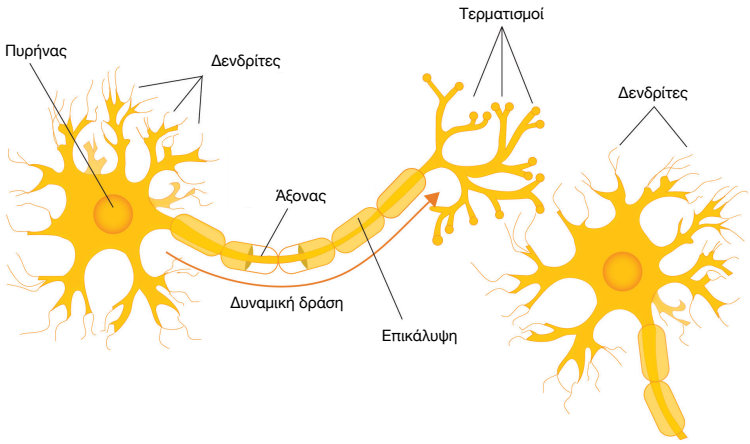
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

# Νευρωνικά Δίκτυα και Ευφυή Υπολογιστικά Συστήματα

## ΔΙΚΤΥΑ PERCEPTRON

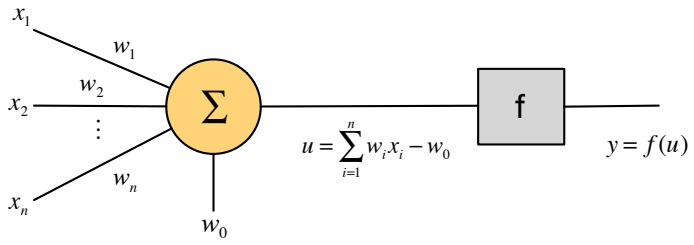
## Μοντέλο βιολογικού νευρώνα






## Μονοστρωματικό perceptron






- $x_i, w_i, y$ Είσοδοι, συναπτικά βάρη, έξοδος
- $u$ Διέγερση (γινόμενο διανυσμάτων εισόδου-βαρών)
- $f$ Συνάρτηση ενεργοποίησης
- $w_0$ Κατώφλι ενεργοποίησης

## Διέγερση perceptron



Συνθήκες διέγερσης

$$u > 0, \quad \alpha\nu \quad \sum_{i=1}^{n} w_i x_i > w_0$$

$$u = 0, \quad \alpha\nu \quad \sum_{i=1}^{n} w_i x_i = w_0$$

$$u < 0, \quad \alpha\nu \quad \sum_{i=1}^{n} w_i x_i < w_0$$

Συνάρτηση ενεργοποίησης

Κλασική

$$f(u) = \begin{cases} 1, & \alpha\nu \ u > 0 \\ 0, & \alpha\nu \ u \le 0 \end{cases}$$

Διπολική

$$f(u) = \begin{cases} 1, & \alpha\nu \ u > 0 \\ -1, & \alpha\nu \ u \le 0 \end{cases}$$

## Διέγερση perceptron

5

Διανύσματα βαρών και εισόδου

$$\mathbf{w} = [w_0, w_1, \ldots, w_n]$$

$$\mathbf{x} = [x_0, x_1, \ldots, x_n]$$

- Κατώφλι $w_0$
- Πόλωση $x_0 = -1$

Γράφουμε: $\boxed{u = \mathbf{w}^T \mathbf{x}}$

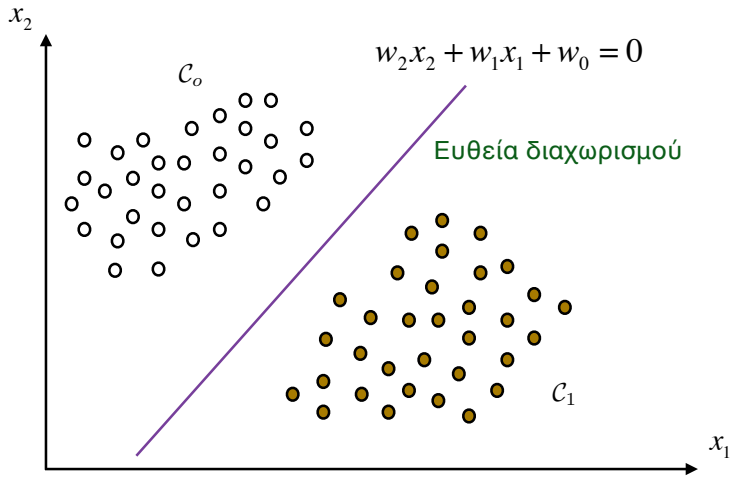
Παρατήρηση

Η εξίσωση της συνάρτησης ενεργοποίησης χωρίζει σε δύο υπερεπίπεδα το χώρο $\mathbb{R}^n$:

$$u > 0 \quad u < 0$$

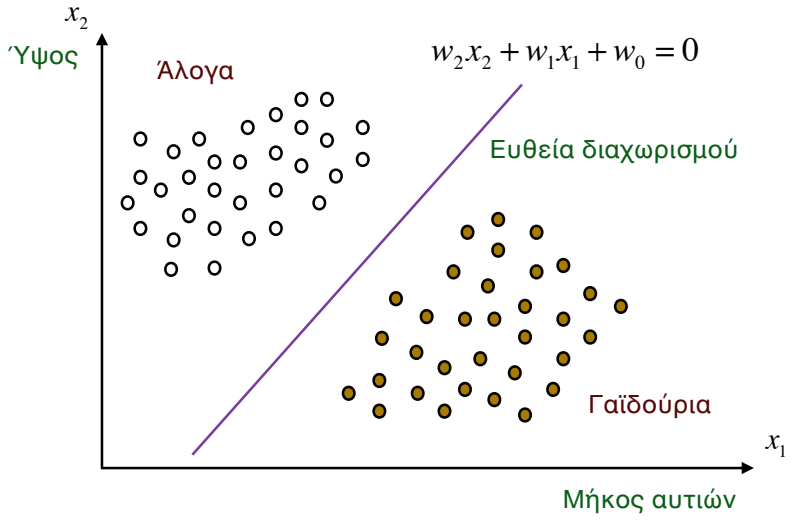
## Ταξινόμηση με perceptron

6




## Ταξινόμηση με perceptron - Παράδειγμα

7




## Εκπαίδευση perceptron

8

Τυπική διατύπωση προβλήματος

Δίνεται ένα σύνολο ζευγών $(\mathbf{x}_1, d_1), \ldots, (\mathbf{x}_p, d_p)$
με $d_i = -1$ αν $\mathbf{x}_i \in \mathcal{C}_o$ και $d_i = 1$ αν $\mathbf{x}_i \in \mathcal{C}_1$

Ζητάμε την εύρεση των βαρών **w** και του κατωφλίου $w_o$, έτσι ώστε:
$\mathbf{w}^\top \mathbf{x}_i + w_o \geq 0$ αν $d_i = 1$ $(\mathbf{x}_i \in \mathcal{C}_o)$
$\mathbf{w}^\top \mathbf{x}_i + w_o < 0$ αν $d_i = -1$ $(\mathbf{x}_i \in \mathcal{C}_1)$

Υπόθεση

*Υπάρχει* τέτοια ευθεία (οι κλάσεις είναι γραμμικά διαχωρίσιμες)

# Διαδικασία εκπαίδευσης



## Διαδικασία ανάλυσης προτύπων

- Τα δεδομένα παρουσιάζονται επαναληπτικά σε κυκλική σειρά
  - Κάθε κύκλος χρήσης των δεδομένων ονομάζεται *εποχή*
- Σε κάθε βήμα ελέγχουμε αν υπάρχει *σφάλμα ταξινόμησης*
- Τροποποιούμε (σε κάθε βήμα) το επαυξημένο διάνυσμα βαρών, αν υπάρχει σφάλμα ταξινόμησης

| Επανάληψη k | 1 | 2 | … | P | P+1 | P+2 | … | 2P | 2P+1 | … |
|---|---|---|---|---|---|---|---|---|---|---|
| Πρότυπο p | 1 | 2 | … | P | 1 | 2 | … | P | 1 | … |
| | Εποχή 1 | | | | Εποχή 2 | | | | … | |

# Διαδικασία εκπαίδευσης



## Κανόνας σταθερής αύξησης βαρών

Κατά την επανάληψη *k* έχουμε:

Πρότυπο στην είσοδο: $(\mathbf{x}, d)$

Έξοδος perceptron: $y = f\left(\mathbf{w}(k-1)^T \mathbf{x}\right)$

Τότε, τα νέα βάρη δίνονται από τη σχέση:

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \beta(d-y)\mathbf{x}$$

βήμα εκπαίδευσης

# Αλγόριθμος εκπαίδευσης perceptron



- ΒΗΜΑ 1: 1. Αρχικοποίησε το διάνυσμα βαρών τυχαίες τιμές
  2. Δώσε μία μικρή θετική τιμή στο βήμα εκπαίδευσης
  3. Όρισε το μέγιστο αριθμό εποχών
- ΒΗΜΑ 2: Επανέλαβε έως ότου δεν γίνει καμμία αλλαγή βαρών ή έχει συμπληρωθεί ο μέγιστος αριθμός εποχών
  Για κάθε ένα από τα *P* πρότυπα εκτέλεσε
  1. Υπολόγισε την έξοδο του perceptron
  2. Αν υπάρχει σφάλμα προσάρμοσε τα βάρη ως:

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \beta(d-y)\mathbf{x}$$

- ΒΗΜΑ 3: Αν έχει συμπληρωθεί ο μέγιστος αριθμός εποχών επέστρεψε ΣΦΑΛΜΑ, αλλιώς επέστρεψε τα βάρη

# Βελτίωση επίδοσης perceptron



## Ιδιότητα κανόνα σταθερής αύξησης βαρών

- Αν ένα πρότυπο ταξινομήθηκε λανθασμένα, μετά τη διόρθωση των βαρών, την επόμενη φορά ή θα ταξινομηθεί σωστά, ή θα πλησιάζει περισσότερο στο να ταξινομηθεί σωστά

$$\begin{aligned} u_{k,new} &= \mathbf{w}(k)^T \mathbf{x} \\ &= \mathbf{w}(k-1)^T \mathbf{x} + \beta(d-y)\mathbf{x}^T \mathbf{x} \\ &= u_{k,old} + \beta(d-y)\|\mathbf{x}\|^2 \end{aligned}$$

$$\begin{cases} -2, & if\ d=-1, y=1 \\ 2, & if\ d=1, y=-1 \end{cases}$$

## Θεώρημα σύγκλισης

13

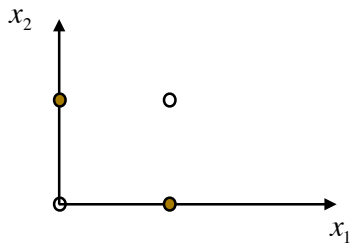
Σύγκλιση του κανόνα perceptron

- Αν το πρόβλημα είναι γραμμικά διαχωρίσιμο, τότε ο κανόνας perceptron συγκλίνει σε πεπερασμένο αριθμό επαναλήψεων

## Παράδειγμα XOR

14

Πρόβλημα μη γραμμικά διαχωρίσιμο




- Συγκλίνει ο αλγόριθμος;
- Βρίσκει λύση;

## Κανόνας ADALINE

15

Adaptive linear element

- Δεν χρησιμοποιείται η μη-γραμμική συνάρτηση ενεργοποίησης
  - Η έξοδος παίρνει συνεχείς τιμές
  - Οι στόχοι μπορούν να παίρνουν συνεχείς τιμές

$$\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(P)}]^T \qquad \mathbf{d} = [d^{(1)}, d^{(2)}, ..., d^{(P)}]^T$$

$$\mathbf{Xw} = \mathbf{d}$$

Επίλυση συστήματος $P$ εξισώσεων με $n$+1 αγνώστους

## Κανόνας ADALINE

16

Adaptive linear element

- Αν $P>n$+1 (συνήθης περίπτωση) το σύστημα μπορεί να μην έχει λύση
- Στην περίπτωση αυτή αναζητούμε προσεγγιστική λύση, χρησιμοποιώντας κριτήριο απόστασης από όλα τα πρότυπα
  - Ελάχιστο τετραγωνικό σφάλμα

$$J = \sum_{i=1}^{P} \left( d^{(i)} - \mathbf{w}^T \mathbf{x}^{(i)} \right)^2$$

Ο αλγόριθμος τερματίζει αν το σφάλμα γίνει μικρότερο από $\varepsilon$

## Αλγόριθμος εκπαίδευσης ADALINE

17

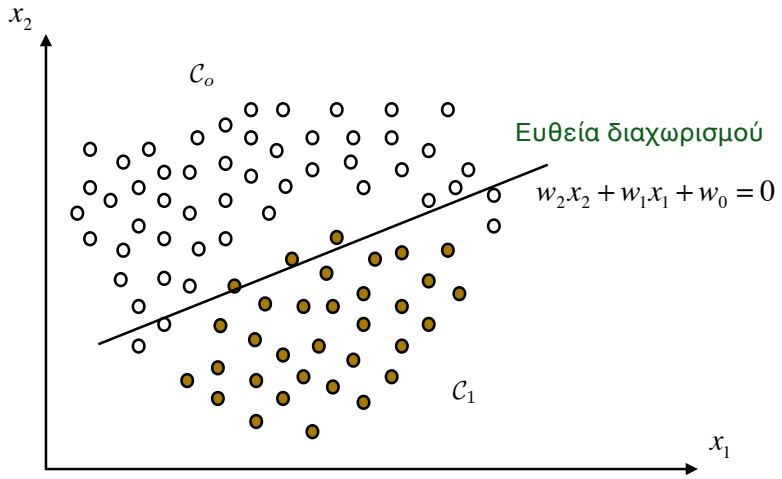
- ΒΗΜΑ 1: 1. Αρχικοποίησε το διάνυσμα βαρών τυχαίες τιμές
  2. Δώσε μία μικρή θετική τιμή στο βήμα εκπαίδευσης
  3. Όρισε ένα όριο *ε* για το σφάλμα εκπαίδευσης
  3. Όρισε το μέγιστο αριθμό εποχών
- ΒΗΜΑ 2: Επανέλαβε έως το μέγιστο αριθμός εποχών
  Για κάθε ένα από τα *P* πρότυπα εκτέλεσε
  1. Υπολόγισε την έξοδο του perceptron
  2. Αν υπάρχει σφάλμα προσάρμοσε τα βάρη ως:

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \beta(d-y)\mathbf{x}$$

  Επέστρεψε τα βάρη αν ισχύει ότι: $\sum_{i=1}^{P}\left(d^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)}\right)^2 < \varepsilon$
- ΒΗΜΑ 3: Επέστρεψε ΣΦΑΛΜΑ

## Ταξινόμηση με ADALINE

18




## Παρατηρήσεις

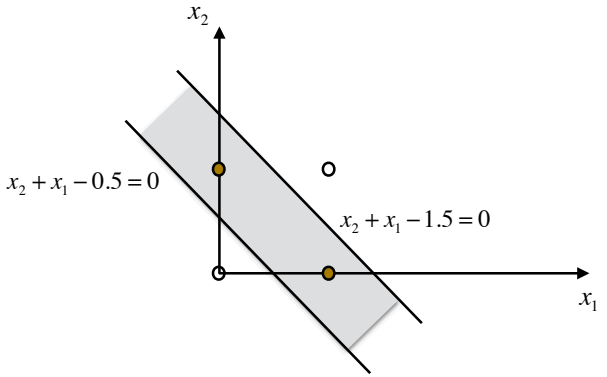
19

- Αν το πρόβλημα είναι γραμμικά διαχωρίσιμο το perceptron βρίσκει πάντα λύση, σε αντίθεση με το adaline
- Αν το πρόβλημα δεν είναι γραμμικά διαχωρίσιμο, το perceptron δεν συγκλίνει, ενώ το adaline *μπορεί* να συγκλίνει επιτρέποντας λανθασμένες ταξινομήσεις
- Το adaline συγκλίνει σε περιπτώσεις μη-γραμμικά διαχωρίσιμων προβλημάτων *μόνο* όταν το πρόβλημα είναι *σχεδόν* γραμμικά διαχωρίσιμο (κάτω από το σφάλμα *ε*)

