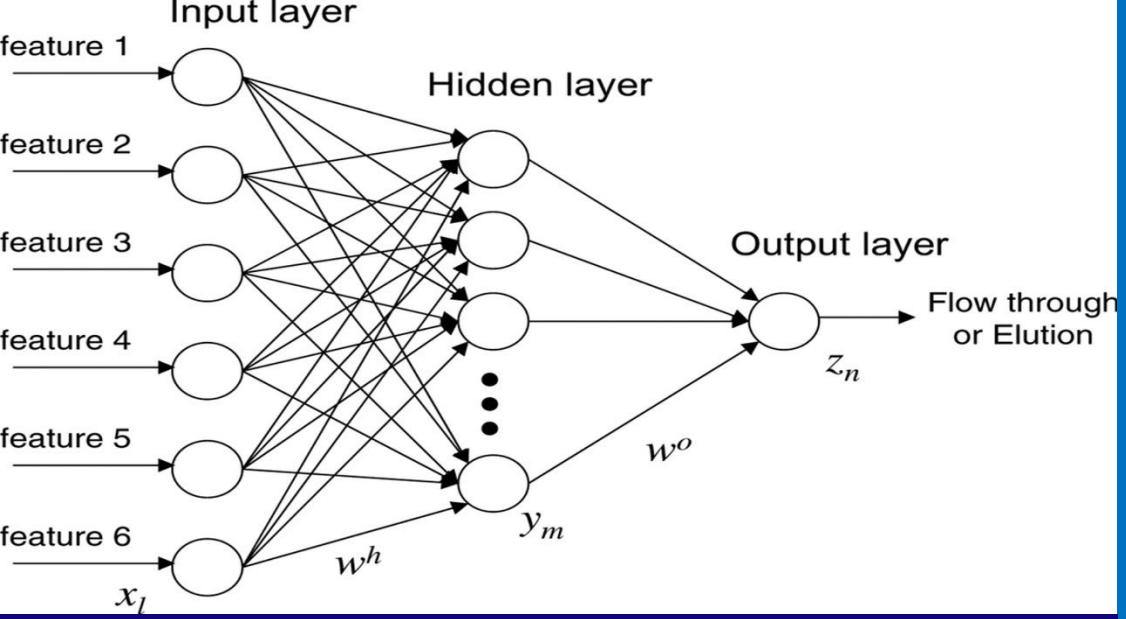
ΤΕΧΝΗΤΗ *ΝΟΗΜΟΣΥΝΗ* – *ΥΠΟΛΟΓΙΣΤΙΚΗ ΝΟΗΜΟΣΥΝΗ*

Λάζαρος Ηλιάδης Καθηγητής ΔΠΘ
Email: *liliadis@civil.duth.gr*

MLFF - BACK PROPAGATION!!



- ❖ Ενισχυόμενη μάθηση (reinforcement learning)
  - ➢ Βασίζεται σε διαδικασίες ανταμοιβής σε σχέση με το αποτέλεσμα.

✓ Προσαρμογή βαρών

- ❖ Η εκπαίδευση αποσκοπεί στην αναπροσαρμογή των βαρών των συνδέσεων μεταξύ των νευρώνων του ΤΝΔ, σύμφωνα με κάποιο κανόνα μετατροπής (Hebbian Learning Rule).

– Εκπαίδευση

Οι μέθοδοι εκπαίδευσης μπορούν να διαχωριστούν σε:

✓ **Επιβλεπόμενη (supervised learning)**

Το ΤΝΔ εκπαιδεύεται με συγκεκριμένες εισόδους οι οποίες ταιριάζουν με τις εξόδους.

✓ **Μη επιβλεπόμενη (unsupervised learning)**

Μια μονάδα εξόδου εκπαιδεύεται να ανταποκρίνεται σε ομάδες προτύπων που υπάρχουν στην είσοδο..

✓ Η εκπαίδευση δύο σταδίων **(semi-supervised training)**

Πρόκειται για τον συνδυασμό των παραπάνω μεθόδων εκπαίδευσης και εκτελείται σε δύο στάδια.

Στο 1ο στάδιο, γίνεται εκπαίδευση χωρίς επίβλεψη, η οποία ομαδοποιεί τα δεδομένα εισόδου.

Στο 2ο στάδιο γίνεται εκπαίδευση με επίβλεψη και δημιουργείται η συνάρτηση απεικόνισης εισόδου – εξόδου.

