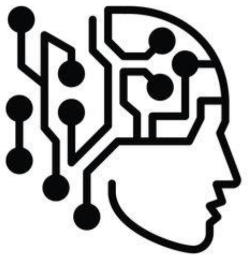


Artificial Intelligence

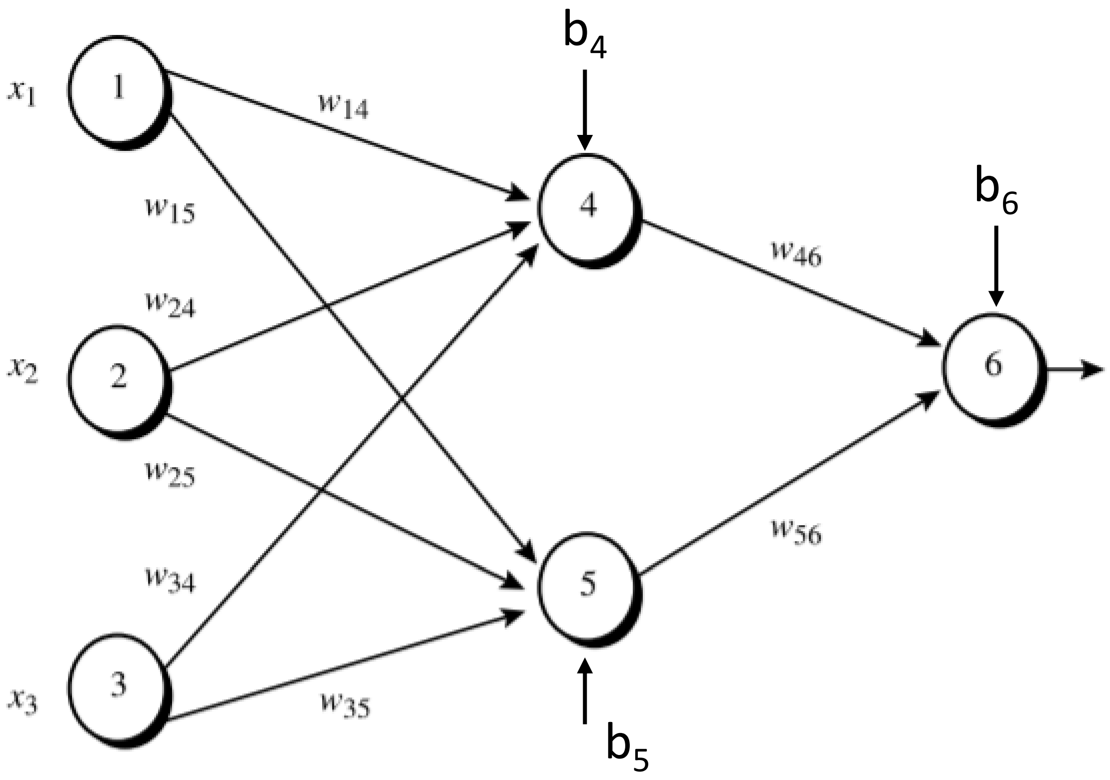
Neural Network

Artificial Neural Network Learning

- <u>How does a perceptron learn the appropriate weights?</u>

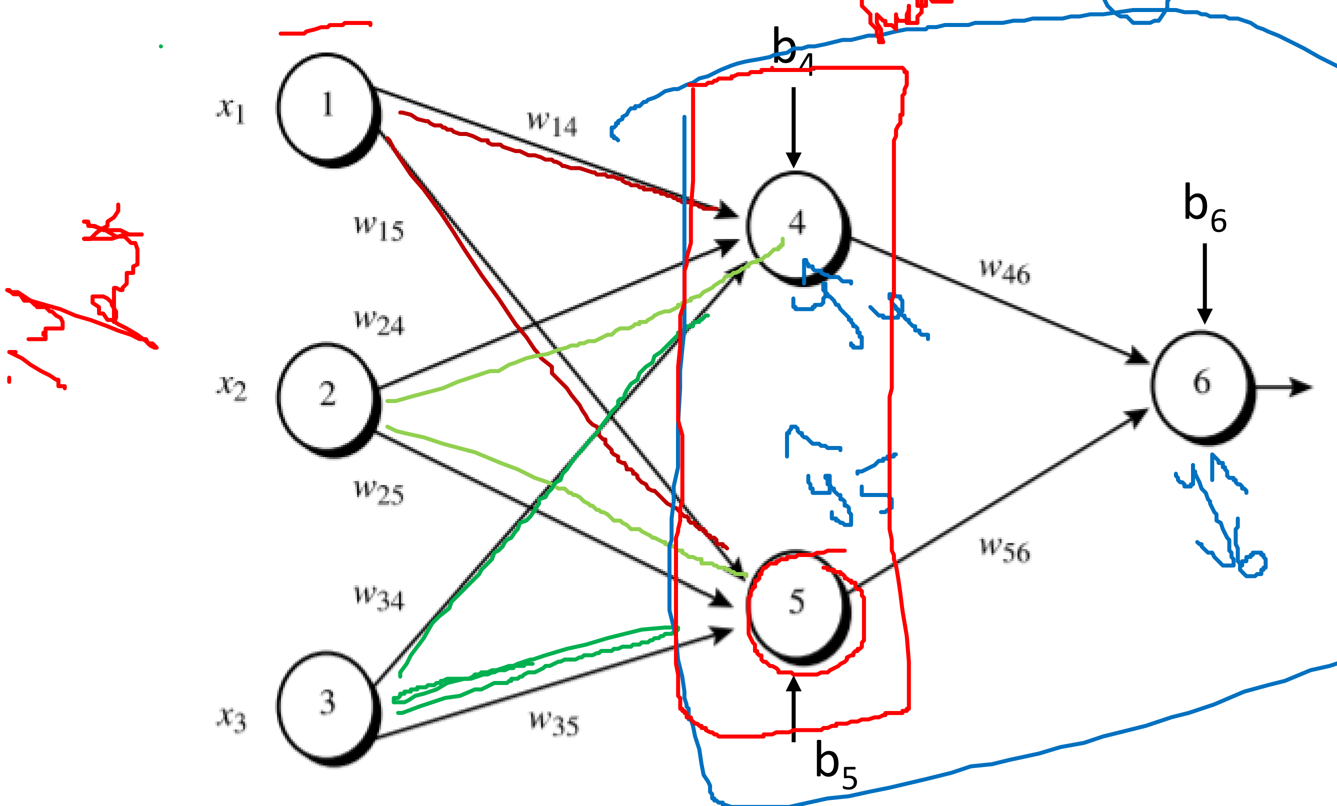
Example

x1	x2	x3	w14	w15	w24	w25	w34	w35	w46	w56	b4	b5	b6
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1



Example

x1	x2	x3	w14	w15	w24	w25	w34	w35	w46	w56	b4	b5	b6
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1



Feed-Forward

$$\sigma = \frac{1}{1+e^{-y}}$$

$$y_4 = (x_1 \cdot w_{14} + x_2 \cdot w_{24} + x_3 \cdot w_{34}) + b_4$$

$$= 1 \cdot 0.2 + 0 \cdot 0.4 + 1 \cdot (-0.5) + (-0.4)$$

$$\hat{y}_4 = \sigma(-0.7) \Rightarrow 0.332$$

$$y_5 = x_1 \cdot w_{15} + x_2 \cdot w_{25} + x_3 \cdot w_{35} + b_5$$

$$= 0.1$$

$$\hat{y}_5 = \sigma(0.1) = 0.524$$

$$y_6 = \hat{y}_4 \cdot w_{46} + \hat{y}_5 \cdot w_{56} + b_6$$

$$= 0.044$$

$$\hat{y}_6 = \sigma(0.00) = 0.5$$

Backpropagation

- Error = ½ (Target – output)^2 =>**Squared Error Function**
- Derivate of Squared Error Function= (Target – output)

- Sigmoid Function: σ
- Derivate of Sigmoid Function: σ (1-σ)

