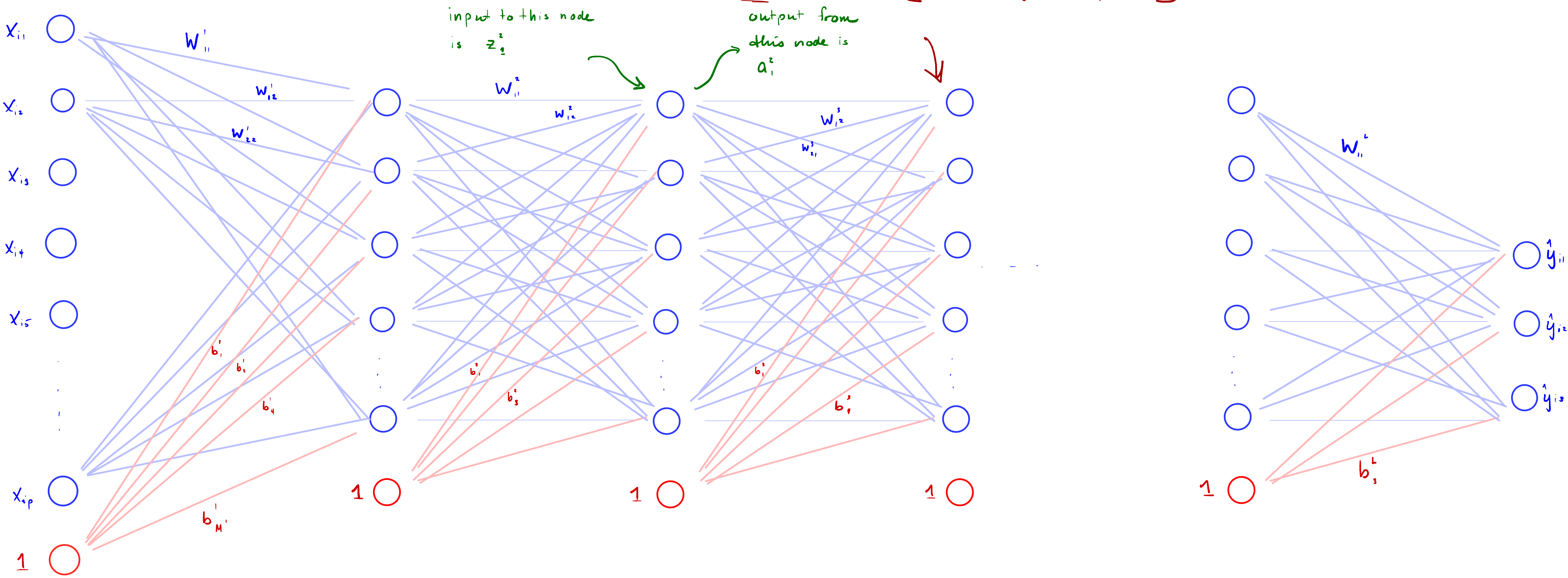
What is $\nabla_b f_{i_t}(b^{(t)}) = \left.\frac{df_{i_t}}{db}\right|_{b^{(t)}}$?

$$\left.\frac{df_i}{db}\right|_{b^{(t)}} = \frac{df_i}{d\hat{y}_i}\cdot\frac{d\hat{y}_i}{dz_i}\cdot\left.\frac{dz_i}{db}\right|_{b^{(t)}}$$

$$= \delta_i \cdot 1$$

More generally

$$a^{\ell} \qquad z^{\ell+1} = W^{\ell+1} a^{\ell} + b^{\ell+1}$$



$\underline{X} = a^0$ $\qquad W^1, b^1 \qquad W^2, b^2 \qquad W^3, b^3 \qquad \ldots \qquad W^L, b^L$

$$z^1 = W^1 a^0 + b^1 \qquad z^2 = W^2 a^1 + b^2 \qquad z^3 = W^3 a^2 + b^3 \qquad z^L = W^L a^{L-1} + b^L$$

$$a^1 = \sigma(z^1) \qquad a^2 = \sigma(z^2) \qquad a^3 = \sigma(z^3) \qquad a^L = \sigma(z^L) = \hat{y}$$

let M^{ℓ} = # of blue nodes in layer ℓ

$W^{\ell} \sim M^{\ell} \times M^{\ell-1}$,

$\underline{S}^{\ell} \sim M^{\ell} \times 1$,

$\underline{a}^{\ell}, \underline{z}^{\ell} \sim M^{\ell} \times 1$,

$\underline{b}^{\ell} \sim M^{\ell} \times 1$

z^ℓ is input to nodes in layer ℓ, a^ℓ is output of those nodes

$W^\ell_{j,k}$ = weight in layer ℓ, applied to layer $\ell-1$ output $(a_k^{\ell-1})$, feeding into next layer (z_j^ℓ)