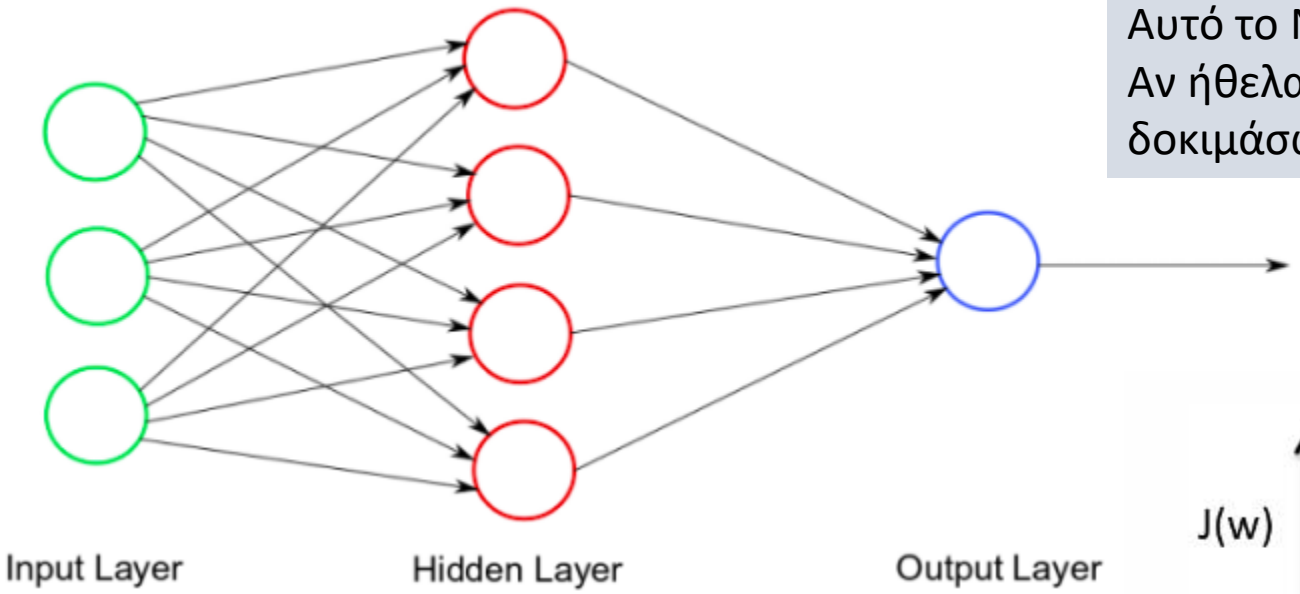
Οι ομάδες του 1ου σταδίου χρησιμεύουν για να αρχικοποιηθεί το ΤΝΔ στο 2ο στάδιο.

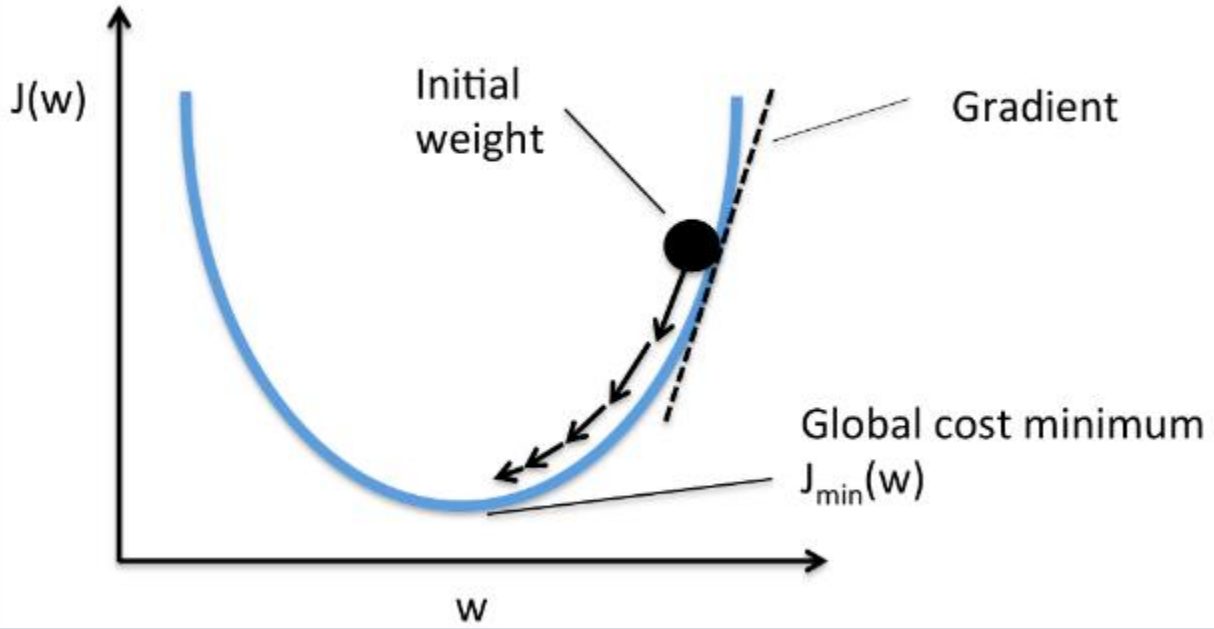
Πως θα βρω τα βέλτιστα ΒΑΡΗ;

Αυτό το Νευρωνικό δίκτυο έχει 16 βάρη!!

Αν ήθελα ακρίβεια 3 δεκαδικών ψηφίων και αν θα έπρεπε να δοκιμάσω όλους τους συνδυασμούς βαρών τότε θα προέκυπταν:

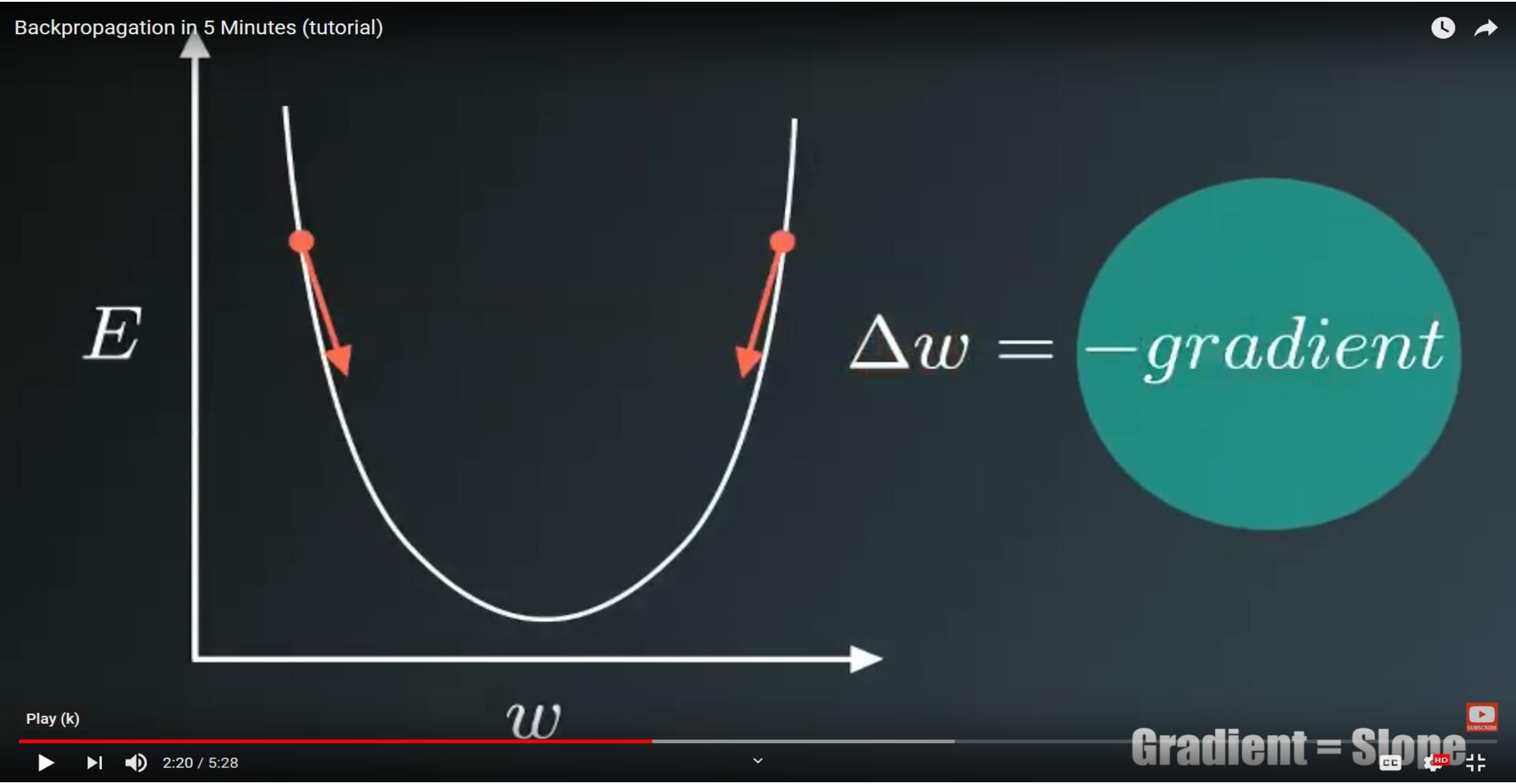
$1000^{16} = 10^{48}$ **Διαφορετικοί συνδυασμοί βαρών !!!**



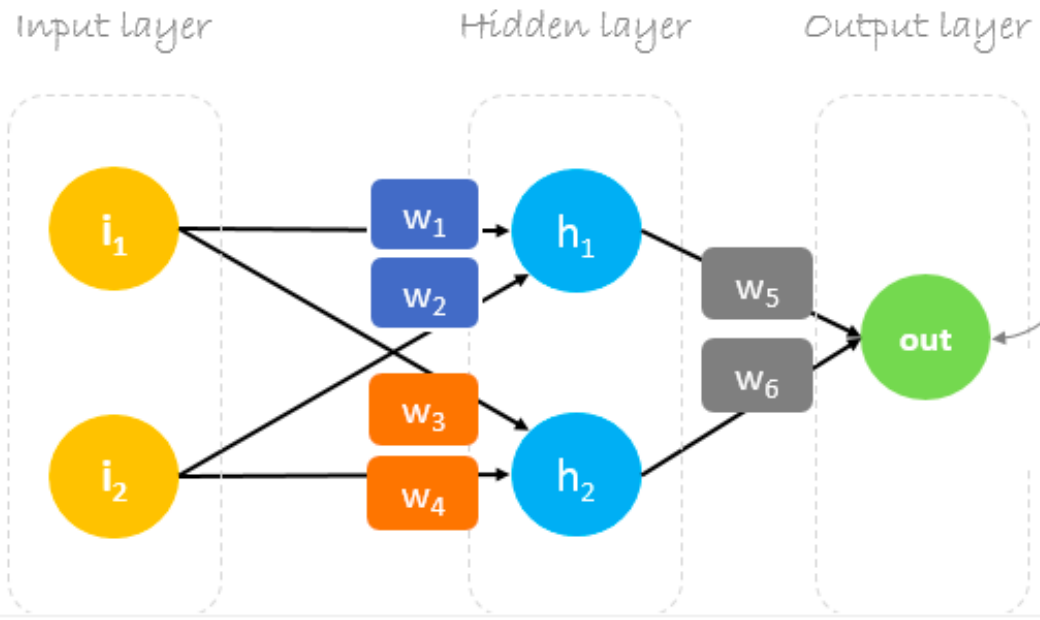


Έστω ότι αυτή είναι η συνάρτηση σφάλματος! Θα έπρεπε να μειώνω συνέχεια τα βάρη ώστε να ελαχιστοποιηθεί το σφάλμα. (ΟΛΙΚΟ ΜΕΓΙΣΤΟ). Αν συνεχίσω να μειώνω τα βάρη, μετά από κάποιο σημείο τα βάρη θα αρχίσουν να αυξάνονται και επίσης από κάποιο σημείο θα αρχίσει και το σφάλμα να αυξάνεται. Πρέπει να σταματήσω στο κατάλληλο σημείο όσο έχω αρνητικό Gradient Descent (δηλαδή όσο το σφάλμα μειώνεται). Η κλίση της γωνίας της εφαπτόμενης (παράγωγος) σημαίνει ότι το σφάλμα συνεχίζει να μειώνεται!! Αλλιώς αν γίνει θετική το σφάλμα θα αρχίσει να μεγαλώνει. Negative gradient descent

GRADIENT DESCENT



BACK PROPAGATION



Weights, weights, weights

Η εκπαίδευση του ΤΝΔ στοχεύει στο να βρει τα βάρη που θα ελαχιστοποιήσουν το σφάλμα πρόγνωσης. Μετά ο αλγόριθμος BP χρησιμοποιείται για να αλλάξει τα βάρη ώστε να απεικονίζονται οι εισαγόμενες τιμές ΣΩΣΤΑ στα αποτελέσματα.

