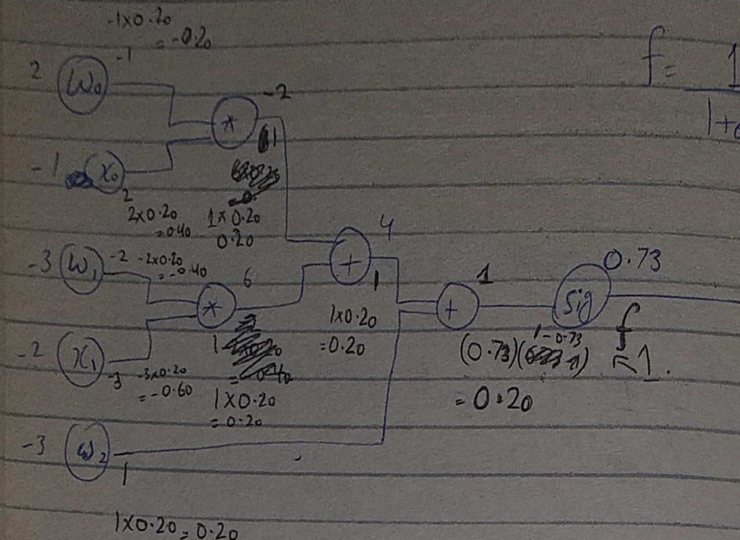
11-10-2021 Monday

week # 7 Lecture # 9

$$f = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



How to Code it?

w [2 -3 -3]

x [-1 , -2]

Lambda function:

Unnamed function

~~def sig(x):~~

~~return sig = lambda x =~~ $\frac{1}{1+e^{-x}}$

def sig(x):

return $\frac{1}{1+np.exp(-x)}$

N

$dot = w[0] \times x[0] + w[1] * x[1] + w[2]$

$f = sig(dot)$ // Until Here it is forward Pass

$$\boxed{\begin{array}{l} dx = \frac{df}{dx} \text{ final derivative} \\ dw = \frac{df}{dw} \end{array}}$$

$ddot = f * (1-f)$ // $\frac{df}{ddot}$

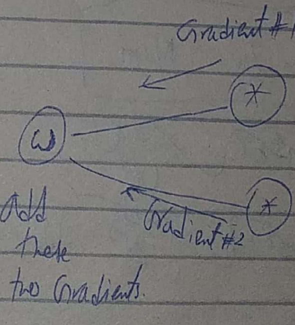
$dw[0] = x[0] * ddot$ // $\frac{df}{dw} = \overset{ddot}{\boxed{\frac{df}{ddot}}} \times \overset{x[0]}{\boxed{\frac{ddot}{dw}}}$

$dw[1] = x[1] * ddot$

$dw[2] = 1 * ddot$

$dx[0] = w[0] * ddot$

$dx[1] = w[1] * ddot$



Example.

$$f(x,y) = \frac{\sigma(y) + x}{\sigma(x) + (x+y)^2}$$

Solution.

$x = 3$

$y = -4$

T

sigy = sig(y)

sigx = sig(x)

Num = sigy + x

~~Denum = sigx + sqrt(x+y)~~

xpy = x+y

sxpy = xpy ** 2

Den = sigx + sxpy

Den_Inv = 1/Den

f = Num * Den_Inv

$\boxed{\text{dNum} = \text{Den\_Inv}}$ //Gradients calculated //here

$\boxed{\text{dDen\_Inv} = \text{Num}}$

~~dNum1x = sigy~~

~~dNum2y = sig(y) * (1 - sigy)~~

~~dden_Inv = -1/sqr(den)~~ ~~d_Den_Inv =~~

$$dden = \frac{-1}{(den)^2} * dden\_Inv$$

$$\boxed{dden = \frac{-1}{(den)^2} * Num}$$

Local Gradient w.r.t ~~in~~ input

H

$\cancel{dsigx} = \frac{dden}{dsigx} = \frac{d}{dsigx}\left(sigx + \cancel{ss}\, s\cancel{q}xpy\right)$

$\boxed{dsigx = 1 * dden}$

$dsxpy = \frac{dden}{dsxpy} = \frac{d}{dsxpy}\left(sigx + sxpy\right)$

$\boxed{dsxpy = 1 * dden}$

$dxpy = \frac{dsxpy}{dxpy} \cancel{df} = \frac{d}{dxpy}\left(xpy * *2\right)$

$\boxed{dxpy = 2\, xpy \times dsxpy}$

$dx = \frac{dxpy}{dx} = \frac{d}{dx}\left(x+y\right)$

$\boxed{dx = 1\times}\, dxpy$

$dy = \frac{dxpy}{dy} = \frac{d}{dy}\left(x+y\right)$

$\boxed{dy = 1 \times dxpy}$

$dx \mathrel{+}= \cancel{\{}\ sigx * (1 - sigx) * dsigx$

$$d\,sigy = \frac{d\,num}{d\,sigy} \quad \frac{d}{d\,sigy}(sigy + x)$$

$$\boxed{d\,sigy = 1 * d\,Num}$$

$$dx = \frac{d\,Num}{dx} = \frac{d}{dx}(sigy + x)$$

$$\boxed{dx \mathrel{+}= 1 * d\,Num}$$

$$dy = \frac{d\,sigy}{dy} = \frac{d}{dy}(sigy)$$

$$\boxed{dy \mathrel{+}= sig(y) * (1 - sigy) * d\,sigy}$$

+1

In General:

function = local variables

d local variable = derivative w.r.t to local variables in function * derivative of function.

L

```
for i in range (epochs):
    dw = Evaluate_Gradient (X, Y, w)
    w = w - αdw
    print (loss)
```

```
for i in range (epochs):
for ix in X:
    dw = evaluate_Gradient (x, y, w)
    w = w - αdw
    Print (loss)
```

Stochastic Gradient Decent
Slow.
More fluctuations in weights.

```
for i in range (epochs):
    dw = evaluate_Gradient (X, Y, w)   // take avg of
                                       // Gradients
    W = w - αdw
                                       // dw = avg of Gradients
    Print (loss)                       across all the imgs
```

Batch Gradient Decent

Slower, but gives good Gradients.

~~for~~ 2 ~~one~~

Mini Batch ~~&~~ Gradient Decent:

e.g 256 all images

32 Size batches

32, 32, 32, 32, ~~32~~

32, 32, 32, 32

Calculate gradients in 32 imgs & then update one time in weights.

Shuffle the images