Backpropagation

- Error = ½ (Target – output)^2 =>**Squared Error Function**
- Derivate of Squared Error Function= (Target – output)
- Sigmoid Function: σ $= \frac{1}{1+e^{-y}}$
- Derivate of Sigmoid Function: σ (1-σ)

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

$2 \cdot \frac{1}{2} (T-O)$

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right] \\
&= \frac{d}{dx}\left(1+\mathrm{e}^{-x}\right)^{-1} \\
&= -(1+e^{-x})^{-2}(-e^{-x}) \\
&= \frac{e^{-x}}{\left(1+e^{-x}\right)^{2}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{(1+e^{-x})-1}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \\
&\quad \cdot \left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \\
&= \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right. \\
&= \sigma(x) \cdot (1-\sigma(x))
\end{aligned}
$$

Review: Partial Derivate Example

Rule	$f(x)$	Scalar derivative notation with respect to x	Example
Constant	c	0	$\frac{d}{dx}99 = 0$
Multiplication by constant	cf	$c\frac{df}{dx}$	$\frac{d}{dx}3x = 3$
Power Rule	x^n	nx^{n-1}	$\frac{d}{dx}x^3 = 3x^2$
Sum Rule	$f + g$	$\frac{df}{dx} + \frac{dg}{dx}$	$\frac{d}{dx}(x^2 + 3x) = 2x + 3$
Difference Rule	$f - g$	$\frac{df}{dx} - \frac{dg}{dx}$	$\frac{d}{dx}(x^2 - 3x) = 2x - 3$
Product Rule	fg	$f\frac{dg}{dx} + \frac{df}{dx}g$	$\frac{d}{dx}x^2x = x^2 + x2x = 3x^2$
Chain Rule	$f(g(x))$	$\frac{df(u)}{du}\frac{du}{dx}$, let $u = g(x)$	$\frac{d}{dx}ln(x^2) = \frac{1}{x^2}2x = \frac{2}{x}$

