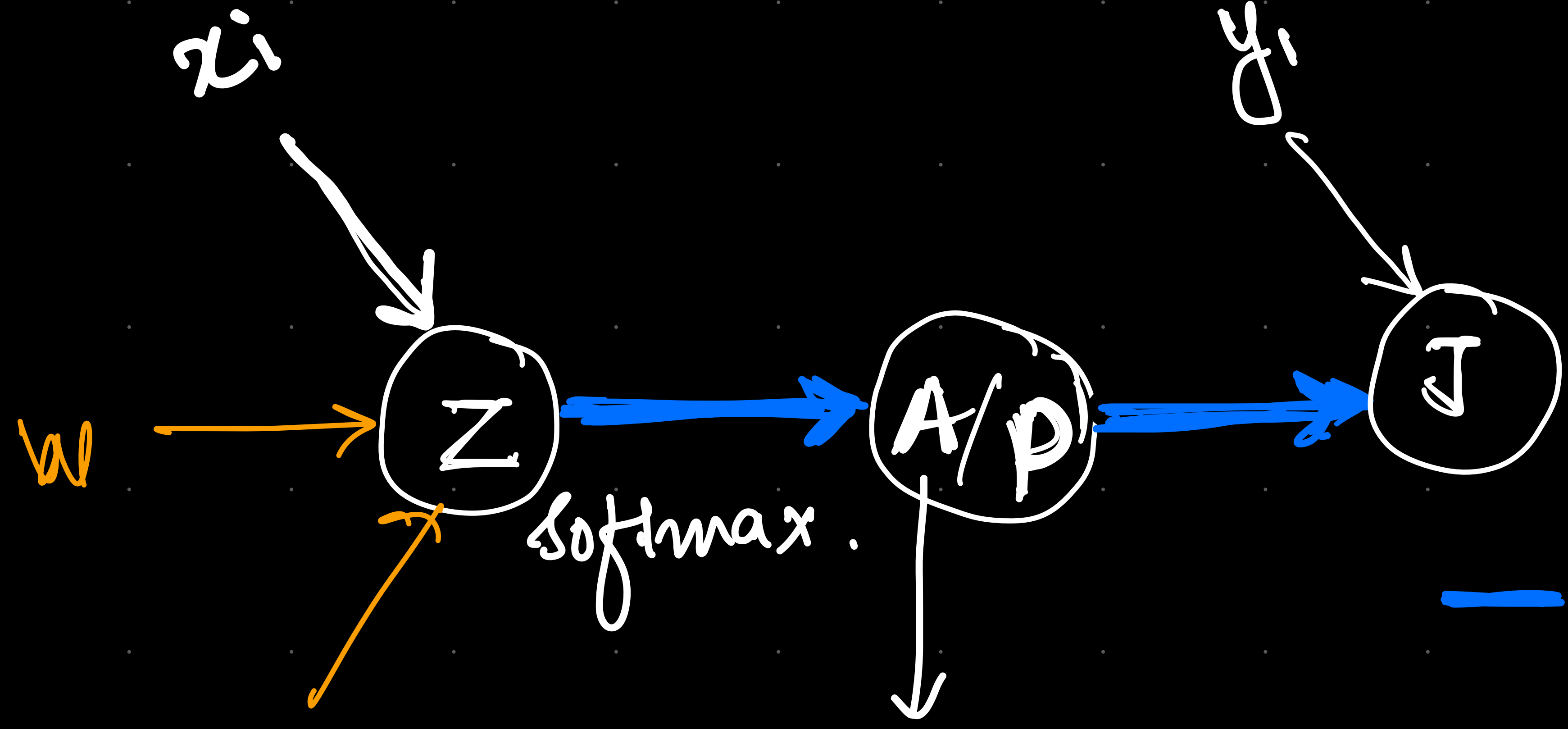
NN Lecture - 3

Backward Prop - Softmax classifier.

N - layer Neural Network.

Computational graph for softmax classifier

K = # classes.



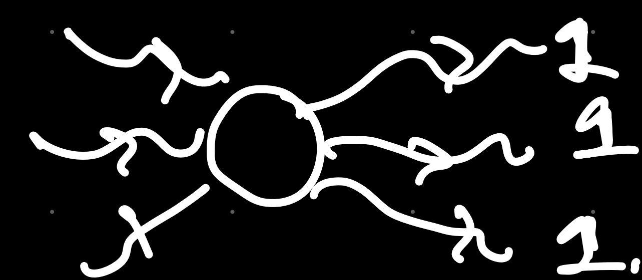
$$\text{CE loss} = \sum_{1}^{K} y_k \log p_k$$

— Forward. Propogation.

xw+b.

