Τεχνητή Νοημοσύνη - Νευρωνικά Δίκτυα

Δημήτριος Κοσμόπουλος

Πανεπιστήμιο Πατρών
Τμήμα Μηχανικών ΗΥ κ Πληροφορικής

11 Δεκεμβρίου 2024

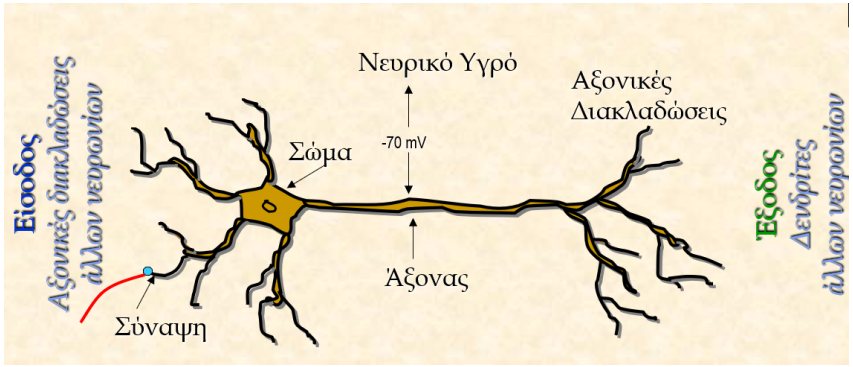
Εισαγωγή

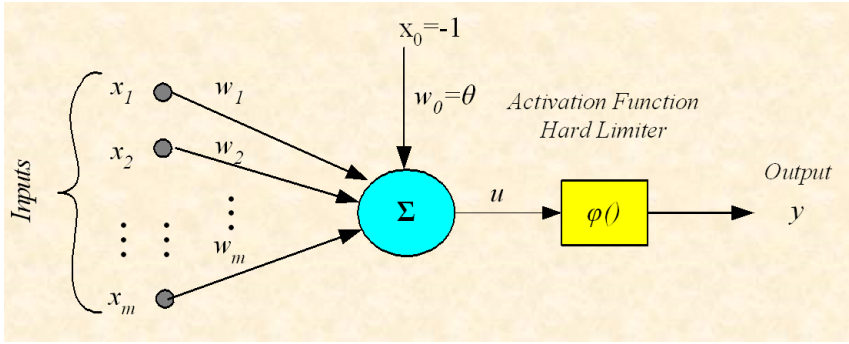
- Στόχος αυτής της παρουσίασης είναι να αποτελέσει μια πρώτη γνωριμία με το χώρο των Τεχνητών Νευρωνικών Δικτύων (ΤΝΔ) και να σας εισάγει στις βασικές αρχιτεκτονικές ΤΝΔ και στους βασικούς αλγορίθμους εκπαίδευσης τους.
- Τα ΤΝΔ, μαζί με άλλους αλγορίθμους εμπνευσμένους από τη φύση, αποτελούν την επιστημονική περιοχή της Υπολογιστικής Νοημοσύνης (Computational Intelligence), που είναι μέρος της Τεχνητής Νοημοσύνης.

Τι είναι τα ΤΝΔ

- Είναι απλοποιημένα μοντέλα του Εγκεφάλου
- Δίκτυα, από διασυνδεδεμένα νευρωνικά υπολογιστικά στοιχεία (νευρώνες), που έχουν την ικανότητα να ανταποκρίνονται σε ερεθίσματα που δέχονται στην είσοδό τους και να μαθαίνουν να προσαρμόζονται στο περιβάλλον τους

Ο βασικός νευρώνας





Σύνοψη

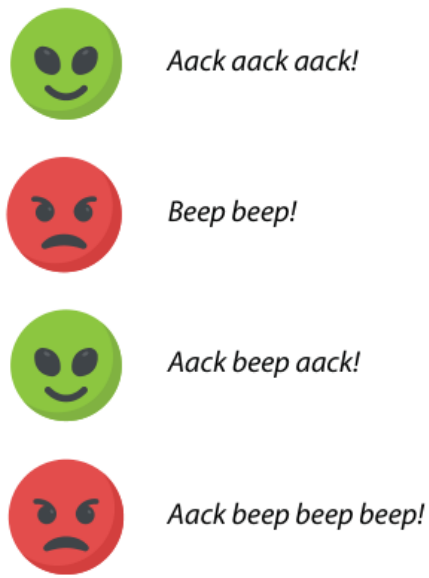
- Δίκτυο perceptron
- Δίκτυο multilayer perceptron

Σενάριο 1: μια πρωτόγονη γλώσσα

- Βρισκόμαστε σε άγνωστο πλανήτη
- Οι κάτοικοι του μιλούν μια άγνωστη σε εμάς γλώσσα με πολύ απλό λεξιλόγιο και σύνταξη
- ανάλογα με τις λέξεις που χρησιμοποιούν θα πρέπει να φτιάξουμε ένα ταξινομητή που θα διακρίνει αν είναι χαρούμενοι ή όχι

Σενάριο 1: μια πρωτόγονη γλώσσα

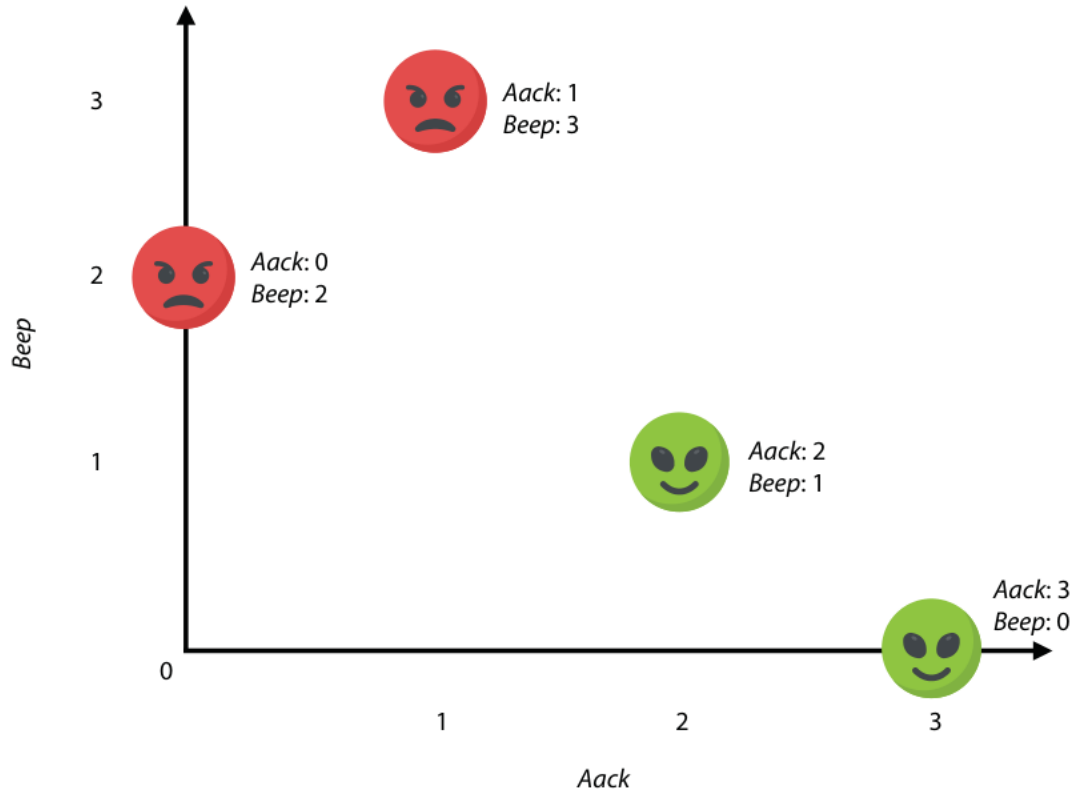
Δεδομένα



Πρόβλεψη

Σενάριο 1: μια πρωτόγονη γλώσσα

Sentence	*Aack*	*Beep*	Mood
Aack aack aack!	3	0	Happy
Beep beep!	0	2	Sad
Aack beep aack!	2	1	Happy
Aack beep beep beep!	1	3	Sad



