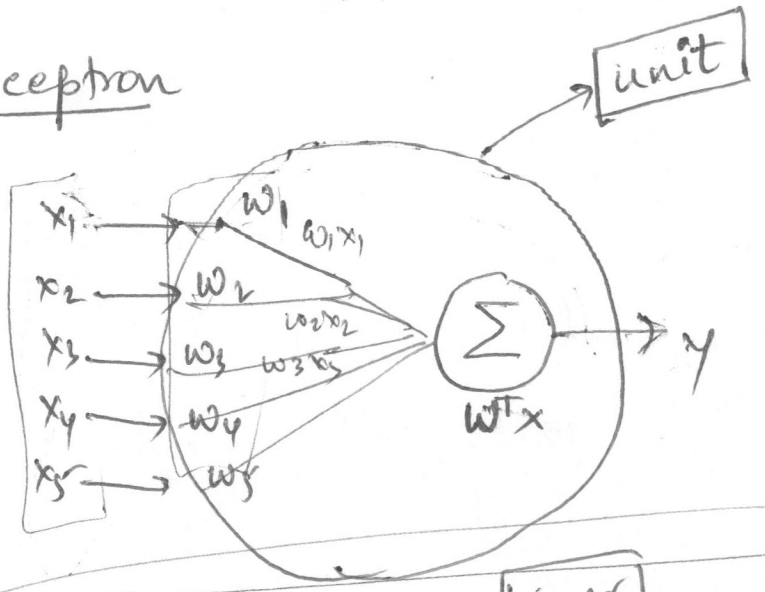
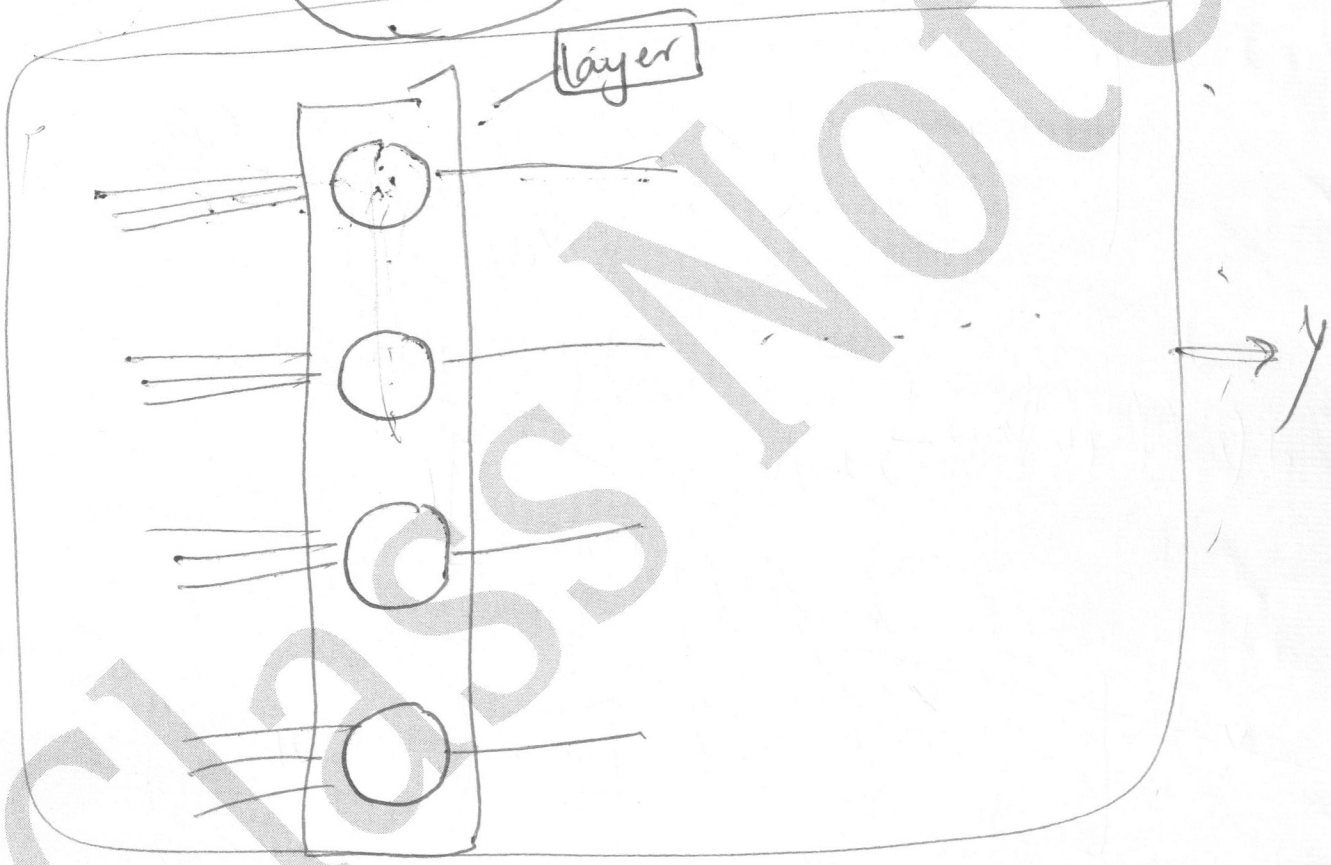
Perceptron



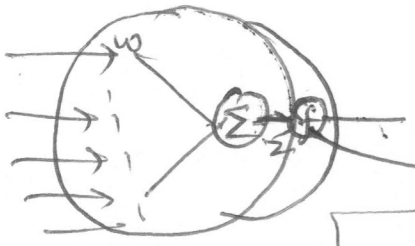
Linear unit.