A - $\boxed{\text{Activation}}$

P - probability

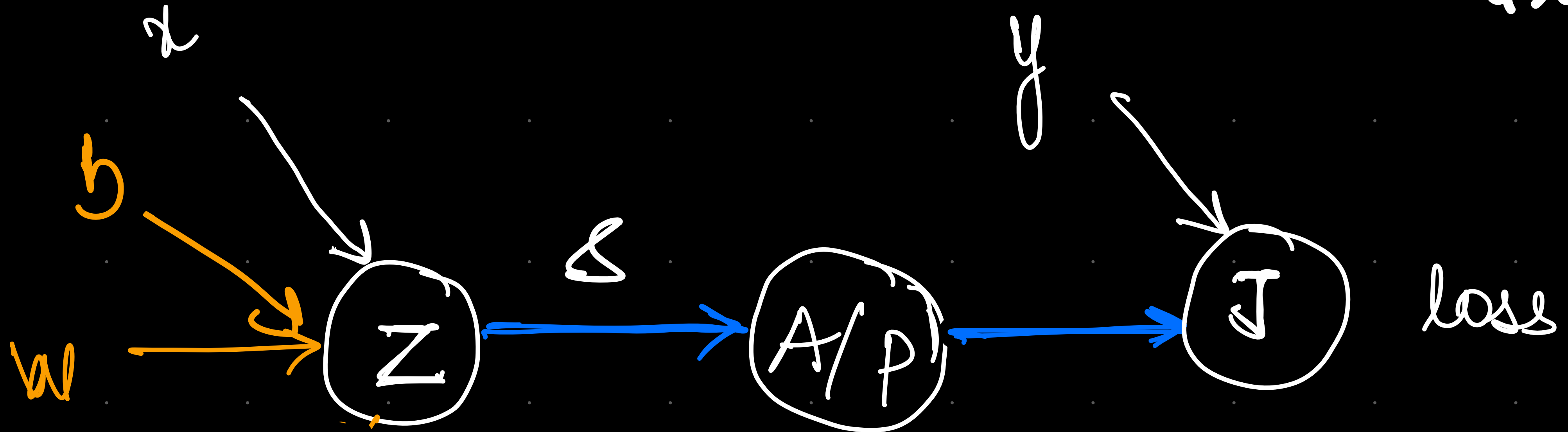


Activated.

Computational graph.

Backward Propogation $\left(\frac{\partial J}{\partial w}, \frac{\partial J}{\partial b}\right)$

Gradients.



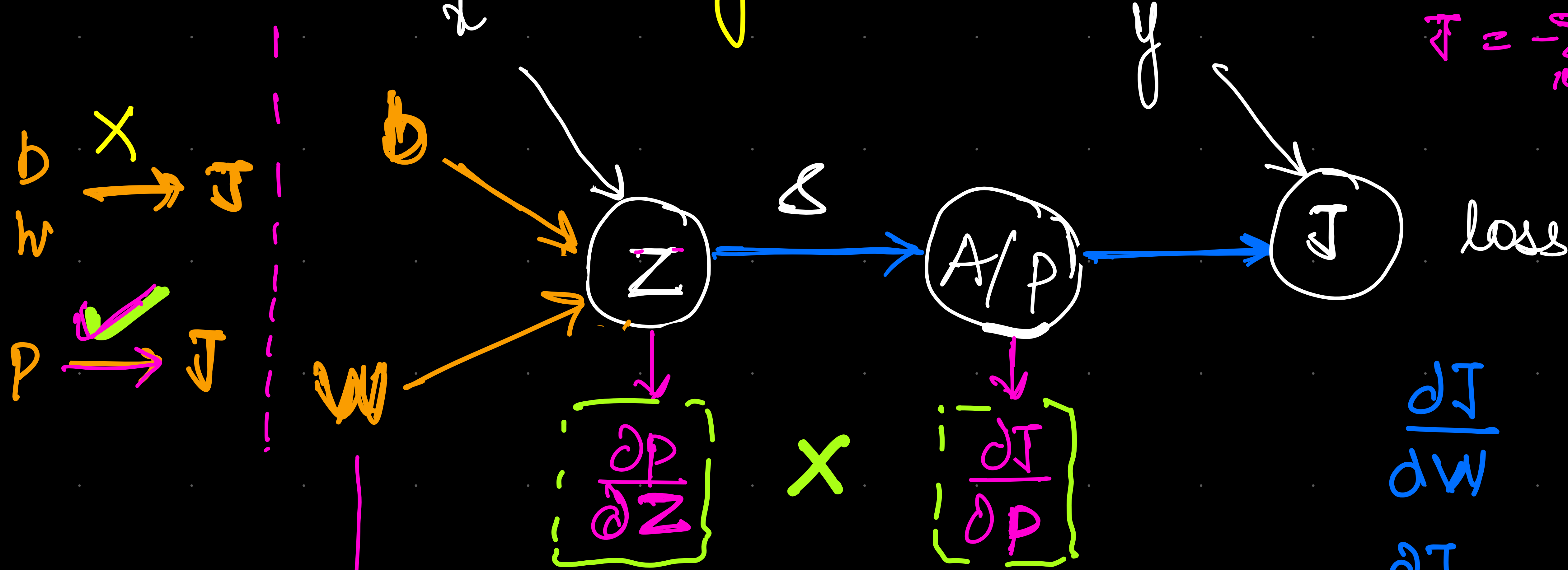
Goal : calculate $\frac{\partial J}{\partial w}, \frac{\partial J}{\partial b}$