To update W^ℓ, need to compute $\nabla f(W^\ell)$

$$\frac{df}{dW^\ell_{j,k}} = \frac{df}{dz_j^\ell} \cdot \frac{dz_j^\ell}{dW^\ell_{j,k}} = \delta_j^\ell \cdot a_k^{\ell-1} \implies \nabla f(W^\ell) = \underline{\delta}^\ell (\underline{a}^{\ell-1})^T$$

$$\delta_j^\ell := \frac{df}{dz_j^\ell} = \begin{cases} \frac{df}{da_j^\ell} \cdot \frac{da_j^\ell}{dz_j^\ell} = \left[(W^{\ell+1})^T \delta^{\ell+1} \right]_j \cdot \sigma'(z_j^\ell) & \ell < L \\ \frac{df}{a_j^\ell} \frac{da_j^\ell}{dz_j^\ell} & \ell = L \end{cases}$$

$$\underline{\delta}^\ell = \begin{cases} \left((W^{\ell+1})^T \underline{\delta}^{\ell+1} \right) \odot \sigma'(\underline{z}^\ell) \\ \nabla f(\underline{a}^L) \odot \sigma'(\underline{z}^\ell) \end{cases}$$

element-wise multiplication

$$\frac{df}{da_j^\ell} = \sum_{k=1}^{M^{\ell+1}} \frac{df}{dz_k^{\ell+1}} \frac{dz_k^{\ell+1}}{da_j^\ell} = \sum_{k=1} \delta_k^{\ell+1} W_{kj}^{\ell+1} = \left[(W^{\ell+1})^T \delta^{\ell+1} \right]_j$$

$$\frac{da_j^\ell}{dz_j^\ell} = \sigma'(z_j^\ell)$$

$$\frac{dz_j^\ell}{dW^\ell_{j,k}} = a_k^{\ell-1} \quad \text{because } z_j^\ell = (W^\ell a^{\ell-1} + b^\ell)_j = \sum_k W^\ell_{jk} a_k^{\ell-1} + b_j^\ell$$

$$\frac{df}{db_j^\ell} = \frac{df}{dz_j^\ell} \cdot \frac{dz_j^\ell}{db_j^\ell} = \delta_j^\ell \cdot 1 \implies \nabla f(b^\ell) = \delta_j^\ell$$

Consider $f(x_i) = \sum_{k=1}^{q} (\hat{y}_{ik} - y_{ik})^2 = \|\hat{y}_i - y_i\|_2^2$

$$\frac{df}{da_j^L} = \frac{d}{da_j^L} \left(\sum_{k=1}^{q} (a_k^L - y_{ik})^2 \right)$$

$$= \frac{d}{da_j^L} (a_j^L - y_{ij})^2$$

$$= 2(a_j^L - y_{ij})$$

$a^L = \hat{y}_i \implies a_j^L = \hat{y}_{ij}$

Backpropagation Algorithm

for $t = 1, 2, 3, \dots$

select $i_t \sim \text{unif}(1, 2, \dots, n)$

\# forward pass:

$$\underline{a}^0 = \underline{x}_{i_t}$$

for $l = 1, 2, \dots, L$

$$\underline{z}^l = W^{l(t)} \underline{a}^{l-1} + \underline{b}^{l(t)}$$

$$\underline{a}^l = \sigma(\underline{z}^l)$$

end

forward pass

\# backprop

$$\underline{\delta}^L = \nabla f_{i_t}(a^L) \cdot \sigma'(z^L)$$

for $l = L-1, L-2, \dots, 1$

$$\underline{\delta}^l = \left[(W^{l+1\,t})^T \delta^{l+1} \right] \odot \sigma'(\underline{z}^l)$$

$$\nabla f(W^{lt}) = \underline{\delta}^l (\underline{a}^{l-1})^T$$

$$\nabla f(b^{lt}) = \underline{\delta}^l$$

$$W^{l\,t+1} = W^{lt} - \tau \nabla f(W^{lt})$$

$$b^{l\,t+1} = b^{lt} - \tau \nabla f(b^{lt})$$

end

backprop.