Width of a layer → # units.

Class Notes

Instead of linear unit, we use a non-linear unit.

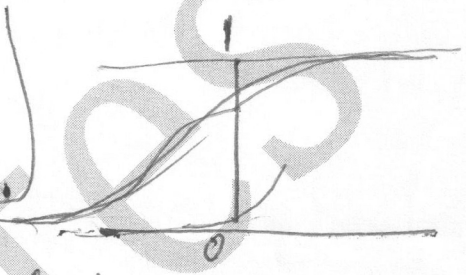


$z = w^T x$

activation fn.

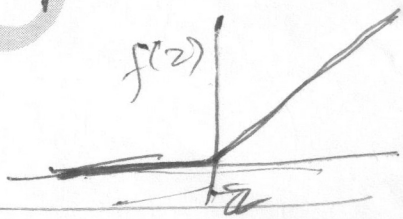
Sigmoid fn.

$$f(z) = \frac{1}{1 + \exp(-z)}$$



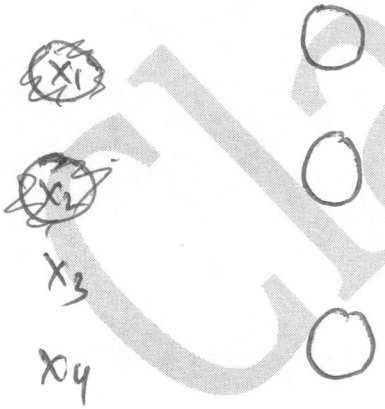
tanh unit $f(z) = \dfrac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)}$

