Lec 7. COMPUTATION GRAPH

why we have BackProp

Forward pass → Forward Propagation

then Backward Propagation.

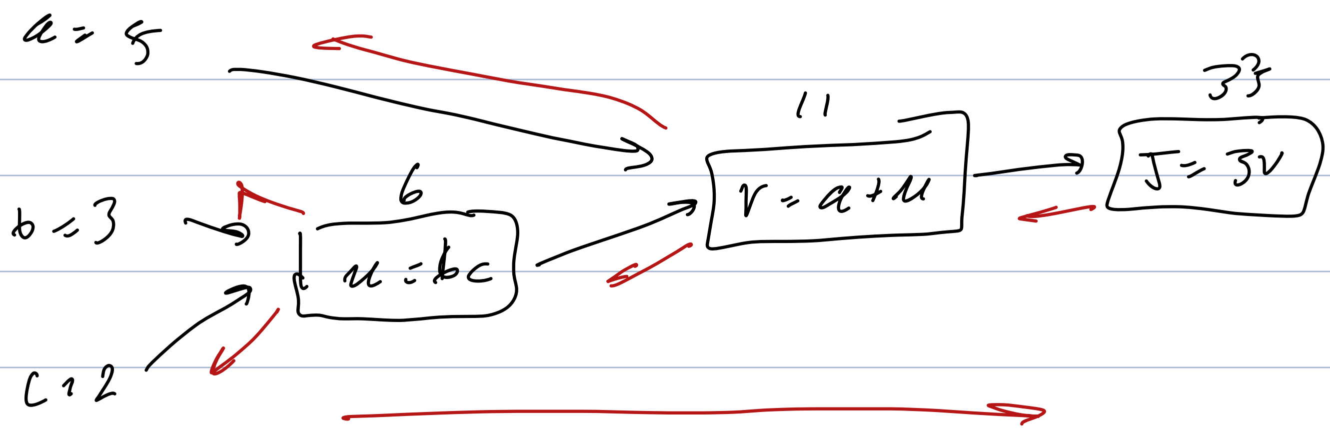
Computation Graph example

$$J(a,b,c) = \underbrace{3(\underbrace{a + \underbrace{bc}_{u}}_{v})}_{J}$$

$u = bc$

$v = a + u$

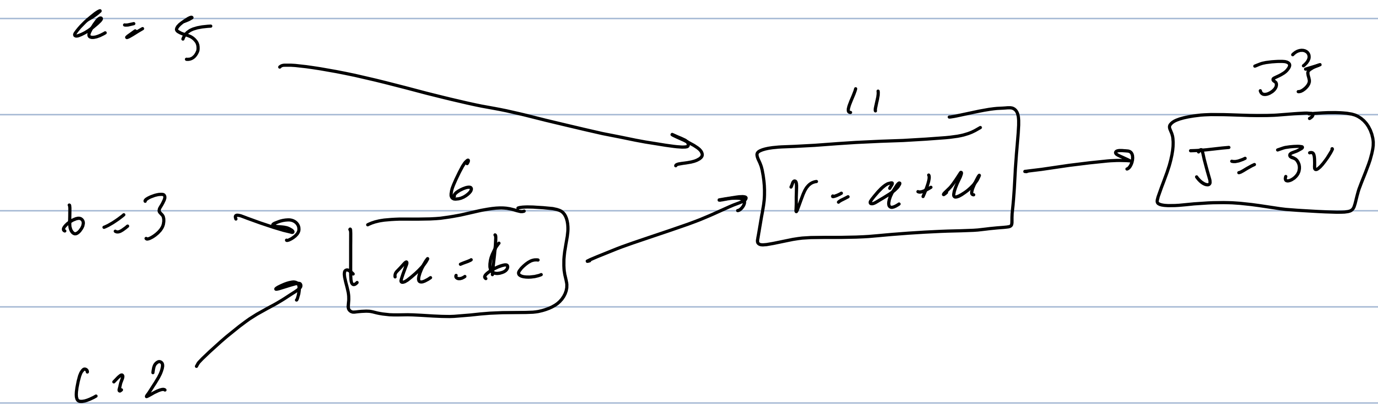
$J = 3v$



Left to Right, we use to compute J
Pass.

Compute Derivative Do a Right to Left Pass

Sec 8 Derivatives w/ a Computation Graph



$\frac{dJ}{dv} = ?$ $\quad J = 3v \quad \frac{dJ}{dv} = 3$

$$\frac{dJ}{da} = \frac{\partial v}{\partial a} = 3(a + u)$$

Chain Rule. $a \to v \to J$.

$$\frac{dJ}{da} = \frac{\partial J}{\partial v} \frac{\partial v}{\partial a}.$$

$$\frac{dJ}{dc} = \frac{\partial J}{\partial v} \cdot \frac{\partial v}{\partial u} \cdot \frac{\partial u}{\partial c}$$

$$= (3)(1)(b)$$

$$= 6$$

For Code → Final var we care about → last node in computation graph is J.

$$\frac{d\ \text{Final output var}}{d\ \text{var}} =$$

$\frac{dJ}{d\text{var}}$ too long, always w.r.t J.