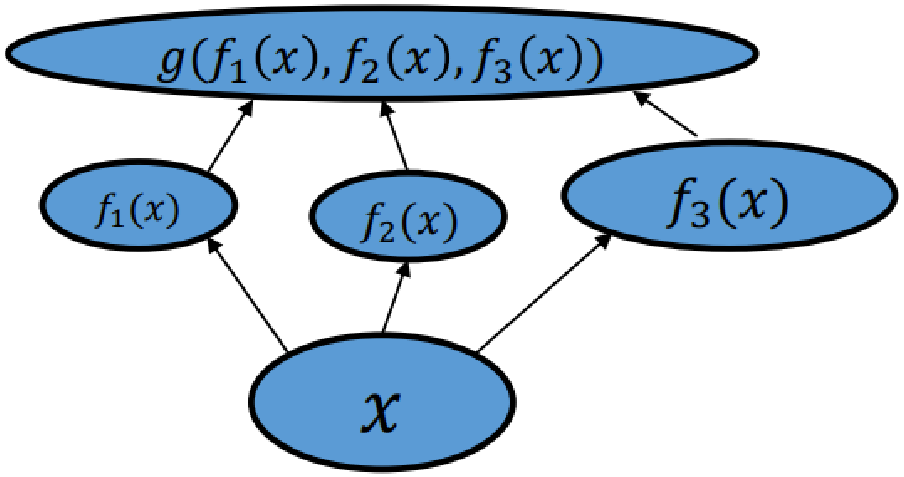
Review: Chain-Rule

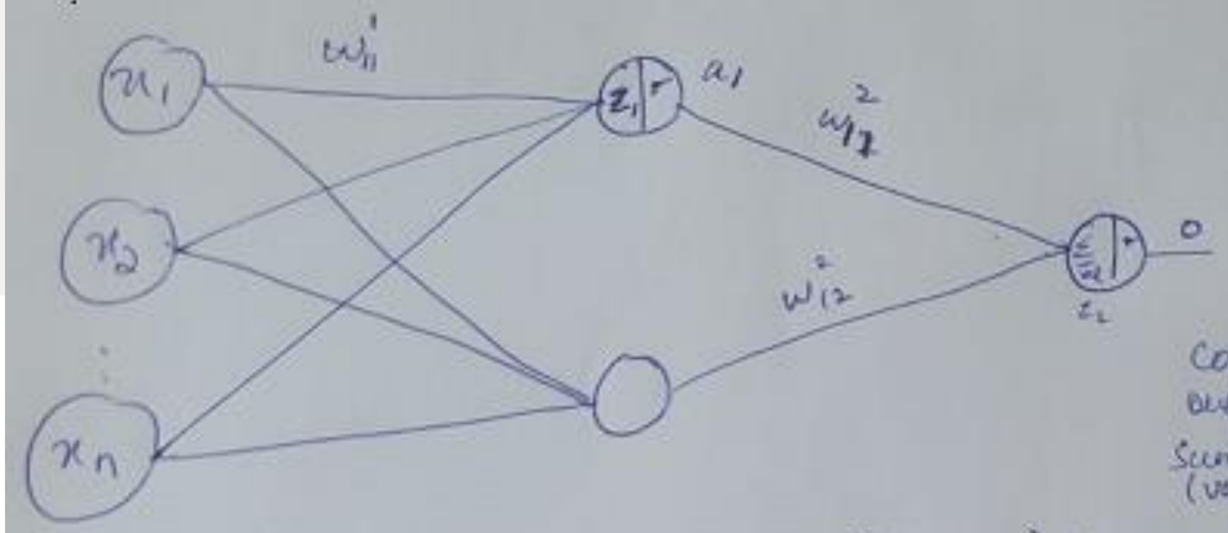
- Univariate Chain Rule

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx}$$

- Multivariate Chain Rule

$$\frac{\partial g}{\partial x} = \sum \frac{\partial g}{\partial f_i}\frac{\partial f_i}{\partial x}$$





cost $\rightarrow$ output O
output $\rightarrow$ summation (val)
Summ (val) $\rightarrow$ w_{11}^2

$$\frac{\partial C}{\partial w_{11}^2} = \frac{\partial C}{\partial O}\cdot\frac{\partial O}{\partial val(z_2)}\cdot\frac{\partial val(z_2)}{\partial w_{11}^2}$$

$$-(y-\hat{y})\cdot O(1-O)\cdot a_1$$

$$\frac{\partial C}{\partial w_{11}^1} = \frac{\partial C}{\partial O}\cdot\frac{\partial O}{\partial val_{z_2}}\cdot\frac{\partial val^{z_2}}{\partial a_1}\cdot\frac{\partial a_1}{\partial z_1}\cdot\frac{\partial z_1}{\partial w_{11}^1}$$

$$-(y-\hat{y})\cdot O(1-O)\cdot w_{11}\cdot(a_1(1-a_1))\cdot x_1$$

Compute Gradient

Output layer

$$Err_j = O_j(1 - O_j)(T_j - O_j)$$

Hidden layer

$$Err_j = O_j(1 - O_j)\sum_k Err_k w_{jk},$$

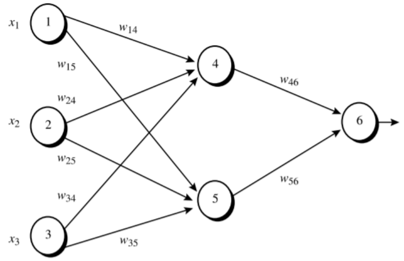
$$Error\ at\ unit\ 6 = \sigma(1 - \sigma)(T - \hat{y}_6\ \ )$$

$$\text{Error at unit 6} = \hat{y}_6\ \ (1 - \hat{y}_6\ \ )(1 - \hat{y}_6\ \ )$$

$0.5\ (1-0.5)(1-0.5) = 0.125$

Error at unit 6

ΔE_6



$$Err\ at\ unit\ 5 = \frac{\partial E}{\partial \hat{y}_6} \cdot \frac{\partial \hat{y}_6}{\partial y_6} \cdot \frac{\partial y_6}{\partial \hat{y}_5} \cdot \frac{\partial \hat{y}_5}{\partial y_5}$$