RELU unit. $f(z) = \max(0, z)$

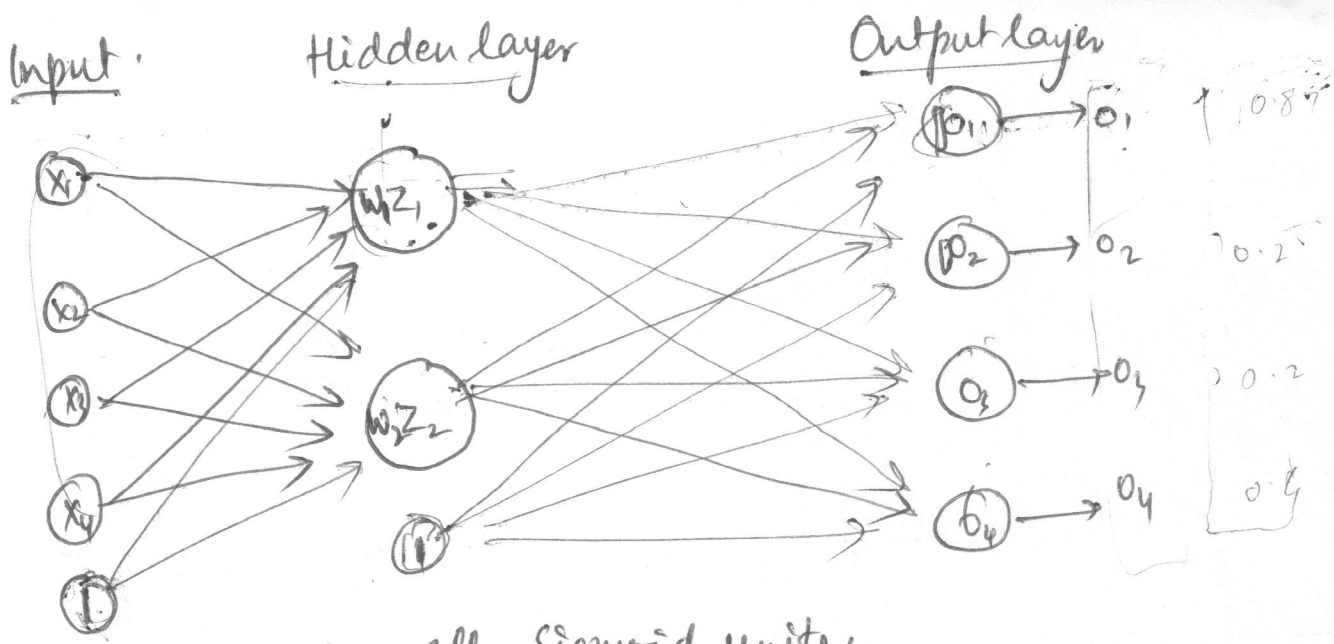


Multi-layered Perceptron (MLP) → K-way classification
K=4

$x \in R^4$ Input layer Hidden layer Output layer

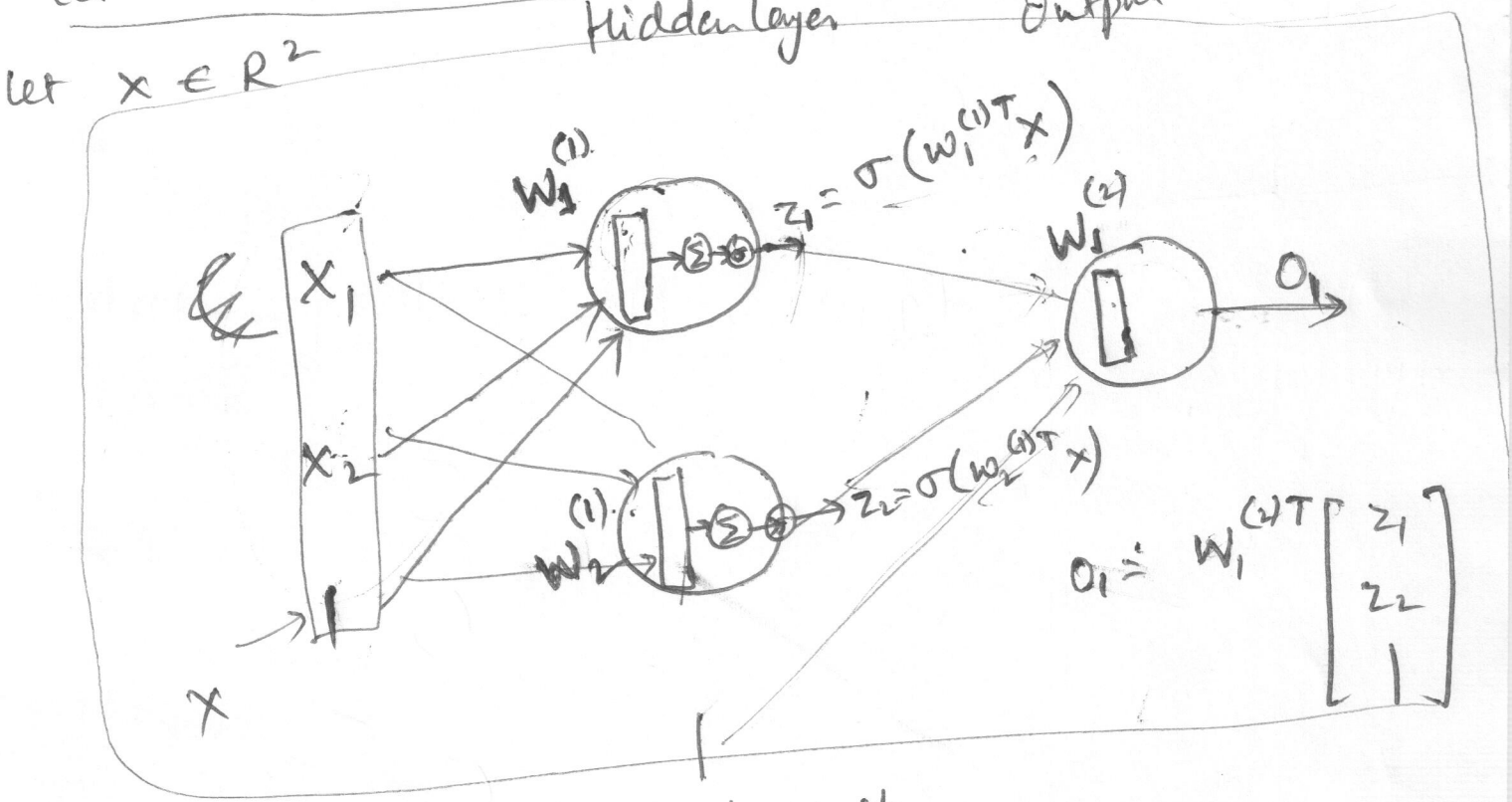


Class Notes

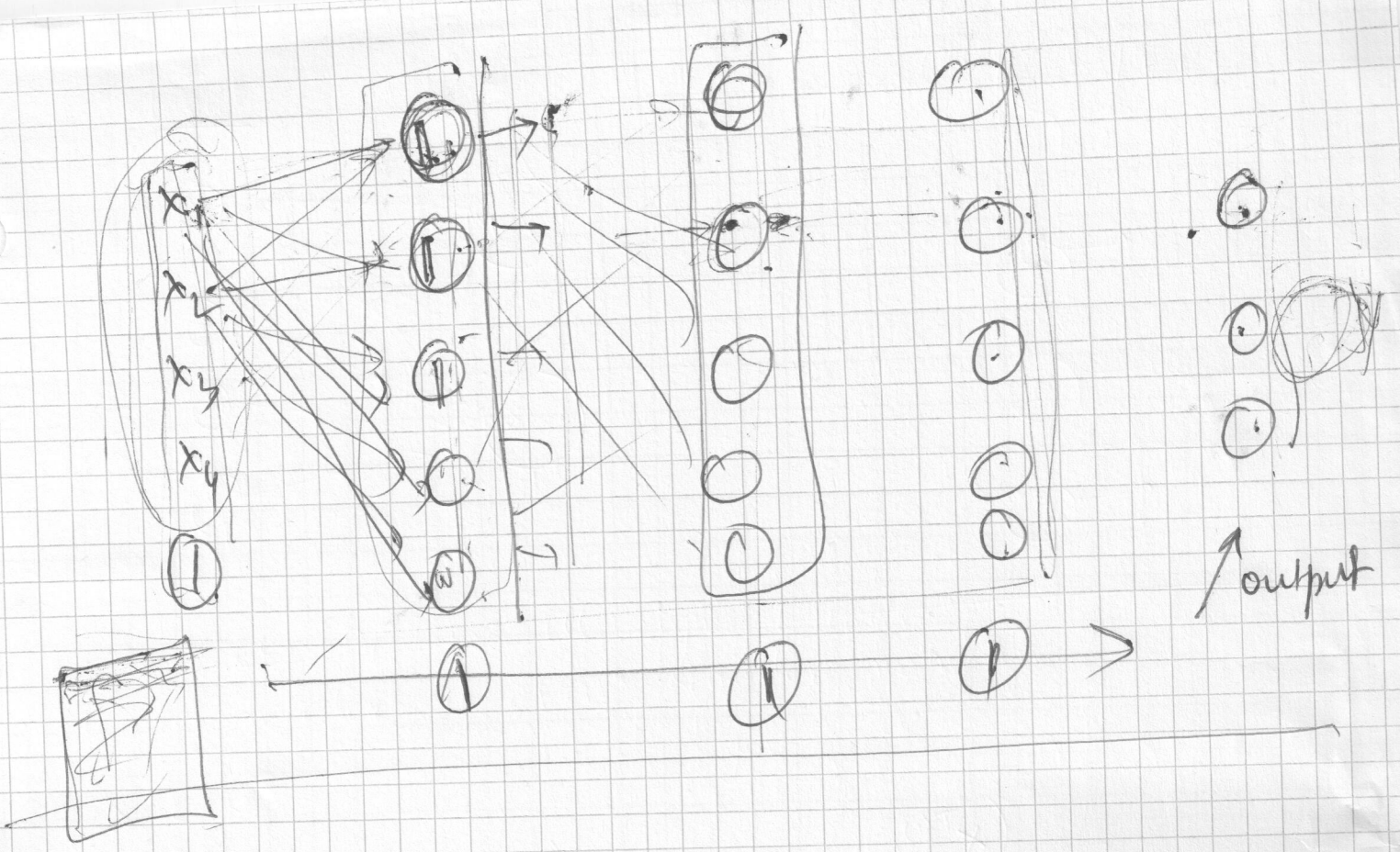


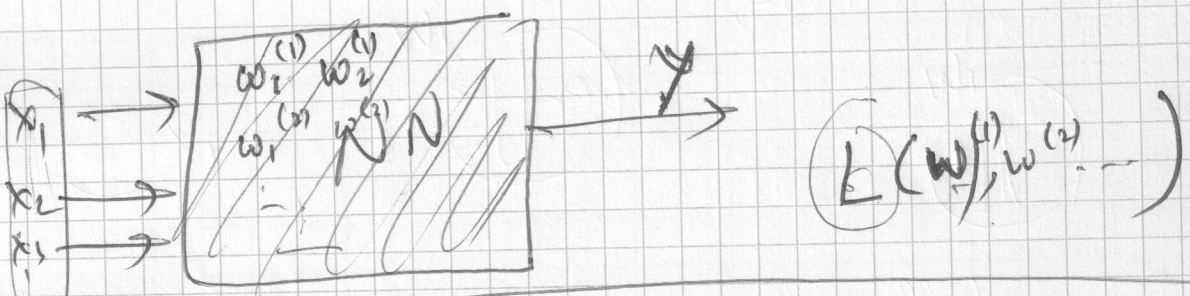
$z_1, z_2, o_1 \dots o_4$ are all sigmoid units.

Let us consider an even simpler network.



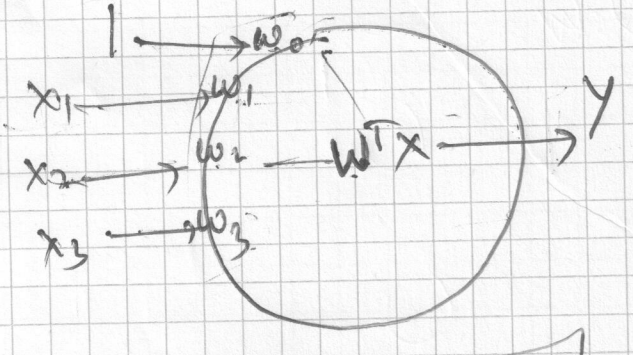
activation - output of a unit





Assuming we have only 1 training example

$$x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad y$$



Initialize the weight vector.