Backward Propogation

$z = wx + b$

$p = e^{z} / \sum e^{z}$

$J = -\sum_{k} y_k \log p_k$



$\boxed{\frac{\partial z}{\partial w}} \times \boxed{\frac{\partial J}{\partial z}}$

$\frac{\partial J}{\partial w}$ ★★★ Goal-1 computed.

$\frac{\partial J}{\partial w}$

$\frac{\partial J}{\partial b}$.

Backward Propogation chain rule

Chain rule for $\partial J/\partial W$:

$$\frac{\partial J}{\partial W} = \frac{\partial J}{\partial p}\frac{\partial p}{\partial z}\frac{\partial z}{\partial W}$$

Chain rule for $\partial J/\partial b$:

$$\frac{\partial J}{\partial b} = \frac{\partial J}{\partial p}\frac{\partial p}{\partial z}\frac{\partial z}{\partial b}$$

Backward Propogation - shorthands

Chain rule for $\partial J/\partial W$

$$\frac{\partial J}{\partial W} = \overset{1}{\frac{\partial J}{\partial p}} \; \overset{2}{\frac{\partial p}{\partial z}} \; \overset{3}{\frac{\partial z}{\partial W}}$$

$\partial J/\partial \square$ $\not\!\!/$ x

Rule: All derivatives for $\boxed{\partial J/\partial \square}$ can be written as $\boxed{\partial \square}$

$$\downarrow \qquad \downarrow \qquad \downarrow$$

∂W ✓ $\qquad \partial p$ ✓ $\qquad \partial z$ ✗ $\qquad \partial W$ ✗

Backward Propogation - shorthands

chain rule for $\partial J / \partial w$

$$\frac{\partial J}{\partial w} = \boxed{\frac{\partial J}{\partial p} \frac{\partial p}{\partial z}} \frac{\partial z}{\partial w}$$

$$\frac{\partial J}{\partial w} = \frac{\partial J}{} \frac{\partial z}{}$$

★★★

$$\boxed{\partial w = \partial z \; \partial z / \partial w}$$

→ Very easy.

$$\frac{\partial z}{\partial w} = \frac{\partial (wx+b)}{\partial w}$$
$$= x$$

This is only for coding.

↓ V.V. Nice solution.

Backward Propogation - ∂z, $\partial z/\partial w$ ***

① $\frac{\partial z}{\partial w} = \frac{\partial(wx+b)}{\partial w} = x$

$dw = (p-y)x$

Near solution.

② $\partial z = P_k - \boxed{I(k=y)}$ ← Indicator function.

$k \neq y \rightarrow 0$

$k = y \rightarrow 1$

Probab. of class (k)

code :

$$\boxed{\partial z = p - y}$$

Prob vector   Ground. T. vector.

$$\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

** Residual (like)

How? Derivation in post-read (Difficult)

Not expected.

But why are we subtracting 1 ?

Interestingly

Ground Truth $y = [0 \; 1 \; 0]$

$$\begin{array}{cccc} & p_1 & p_2 & p_3 \\ \text{Predicted probs} = & [0.2, & 0.3, & 0.5] \end{array}$$

$\boxed{dz = p - y}$ ★★★

$\frac{\partial J}{\partial z} = \boxed{\partial z} = [0.2, -0.7, 0.5]$

error.

