

# Υπολογιστική Νοημοσύνη

Ιωάννης Γ. Τσούλος

Τμήμα Πληροφορικής και τηλεπικοινωνιών  
Πανεπιστήμιο Ιωαννίνων

2024

# Περίληψη

- 1 MLP
  - Βασικά στοιχεία
- 2 Μέθοδοι μάθησης
  - Gradient Descent
  - Η μέθοδος RPROP
  - Η μέθοδος ADAM
  - Η μέθοδος Simulated Annealing
- 3 Ειδικά θέματα
  - Κανονικοποίηση τιμών
  - Αρχικοποίηση βαρών
  - Folding

# Ορισμός

- 1 Είναι τεχνητά νευρωνικά δίκτυα τα οποία διαθέτουν ένα ή περισσότερα ενδιάμεσα επίπεδα επεξεργασίας.
- 2 Απαραίτητο στοιχείο τους είναι οι συναρτήσεις ενεργοποίησης
- 3 Μπορούν να χρησιμοποιηθούν για μάθηση συναρτήσεων αλλά και για εύρεση κατηγοριών.
- 4 Επιλύουν προβλήματα που ένα απλό Perceptron δεν μπορεί να λύσει.
- 5 Αποτελούν την βάση για τα βαθιά νευρωνικά δίκτυα.

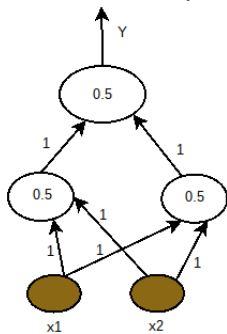
# Το πρόβλημα XOR

- 1 Τα δίκτυα PERCEPTRON και ADALINE δεν είναι σε θέση να μάθουν πολύπλοκες συναρτήσεις όπως το κλασικό πρόβλημα XOR.
- 2 Το πρόβλημα αυτό δίνεται στον επόμενο πίνακα.

A	B	$\Upsilon$
0	0	0
0	1	1
1	0	1
1	1	0

# Επίλυση του XOR με MLP

Μια ενδεικτική υλοποίηση με χρήση MLP



Για τα δίκτυα MLP έχει αναπτυχθεί και αποδειχθεί η θεωρία της παγκόσμιας προσέγγισης:

- 1 Ένα δίκτυο δύο στρωμάτων (είσοδος - επεξεργασία - έξοδος) μπορεί να προσεγγίσει μια οποιαδήποτε συνάρτηση.
- 2 Δεν χρειάζεται πάνω από ένα στρώμα επεξεργασίας
- 3 Οι νευρώνες του κρυφού στρώματος έχουν σαν συνάρτηση ενεργοποίησης την σιγμοειδή.
- 4 Ο νευρώνας εξόδου έχει σαν έξοδο την γραμμική συνάρτηση ενεργοποίησης.

## Το MLP σαν συνάρτηση

Ένα νευρωνικό δίκτυο MLP μπορεί να αναπαρασταθεί με πολλούς τρόπους αλλά ο πιο απλός είναι η ακόλουθη εξίσωση:

$$N(\vec{x}, \vec{w}) = \sum_{i=1}^H w_{(d+2)i-(d+1)} \sigma \left( \sum_{j=1}^d x_j w_{(d+2)i-(d+1)+j} + w_{(d+2)i} \right) \quad (1)$$

όπου

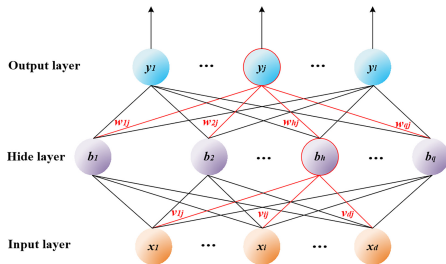
- 1  $H$  είναι ο συνολικός αριθμός μονάδων επεξεργασίας του νευρωνικού δικτύου
- 2  $d$  είναι η διάσταση του προβλήματος εισόδου.
- 3  $\vec{w}$  είναι τα βάρη του ΤΝΔ.
- 4 Η πόλωση κάθε μονάδας επεξεργασίας:  $w_{(d+2)i}$ .
- 5 Αν  $d=3$  και έχουμε 2 κόμβους επεξεργασίας τότε συνολικά θα υπάρχουν  $(d+2)H = 10$  συνολικά παράμετροι στο ΤΝΔ.