$$L(w) = \frac{1}{2}(y - w^T x)^2$$

$$\frac{d}{dw_0} L(w) = -(y - w^T x)$$

$$\frac{d}{dw_1} L(w) = -(y - w^T x)\, x_1$$

$$\vdots$$

$w^T x$

$= w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$

new weight

$$w_0 = w_0 - \eta \frac{d}{dw_0} L(w)$$

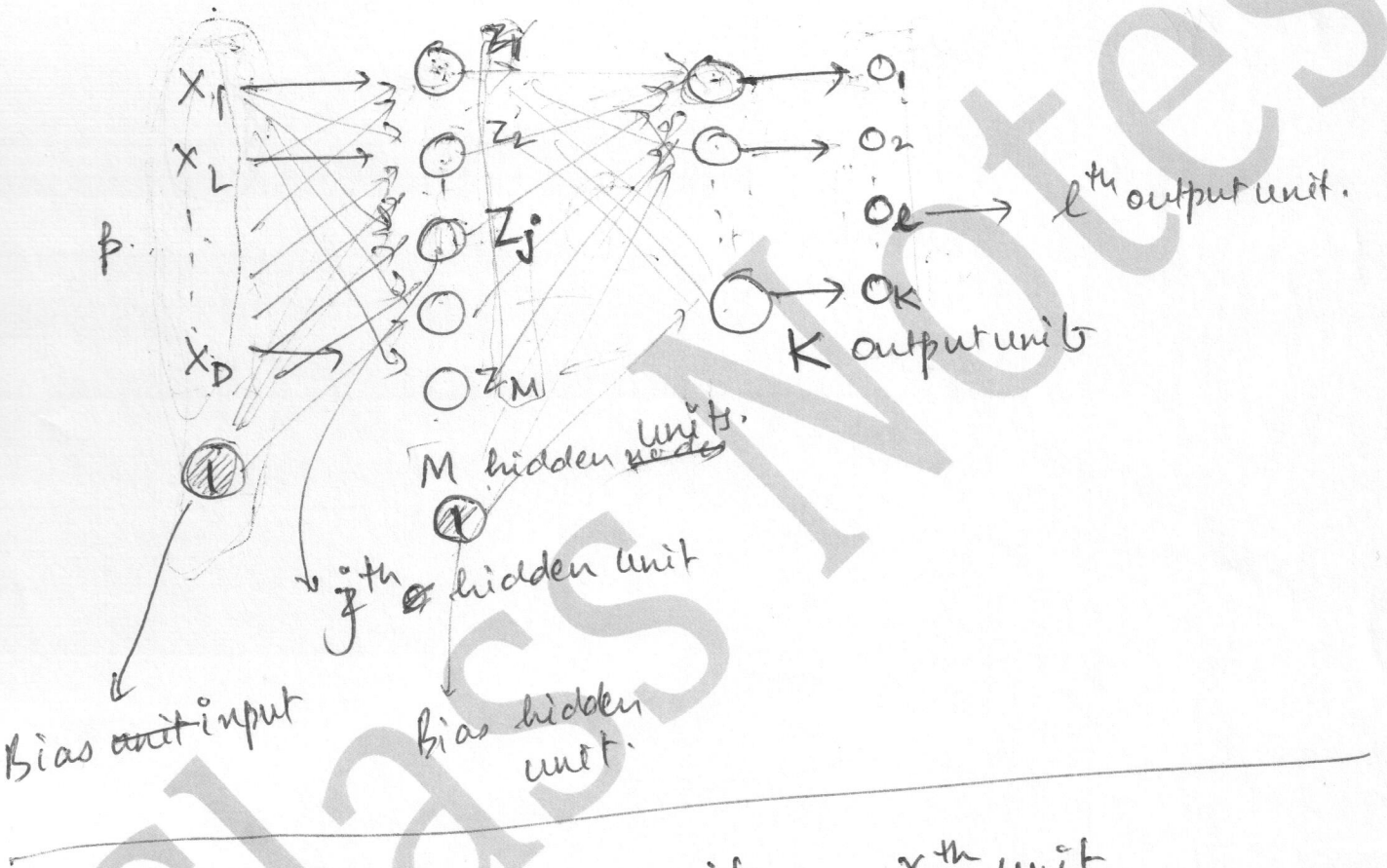
$$= w_0 - \eta\,(-(y - w^T x))$$

$$= w_1 - \eta\,(-(y - w^T x)\, x_1)$$

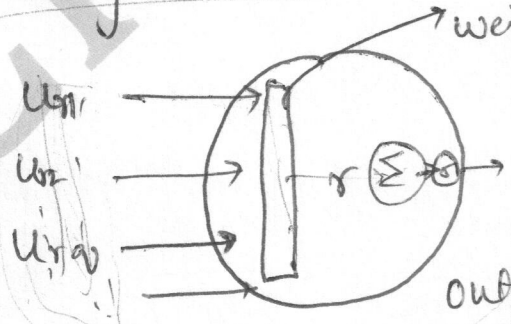
How to train a multi-layered perceptron? (MLP)

For k-way classification

~~$x \in \mathbb{R}^D$~~ $x \in \mathbb{R}^D$ $y \in \{1, \ldots, k\}$



Consider a general unit. r^{th} unit



$$net_r = w_r^T u_r = \sum_q w_{rq} u_{rq}$$

output of r^{th} unit will be $\sigma(net_r)$

$$\sigma(net_r) = \frac{1}{1 + \exp(-net_r)}$$

Class Notes

For the j^{th} hidden unit

$$z_j = \sigma(net_j) = \sigma(w_j^T x) \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \\ 1 \end{bmatrix}$$

For the l^{th} output unit

$$o_l = \sigma(net_l) = \sigma(w_l^T z) \qquad z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_M \\ 1 \end{bmatrix}$$

Predicted class = $\text{argmax}[o_1, o_2, \cdots o_K]$

For the j^{th} hidden unit: w_j is a $(D+1)$ length vector

For the l^{th} output unit: w_l is a $(M+1)$ length vector

Training: Given N training examples

$$\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_N & y_N \end{bmatrix}$$

~~to find~~ learn w_j's and w_l's.

