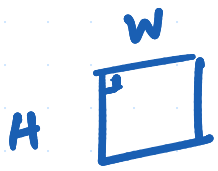
Lecture - 4

vectors → capital letters V, X, Y

Scalars → small letters v, x, y.

$$\mathcal{D} = \{x, y\}_{i=1}^{n}$$

$$x_1^{(1)} \begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \\ x_3^{(1)} \\ \vdots \\ x_n^{(1)} \end{pmatrix} \rightarrow y_1^{(1)} \begin{pmatrix} y_1^{(1)} \\ y_2^{(1)} \\ y_3^{(1)} \end{pmatrix}$$

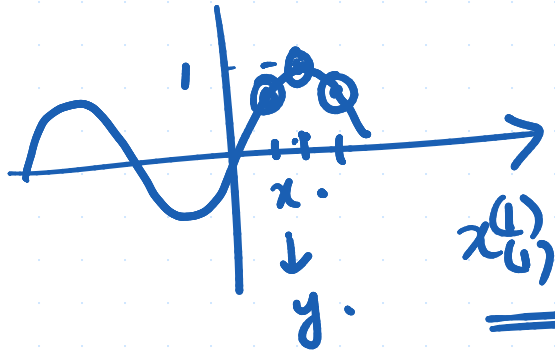


grayscale.

0-255 → flatten.

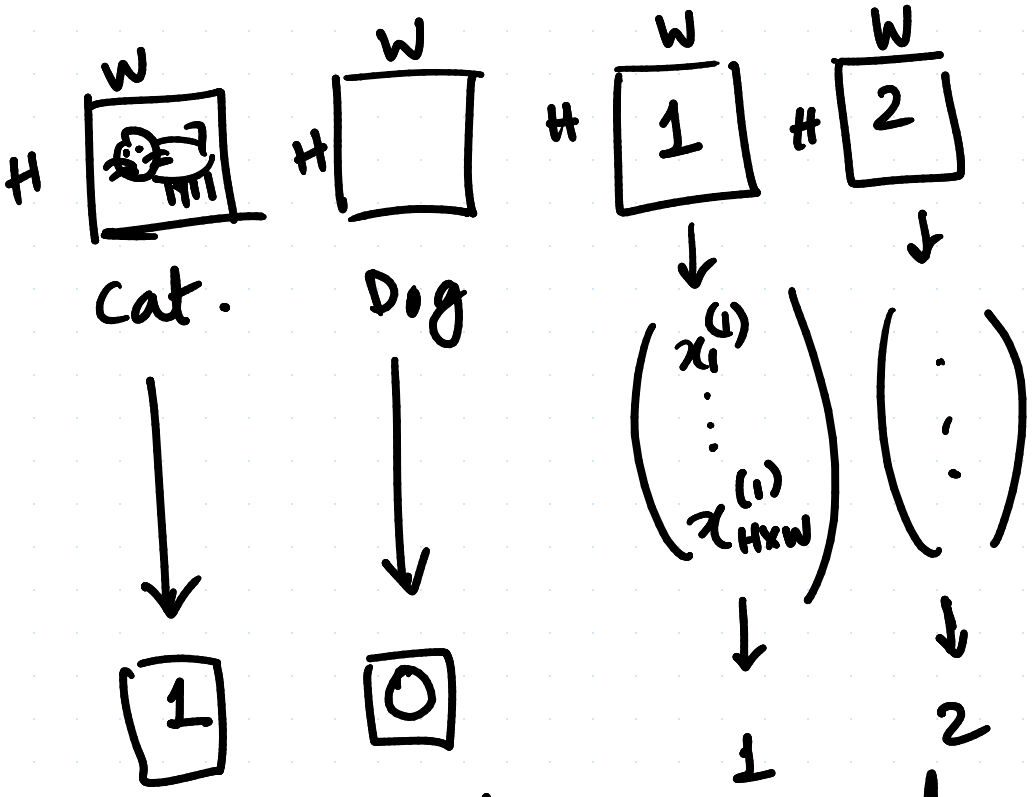
$$\begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_{H \times W}^{(1)} \end{pmatrix}$$

input using vectors.