Σενάριο 1: μια πρωτόγονη γλώσσα

- *beep*: -1
- *aack*: +1

Υπολογίζουμε το σκορ της πρότασης προσθέτοντας τα σκορ όλων των λέξεων που περιέχει ως εξής:

- Αν το σκορ είναι θετικό ή μηδενικό, προβλέπουμε ότι το *alien* είναι χαρούμενο.

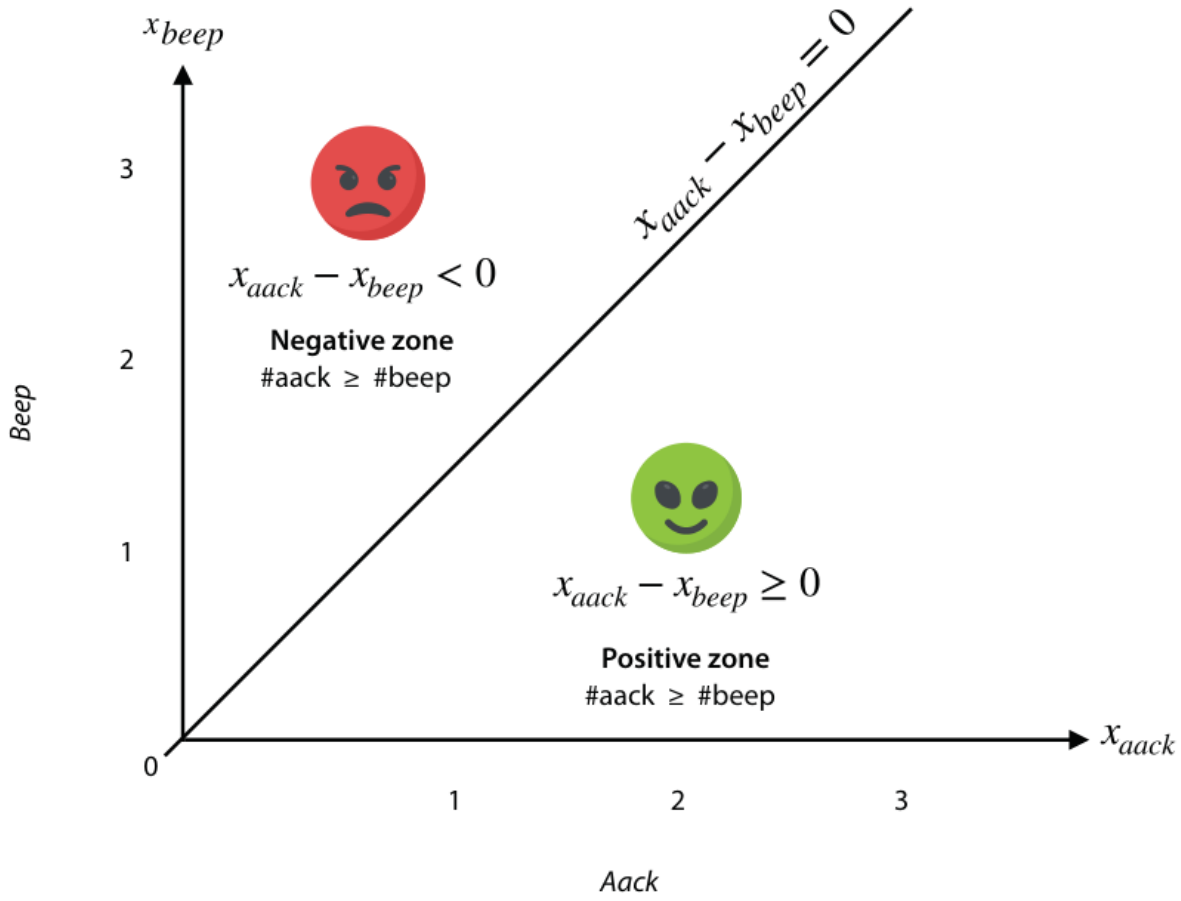
$$x_{aack} - x_{beep} \geq 0$$

- Αν το σκορ είναι αρνητικό, προβλέπουμε ότι το *alien* είναι λυπημένο.

$$x_{aack} - x_{beep} < 0$$

Σενάριο 1: μια πρωτόγονη γλώσσα

Ορισμός της ευθείας "απόφασης"



Σενάριο 2: μια διαφορετική πρωτόγονη γλώσσα

Sentence	*Crack*	*Doink*	Mood
Crack!	1	0	Sad
Doink doink!	0	2	Sad
Crack doink!	1	1	Sad
Crack doink crack!	2	1	Sad
Doink crack doink doink!	1	3	Happy
Crack doink doink crack!	2	2	Happy
Doink doink crack crack crack!	3	2	Happy
Crack doink doink crack doink!	2	3	Happy

Σενάριο 2: μια διαφορετική πρωτόγονη γλώσσα

- *crack*: $+1$
- *doink*: $+1$

Υπολογίζουμε τη βαθμολογία της πρότασης προσθέτοντας το σκορ όλων των λέξεων που περιέχει ως εξής:

- Αν το σκορ είναι μεγαλύτερο από 3.5, προβλέπουμε ότι το *alien* είναι χαρούμενο.

$$x_{crack} + x_{doink} \geq 3.5$$

- Αν το σκορ είναι μικρότερο από 3.5, προβλέπουμε ότι το *alien* είναι λυπημένο.

$$x_{crack} + x_{doink} < 3.5$$

Σενάριο 2: μια διαφορετική πρωτόγονη γλώσσα

Η ευθεία απόφασης είναι η εξής:

$$x_{crack} + x_{doink} - 3.5 = 0$$

Ο σταθερός όρος -3.5: *bias*. Αν το *alien* δεν μιλά σημαίνει ότι είναι λυπημένο.

- Θετικό *bias*: αριθμός λέξεων σε αξιολογήσεις (συνήθως οι θετικές είναι επιγραμματικές και οι αρνητικές κριτικές έχουν μεγάλη έκταση)
- Αρνητικό *bias*: αριθμός λέξεων σε προσωπικές συνομιλίες (συνήθως οι δυσαρεστημένοι δεν επικοινωνούν)

Γενίκευση

Μια πιο γενική ευθεία απόφασης είναι η εξής:

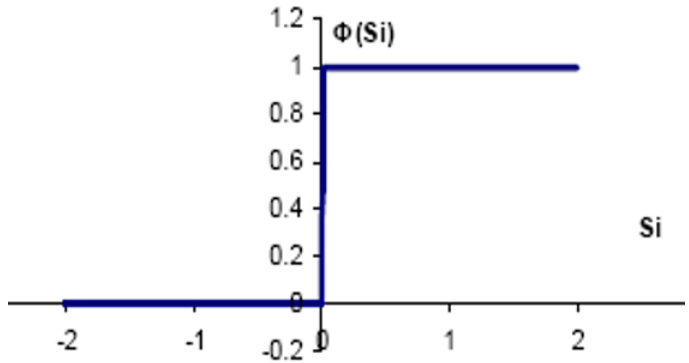
$$a \cdot x_1 + b \cdot x_2 + c = 0$$

π.χ.

$$5 \cdot x_1 - 12 \cdot x_2 + 15 = 0$$

Ερμηνεία: η λέξη 1 έχει σκορ 5, η λέξη 2 έχει σκορ -12, το *bias* είναι 15.