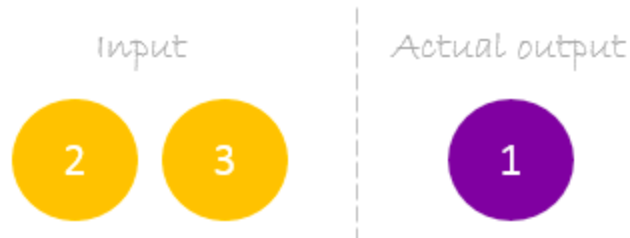
Έστω ότι τα αρχικά βάρη είναι:

`w1 = 0.11` , `w2 = 0.21` , `w3 = 0.12` , `w4 = 0.08` , `w5 = 0.14` `w6 = 0.15`

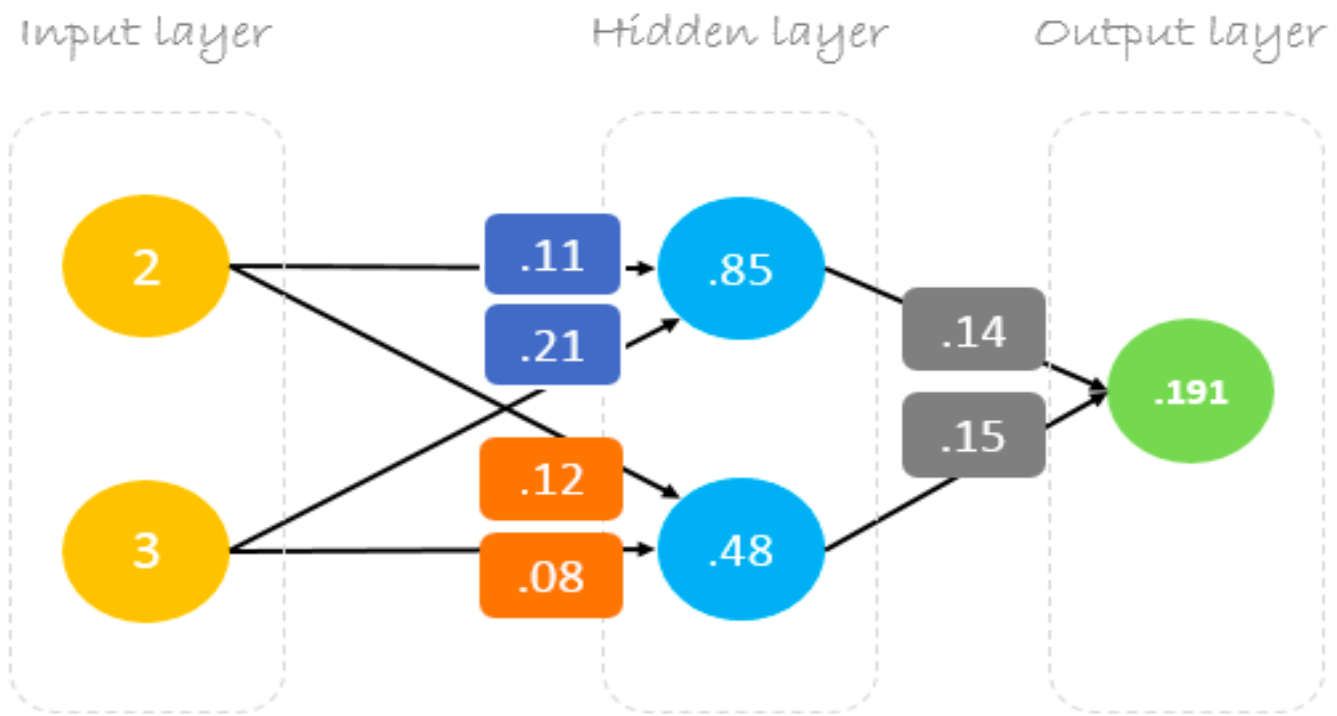
Dataset

Το σύνολο δεδομένων έχει 2 νευρώνες Εισόδου I1 και I2 και 1 νευρώνα εξόδου (τιμή Z)

Έστω ότι στην είσοδο έχουμε τιμές Inputs=[2, 3] και στην έξοδο output=[1]



FORWARD PASS ΠΡΟΩΘΗΣΗ ΠΡΟΣ ΤΟ ΕΠΙΠΕΔΟ ΕΞΟΔΟΥ



Forward Pass

Matrix multiplication

$$[2 \quad 3] \cdot \begin{bmatrix} 0.11 & 0.12 \\ 0.21 & 0.08 \end{bmatrix} = [0.85 \quad 0.48] \cdot \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} = [0.191]$$

Details

2 x .11 + 3 x .21 = .85

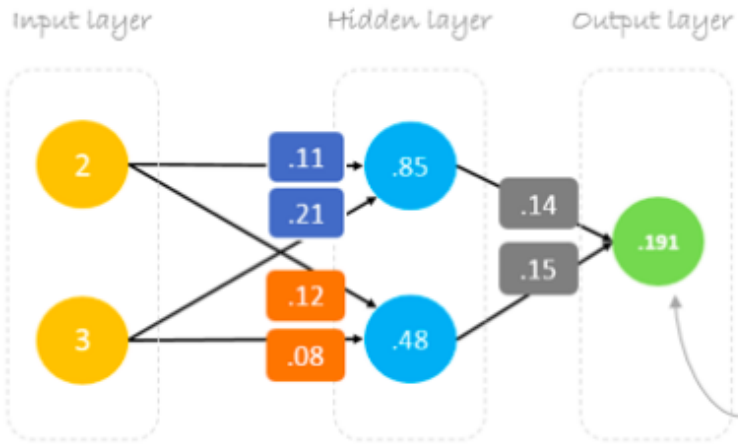
2 x .12 + 3 x .08 = .48

.85 x .14 + .48 x .15 = .191

ΕΥΡΕΣΗ ΣΦΑΛΜΑΤΟΣ!