# Μάθηση κατηγοριών με πολλές εξόδους

- 1 Χρησιμοποιούνται τόσες εξοδοί όσες και το πλήθος των κατηγοριών του προβλήματος (πχ στο wine 3)
- 2 Το αποτέλεσμα του ΤΝΔ θεωρείται εκείνη η έξοδος με την μεγαλύτερη τιμή εξόδου
- 3 Απαιτείται μεγαλύτερος αριθμός βαρών και πιο αργοί χρόνοι εκπαίδευσης



## ΤΝΔ πολλών εξόδων



## Κατώφλια στην μάθηση κατηγοριών

- 1 Για τις περιπτώσεις που θέλουμε το παραπάνω δίκτυο να κάνει πρόβλεψη κατηγορίας μπορούν να χρησιμοποιηθούν κατώφλια στις τιμές.
- 2 Για παράδειγμα έστω ο ακόλουθος ενδεικτικός πίνακας:

$y(x)$	$N(x)$	CLASS
1	2.4	1
0	0.4	0
0	1.3	1
1	0.8	1

- 3 Στην στήλη  $y(x)$  είναι η πραγματική έξοδος και στην στήλη  $N(x)$  η έξοδος του ΤΝΔ.
- 4 Με την χρήση κατωφλιών (κοντινότερη κατηγορία) παράγεται η στήλη CLASS όπου βλέπουμε πως το ΤΝΔ “μαντεύει” σωστά τις 3 από τις 4 πραγματικές εξόδους.

# Μέσο τετραγωνικό σφάλμα σε μάθηση συναρτήσεων

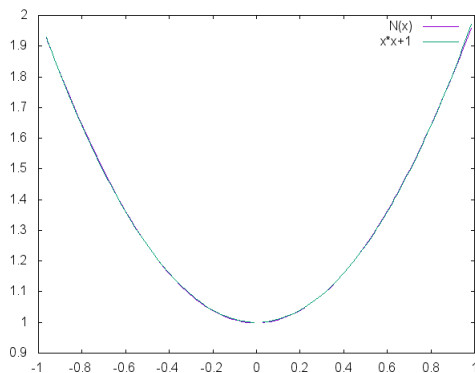
- 1 Έστω  $x = (x_1, x_2, \dots, x_M)$  τα πρότυπα εισόδου
- 2 Έστω  $y = (y_1, y_2, \dots, y_M)$  οι πραγματικές έξοδοι
- 3 Το σφάλμα ορίζεται

$$E = \frac{1}{M} \sum_{i=1}^M (N(x_i) - y_i)^2$$

- 4 Μπορεί να χρησιμοποιηθεί άμεσα σε μεθόδους βελτιστοποίησης

# Παράδειγμα μάθησης συνάρτησης

Στο σχήμα παρατίθεται η μάθηση της συνάρτησης  $x^2 + 1$  την οποία το δίκτυο Adaline δεν κατόρθωσε να μάθει.



- 1 Έστω  $x = (x_1, x_2, \dots, x_M)$  τα πρότυπα εισόδου
- 2 Έστω  $y = (y_1, y_2, \dots, y_M)$  οι πραγματικές έξοδοι
- 3 Το σφάλμα ορίζεται

$$E = \frac{1}{M} \sum_{i=1}^M (C(N(x_i)) \neq y_i)$$

- 4 Η συνάρτηση  $C(X)$  βρίσκει και επιστρέφει την κοντινότερη κατηγορία στην οποία είναι η τιμή  $X$ .

## Σφάλμα κατηγοριοποίησης σε imbalanced data

- 1 Είναι δεδομένα στα οποία τα πρότυπα που ανήκουν σε κάποιες κατηγορίες είναι πολύ λιγότερα από τον μέσο όρο.
- 2 Σε αυτήν την περίπτωση το ΤΝΔ τείνει να μάθει μόνο τα δεδομένα που ανήκουν στην κατηγορία με τα περισσότερα πρότυπα.
- 3 Μια καλύτερη μέθοδος υπολογισμού
  - 1 Έστω οι κατηγορίες  $C = (c_1, c_2, \dots, c_K)$
  - 2 Υπολογίζουμε το σφάλμα κατηγοριοποίησης μεμονωμένο για κάθε κατηγορία  $E(c_i)$
  - 3 Το σφάλμα ορίζεται ως

$$E = \frac{1}{K} \sum_{i=1}^K E(c_i)$$

- 1 Οι περισσότερες μέθοδοι εκπαίδευσης χρησιμοποιούν παράγωγο του σφάλματος
- 2 Εμπλέκεται και η παράγωγος της συνάρτησης ενεργοποίησης.
- 3 Συνήθως η συνάρτηση ενεργοποίησης είναι η σιγμοειδής
- 4  $\sigma(x) = \frac{1}{1+\exp(-x)}$ ,  $\sigma'(x) = \sigma(x) \times (1 - \sigma(x))$
- 5 Υπολογίζονται εύκολα, χωρίς πολλές πράξεις