## Παράδειγμα XOR

20

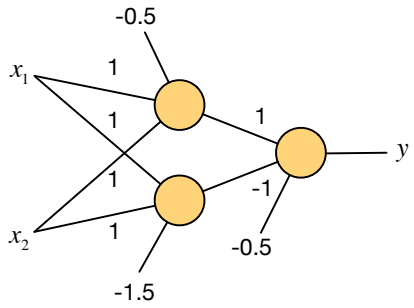
Χρήση περισσότερων νευρώνων



## Παράδειγμα XOR

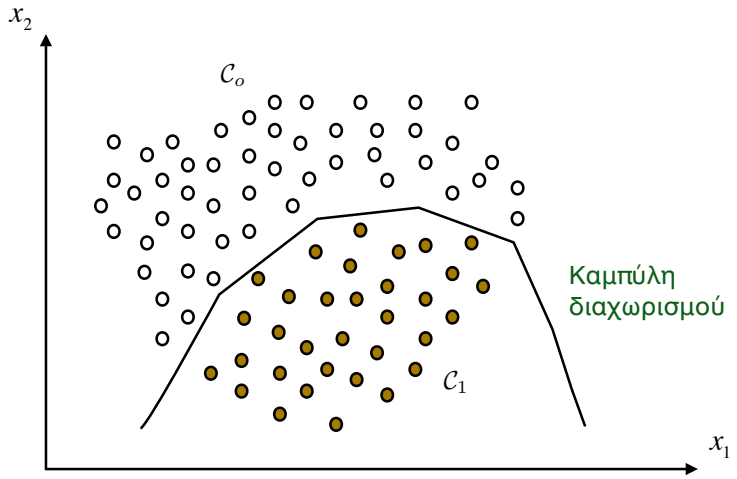
21

Ταξινόμηση με δίκτυο δύο επιπέδων νευρώνων perceptron




## Πολυστρωματικό perceptron (MLP)

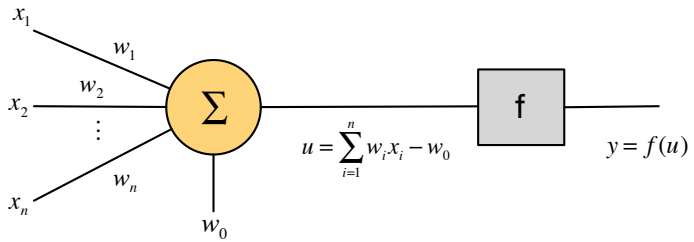
22




## Πολυστρωματικό perceptron

23

Μορφή νευρώνων




- $x_i, w_i, y$ Είσοδοι, συναπτικά βάρη, έξοδος
- $u$ Διέγερση (γινόμενο διανυσμάτων εισόδου-βαρών)
- $f$ Συνάρτηση ενεργοποίησης (συνήθως σιγμοειδής)
- $w_0$ Κατώφλι ενεργοποίησης

## Πολυστρωματικό perceptron (MLP)

24

Συνάρτηση ενεργοποίησης

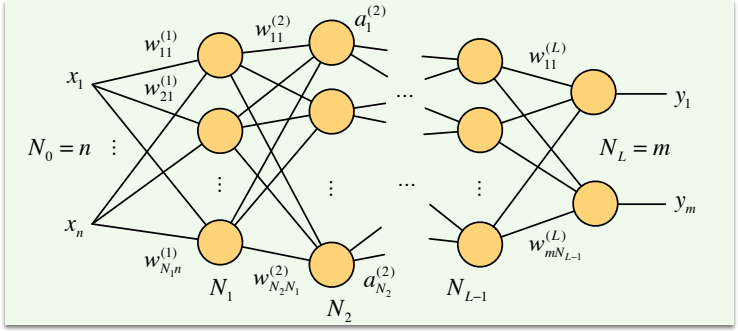
Σιγμοειδής $f(u) = 1/(1+e^{-u})$

Υπερβολική εφαπτομένη $f(u) = \dfrac{e^{u} - e^{-u}}{e^{u} + e^{-u}}$

"Μαλακές", παραγωγίσιμες συναρτήσεις που δημιουργούν ομαλές επιφάνειες εξόδου

## Τοπολογία MLP

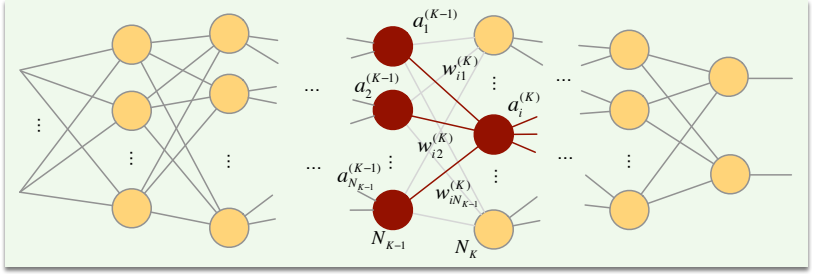
25




- $N_i$ Πλήθος νευρώνων στο στρώμα *i*
- $w_{ij}^{(K)}$ Βάρος από το νευρώνα *j* του στρώματος *K*-1 στο νευρώνα *i* του στρώματος *K*
- $a_i^{(K)}$ Ενεργοποίηση νευρώνα *i* του στρώματος *K*

## Ανάκληση δικτύων MLP

26




Ενεργοποίηση νευρώνα *i* του στρώματος *K*

$$a_i^K = f\left(\sum_{j=1}^{N_{K-1}} w_{ij}^K a_j^{K-1} + w_{i0}^K\right)$$

## Διαδικασία υπολογισμού εξόδου MLP

27

**1.** Επανέλαβε για κάθε στρώμα *K* του δικτύου, από *1* έως *L*

**1.1.** Επανέλαβε για κάθε νευρώνα *i* του στρώματος *K*

**1.1.1.** Υπολόγισε την έξοδο του νευρώνα από τη σχέση:

$$a_i^K = f\left(\sum_{j=1}^{N_{K-1}} w_{ij}^K a_j^{K-1} + w_{i0}^K\right)$$

**1.1.2.** Καταχώρησέ τη στο διάνυσμα:

$$\left[a_1^K, a_2^K, ..., a_{\mathrm{N_K}}^K\right]$$

**2.** Επέστρεψε ως έξοδο το διάνυσμα: $\left[a_1^L, a_2^L, ..., a_m^L\right]$

## Θεώρημα προσέγγισης

28

Έστω $\phi$ μία συνάρτηση *n* μεταβλητών, ορισμένη στο διάστημα [0,1]. Τότε υπάρχει κάποιος ακέραιος αριθμός *k* τέτοιος ώστε:

$$\left|\phi(x_1, x_2, ..., x_n) - f\left(\sum_{j=1}^{n} w_{ij} x_j + w_0\right)\right| < \varepsilon$$