ΔE_5

$$\boxed{\hat{y}_5(1 - \hat{y}_5)\Delta E_6 \omega_{56}}$$

Error at unit 5

-0.0349

$$Err\ at\ unit\ 4 = \frac{\partial E}{\partial \hat{y}_6} \cdot \frac{\partial \hat{y}_6}{\partial y_6} \cdot \frac{\partial y_6}{\partial \hat{y}_4} \cdot \frac{\partial \hat{y}_4}{\partial y_4}$$

ΔE_4

$$\boxed{\hat{y}_4(1 - \hat{y}_4)\Delta E_6 \omega_{46}}$$

Error at unit 4

-0.0165

Weights Update

$\Delta w_{ij} = \eta \delta_j o_i$

$\eta = 0.01$

y_5

ΔE_6. (~~input~~ at unit 6 or Output unit 5)

$\omega_{56(new)} = \omega_{56} + \eta \ \Delta E_6 \hat{y}_5 = \cancel{0.195}$ -0.197

$\omega_{46(new)} = \omega_{46} + \eta \ \Delta E_6 \hat{y}_4 = -0.298$

$\omega_{14\ (new)} = \omega_{14} + \eta \cdot \Delta E_4 \cdot (x_1)$

$\omega_{15(new)} = \omega_{15} \times \eta \cdot \Delta E_5 \ (x_1)$

$\omega_{24(new)} = \omega_{24} + \Delta E_4 x_2$

$\omega_{25(new)} = \omega_{25} + \Delta E_5 x_2$

$\omega_{34(new)} = \omega_{34} + \Delta E_4 x_3$

$\omega_{35(new)} = \omega_{35} + \Delta E_5 x_3$

Weights Update

$\eta = 0.01$

0.5

ΔE_6. (input at unit 6 and Output unit 5)

$\omega_{56} = \omega_{56} + \eta \ \ \Delta\omega_{56}$ $= \cancel{0.195}$ -0.197