$j = 1 \cdots M$ $\quad$ $l = 1 \cdots K$

$J(w_j\text{s} \;\&\; w_l\text{s}) =$ _____

If $y = 2$

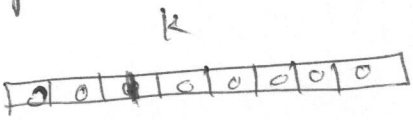
o_1
o_2
$\vdots$
o_l
$\vdots$
o_K

Class Notes

Convert y_i into a vector

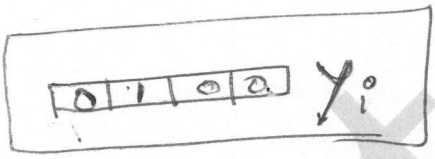
y_i ↑ label for i^{th} example.

One-Hot-Encoding
1 of k Encoding.



example let $k = 4$

$y_i = 2$



loss for the i^{th} example:

$$J_i(\quad) = \frac{1}{2}\sum_{\ell=1}^{k}(y_{i\ell} - O_{i\ell})^2$$

$O_{i\ell}$ → output at ℓ^{th} output unit for i^{th} training example.

Squared loss

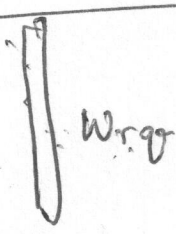
Total loss $J = \sum_{i=1}^{N} \frac{1}{2} \sum_{\ell=1}^{k} (y_{i\ell} - O_{i\ell})^2$

We will assume that $N = 1$, drop the subscript i.

$$J = \frac{1}{2} \sum_{\ell=1}^{k} (y_\ell - O_\ell)^2$$

---

for a general unit



We need $\frac{\partial J}{\partial w_{rq}}$

$$w_{rq} \leftarrow w_{rq}^{(old)} - \eta \frac{\partial J}{\partial w_{rq}}$$

Class Notes

Obs. 1

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

Not the same z as before

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

$$= \left(\frac{e^{-z}}{1+e^{-z}}\right)\left(\frac{1}{1+e^{-z}}\right)$$

$$= \left(1-\frac{1}{1+e^{-z}}\right)\left(\frac{1}{1+e^{-z}}\right)$$

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

Obs. 2

For any r^{th} unit.

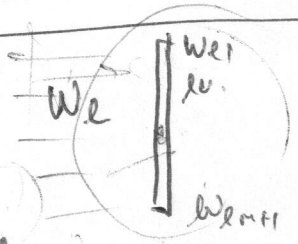
$$\frac{\partial J}{\partial w_{rq}} = \frac{\partial J}{\partial net_r} * \frac{\partial net_r}{\partial w_{rq}}$$

$$net_r = \sum_q w_{rq} u_{rq} = w_{r1}u_{r1} + w_{r2}u_{r2} + \ldots w_{rq}u_{rq} + \ldots$$

$$\boxed{\frac{\partial net_r}{\partial w_{rq}} = u_{rq}}$$

$$\frac{\partial J}{\partial w_{rq}} = u_{rq}\frac{\partial J}{\partial net_r}$$

Consider the ℓ^{th} output unit.



We want $\frac{\partial J}{\partial w_{\ell j}} = z_j \frac{\partial J}{\partial net_\ell}$

$o_\ell = \sigma(net_\ell)$

$$\frac{\partial J}{\partial net_k} = \frac{\partial J}{\partial o_k} \frac{\partial o_k}{\partial net_k} \rightarrow o_k(1-o_k)$$

$$= \frac{\partial J}{\partial o_k} o_k(1-o_k)$$

$$\frac{\partial J}{\partial o_k} = \frac{\partial}{\partial o_k}\left[\frac{1}{2}\sum_{k=1}^{k}(y_k - o_k)^2\right]$$