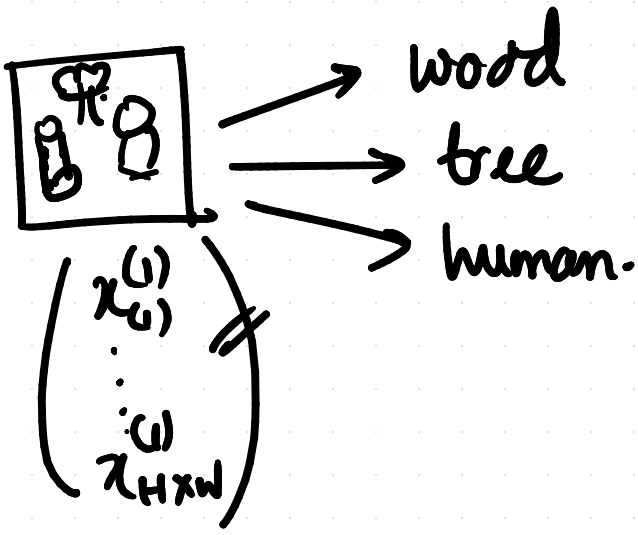


Scalar input, scalar o/p.

$x_{(1)}^{(1)} \rightarrow y_{(1)}^{(1)}$.



output is scalar, i.e; single class.



(1, 2, 3)

x^i → i^{th} input vector.

o^i → i^{th} output scalar

net^i → $W.x^i$ (n - input/output).

Observations.

W → weight vector.

W & x^i has m+1 components.

$$\boxed{W} = \langle w_m, w_{m-1}, \ldots, w_2, w_1, w_0 \rangle$$

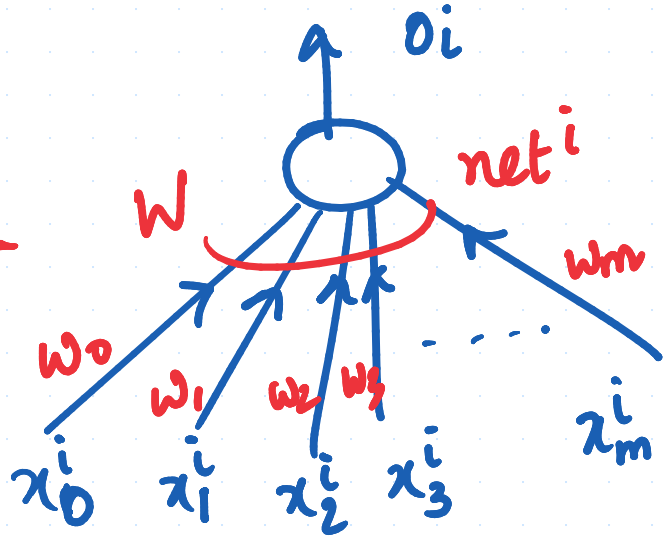
$$x^i = \langle x^i_m, x^i_{m-1}, \ldots, x^i_2, x^i_0 \rangle.$$

$$net_i = net^i$$

Derivative of Sigmoid.

$$o^i = \frac{1}{1+e^{-net^i}}$$

$$net^i = \sum_i \sum_j x^i_j w_j$$



$$o^i = \frac{1}{1+e^{-net_i}}$$ → for ith input.

$$\ln o^i = -\ln\left(1+e^{-net_i}\right)$$ $\boxed{\ln = \log_e}$

$\frac{\partial}{\partial net_i}$ on both sides

$$\frac{1}{o^i}\frac{\partial o^i}{\partial net_i} = -\frac{1}{\left(1+e^{-net_i}\right)}\left(-e^{-net_i}\right)$$