$\omega_{46} = \omega_{46} + \eta \ \ \Delta\omega_{46}$ $= -0.298$

$\omega_{14} =$ $w_{14} + \eta \cdot \Delta E_4 \cdot (x_1)$

$\omega_{15} =$ $w_{15} + \eta \cdot \Delta E_5 \ (x_1)$

$\omega_{24} = \omega_{24} + \Delta E_4 x_2$

$\omega_{25} = \omega_{25} + \Delta E_5 x_2$

$\omega_{34} = \omega_{34} + \Delta E_4 x_3$

$\omega_{35} = \omega_{35} + \Delta E_5 x_3$

Bias Update

- $b_6 = b_6 + \eta . \Delta b_6$ $\Rightarrow \Delta b_6$ = ΔError at unit 6

- $b_5 = b_5 + \eta . \Delta b_5$

  ΔE_5

- $b_4 = b_4 + \eta . \Delta b_4$

  ΔE_4

1. Compute gradients

$$Err_j = O_j(1 - O_j)(T_j - O_j),$$

$$Err_j = O_j(1 - O_j)\sum_k Err_k w_{jk},$$

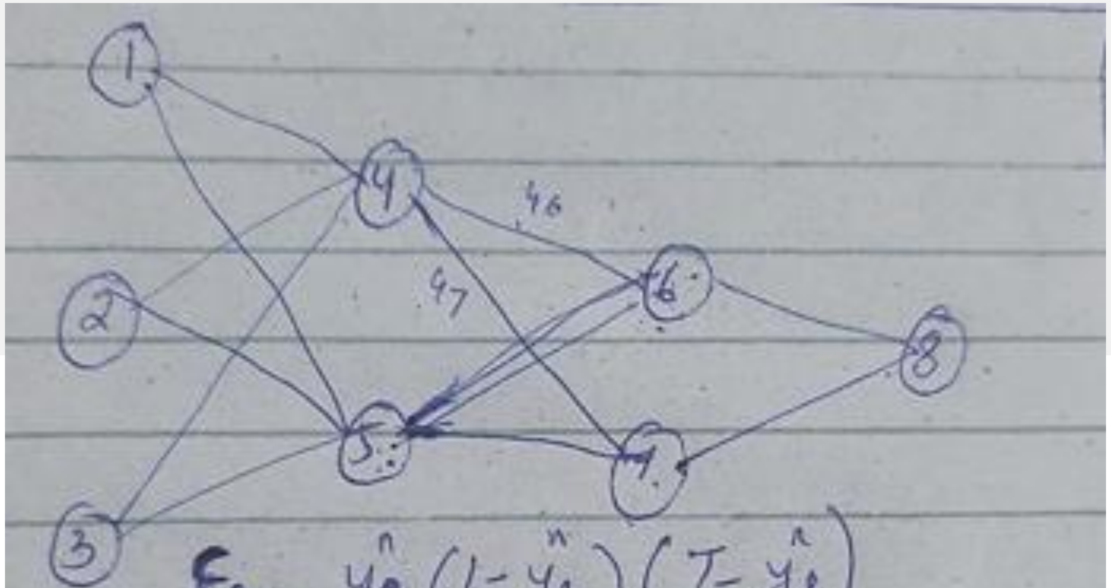
2. Update weights

$w_{new} = w_{old} + \Delta w_{ij}$

$$\Delta_{w_{ij}} = \eta \delta_j o_i$$

3. Update Bias

$b_n = b_n + \eta . \Delta b_n \quad \Rightarrow \Delta b_n = \Delta E$rror at unit n



$E_8 = \hat{y}_8 (1 - \hat{y}_8)(T - \hat{y}_8)$

$E_6 = \hat{y}_6 (1 - \hat{y}_6) E_8 w_{68}$

$E_7 = \hat{y}_7 (1 - \hat{y}_7) E_8 w_{78}$

$E_4 = \hat{y}_4 (1 - \hat{y}_4)(E_6 w_{46} + E_7 w_{47})$

$E_5 = \hat{y}_5 (1 - \hat{y}_5)(E_6 w_{56} + E_7 w_{57})$

Update weight

$\Delta w_{68} = \eta . E_8 \times \hat{y}_6$ | $\Delta w_{46} = \eta \times E_6 \times \hat{y}_4$

Δw [illegible] $= \eta \times E_6 \times x_1$

1. Compute gradients

$$Err_j = O_j(1-O_j)(T_j - O_j),$$

$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk},$$

2. Update weights

$$w_{new} = w_{old} + \Delta w_{ij}$$

3. Update Bias

$$b_n = b_n + \eta . \Delta b_n \qquad \Rightarrow \Delta b_n = \Delta Error \text{ at unit n}$$

1 Compute gradients/error at each unit

$E_6 = O_6(1-O_6)(Target - O_6)$

$= \hat{y}_6(1-\hat{y}_6)(T-\hat{y}_6)$

$E_4 = \hat{y}_4(1-\hat{y}_4)\, E_6\, w_{46}$

$\boxed{E_5 = \hat{y}_5(1-\hat{y}_5)\, E_6\, w_{56}}$

② Update weights

$\Delta w_{ji} = \eta\, \delta_j\, O_i$

$\Delta w_{46} = \eta . E_6 . \hat{y}_4$

$\Delta w_{56} = \eta . E_6 . \hat{y}_5$

$\Delta w_{35} = \eta . E_5 . x_3$

$\Delta w_{34} = \eta . E_5 . x_3$

$\Delta w_{24} = \eta . E_4 . x_2$

$\boxed{w_{24}^{new} = w_{24}^{old} + \Delta w_{24}}$

③ Update Bias

$b_4^{new} = b_4 + \eta E_4$

$b_5^{N} = b_5 + E_5$

$b_6^{N} = b_6 + \eta E_6$

Learning Rate

- This is a subtle art.
- Too small: can take days instead of minutes to converge
- Too large: diverges (MSE gets larger and larger while the weights increase and usually oscillate)
- Sometimes the “just right” value is hard to find.

Train ANN

Training neural nets:

Loop until convergence:

- for each example n
  1. Given input $\mathbf{x}^{(n)}$, propagate activity forward $(\mathbf{x}^{(n)} \rightarrow \mathbf{h}^{(n)} \rightarrow o^{(n)})$ (**forward pass**)
  2. Propagate gradients backward (**backward pass**)
  3. Update each weight (via gradient descent)