$$= \frac{1}{2}\frac{\partial}{\partial o_k}(y_k - o_k)^2$$

$$= \frac{1}{2} * 2(y_k - o_k)(-1)$$

$$= -(y_k - o_k)$$

$$\boxed{\frac{\partial J}{\partial w_{kj}} = -z_j\, o_k(1-o_k)(y_k - o_k)}$$

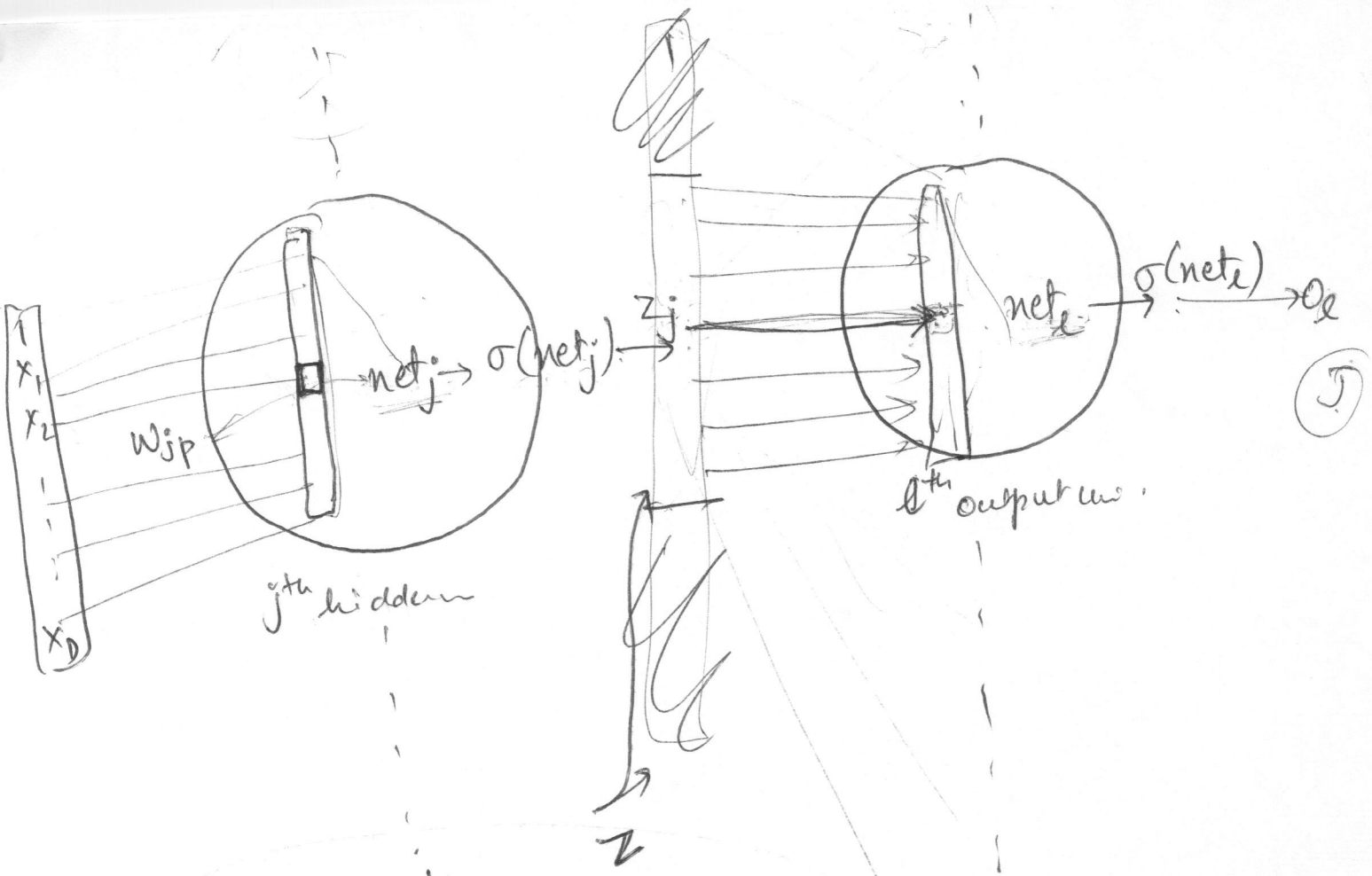
We will denote $o_k(1-o_k)(y_k - o_k) = \delta_k$ ← error at k^{th} output unit

$$\frac{\partial J}{\partial w_{kj}} = -\delta_k z_j$$

Update rule:

$$w_{kj} = w_{kj}^{(old)} - \eta \frac{\partial J}{\partial w_{kj}}$$

$$= w_{kj}^{(old)} + \eta \delta_k z_j$$



$$\frac{\partial J}{\partial net_j} = \sum_{\ell=1}^{k} \frac{\partial J}{\partial net_\ell} \cdot \frac{\partial net_\ell}{\partial net_j}$$

$$\frac{\partial J}{\partial net_\ell} = -\delta_\ell = -\delta_\ell$$

$$\frac{\partial J}{\partial net_j} = -\sum_{\ell=1}^{k} \delta_\ell \frac{\partial net_\ell}{\partial net_j}$$

$$= -\sum_{\ell=1}^{k} \delta_\ell \frac{\partial net_\ell}{\partial z_j} \frac{\partial z_j}{\partial net_j} \qquad z_j = \sigma(net_j)$$

$$= -\sum_{\ell=1}^{k} \left[\delta_\ell \left(\frac{\partial net_\ell}{\partial z_j} \right) z_j (1 - z_j) \right]$$

$$= -z_j(1-z_j) \sum_{\ell=1}^{k} \delta_\ell \frac{\partial net_\ell}{\partial z_j}$$

What is $\frac{\partial net_k}{\partial z_j}$ $\quad net_k = \sum_{j=1}^{M} w_{kj} z_j$

$$= w_{kj}$$

$$\frac{\partial J}{\partial net_j} = -z_j(1-z_j) \sum_{k=1}^{K} \delta_k w_{kj}$$

let $\delta_j = z_j(1-z_j) \sum_{k=1}^{K} \delta_k w_{kj}$

$$\frac{\partial J}{\partial net_j} = -\delta_j \longrightarrow \text{error at the hidden units.}$$

$$\boxed{\frac{\partial J}{\partial w_{jp}} = -\delta_j x_p} \longrightarrow p^{th} \text{ feature value of the input}$$