Βηματική συνάρτηση



$$\phi(x) = \begin{cases} 1, & \text{αν } x \geq 0, \\ 0, & \text{αν } x < 0. \end{cases}$$

Για την περίπτωσή μας η έξοδος θα είναι:

$$\hat{y} = \phi(a \cdot x_1 - b \cdot x_2 + c)$$

Γενίκευση

Μια γενική ευθεία απόφασης είναι η εξής:

$$w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_n \cdot x_n + b = 0$$

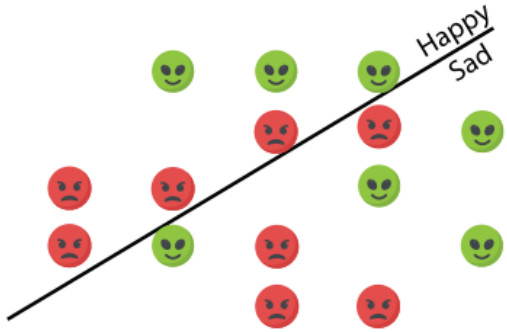
Άρα η πρόβλεψη θα δίνεται από:

$$\hat{y} = \phi(w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_n \cdot x_n + b)$$

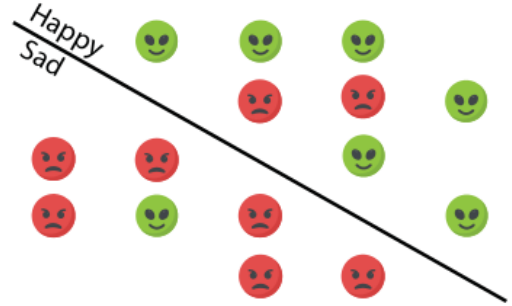
Η επιφάνεια απόφασης θα ανήκει στον n-διάστατο χώρο και δεν οπτικοποιείται (παρά μόνο οι προβολές της).

Συνάρτηση σφάλματος

Πώς συγκρίνουμε ταξινομητές; Με κατάλληλη συνάρτηση σφάλματος.



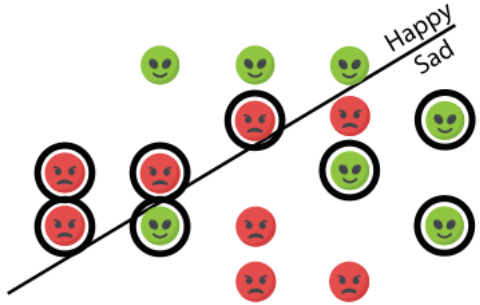
Bad classifier



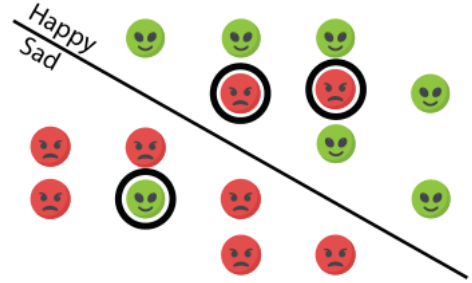
Good classifier

Συνάρτηση σφάλματος

Συνάρτηση σφάλματος 1: Μέτρηση πλήθους σφαλμάτων



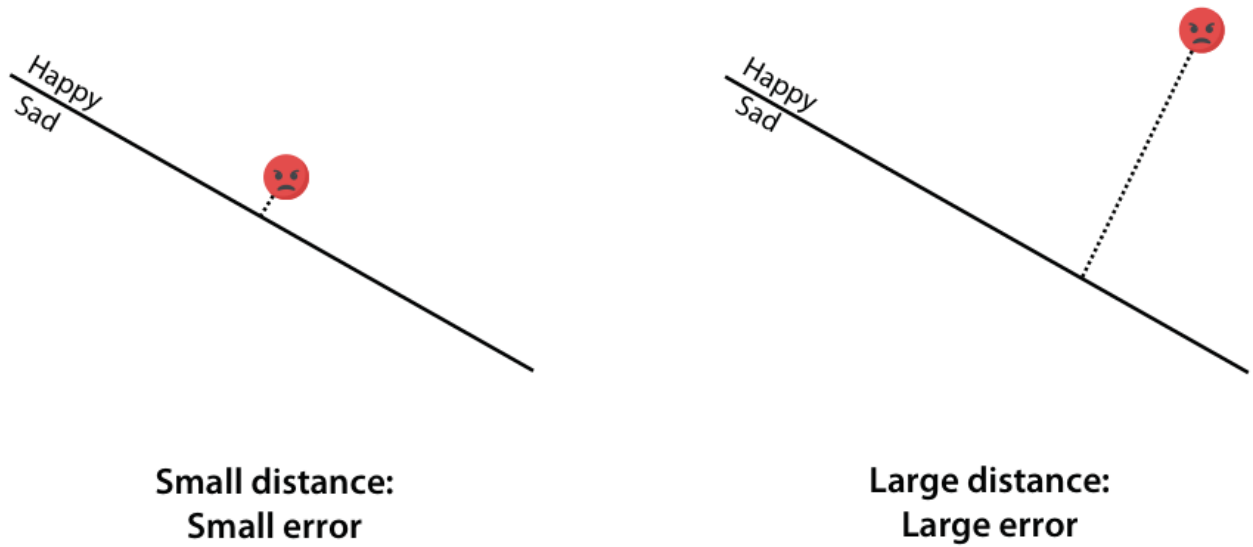
Bad classifier
Error: 8



Good classifier
Error: 3

Συνάρτηση σφάλματος

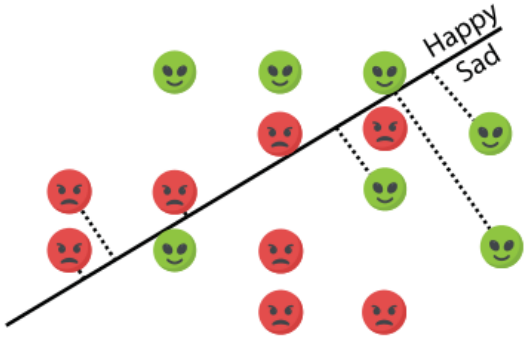
Συνάρτηση σφάλματος 2: Μέτρηση απόστασης από ευθεία απόφασης



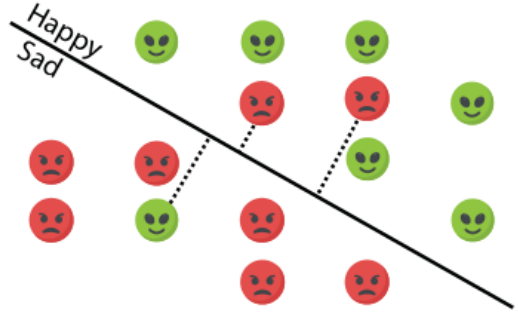
Χρήσιμο σφάλμα για βελτιστοποίηση του σφάλματος με επαναληπτικές μεθόδους π.χ. gradient descent.