Υπολογισμός του σφάλματος

Τώρα θα βρεθεί πόσο καλά πήγε το ΤΝΔ βρίσκοντας το σφάλμα. Το αποτέλεσμα που πήραμε δνε είναι ούτε καν κοντά σε αυτό που θέλουμε να πάρουμε.



Είσοδος

2 3

Σωστή τιμή-Actual value

1

Πρόβλεψη

actual

ΣΦΑΛΜΑ ΕΙΝΑΙ 0 ΟΤΑΝ ΕΙΝΑΙ ΣΩΣΤΗ Η ΠΡΟΒΛΕΨΗ

$$\text{Error} = \frac{1}{2}(\text{prediction} - \text{actual})^2$$

Το σφάλμα είναι πάντα θετικό λόγω ύψωσης στο τετράγωνο

το $\frac{1}{2}$ πολλαπλασιάζεται για να διευκολύνει τον υπολογισμό της παραγώγου στη συνέχεια

$$\text{Error} = \frac{1}{2}(0.191 - 1.0)^2 = 0.327$$

GRADIENT DESCENT ΑΛΛΑΓΗ ΒΑΡΩΝ

Πρέπει λοιπόν να αλλάξουμε τα βάρη και το ερώτημα είναι πως;
Η απάντηση λέγεται Backpropagation

Πρόκειται για αντίστροφη μετάδοση του σφάλματος. Είναι ένας μηχανισμός που χρησιμοποιεί τον αλγόριθμο Gradient Descent! Υπολογίζει την μερική παράγωγο του Σφάλματος ως προς τα βάρη του ΤΝΔ

Το Gradient Descent είναι ένας αλγόριθμος βελτιστοποίησης που χρησιμοποιεί επαναλήψεις για να βρει την ελάχιστη τιμή μιας συνάρτησης.

Παλιό Βάρος

Παράγωγος Σφάλματος ως προς το βάρος

Derivative of Error with respect to weight

$$^{*}W_x = W_x - a\left(\frac{\partial Error}{\partial W_x}\right)$$

Νέο Βάρος

Ρυθμός Μάθησης

Learning rate

ΜΑΘΗΜΑΤΙΚΑ ΕΠΑΝΑΠΡΟΣΔΙΟΡΙΣΜΟΥ ΒΑΡΟΥΣ

Η παράγωγος της συναρτησης σφάλματος υπολογίζεται με τον πιο κάτω κανόνα της αλυσίδας

ΕΑΝ Z είναι μια σύνθετη συνάρτηση της μορφής: $z = f(x,y)$ **όπου** $x = x(t)$ **και** $y = y(t)$

τότε $\frac{dz}{dt} = \frac{\partial z}{\partial x}.\frac{dx}{dt} + \frac{\partial z}{\partial y}.\frac{dy}{dt}$ where Z=Error t=W6 X=prediciton and Y=actual

Y είναι σταθερά πραγματική τιμή

ΝΕΟ ΒΑΡΟΣ $^{*}W_6 = W_6 - a\left(\frac{\partial Error}{\partial W_6}\right)$ *learning rate.*

Prediction=(i1 w1+i2w2)w5 +(i1w3+i2W4)w6

$$\frac{\partial Error}{\partial W_6} = \frac{\partial Error}{\partial prediction} * \frac{\partial prediction}{\partial W_6}$$ chain rule

Error = $\frac{1}{2}$(prediction – actual)^2

prediction = $(i_1 w_1 + i_2 w_2) w_5 + (i_1 w_3 + i_2 w_4) w_6$

0 συντελεστής

i1 και i2 είναι οι είσοδοι

$$\frac{\partial Error}{\partial W_6} = \frac{\frac{1}{2}(predictoin - actula)^2}{\partial prediciton} * \frac{\partial (i_1 w_1 + i_2 w_2) w_5 + (i_1 w_3 + i_2 w_4) w_6}{\partial W_6}$$

$$\frac{\partial Error}{\partial W_6} = 2 * \frac{1}{2}(predictoin - actula)\frac{\partial (predictoin - actula)}{\partial prediciton} * (i_1 w_3 + i_2 w_4)$$ $h_2 = i_1 w_3 + i_2 w_4$

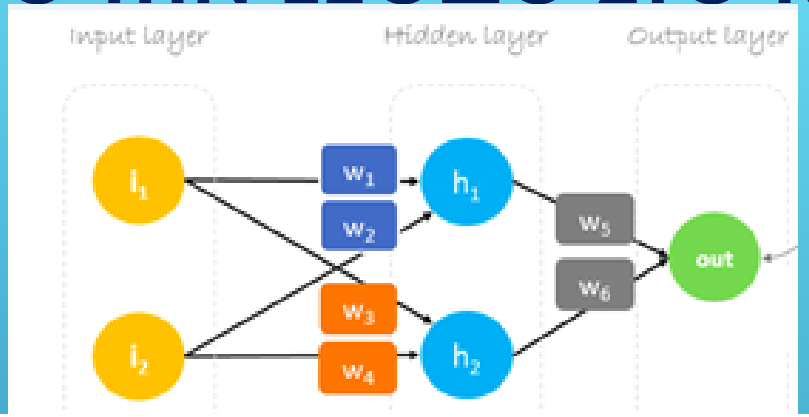
$$\frac{\partial Error}{\partial W_6} = (predictoin - actula) * (h_2)$$