Update rule

$$w_{jp} = w_{jp}^{(old)} - \eta \frac{\partial J}{\partial w_{jp}}$$

$$= w_{jp}^{(old)} + \eta \delta_j x_p$$

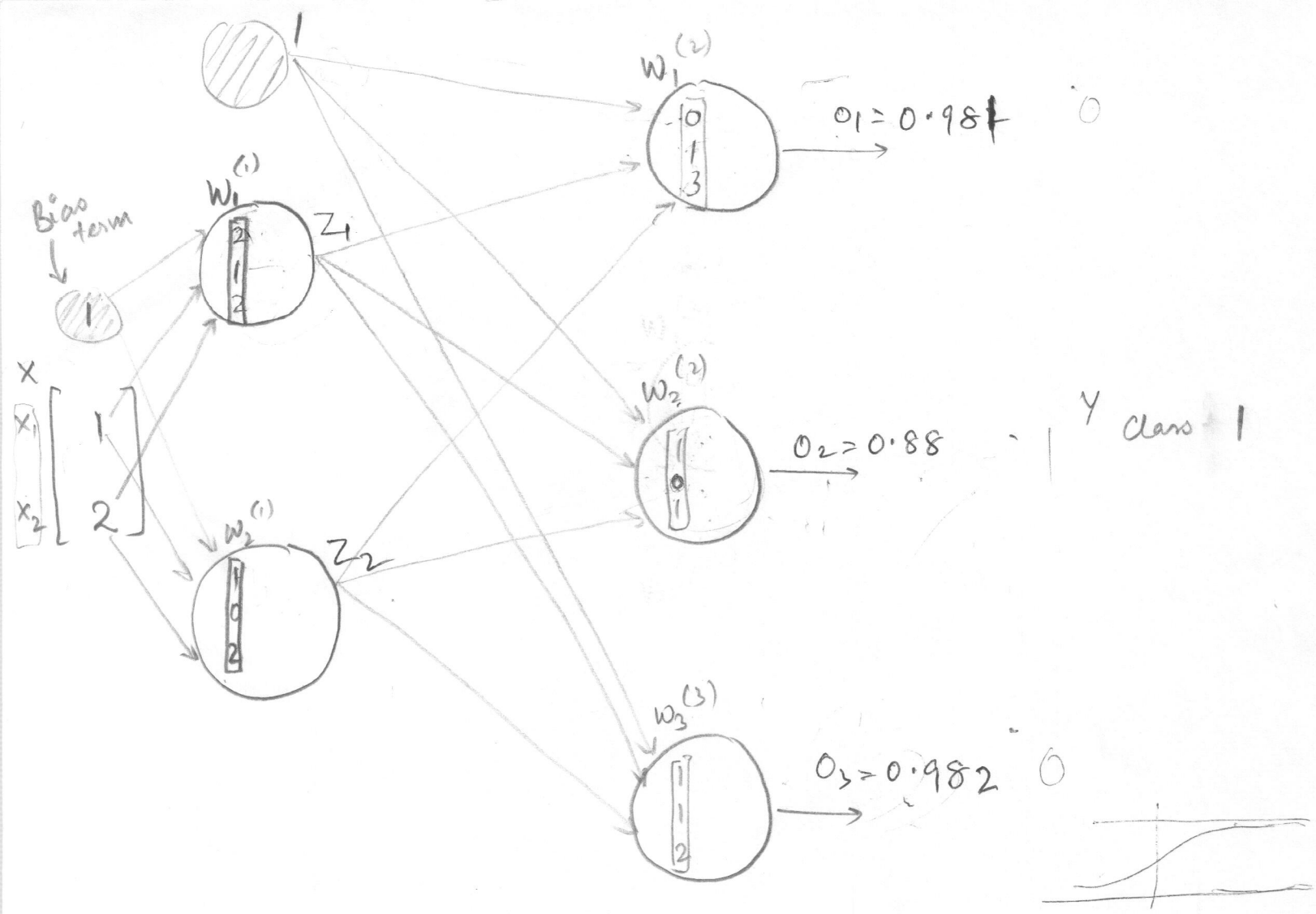
For the toy example:

output layer errors

$\delta_1 = o_1(1-o_1)(y_1-o_1) = 0.981 * (1-0.981) * (0-0.981)$

$\delta_2 = o_2(1-o_2)(y_2-o_2) =$

$\delta_3 = o_3(1-o_3)(y_3-o_3) =$



3-way classification, $\{1, 2, 3\}$

At first hidden unit: $net_1^{(1)} = w_1^{(1)T} x = 7$

$$z_1 = \sigma(net_1^{(1)}) = \sigma(7) = \frac{1}{1+e^{-7}}$$

At second hidden unit: $net_2^{(1)} = w_2^{(1)T} x = 5$

$$z_2 = \sigma(net_2^{(1)}) = \sigma(5) = 0.993$$

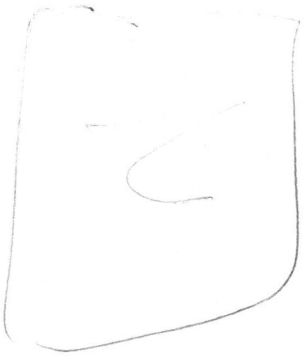
At first output unit: $net_1^{(2)} = w_1^{(2)T} z = 3.976$

$$o_1 = \sigma(net_1^{(2)}) = 0.981$$

second output unit $o_2 = \sigma(1.993) = 0.88$

third output unit $o_3 = \sigma(3.983) = 0.982$

What will be the error if $y = 2$



O_1 0.981

O_2 0.8.8

O_3 0.982

$$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \leftarrow y = 2 \quad \text{one-hot-encoding}$$

$$J = \frac{1}{2}\sum(O_e - y_e)^2 = \frac{1}{2}\left[0.981^2 + (-0.12)^2 + 0.982^2\right]$$