Συνάρτηση σφάλματος

Συνάρτηση σφάλματος 2: Μέτρηση απόστασης από ευθεία απόφασης



Bad classifier
Error: .....



Good classifier
Error: .....

Συνάρτηση σφάλματος

Συνάρτηση σφάλματος 3: ιδιότητες

1. Τα σημεία που βρίσκονται στο όριο έχουν σκορ ίσο με 0.
2. Τα σημεία στη θετική ζώνη έχουν θετικό σκορ.
3. Τα σημεία στη αρνητική ζώνη έχουν αρνητικό σκορ.
4. Τα σημεία που βρίσκονται κοντά στο όριο έχουν σκορ με μικρή απόλυτη τιμή.
5. Τα σημεία που βρίσκονται μακριά από το όριο έχουν σκορ με μεγάλη απόλυτη τιμή.

Συνάρτηση σφάλματος

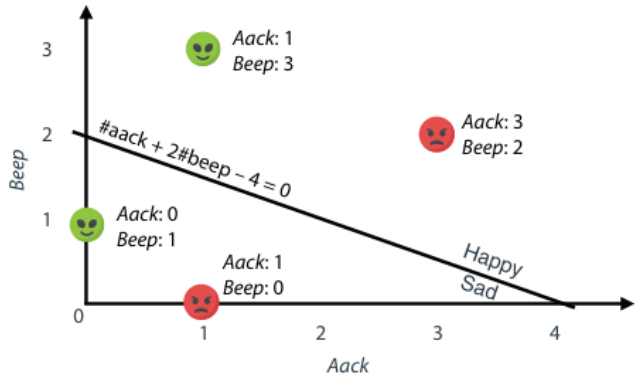
Συνάρτηση σφάλματος 3: ιδιότητες

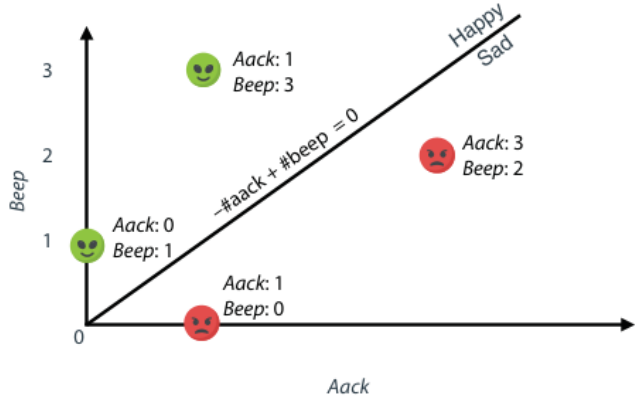
1. Αν η πρόταση / δείγμα ταξινομείται σωστά τότε το σφάλμα είναι μηδέν.
2. Αν η πρόταση / δείγμα δεν ταξινομείται σωστά τότε το σφάλμα είναι:

$$\|w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_n \cdot x_n + b\|$$

3. Συνολικά υπολογίζουμε τη μέση τιμή των σφαλμάτων για όλο το σύνολο δεδομένων.

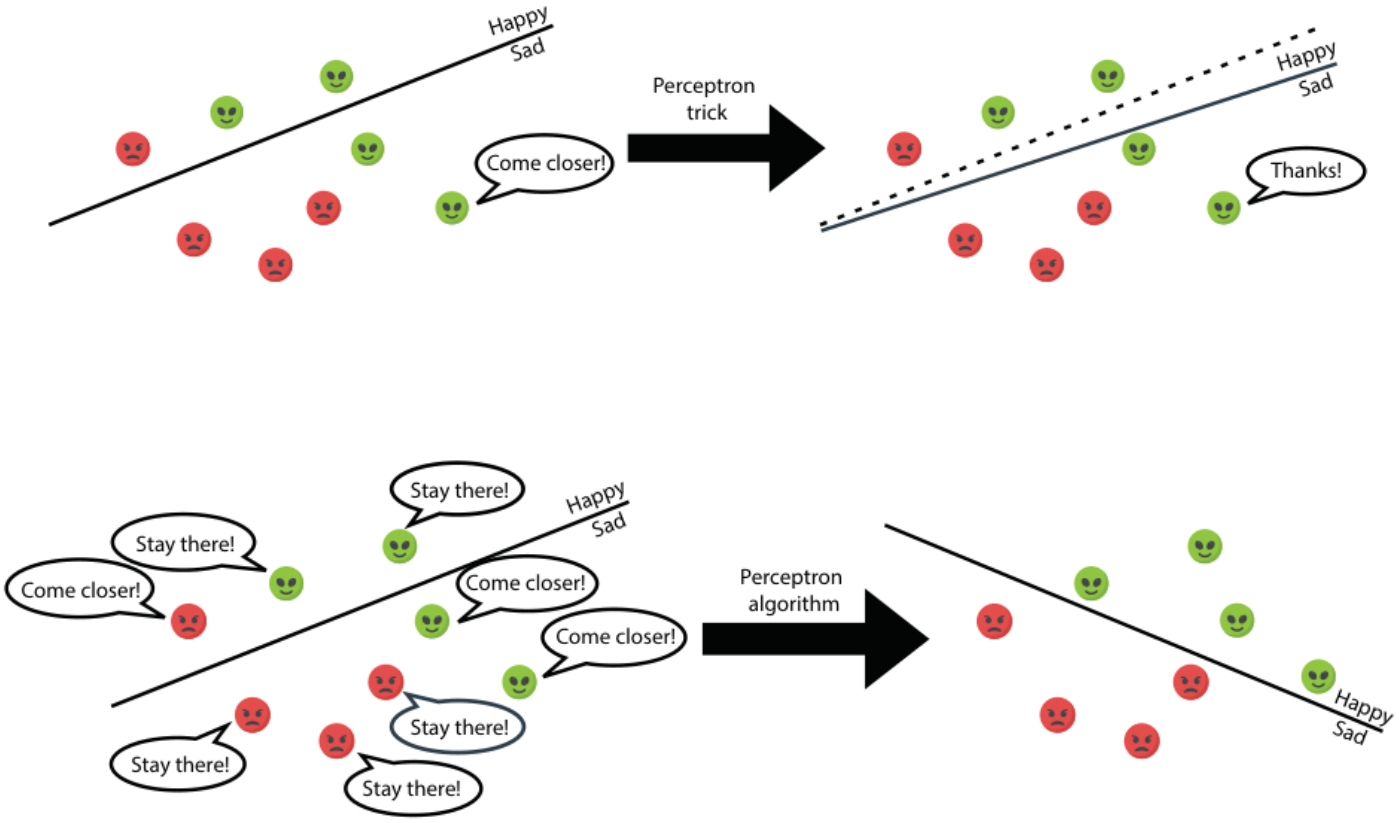
Συνάρτηση σφάλματος: Παράδειγμα





Sentence (x_{aack}, x_{beep})	**Label** y	**Classifier 1 score** $1x_{aack} + 2x_{beep} - 4$	**Classifier 1 prediction** $step(1x_{aack} + 2x_{beep} - 4)$	**Classifier 1 error**	**Classifier 2 score** $-x_{aack} + 2x_{beep}$	**Classifier 2 prediction** $step(-x_{aack} + 2x_{beep})$	**Classifier 2 error**
(1,0)	Sad (0)	–3	0 (correct)	0	–1	0 (correct)	0
(0,1)	Happy (1)	-2	0 (incorrect)	2	1	1 (correct)	0
(1,3)	Happy (1)	3	1 (correct)	3	2	1 (correct)	0
(3,2)	Sad (0)	3	1 (incorrect)	0	–1	0 (correct)	0
Mean perceptron error				1.25			0