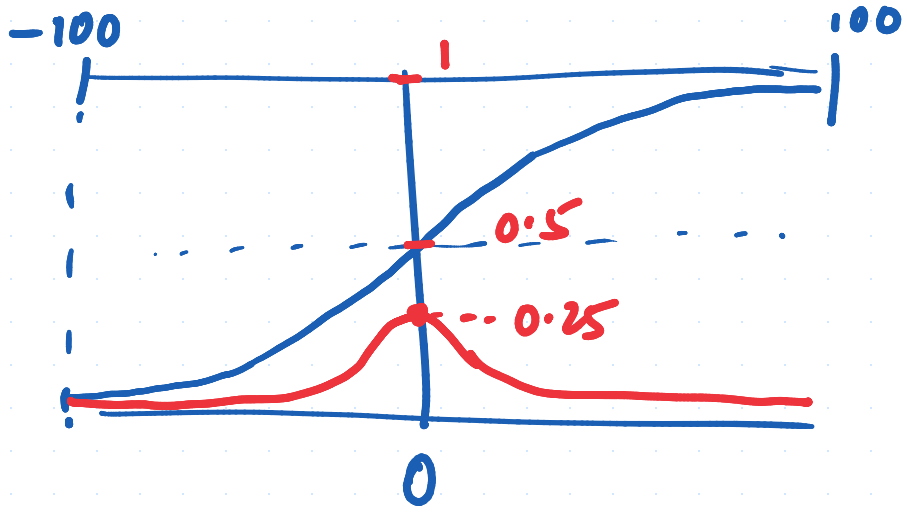
$$= \frac{e^{-net_i}}{1+e^{-net_i}} = (1-o^i)$$

$$\frac{1}{o^i}\frac{\partial o^i}{\partial net_i} = (1-o^i)$$

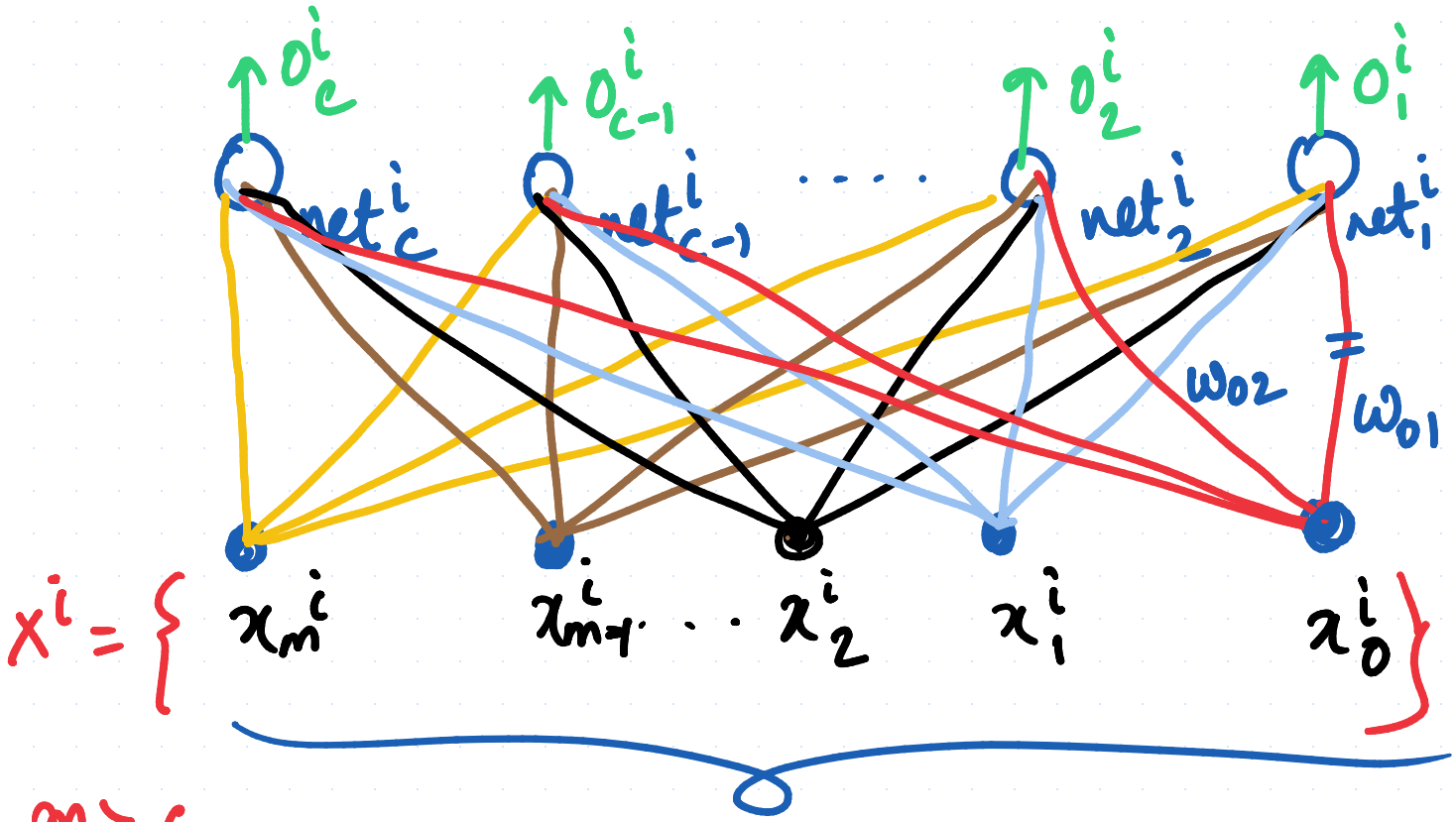
$$\boxed{\frac{\partial o^i}{\partial net_i} = o^i(1-o^i)}$$