The following slides show backpropagation for a 2-layer network using slightly different notation. The concepts are no different than the above slides, but some students find it easier to study the 3-layer case first.

2-layer network (1 hidden layer)

w_{kj} = weight on j^{th} element of x_i on hidden node k

$\Rightarrow h_{ik}$ = output of k^{th} hidden node if x_i = input

$$= \sigma(W \underline{x}_i) = \sigma\left(\sum_{j=1}^{p} w_{kj} x_{ij}\right)$$

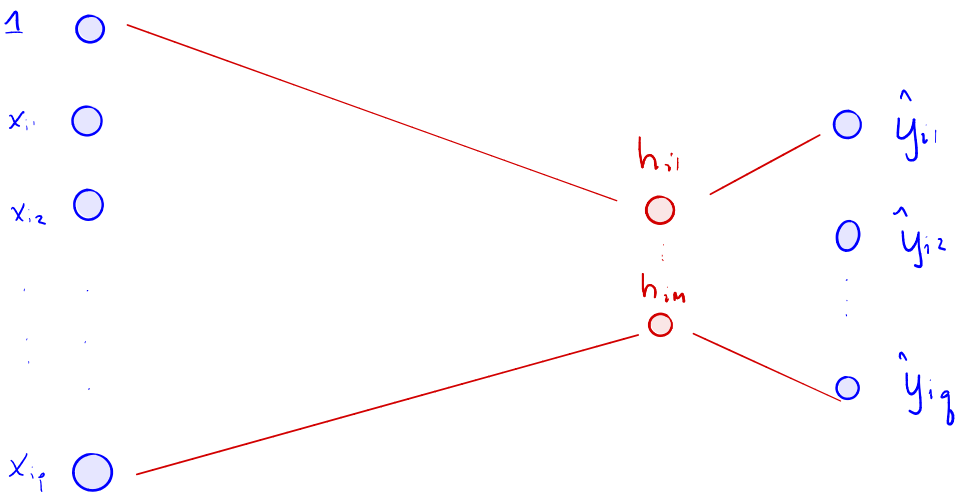
$$\Rightarrow \underline{h}_i = \begin{bmatrix} h_{i0} \\ h_{i1} \\ \vdots \\ h_{iM} \end{bmatrix}$$

V_{kj} = weight on j^{th} hidden node output on predictor k

$\Rightarrow \hat{y}_{ik}$ = output k for input i

$$= \sigma(V \underline{h}_i) = \sigma\left(\sum_{m=0}^{M} v_{km} h_{im}\right)$$

Ex 2-layer network

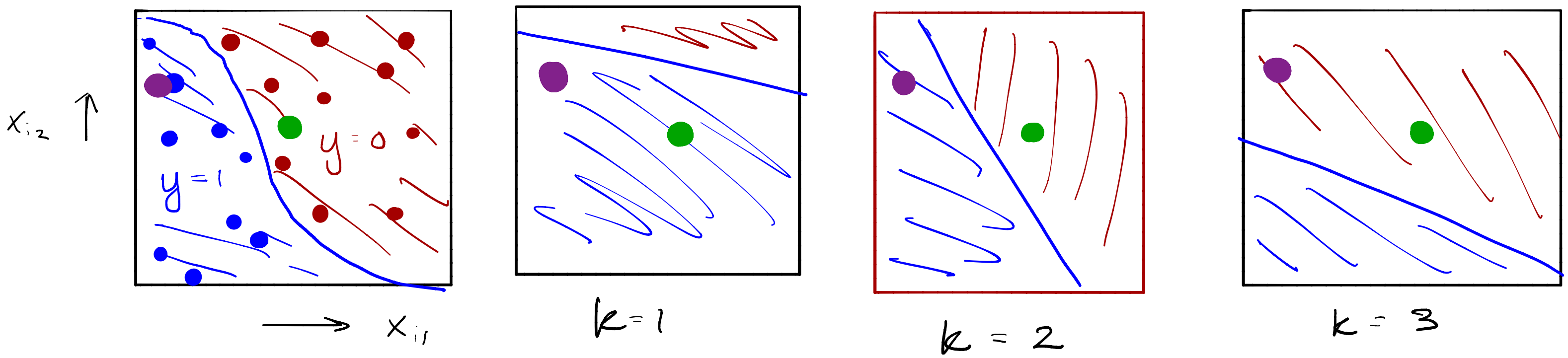


when M is small (i.e. small # hidden nodes @ some layer), then we can take the vector