για οποιοδήποτε $\varepsilon$>0 και οποιαδήποτε $x_1, x_2, ..., x_n \in [0,1]$

- Αυτό σημαίνει ότι *υπάρχει* πάντα κάποιο MLP με δύο στρώματα που προσεγγίζει μία οποιαδήποτε συνάρτηση, με όσο μικρό σφάλμα θέλουμε

## Εκπαίδευση δικτύων MLP



Τυπική διατύπωση προβλήματος

Δίνεται ένα σύνολο δεδομένων εισόδου-επιθυμητής εξόδου

$$(\mathbf{x}^{(1)},\mathbf{d}^{(1)}),(\mathbf{x}^{(2)},\mathbf{d}^{(2)}),...,(\mathbf{x}^{(P)},\mathbf{d}^{(P)})$$

όπου $\mathbf{x}^{(i)}=[x_1^{(P)},x_2^{(P)},...,x_n^{(P)}], \mathbf{d}^{(i)}=[d_1^{(P)},d_2^{(P)},...,d_m^{(P)}]$

Ζητάμε την εύρεση όλων των βαρών και κατωφλίων ενός δικτύου MLP, έτσι ώστε:

$$J=\frac{1}{P}\sum_{i=1}^{P}\left\|\mathbf{d}^{(i)}-\mathbf{y}^{(i)}\right\|^2=\frac{1}{P}\sum_{i=1}^{P}\sum_{j=1}^{m}\left(d_j^{(i)}-y_j^{(i)}\right)^2<\varepsilon$$

όπου $\mathbf{y}=[y_1,y_2,...,y_m]$ η έξοδος του δικτύου

## Παρατηρήσεις



- Έστω ότι τα δεδομένα αποτελούν τιμές (για τις αντίστοιχες εισόδους) μίας πραγματικής συνάρτησης που έχουμε να προσεγγίσουμε
  - Τότε, με βάση το θεώρημα προσέγγισης, υπάρχει κάποιο δίκτυο MLP που μπορεί να προσεγγίσει τη συνάρτηση αυτή
- *Προσοχή*
  - Αν η τοπολογία του δικτύου θεωρείται *δεδομένη,* δεν είμαστε σίγουροι ότι *για αυτή την τοπολογία* υπάρχουν βάρη που αποτελούν λύση
  - Το ότι υπάρχουν βάρη τέτοια ώστε να λύνεται το πρόβλημα, δεν σημαίνει απαραίτητα ότι μπορούμε να καταστρώσουμε μία μέθοδο που βρίσκει τα βάρη αυτά σε *κάθε* περίπτωση

## Ιδέα αλγόριθμου back propagation



- Καταστρώνουμε ένα οργανωμένο πλάνο μικρής τροποποίησης κάθε βάρους του δικτύου, λαμβάνοντας υπόψην το σφάλμα που έχουμε για μία συγκεκριμένη είσοδο, την αντίστοιχη επιθυμητή έξοδο και την ανάκληση του δικτύου
- Εφαρμόζουμε το πλάνο σε *εποχές*, όπως και στο απλό perceptron διατρέχοντας με οργανωμένο τρόπο όλα τα δεδομένα
- Προσαρμόζουμε τα βάρη *ανάλογα με το πόσο συνεισέφεραν στο συνολικό σφάλμα* του δικτύου (credit asignment)
  - Για τον καθορισμό της συνεισφοράς αυτής χρησιμοποιούμε την *παράγωγο* του μέσου τετραγωνικού σφάλματος ως προς το συγκεκριμένο βάρος (gradient descent)

## Μέθοδος κατάβασης δυναμικού



Μεταβολή βαρών

$$\frac{dw_{ij}}{dt}=-\beta\frac{\partial J}{\partial w_{ij}}$$

Για διακριτό χρόνο

$$w_{ij}^{(l)}(k+1)-w_{ij}^{(l)}(k)=-\beta\frac{\partial J}{\partial w_{ij}^{(l)}(k)}$$

όπου $w_{ij}^{(l)}(k)$ το βάρος σύνδεσης του νευρώνα *j* του στρώματος *l-1* με το νευρώνα *i* του στρώματος *l* τη χρονική στιγμή *k*

## Υπολογισμός παραγώγου

33

Απόκριση νευρώνα

$$a_i^{(l)}(k) = f\left(u_i^{(l)}(k)\right)$$

όπου $u_i^{(l)}(k) = \sum_{v=1}^{N_{l-1}} w_{iv}^{(l)}(k) a_v^{(l-1)}(k) + w_{i0}^{(l)}(k)$