Derivative of sigmoid

Maximum value of derivative of sigmoid = 0.25



Softmax using FCNN, DNN, layer Dense



$m > c$.

MNIST → 28×28 → 784 → m

classes → (0-10) → 10 → c

$m >> c$

Output for class c, c: 1 to C.

$$O_c^i = S(NET^i)_c = \frac{e^{net_c^i}}{\sum_{k=1}^{C} e^{net_k^i}}.$$

$\mathcal{D}$

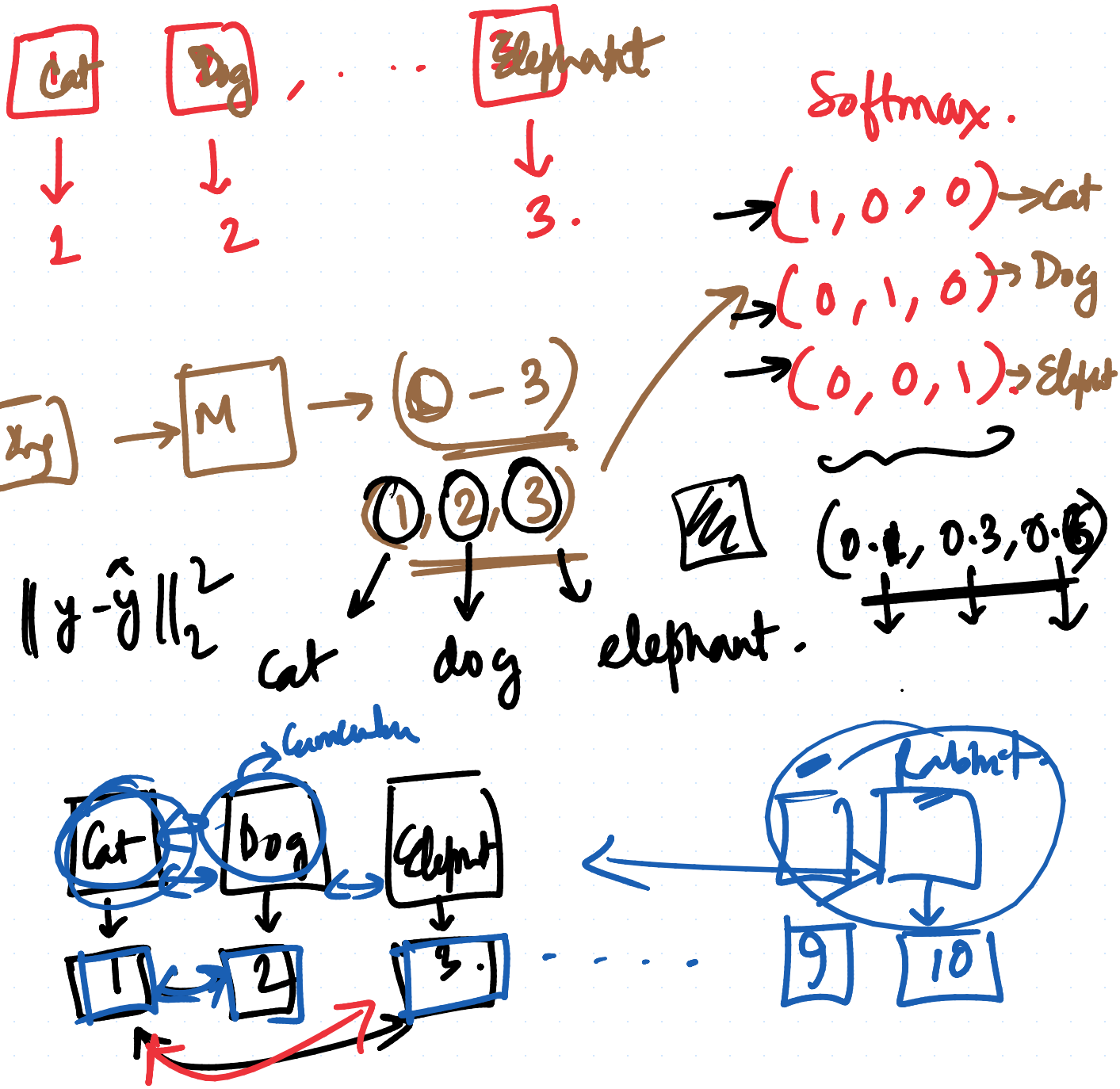
$[x_0^0, x_1^0, \ldots, x_m^0] \rightarrow [O_0^0, O_1^0, \ldots, O_C^0]$