Δ = prediction – actual ← delta

$$\frac{\partial Error}{\partial W_6} = \Delta h_2$$

h2=i1w3+i2w4

ΑΛΛΑΓΗ ΒΑΡΩΝ ΑΠΌ ΤΗΝ ΕΞΟΔΟ ΣΤΟ ΚΡΥΦΟ



Για να αλλάξω το W6 εφαρμόζω τον τύπο $^*W_6 = W_6 - a\,\Delta h_2$ παρόμοια για κάθε βάρος από το επίπεδο εξόδου προς το κρυφό

$$^*W_5 = W_5 - a\,\Delta h_1$$

Έτσι όπως πηγαίνουμε προς τα πίσω για να αλλάξουμε τα `w1` , `w2` , `w3` `w4` που υπάρχουν μεταξύ του επιπέδου Εισόδου και Κρυφού ενδεικτικά η μερική παράγωγος της συνάρτησης σφάλματος όσον αφορά στο `w1` είναι η εξής!

$$\frac{\partial Error}{\partial W_1} = \frac{\partial Error}{\partial prediction} * \frac{\partial prediction}{\partial h_1} * \frac{\partial h_1}{\partial W_1}$$ ← chain rule

$$\frac{\partial Error}{\partial W_1} = \frac{\partial \frac{1}{2}(predictoin - actula)^2}{\partial prediciton} * \frac{\partial (h_1)\,w_5 + (h_2)\,w_6}{\partial h_1} * \frac{\partial\, i_1 w_1 + i_2 w_2}{\partial w_1}$$

$$\frac{\partial Error}{\partial W_1} = 2 * \frac{1}{2}(\mathbf{predictoin - actula}) \frac{\partial (predictoin - actula)}{\partial prediciton} * (w_5) * (i_1)$$

Error = $\frac{1}{2}$(prediction − actual)^2

prediction = $(h_1)\,w_5 + (h_2)\,w_6$

$h_1 = i_1 w_1 + i_2 w_2$

ΝΕΑ ΒΑΡΗ

Το a Learning Rate είναι πολύ σημαντικό

updated weights

ΝΕΑ ΒΑΡΗ

$$
\begin{aligned}
{}^{*}w_6 &= w_6 - a\,(h_2 \,.\, \Delta) \\
{}^{*}w_5 &= w_5 - a\,(h_1 \,.\, \Delta) \\
{}^{*}w_4 &= w_4 - a\,(i_2 \,.\, \Delta w_6) \\
{}^{*}w_3 &= w_3 - a\,(i_1 \,.\, \Delta w_6) \\
{}^{*}w_2 &= w_2 - a\,(i_2 \,.\, \Delta w_5) \\
{}^{*}w_1 &= w_1 - a\,(i_1 \,.\, \Delta w_5)
\end{aligned}
$$

Μπορούμε να παραστήσουμε τους πιο πάνω υπολογισμούς με πράξεις πινάκων

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - a\,\Delta \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \begin{bmatrix} a h_1 \Delta \\ a h_2 \Delta \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - a\,\Delta \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} . \begin{bmatrix} w_5 & w_6 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \begin{bmatrix} a\,i_1 \Delta w_5 & a\,i_1 \Delta w_6 \\ a\,i_2 \Delta w_5 & a\,i_2 \Delta w_6 \end{bmatrix}$$

ΡΥΘΜΟΣ ΜΑΘΗΣΗΣ

Learning rate: **Είναι υπερπαράμετρος την τιμή της οποίας καλούμαστε να εκτιμήσουμε**

$\Delta = 0.191 - 1 = -0.809$ ← Delta = prediction - actual

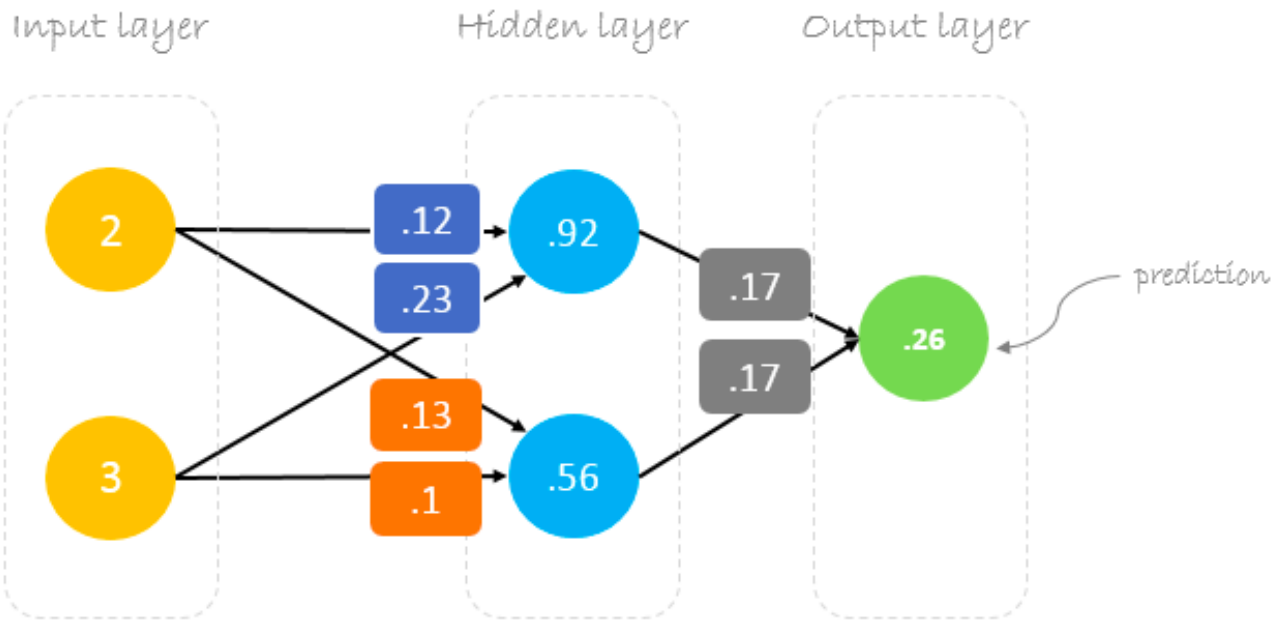
$a = 0.05$ ← Learning rate, **αρχικοποιούμε την τιμή του εμείς**