$$\begin{bmatrix} h_{i1} \\ \vdots \\ h_{iM} \end{bmatrix}$$

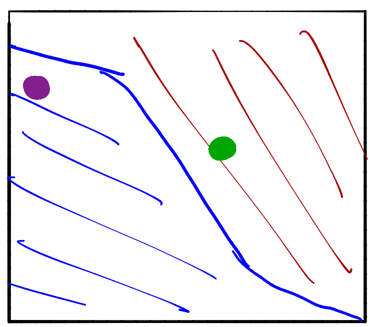
and this can considered as latent low-dimensional features.

Output of hidden node k : $x_i \longrightarrow h_k$



1^{st} set of weights define these intermediate linear classifiers

final output = weighted sum of h_k's + thresholding



2^{nd} set of weights let us combine the linear classifiers

SGD for 2-layer network:

$$\hat{y}_{ik} = \sigma(V h_i) = \sigma(V \sigma(W \underline{x}_i)) = \sigma\left(\sum_{m=0}^{M} V_{km}\, \sigma\left[\sum_{j=0}^{P} w_{mj}\, x_{ij}\right]\right)$$

Backpropagation algorithm

- choose initial weights $W^{(1)}$ and $V^{(1)}$
- for $t = 1, 2, \dots$
  - choose i_t
  - calculate $\hat{y}_{i_t k}$ and $h_{i_t m}$ using $W^{(t)}$ and $V^{(t)}$ (forward pass)

$$\hat{y}_{i_t k} = \sigma(V^{(t)} h_i) = \sigma(V^{(t)} \sigma(W^{(t)} \underline{x}_i)) = \sigma\left(\sum_{m=0}^{M} V^{(t)}_{km}\, \sigma\left[\sum_{j=0}^{P} w^{(t)}_{mj}\, x_{ij}\right]\right)$$

  - update weights for each layer, starting with deepest layer (closest to output) and working back to shallowest

$$V^{(t+1)} = V^{(t)} - \tau\, \underline{h}_{i_t}\, \underline{\delta}_{i_t}^T \quad \text{where } \underline{\delta}_i^T = [\delta_{i,0}\ \ \delta_{i,1}\ \cdots\ \delta_{i,K}]$$

$$W^{(t+1)} = W^{(t)} - \tau\, \underline{x}_{i_t}\, \gamma_{i_t}^T \quad \text{where } \underline{\gamma}_{i_t}^T = [\gamma_{i_t,0}, \dots, \gamma_{i_t,M}]$$

To update $V^{(t)}$:

$$\frac{df_i}{dV_{km}} = \frac{df_i}{d\hat{y}_i} \cdot \frac{d\hat{y}_i}{dV_{k,m}}$$

$$= 2(\hat{y}_{ik} - y_{ik})\,\sigma'\left(V^{(t)}\underline{h}_i\right) h_{i,m}$$

$$= \underbrace{2(\hat{y}_{ik} - y_{ik})\,\hat{y}_{i,k}(1 - \hat{y}_{ik})}_{= \delta_{i,k}}\, h_{i,m}$$