Αλγόριθμος *perceptron*



Αλγόριθμος *perceptron*

Inputs:

- A perceptron with weights a, b, and bias c
- A point with coordinates (x_1, x_2) and label y
- A small value η (the learning rate)

Output:

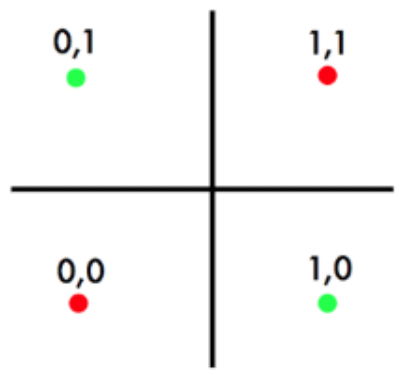
- A perceptron with new weights a', b', and bias c'

Procedure:

- The prediction the perceptron makes at the point is $\hat{y} = step(ax_1 + bx_2 + c)$.
- **Return** the perceptron with the following weights and bias:
  - $a' = a + \eta(y - \hat{y})x_1$
  - $b' = b + \eta(y - \hat{y})x_2$
  - $c' = c + \eta(y - \hat{y})$

Περιορισμοί Αλγόριθμου *perceptron*

- Ο αλγόριθμος λειτουργεί όταν το πρόβλημα είναι γραμμικά διαχωρίσιμο.
- Τι συμβαίνει όμως όταν αυτό δεν ισχύει; π.χ. *XOR*

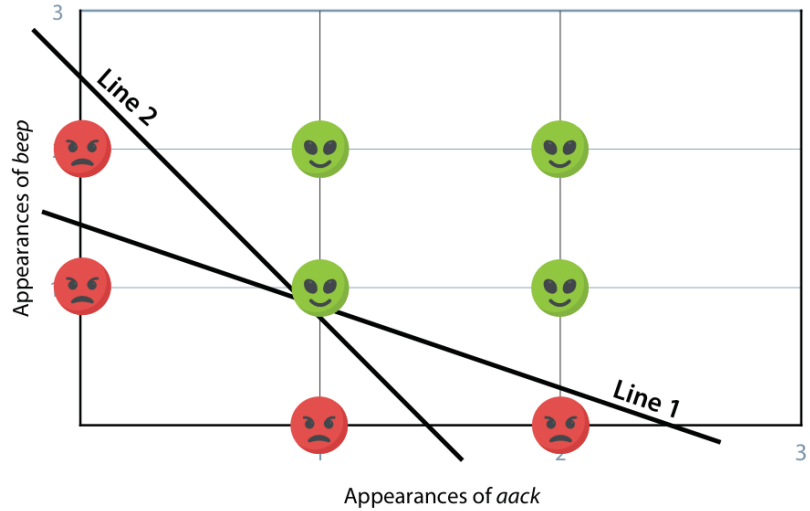


XOR

Σενάριο 3: μια άλλη διάλεκτος

Sentence	*Aack*	*Beep*	Mood
"Aack"	1	0	Sad
"Aack aack"	2	0	Sad
"Beep"	0	1	Sad
"Beep beep"	0	2	Sad
"Aack beep"	1	1	Happy
"Aack aack beep"	2	1	Happy
"Beep aack beep"	1	2	Happy
"Beep aack beep aack"	2	2	Happy

Σενάριο 3: μια άλλη διάλεκτος



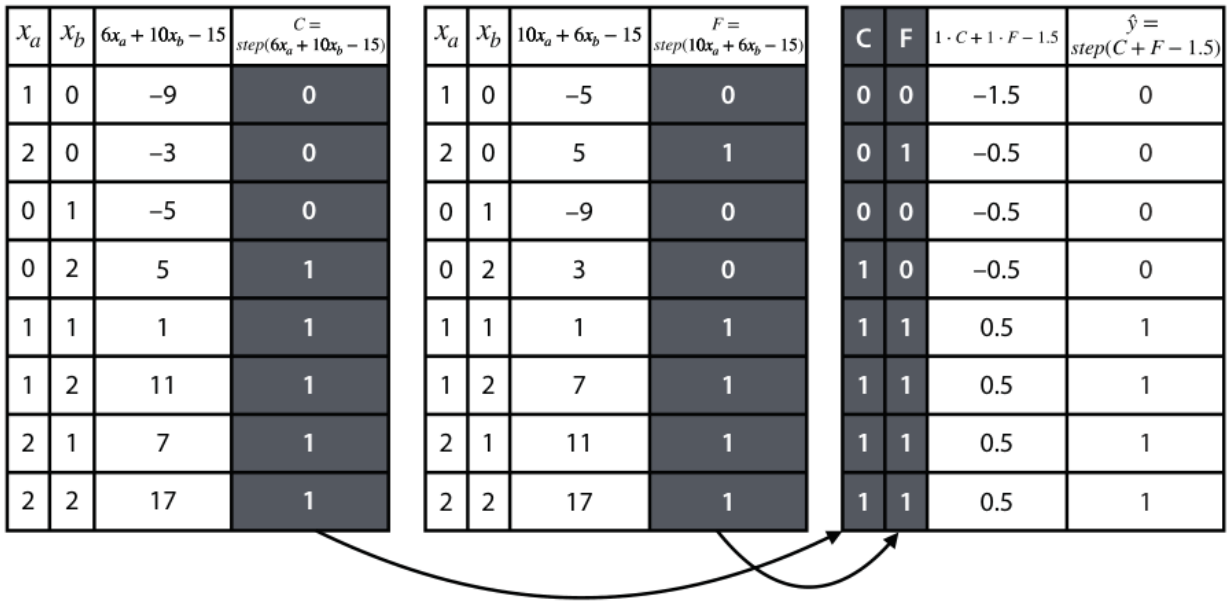
Γραμμή 1:

$$6 \cdot x_a + 10 \cdot x_b - 15 = 0$$

Γραμμή 2:

$$10 \cdot x_a + 6 \cdot x_b - 15 = 0$$

Σενάριο 3: μια άλλη διάλεκτος



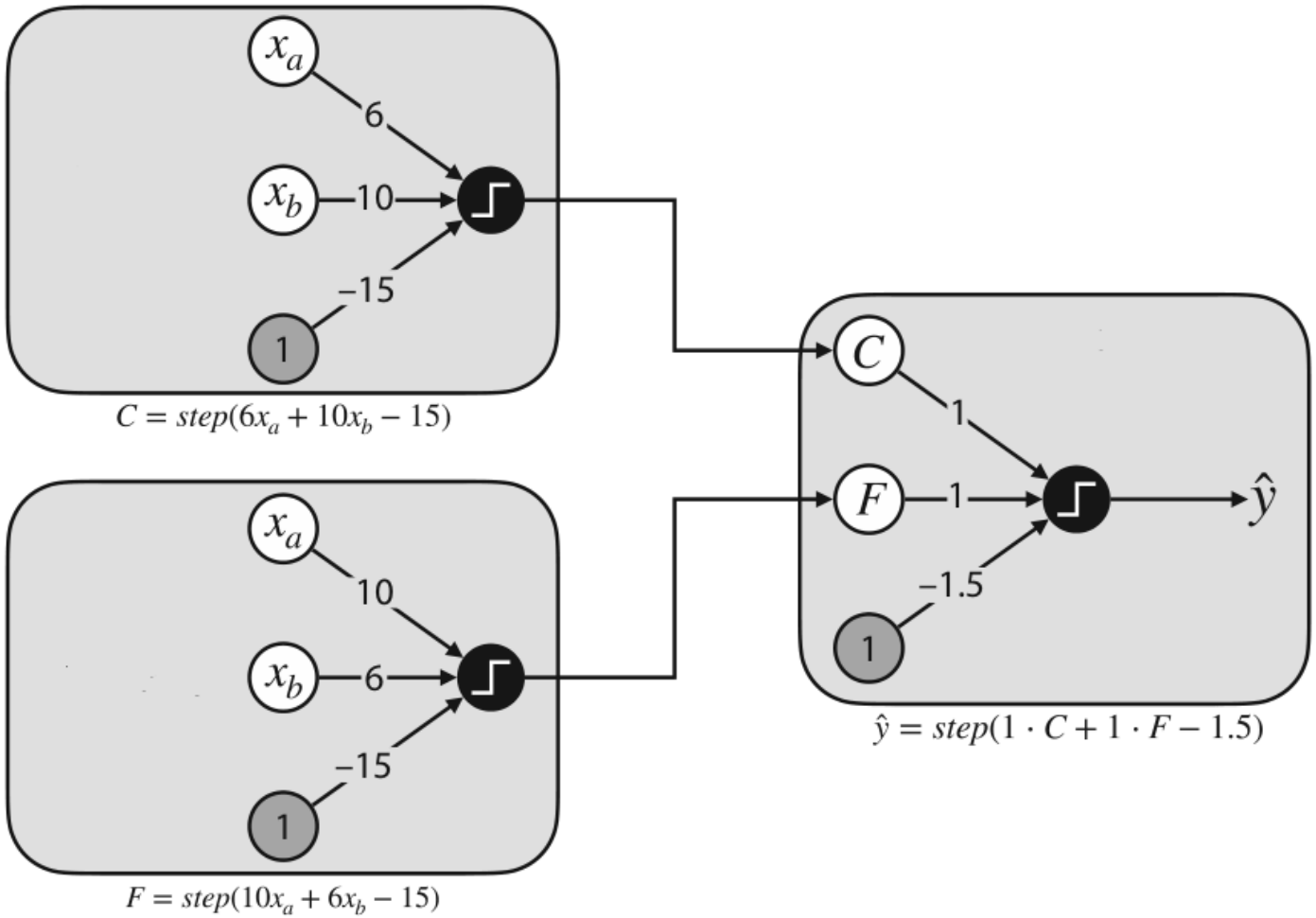
x_a	x_b	$6x_a + 10x_b - 15$	$C = step(6x_a + 10x_b - 15)$
1	0	−9	0
2	0	−3	0
0	1	−5	0
0	2	5	1
1	1	1	1
1	2	11	1
2	1	7	1
2	2	17	1

x_a	x_b	$10x_a + 6x_b - 15$	$F = step(10x_a + 6x_b - 15)$
1	0	−5	0
2	0	5	1
0	1	−9	0
0	2	3	0
1	1	1	1
1	2	7	1
2	1	11	1
2	2	17	1

C	F	$1 \cdot C + 1 \cdot F - 1.5$	$\hat{y} = step(C + F - 1.5)$
0	0	−1.5	0
0	1	−0.5	0
0	0	−0.5	0
1	0	−0.5	0
1	1	0.5	1
1	1	0.5	1
1	1	0.5	1
1	1	0.5	1

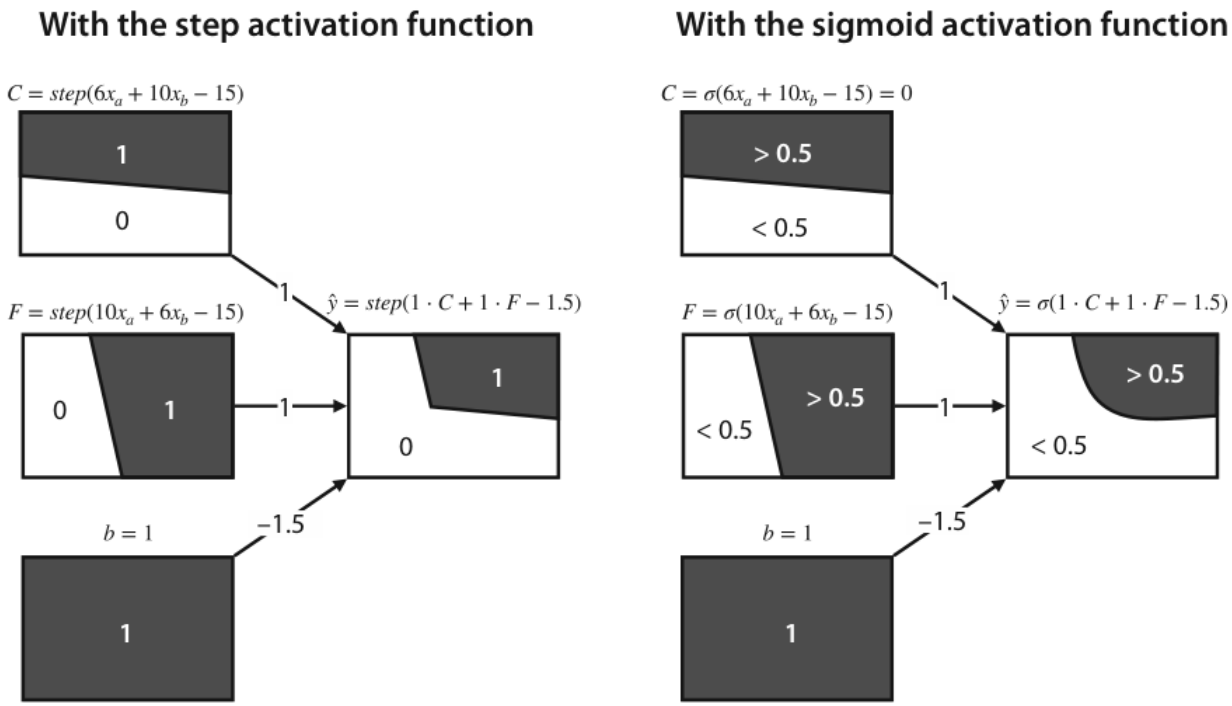
Σενάριο 3: μια άλλη διάλεκτος

Πολυεπίπεδο δίκτυο perceptron (Multi-Layer Perceptron - MLP)