$[x_0^1, x_1^1, \ldots, x_m^1] \rightarrow [O_0^1, O_1^1, \ldots O_C^1]$

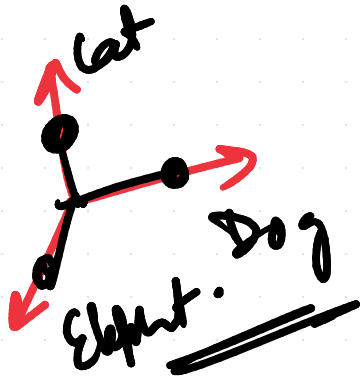
$\vdots$

$[x_0^N, x_1^N, \ldots, x_m^N] \rightarrow [O_0^N, O_1^N, \ldots O_C^N].$

One-hot vectors / encoding



How to compare kilometers with Kgs.



Softmax

$$\begin{Bmatrix} -1 \\ 1 \\ 0 \end{Bmatrix} \rightarrow \begin{Bmatrix} 0.02 \\ 0.5 \\ 0.43 \end{Bmatrix} \xrightarrow{\text{argmax}} \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix}$$

values are between 0 & 1

& Sum of all the values = 1.

Soft plus

$$o_c^i = \frac{e^{net_c^i}}{\sum_{k=1}^{c} e^{net_k^i}}$$, ith i/p pattern.

$$\Rightarrow \ln o_c^i = \log_e e^{net_c^i} - \log\left(\sum_{k=1}^{c} e^{net_k^i}\right)$$

Derivative:- w.r.t. net_c^i we get:-

$$\frac{1}{o_c^i}\frac{\partial o_c^i}{\partial net_c^i} = 1 - \frac{e^{net_c^i}}{\sum_{k=1}^{c} e^{net_k^i}}$$

Case 1:- when c & o & net are the same

$$\frac{1}{o_c^i}\frac{\partial o_c^i}{\partial net_c^i} = 1 - o_c^i.$$

$$\boxed{\frac{\partial o_c^i}{\partial net_c^i} = o_c^i(1 - o_c^i)}$$

Case 2: when c' and net_c^i are diffnt from c & o

$$\frac{1}{o_c^i}\frac{\partial o_c^i}{\partial net_c^i} = 0 - \frac{e^{net_{c'}^i}}{\sum_{k=1}^{c} e^{net_k^i}} = -o_{c'}^i$$

$$\frac{\partial o_c^i}{\partial net_{c'}^i} = -o_c^i o_{c'}^i$$

Maximum Likelihood & Cross Entropy loss:-

Q) How did cross entropy loss come to existence?

input → R.V.