"d var"

$$\frac{dJ}{du} = \frac{\partial J}{\partial v} \cdot \frac{\partial v}{\partial u} = (3)(1) = 3.$$

$$\frac{dJ}{db} = \frac{\partial J}{\partial v} \cdot \frac{\partial v}{\partial u} \cdot \frac{\partial u}{\partial b}$$

$$= (3)(1)(c)$$

$$= 3c \quad \text{if } c = 2 \Rightarrow 6.$$

$$\frac{dJ}{dc} = \frac{\partial J}{\partial v} \cdot \frac{\partial v}{\partial u} \cdot \frac{\partial u}{\partial c} = (3)(1)(b)$$

$$= (3)(1)(3) = 9.$$

So backwards prop allows for the chain rule to compute the derivatives

Lec 9 Logistic Regression Gradient Descent

$$z = w^T x + b$$

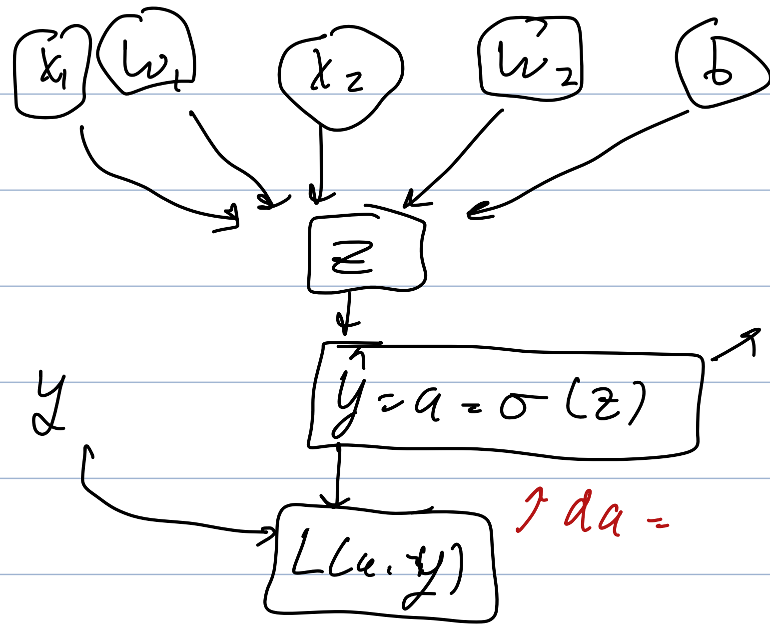
$$\hat{y} = a = \sigma(z)$$

$$\mathcal{L}(a, y) = -(y \log(a) + (1-y)\log(1-a))$$

$\boxed{\text{2 features}}$ x_1 and x_2.

$x_1 \quad x_2$

$\Rightarrow z = w_1 x_1 + w_2 x_2 + b$



$$\frac{da}{dz} = \frac{1}{1+e^{-z}} \Rightarrow a(1-a)$$

Modify w and b to minimize $\mathcal{L}(a, y)$.

da.

$$\frac{\partial L(a,y)}{\partial w_1} = \frac{\partial(L(a,y))}{\partial a}\,\frac{\partial a}{\partial z}\,\frac{\partial z}{\partial w_1}$$

the weight input depend on + activation of last neuron.

$$= -\left(\frac{y}{a} + \frac{(1-y)}{1-a}(-1)\right)\left(\sigma'(z)\right)(x_1)$$

$$= \left(\frac{-y}{a} + \frac{1-y}{1-a}\right)\left(\sigma'(z)\right)(x_1)$$

$$= (a-y)(x_1)$$

$$\frac{dL}{\partial w_1} = \text{"}dw_1\text{"} = x_1 \cdot dz$$

$$\frac{dL}{dw_2} = "dw_2" = x_2 \cdot dz$$

$db = dz$.

α

One step of gradient descent W.R.T one example

Then $\rightarrow$ $w_1 := w_1 - \alpha dw_1$

$w_2 := w_2 - \alpha dw_2$

$b := b - \alpha db$

Lec 9. Gradient Descent on M Examples.

$$J(w,b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(a^{(i)}, y^{(i)})$$

$\leftarrow$ Prediction $\hat{y}^{(i)}$

$a^{(i)} = \hat{y}^{(i)} = \sigma(z^{(i)}) = \sigma(w^T x^{(i)} + b)$

$dw^{(i)}, dw_2^{(i)}, db^{(i)}$ from before just on $(x^{(i)}, y^{(i)})$

$$\therefore \frac{\partial}{\partial w_1} J(w,b) = \frac{1}{m} \sum_{i=1}^{m} \underbrace{\frac{\partial}{\partial w_1} \mathcal{L}(a^{(i)}, y^{(i)})}_{\partial w_1^{(i)}}$$

$\hookrightarrow$ So this is just the avg of these steps, i.e. take into account all of them.

Log Reg on M examples

$J = 0, \quad dw_1 = 0, \quad dw_2 = 0, \quad db = 0.$

For $i = 1$ to m. → $O(m)$ (Sample)

$z^{(i)} = w^T x^{(i)} + b$

$a^{(i)} = \sigma(z^{(i)})$

$J += -[y^{(i)} \log a^{(i)} + (1 - y^{(i)}) \log (1 - a^{(i)})]$

$dz^{(i)} = a^{(i)} - y^{(i)}$

$dw_1 += x_1^{(i)} dz^{(i)}$ ↕ $n = 2$ → 2 features

$dw_2 += x_2 dz^{(i)}$ ↳ for loop

$db += dz^{(i)}$ $O(n)$ (features)

$J /= m$; avg.

$dw_1 /= m$

$dw_2 /= m$

$db /= m$

$\therefore \Rightarrow dw_1 = \frac{\partial J}{\partial w_1}$

↓ to Sum over the entire Set.

1 step of gradient descent.

$w_1 := w_1 - \alpha dw_1$

$w_2 := w_2 - \alpha dw_2$

$$\underbrace{b := b - \alpha\, db}$$

One step of gradient step, would have to repeat all of this for multiple steps, to lower the cost

VECTORIZATION TECHNIQUES TO GET RID OF EXPLICIT FOR LOOPS TO SPEED UP, ELSE WILL TAKE TOO LONG.

Check note on the derivative.