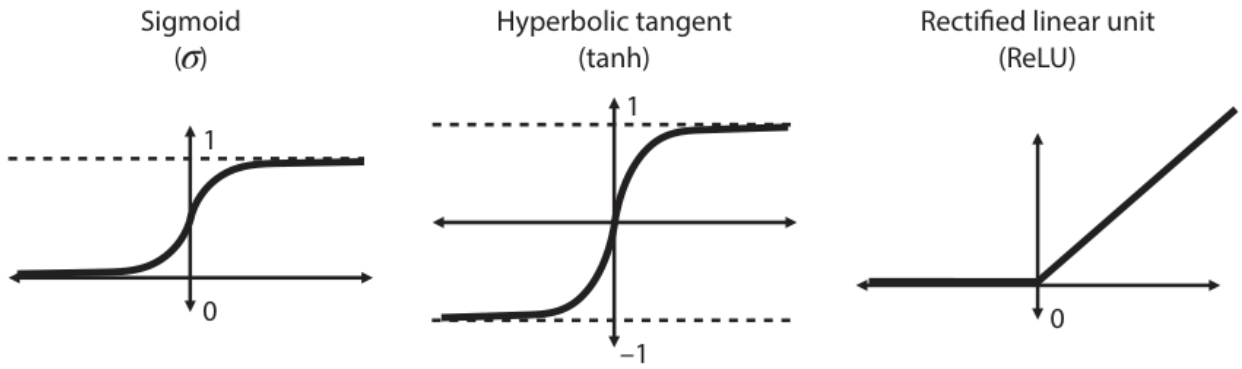


Σενάριο 3: μια άλλη διάλεκτος

Ο ρόλος της συνάρτησης ενεργοποίησης



Συναρτήσεις ενεργοποίησης



Συνάρτηση Σιγμοειδούς

Ορισμός:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Χαρακτηριστικά:

- Επιστρέφει τιμές στο διάστημα: $[0, 1]$.
- Χρήσιμη για προβλήματα δυαδικής ταξινόμησης.
- Μη γραμμική, κατάλληλη για μη γραμμικές σχέσεις.

Μειονεκτήματα:

- Προκαλεί το πρόβλημα του vanishing gradient.
- Οι εξόδοι δεν είναι συμμετρικές γύρω από το 0.

Συνάρτηση *Tanh* (Υπερβολική Εφαπτομένη)

Ορισμός:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Χαρακτηριστικά:

- Επιστρέφει τιμές στο διάστημα: $[-1, 1]$.
- Χρήσιμη όταν οι εισόδοι είναι κανονικοποιημένες γύρω από το 0.
- Μη γραμμική, κατάλληλη για μη γραμμικές σχέσεις.

Μειονεκτήματα:

- Επηρεάζεται από το πρόβλημα του vanishing gradient.

Συνάρτηση ReLU (Rectified Linear Unit)

Ορισμός:

$$f(x) = \begin{cases} x, & \text{αν } x > 0, \\ 0, & \text{αν } x \leq 0. \end{cases}$$

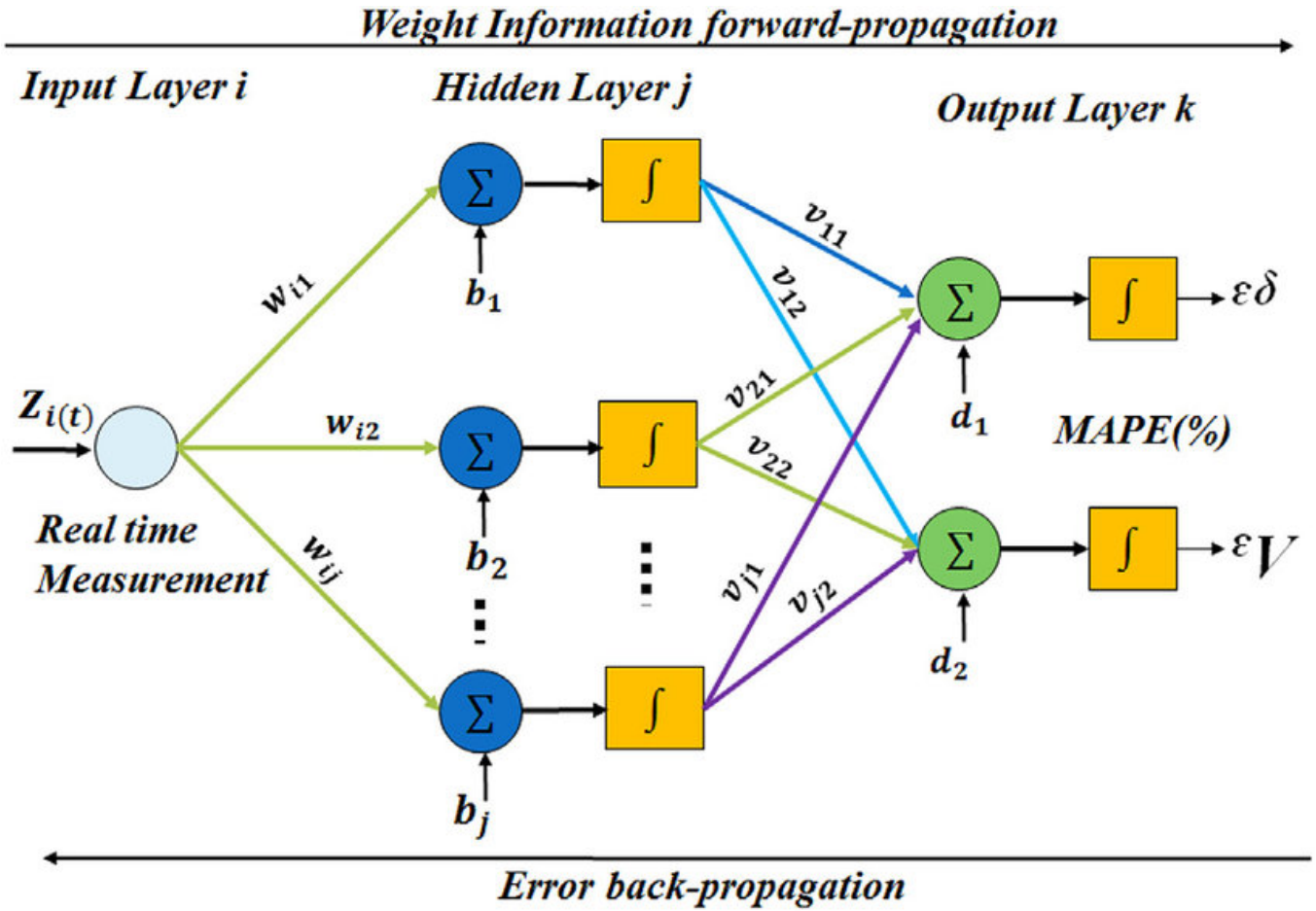
Χαρακτηριστικά:

- Επιστρέφει τιμές στο διάστημα: $[0, \infty)$.
- Πολύ αποδοτική για βαθιά νευρωνικά δίκτυα.
- Εύκολη υπολογιστικά.

Μειονεκτήματα:

- Δεν ενεργοποιείται για $x \leq 0$ (dying ReLU problem).

Αρχιτεκτονική



Εισαγωγή στο *Backpropagation*

Τι είναι το *Backpropagation*;

- Είναι η διαδικασία υπολογισμού των παραγώγων (*gradients*) για την ενημέρωση των βαρών ενός νευρωνικού δικτύου.
- Χρησιμοποιεί τον κανόνα της αλυσίδας (chain rule) για να διαδώσει τα σφάλματα από την έξοδο προς την είσοδο.