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - 0.05(-0.809)\begin{bmatrix} 0.85 \\ 0.48 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - \begin{bmatrix} -0.034 \\ -0.019 \end{bmatrix} = \begin{bmatrix} 0.17 \\ 0.17 \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - 0.05(-0.809)\begin{bmatrix} 2 \\ 3 \end{bmatrix}.[0.14 \quad 0.15] = \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - \begin{bmatrix} -0.011 & -0.012 \\ -0.017 & -0.018 \end{bmatrix} = \begin{bmatrix} .12 & .13 \\ .23 & .10 \end{bmatrix}$$

Έτσι βρίσκω τα νέα βάρη

Τώρα χρησιμοποιώντας τα νέα βάρη πάμε να κάνουμε εμπρόσθιο πέρασμα (forward pass)



Forward Pass

Πολλαπλασιασμός Πινάκων

$$[2 \quad 3] \cdot \begin{bmatrix} 0.12 & 0.13 \\ 0.23 & 0.10 \end{bmatrix} = [0.92 \quad 0.56] \cdot \begin{bmatrix} 0.17 \\ 0.17 \end{bmatrix} = [0.26]$$

Details

2 x .12 + 3 x .23 = .85

2 x .13 + 3 x .10 = .48

.92 x .17 + .56 x .17 = .26

Η πρόγνωση `0.26` **είναι πιό κοντά στο actual output**

Επαναλαμβάνεται το μπρος - πίσω ώσπου να μειωθεί πολύ το σφάλμα

ΣΥΝΑΡΤΗΣΕΙΣ ΜΕΤΑΦΟΡΑΣ –TRANSFER FUNCTIONS

Στο κρυφό επίπεδο πάντοτε χρησιμοποιείται ΜΗ Γραμμική συνάρτηση. Στο επίπεδο εξόδου μπορεί να χρησιμοποιηθεί είτε Γραμμική πχ ReLU είτε Μη Γραμμική ίδια με αυτήν του κρυφού επιπέδου.

Στην αναγνώριση προτύπων – Pattern Recognition όπου οι τιμές είναι διακριτές και το αποτέλεσμα είναι Δυαδικός αριθμός μπορεί να χρησιμοποιηθεί η Sigmoid transfer function η οποία είναι αλλιώς γνωστή ως Logistic που παράγει τιμές από το 0 έως και το +1.