$$= \delta_{ik}\, h_{i,m}$$

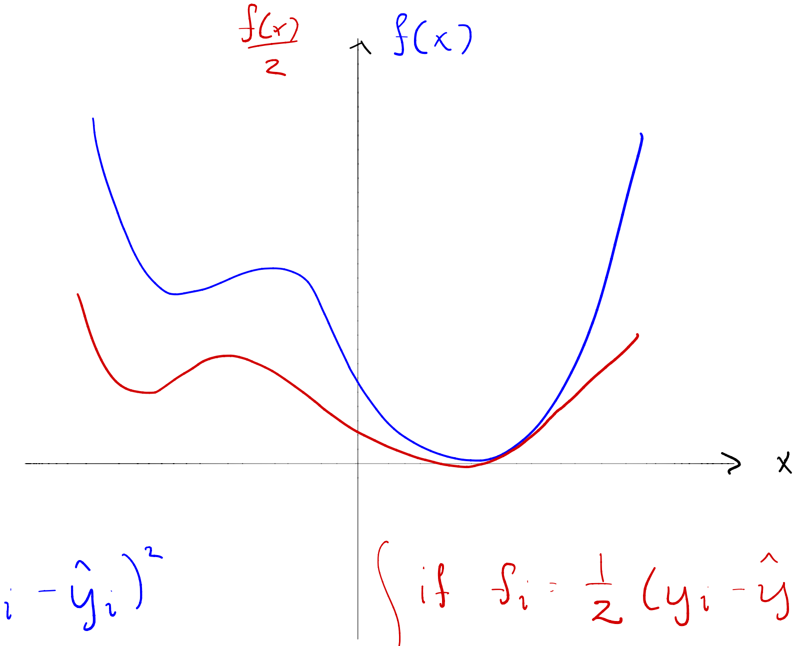
$$\Longrightarrow V^{(t+1)}_{k,m} = V^{(t)}_{k,m} - \tau\, \delta_{i_t,k}\, h_{i_t,m}$$

$$\Longrightarrow \underline{V}_k^{(t+1)} = \underline{V}_k^{(t)} - \tau\, \delta_{i_t,k}\, \underline{h}_{i_t}$$

where $\underline{V}_k = (k^{th} \text{ row of } V)^T$ = vector of weights determining k^{th} predictor

$$\Longrightarrow V^{(t+1)} = V^{(t)} - \tau\, \underline{h}_{i_t}\, \underline{\delta}_{i_t}^T$$

where $\underline{\delta}_i^T = [\delta_{i,0} \quad \delta_{i,1} \quad \cdots \quad \delta_{i,K}]$



if $f_i = (y_i - \hat{y}_i)^2$

then $\dfrac{df_i}{d\hat{y}_i} = -2(y_i - \hat{y}_i)$

if $f_i = \frac{1}{2}(y_i - \hat{y}_i)^2$

then $\dfrac{df_i}{d\hat{y}_i} = -(y_i - \hat{y}_i)$

To update $W^{(t)}$:

$$\frac{df_i}{dW_{m,j}} = \sum_{k=1}^{q} \frac{df_i}{d\hat{y}_{ik}} \cdot \frac{d\hat{y}_{ik}}{dh_{im}} \cdot \frac{dh_{im}}{dW_{m,j}}$$

$$= 2\sum_{k=1}^{q} (\hat{y}_{ik} - y_{ik})\, \sigma'\left(\left(\underline{v}_k^{(t)}\right)^T \underline{h}_i\right) v_{k,m} \cdot \sigma'\left(\left(\underline{w}_m^{(t)}\right)^T \underline{x}_i\right) x_{ij}$$

$$= \sum_{k=1}^{q} \underbrace{2(\hat{y}_{ik} - y_{ik})\,\hat{y}_{ik}(1-\hat{y}_{ik})}_{=\delta_{i,k}}\, v_{k,m}\, h_{im}(1-h_{im})\, x_{ij}$$

$$= \sum_{k=1}^{q} \underbrace{\delta_{ik}\, v_{k,m}\, h_{i,m}(1-h_{i,m})}_{=:\gamma_{im}}\, x_{ij}$$

$$= \gamma_{i,m}\, x_{ij}$$

$$\Rightarrow W_{m,j}^{(t+1)} = W_{m,j}^{(t)} - \tau\, \gamma_{i_t,m}\, x_{i_t,j}$$

$$\Rightarrow \underline{w}_m^{(t+1)} = \underline{w}_m^{(t)} - \tau\, \gamma_{i_t,m}\, \underline{x}_{i_t}$$ where $\underline{w}_m = (m\text{-th row of } W)^T$ = weights going into m^{th} hidden node.

$$\Rightarrow W^{(t+1)} = W^{(t)} - \tau\, \underline{x}_{i_t}\, \gamma_{i_t}^T$$ where $\underline{\gamma}_{i_t}^T = [\gamma_{i_t,0}, \ldots, \gamma_{i_t,M}]$