① $p_2 \uparrow \rightarrow \partial z_2 \downarrow$, $p_2 \downarrow \rightarrow \partial z_2 \uparrow$

② $p_1 \uparrow \rightarrow \partial z_1 \uparrow$, $p_1 \downarrow \rightarrow \partial z_2 \downarrow$

(or p_3)

Lets calculate ∂W

Input Matrix.

$$\boxed{dW = \partial Z \cdot X}$$

$\partial Z \longrightarrow p - y = m \times n - m \times n = (m, n) = (300, 3)$

$X \longrightarrow (m, d) \Rightarrow (300, 2)$

$\partial W \longrightarrow (d, n) \longrightarrow (2, 3)$ #classes

shape.

$\parallel$

W

features

" Neuron

$\boxed{W = W - \alpha\, \partial W}$

same shape.

$(2, 300)(300, 3) \longrightarrow (2, 3)$

$$\boxed{X^T \cdot \partial Z = \partial W}$$

Lets calculate ∂b

Nice

$$\partial b = \partial Z \cdot \partial Z / \partial b$$
$$= (p-y) \cdot 1$$

$$\frac{\partial Z}{\partial b} = \frac{\partial(xw+b)}{\partial b} = 1$$

$$\boxed{\partial b = \partial Z}$$

[0.1 0.5 0.7] → 2 index

argmax → index

0, 1, 2

$\partial Z \rightarrow p-y = (300, 3) = (m, n)$

$\partial b \rightarrow (1, n)$

shape

$b \; (1, n)$

Matrix

$(1, n) = (m, n)$

For m samples.

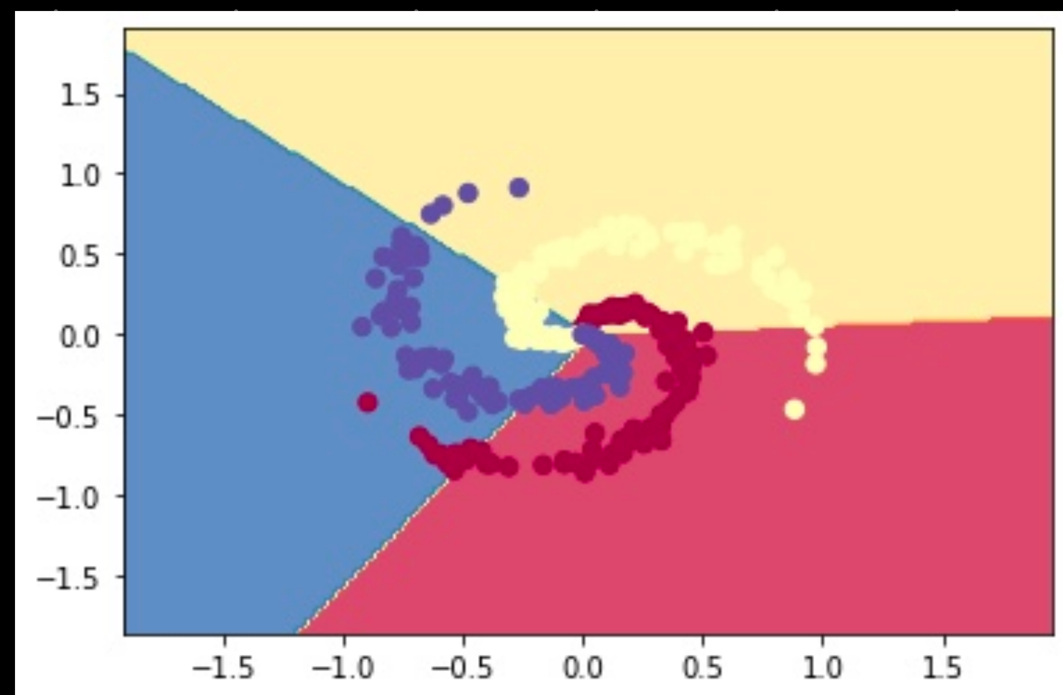
m - 1D array.

Mean of the solution.

★★★

∂b = np.sum(∂Z, axis=0, keepdims=True)

Output of softmax classifier



Task-I

✓ : Adapted LRVs to work for multi-class classification

✗ : Learning non-linear decision boundary