We are interested in modelling probability.

$p(o^i | x^i)$ here o^i = output vector

x^i = input vector.

o_c^i → component of o^i → probability of x^i belonging to the class $C = \{1, 2, \cdots C\}$

C - components, where 1 is redundant.

prob of class c = $1 - \sum$ prob. (class $\neq c$).

<0.1, 0.6, 0.3>

$1 - (0.6 + 0.1)$

$\Rightarrow 1 - 0.7$

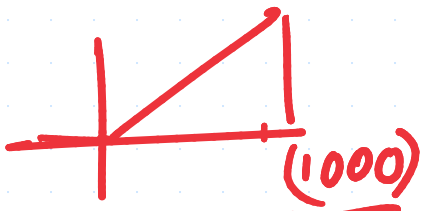
(C-1).

2-class → sigmoid } activation functions.

>2 → softmax }

(0 - 100) → : what activation?

sigmoid



(0-1) → x100.

(0 - 100)

(5, 95)

(0 – 1) → Sigmoid.

(0 – 1) × 90 + 5.

(0 – 90)

(5, 95)

(scaling & translating)

BCE

Likelihood of observation for two classes.

For N o/p – i/p pairs.

Likelihood.

$$\mathcal{L} = \prod_{i=1}^{N} (o^i)^{t^i} (1-o^i)^{(1-t^i)}$$

where target $t^i = 1$ or 0.

Binomial

① How likely an output is belonging to target 1 or 0.

$$\log L = \log\left(\prod_{i=1}^{N} (o^i)^{t^i} (1-o^i)^{(1-t^i)}\right)$$

$0.01 \times 0.7 \times 0.3 \times 0.2$

Likelihood - maximize.

ML → minimization algorithms.

$$-LL = \min \sum_{i=1}^{N} \{ t^i \log o^i + (1-t^i) \log(1-o^i) \}$$

$$= -\frac{1}{N} \sum_{i=1}^{N} \{ y^i \log(\hat{y}_i) + (1-y^i) \log(1-\hat{y}_i) \}$$

Kullback-Leibler Divergence.

$$D_{KL}(p \| q) = \sum_{k=1}^{n} p_k \log\left(\frac{p_k}{q_k}\right)$$

$$D_{KL}(p \| q) = H(p,q) - H(p)$$

cross entropy — entropy

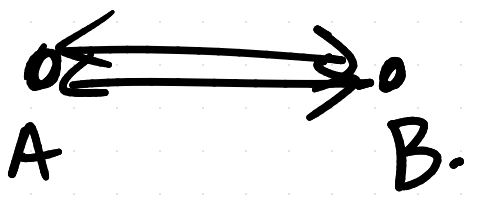
KL- divergence not KL distance

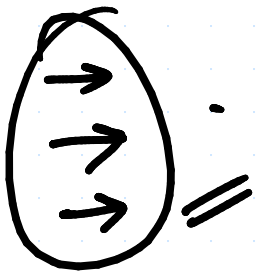
$p_k \log\left(\frac{p_k}{q_k}\right)$ $\quad p_k = 0.1$

$q_k = 0.7$.

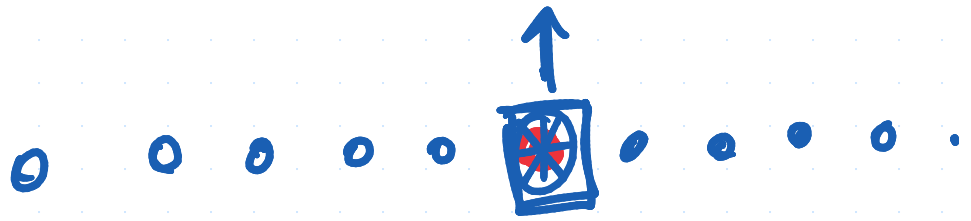
$0.1 \log\left(\frac{0.1}{0.7}\right) \neq 0.7 \log\left(\frac{0.7}{0.1}\right)$.

Not symmetric ; Hence this is not a distance.





Entropy ↑ I.G. ↓



highly likely.