Παράγοντας δέλτα

Επομένως

$$\frac{\partial J}{\partial w_{ij}^{(l)}(k)} = \frac{\partial J}{\partial u_i^{(l)}(k)} \frac{\partial u_i^{(l)}(k)}{\partial w_{ij}^{(l)}(k)} = -\delta_i^{(l)}(k) \frac{\partial u_i^{(l)}(k)}{\partial w_{ij}^{(l)}(k)}$$

$$\frac{\partial u_i^{(l)}(k)}{\partial w_{ij}^{(l)}(k)} = a_i^{(l-1)}(k), \; j \neq 0$$

$$\frac{\partial u_i^{(l)}(k)}{\partial w_{ij}^{(l)}(k)} = 1, \; j = 0$$

Τελικά

$$\frac{\partial J}{\partial w_{ij}^{(l)}(k)} = -\delta_i^{(l)}(k) a_j^{(l-1)}(k)$$

## Υπολογισμός παράγοντα δέλτα

34

Στρώμα εξόδου - νευρώνας *i* - χρονική στιγμή *k*

$$\delta_i^{(L)}(k) = \frac{\partial J}{\partial a_i^{(L)}(k)} \frac{\partial a_i^{(L)}(k)}{\partial u_i^{(L)}(k)} = \frac{\partial J}{\partial a_i^{(L)}(k)} \frac{\partial f(u_i^{(L)}(k))}{\partial u_i^{(L)}(k)}$$

Δηλαδή $\delta_i^{(L)}(k) = \left(d_i^{(L)}(k) - y_i(k)\right) f'\left(u_i^{(L)}(k)\right)$

Παράγωγος συνάρτησης ενεργοποίησης

| | | |
|---|---|---|
| Σιγμοειδής | $f(u) = 1/(1+e^{u})$ | $f'(u) = f(u)(1-f(u))$ |
| Υπερβολική εφαπτομένη | $f(u) = (e^{u}-e^{-u})/(e^{u}+e^{-u})$ | $f'(u) = (1+f(u))(1-f(u))$ |
| Γραμμική | $f(u) = u$ | $f'(u) = 1$ |

## Υπολογισμός παράγοντα δέλτα

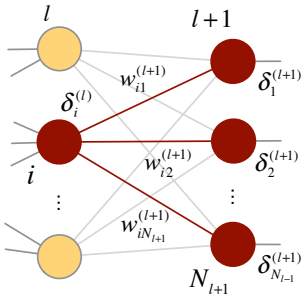
35

Ενδιάμεσο στρώμα *l* - νευρώνας *i* - χρονική στιγμή *k*

$$\delta_i^{(l)}(k) = \frac{\partial J}{\partial u_i^{(l)}(k)} = \sum_{v=1}^{N_{l+1}} \frac{\partial J}{\partial u_v^{(l+1)}(k)} \frac{\partial u_v^{(l+1)}(k)}{\partial a_i^{(l)}(k)} \frac{\partial a_i^{(l)}(k)}{\partial u_i^{(l)}(k)}$$

και μετά από απλές πράξεις

$$\delta_i^{(l)}(k) = \sum_{v=1}^{N_{l+1}} \delta_v^{(l+1)}(k) w_{vi}^{(l+1)}(k) f'\left(u_i^{(l)}(k)\right)$$




Για τον υπολογισμό χρησιμοποιούνται τα σφάλματα του στρώματος *l*+1 (επόμενο στρώμα)

## Αλγόριθμος back-propagation

36

**1.** Αρχικοποίησε όλα τα βάρη του δικτύου σε μικρές τυχαίες τιμές

**2.** Επανέλαβε αυξάνοντας την εποχή, ξεκινώντας από 1

- **2.1.** Επανέλαβε για κάθε πρότυπο, από 1 έως P
  - **2.1.1.** Υπολόγισε τις εξόδους για κάθε στρώμα *K* του δικτύου, από *1* έως *L*
  - **2.1.2.** Υπολόγισε τους παράγοντες δέλτα για τους *m* νευρώνες εξόδου
  - **2.1.3.** Επανέλαβε ανάστροφα για κάθε στρώμα *l* του δικτύου από *L-1* έως *1*
    - **2.1.3.1** Υπολόγισε τους παράγοντες δέλτα για όλους τους νευρώνες του στρώματος, από *1* έως $N_l$
  - **2.1.4.** Ανανέωσε όλα τα βάρη του δικτύου, με βάση τους παράγοντες δέλτα
- **2.2.** Υπολόγισε το σφάλμα εξόδου
- **2.3.** Αν το σφάλμα είναι μικρότερο από *ε* επέστρεψε τα βάρη του δικτύου
- **2.4.** Αν οι εποχή είναι *E* επέστρεψε σφάλμα