Εισαγωγή στο *Backpropagation*

Βασικά Βήματα:

1. **Προώθηση (Forward Pass):**
   - Υπολογισμός της εξόδου του δικτύου.
   - Σύγκριση με την επιθυμητή έξοδο (*label*) για υπολογισμό του σφάλματος.
2. **Διάδοση Πίσω (Backward Pass):**
   - Υπολογισμός των παραγώγων (gradients) του σφάλματος για κάθε βάρος.
3. **Ενημέρωση Βαρών:**
   - Ενημέρωση των βαρών με τη χρήση ενός βελτιστοποιητή (π.χ. Gradient Descent).

$$w \leftarrow w - \eta \cdot \nabla_w$$

όπου η: ρυθμός μάθησης.

Σκοπός:

- Ελαχιστοποίηση της συνάρτησης κόστους (Loss Function).

Βαθιά Νευρωνικά Δίκτυα

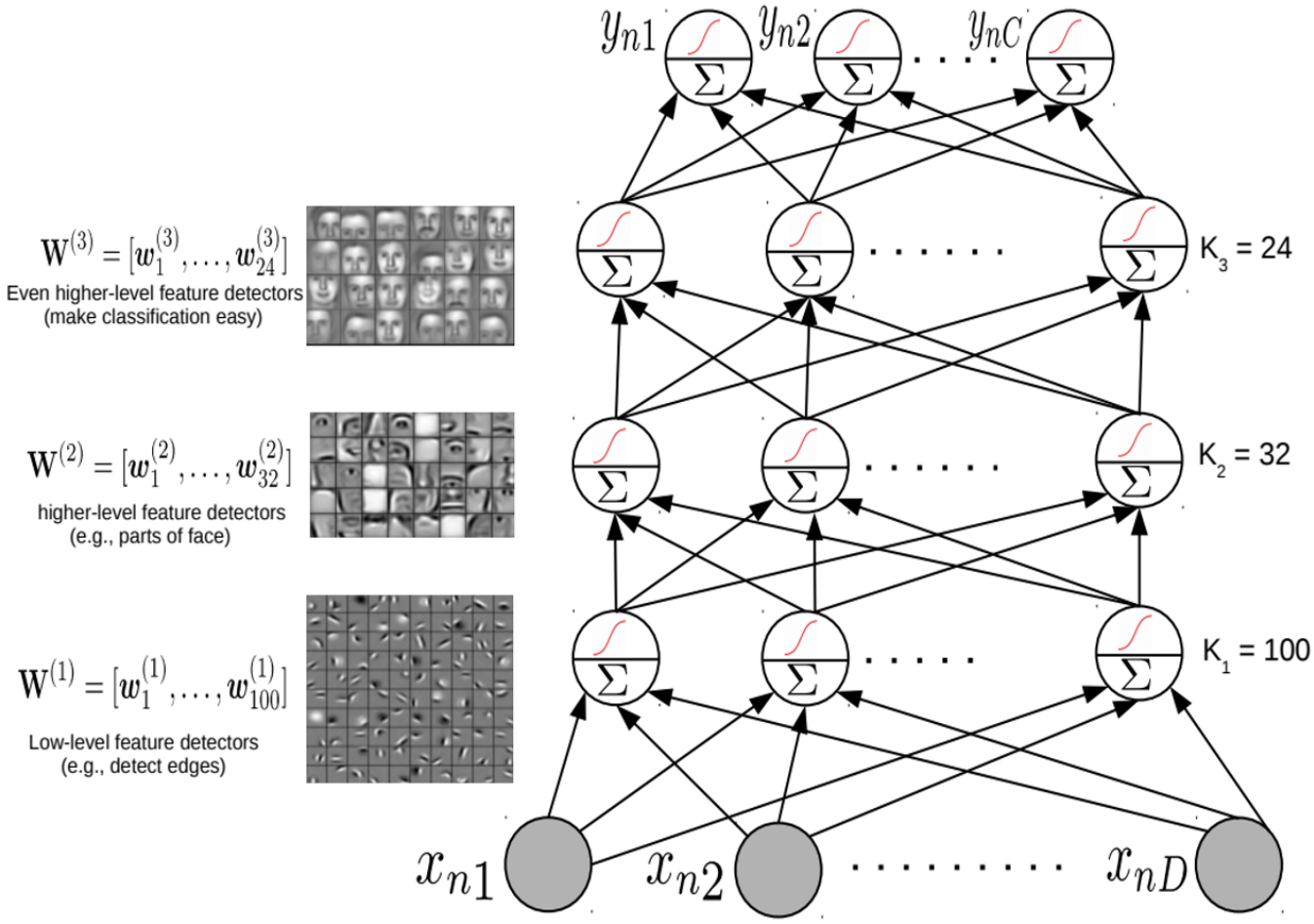
- Με τον όρο Βαθιά Νευρωνικά Δίκτυα (Deep NNs) αναφερόμαστε στα Νευρωνικά Δίκτυα πολλών επιπέδων που στόχο έχουν την προσομοίωση ενός ανθρώπινου εγκεφάλου για την επίλυση ενός συγκεκριμένου προβλήματος.
- Τα νευρωνικά δίκτυα είναι γνωστά από το 1980 και αποτελούν μία πτυχή της Τεχνητής Νοημοσύνης. Η βασική διαφορά τους από τα δίκτυα που χρησιμοποιούνται στο *DL* είναι ότι αποτελούνται από ένα επίπεδο εισόδου (*input*), ένα επίπεδο εξόδου (ουτπυτ) και συνήθως ένα κρυφό (*hidden*) επίπεδο. Αντίθετα, στο *DL* χρησιμοποιούνται δίκτυα με πολλά κρυφά επίπεδα, γεγονός που τα καθιστά πολύ πιο ισχυρά και αποτελεσματικά.
- Στα απλά νευρωνικά δίκτυα ο προγραμματιστής πρέπει να εξάγει κάποια χαρακτηριστικά από τα δεδομένα ώστε να σχηματίσει ένα μοντέλο που περιγράφει και κατηγοριοποιεί τα δεδομένα ενώ στο *DL* η παραπάνω διαδικασία γίνεται αυτόματα από το δίκτυο.

Βαθιά Νευρωνικά Δίκτυα

Γιατί όμως το *DL* άρχισε να χρησιμοποιείται τα τελευταία χρόνια και όχι παλαιότερα;

- Η υπολογιστική πολυπλοκότητα που χρειάζεται για την εκπαίδευση δικτύων με εκατομμύρια νευρώνες είναι πολύ μεγάλη και θα ήταν εξαιρετικά δύσκολο για έναν υπολογιστή της δεκαετίας του 90 να τη φέρει εις πέρας.
  - Στις μέρες μας η εκπαίδευση τέτοιων δικτύων γίνεται πολύ γρηγορότερα με κάρτες γραφικών που έχουν τη δυνατότητα να επεξεργάζονται τα δεδομένα παράλληλα και σε πολύ υψηλές ταχύτητες.
- Η πληθώρα δεδομένων από πηγές όπως το διαδίκτυο μπορεί να χρησιμοποιηθεί για εκπαίδευση.
- Καταλληλότερες συναρτήσεις ενεργοποίησης.

Βαθιά Νευρωνικά Δίκτυα



Βιβλιογραφία

1. Luis Serrano, Grokking Machine Learning, Manning, 2021
2. Σ. Λυκοθανάσης, Σημειώσεις Θεωρία Αποφάσεων