Categorical Cross entropy derivation

For N no. of o/p o/p pairs :-

$$\mathcal{L} = \prod_{i=1}^{N} \prod_{k=1}^{C} (o_k^i)^{t_k^i} \quad ; t_k^i = 1/0.$$

For a pattern i, only one of t_k^i's is 1, rest are 0 :-

$$\max LL = \sum_{i=1}^{N} \sum_{k=1}^{C} t_k^i \log o_k^i .$$

$$\min -LL = -\sum_{i=1}^{N} \sum_{k=1}^{C} t_k^i \log o_k^i$$

For softmax as activation function we maximize the likelihood = minimizing the $-LL$ = cross entropy loss / Error.

Backprop & Gradient Descent.

(Greedy algorithm)

Always in the direction of reducing errors.

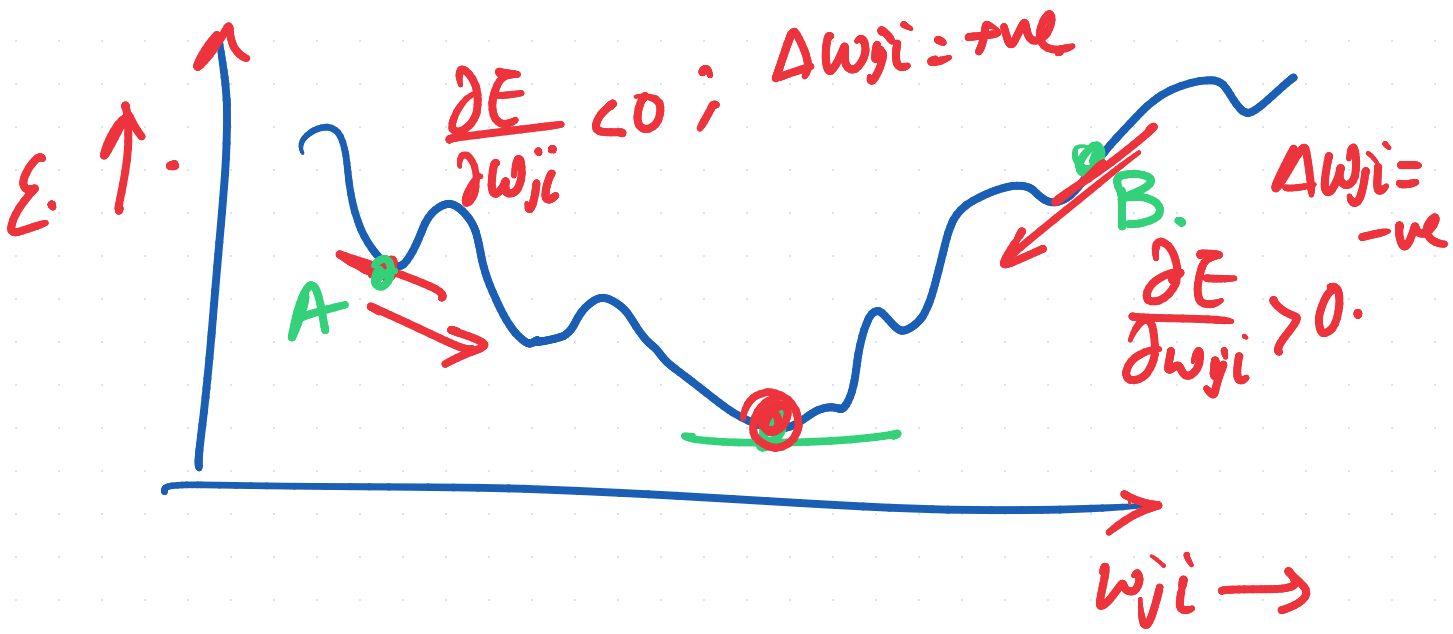
$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}}$$

→ change in weight.

η = learning rate.

E = loss function.

w_{ji} = weight of connection from the i^{th} neuron to the j^{th} neuron.


E ↑
∂E/∂w_ji < 0 ;
Δw_ji = +ve
A
B
Δw_ji = -ve
∂E/∂w_ji > 0.
w_ji →