## Επίδοση back-propagation

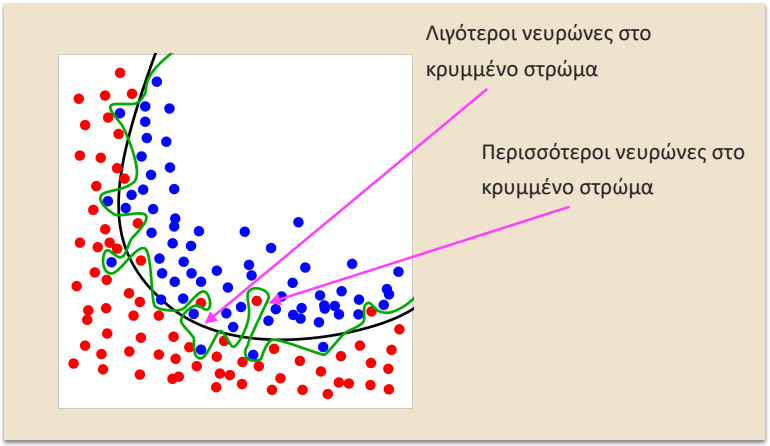
37

- Ο αλγόριθμος back-propagation δεν βρίσκει *πάντα* λύση (παρότι με βάση το θεώρημα προσέγγισης πάντα υπάρχει)
  - Δεν προσεγγίζει *πάντα* τη συνάρτηση εισόδου-εξόδου με σφάλμα μικρότερο του $\varepsilon$
  - Όσο πιο σύνθετη η δομή του δικτύου, τόσο πιο πολλές οι πιθανότητες να βρεθεί λύση (σε κάποιο αριθμό εποχών)
    - Δεν είναι όμως απαραίτητο ότι η λύση αυτή θα είναι πρακτικά καλή (*πρόβλημα overfitting*)
- Ο αλγόριθμος back-propagation δεν βρίσκει *πάντα* τη λύση που δίνει το μικρότερο τετραγωνικό σφάλμα
  - Εξαρτάται από πολλούς παράγοντες όπως η αρχικοποίηση (*πρόβλημα τοπικών ελαχίστων*)

## Πρόβλημα overfitting

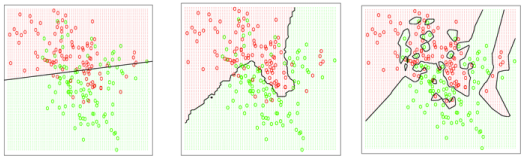
38

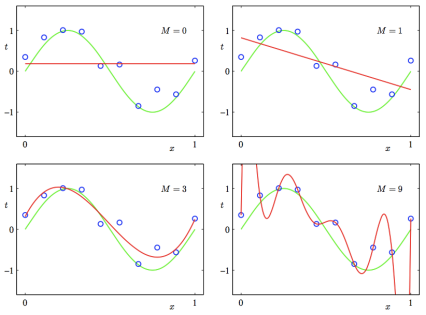
Ταξινόμηση σε δύο κλάσεις

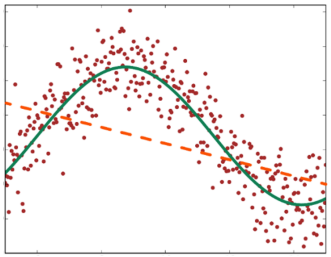



## Πρόβλημα overfitting

39



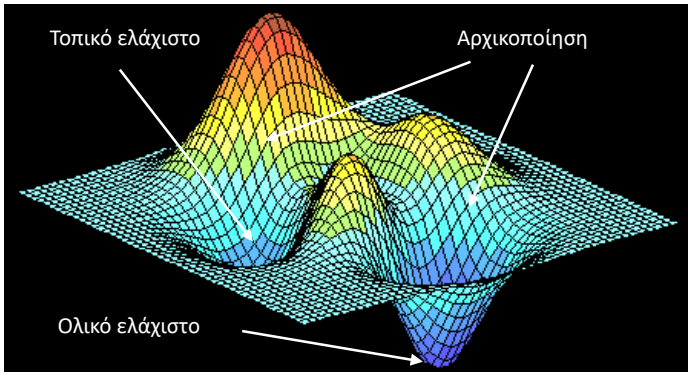


## Πρόβλημα τοπικών ελαχίστων

40

Καμπύλη σφάλματος ως προς τα βάρη του δικτύου



# Παράμετροι που επηρεάζουν



- Δομή δικτύου
  - Αριθμός στρωμάτων
  - Αριθμός νευρώνων σε κάθε στρώμα
- Αρχικές τιμές βαρών
- Βήμα εκμάθησης
- Δεδομένα εκμάθησης
- Εποχές εκμάθησης