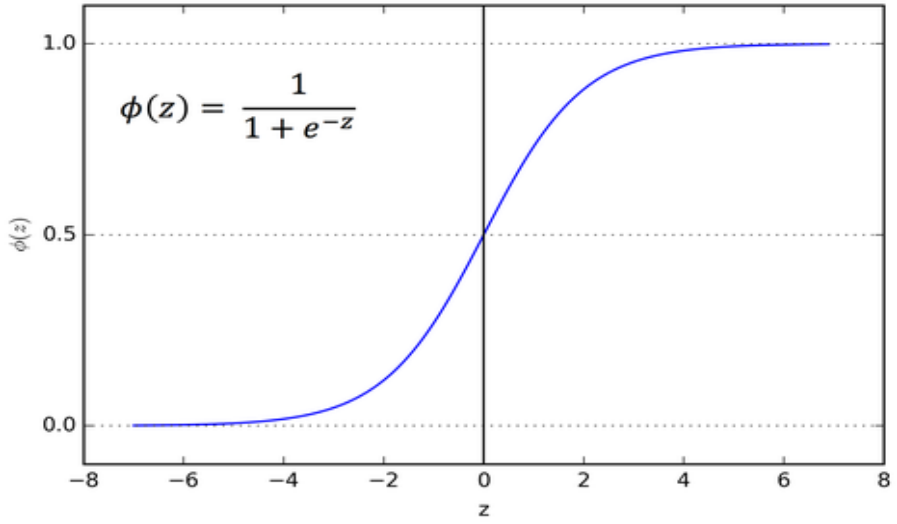


Fig: Sigmoid Function

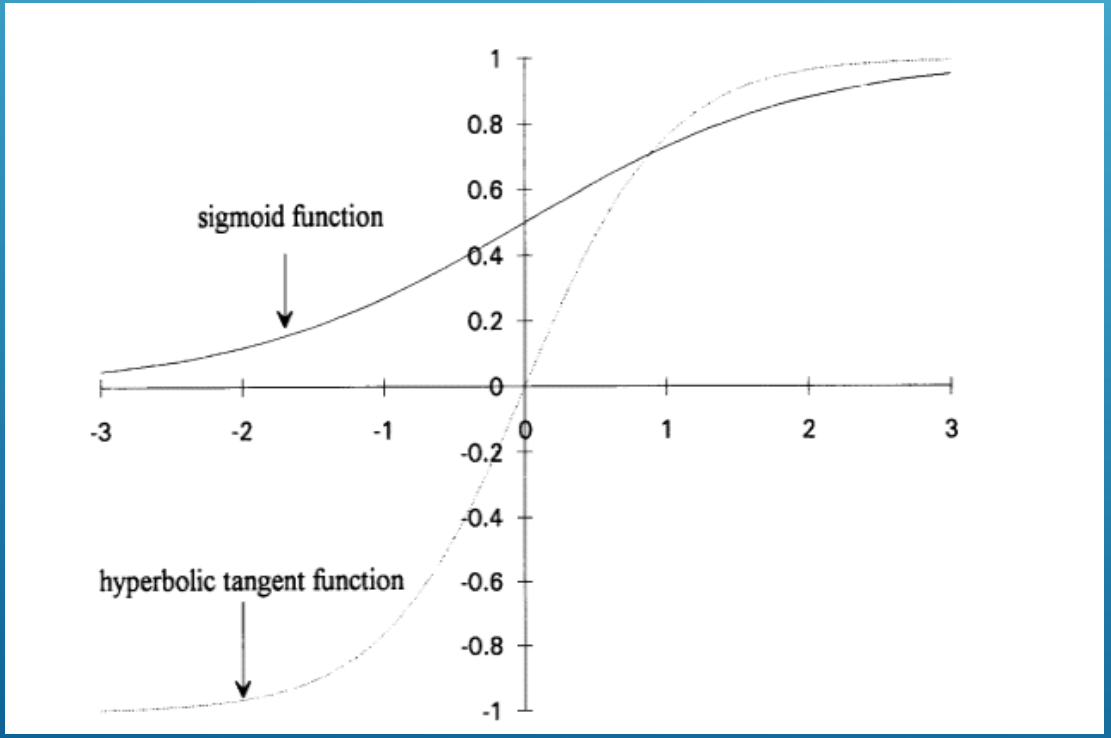
ΣΥΝΑΡΤΗΣΕΙΣ ΜΕΤΑΦΟΡΑΣ –TRANSFER FUNCTIONS

Στην διαδικασία regression / function approximation μπορώ να χρησιμοποιήσω είτε την Sigmoid είτε την TanH (Tangetn Hyperbolic) Εφαπτομένη Υπερβολοειδή **που παράγει τιμές από το -1 έως και το +1**

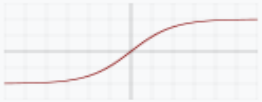
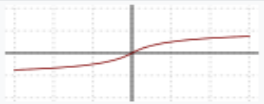
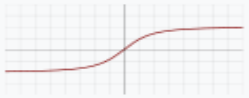
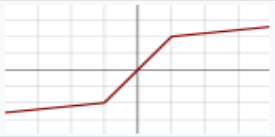
Η Sigmoid θα παράξει τιμές κοντά στο 0 εάν το όρισμα της συνάρτησης είναι ουσιαστικά αρνητικό.

Έτσι, η έξοδος αυτού του hidden neuron θα είναι κοντά στο μηδέν, μειώνοντας έτσι τον ρυθμό εκμάθησης για όλα τα επόμενα βάρη ίσως έτσι επιβραδυνθεί εξαιρετικά η μάθηση (πρακτικά θα σταματήσει)!!!

Προτιμάω την TanH στο Regression. Η TanH έχει μεγαλύτερη κλίση από την Sigmoid άρα μπορεί να αξιολογήσει καλύτερα μικρές διακυμάνσεις στην τιμή της εισόδου. Επίσης παίρνει και αρνητικές τιμές για αρνητικές εισόδους



Πχ για εισόδους 0,1 και 0,2. Η sigmoid δίνει 0,525 και 0,550 διαφορά μόνο 0,025 ενώ η TanH δίνει 0,100 και 0,197 διαφορά 0,097

Name	Plot	Equation	Derivative (with respect to x)
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a. Sigmoid or Soft step)		$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}$ [1]	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$	$f'(x) = 1 - f(x)^2$
SQNL[10]		$f(x) = \begin{cases} 1 & : x > 2.0 \\ x - \frac{x^2}{4} & : 0 \leq x \leq 2.0 \\ x + \frac{x^2}{4} & : -2.0 \leq x < 0 \\ -1 & : x < -2.0 \end{cases}$	$f'(x) = 1 \mp \frac{x}{2}$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
ArSinH		$f(x) = \sinh^{-1}(x) = \ln\left(x + \sqrt{x^2+1}\right)$	$f'(x) = \frac{1}{\sqrt{x^2+1}}$
ElliotSig[11][12] Softsign[13][14]		$f(x) = \frac{x}{1+\lvert x\rvert}$	$f'(x) = \frac{1}{(1+\lvert x\rvert)^2}$
Inverse square root unit (ISRU)[15]		$f(x) = \frac{x}{\sqrt{1+\alpha x^2}}$	$f'(x) = \left(\frac{1}{\sqrt{1+\alpha x^2}}\right)^3$
Inverse square root linear unit (ISRLU)[15]		$f(x) = \begin{cases} \frac{x}{\sqrt{1+\alpha x^2}} & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \left(\frac{1}{\sqrt{1+\alpha x^2}}\right)^3 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Piecewise Linear Unit (PLU)[16]		$f(x) = \max(\alpha(x+c) - c, \min(\alpha(x-c) + c, x))$	$f'(x) = \begin{cases} \alpha & \text{for } \lvert x\rvert > c \\ 1 & \text{for } \lvert x\rvert < c \end{cases}$
Rectified linear unit (ReLU)[17]		$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ 1 & \text{for } x > 0 \end{cases}$