# Ο αλγόριθμος Back Propagation (gradient descent)

Το μέσο τετραγωνικό σφάλμα ορίζεται ως:

$$E(N(\vec{x}, \vec{w})) = \sum (N(\vec{x}_i, \vec{w}) - y_i)^2 \quad (2)$$

Στην μέθοδο αυτή ισχύουν τα ακόλουθα:

- 1 Κινούμαστε αντίθετα από ότι λέει η παράγωγος της εξίσωσης 2. Αν η παράγωγος είναι θετική τότε μειώνεται το αντίστοιχο βάρος αλλιώς αυξάνεται.
- 2 Ο συντελεστής μάθησης είναι θετικός αριθμός και μικρότερος του 1.
- 3 Τα βάρη ενημερώνονται από τον κανόνα

$$w = w - n \frac{\partial E}{\partial w} \quad (3)$$



- 1 Ο αλγόριθμος Back Propagation ουσιαστικά είναι ο αλγόριθμος Gradient Descent
- 2 Έχει πολύ αργή σύγκλιση
- 3 Απαιτείται παραγωγή (σε δίκτυα πολλών επιπέδων είναι δύσκολο).
- 4 Εξαρτάται άμεσα από την επιλογή του  $n$
- 5 Η τιμή του  $n$  μπορεί να υπολογιστεί και με γραμμική αναζήτηση
- 6 Υπάρχει online τρόπος μάθησης (ένα - ένα τα πρότυπα) και offline.
- 7 Τα βάρη ενημερώνονται διαρκώς είτε μέχρι να υπάρξει σύγκλιση είτε μέχρι να φτάσουμε σε μέγιστο αριθμό επαναλήψεων.

# Γραμμική αναζήτηση

## 1 Η εξίσωση

$$x(\lambda) = x_1 + \lambda(x_2 - x_1)$$

δίνει την εξίσωση ευθείας που διέρχεται από τα σημεία  $x_1, x_2$

- 1 Αν μια ευθεία διέρχεται από το  $x_1$  και είναι παράλληλη σε ένα διάνυσμα  $s$  έχει την μορφή

$$x(\lambda) = x_1 + \lambda s$$

## 2 Η διαδικασία

$$\min_{\lambda} f(x + \lambda s)$$

ονομάζεται **γραμμική αναζήτηση**.

- 1 Δίνεται ένα σημείο  $x^{(k)}$
- 2 Υπολογίζεται μια φθίνουσα κατεύθυνση  $s^{(k)}$
- 3 Ελαχιστοποιείται ως προς  $\lambda$  ( $\lambda > 0$ ) η συνάρτηση

$$\phi(\lambda) = f(x^{(k)} + \lambda s^{(k)})$$

και εντοπίζεται η βέλτιστη τιμή  $\lambda^{(k)}$

- 4 Τίθεται  $x^{(k+1)} = x^{(k)} + \lambda^{(k)} s^{(k)}$

Στο σημείο 3 γίνεται η λεγόμενη **γραμμική αναζήτηση**.

## Τεχνικές γραμμικής αναζήτησης

- 1 Γραμμική αναζήτηση χρυσής τομής.
- 2 Γραμμική αναζήτηση Fibonacci.
- 3 Γραμμική αναζήτηση Armijo.

# Η γραμμική αναζήτηση χρυσής τομής

- Επιλέγονται σημεία  $x_1, x_2$  που να ισαπέχουν από τα άκρα του διαστήματος  $[a, b]$ .
- Τα βήματα έχουν ως ακολούθως
  - 1  $\phi = \frac{\sqrt{5}-1}{2}$
  - 2 Για  $i = 1, 2, 3, \dots$  επαναλαμβάνουμε
    - 1  $x_1 = a + (1 - \phi)(b - a)$
    - 2  $x_2 = a + \phi(b - a)$
    - 3 Αν  $f(x_1) < f(x_2)$  τότε  $b = x_2$
    - 4 Αλλιώς  $a = x_1$
    - 5 Τέλος - Αν
  - 3 Τέλος - Για

# Παράδειγμα για την $f(x)=x(x-3)*(x-3)+2$

a	$x_1$	$f(x_1)$	$f(x_2)$	$x_2$	b
<b>2.0000</b>	2.7639	2.1540	2.1803	3.2361	<b>4.0000</b>
<b>2.0000</b>	2.4721	2.6888	2.1540	2.7639	<b>3.2361</b>
• <b>2.4721</b>	2.7639	2.1540	2.0091	2.9443	<b>3.2361</b>
<b>2.7639</b>	2.9443	2.0091	2.0095	3.0557	<b>3.2361</b>
<b>2.8754</b>	2.9443	2.0091	2.0005	2.9868	<b>3.0557</b>
<b>2.9443</b>					<b>3.0557</b>

- Μετά από 6 επαναλήψεις έχει εντοπιστεί ένα διάστημα  $[2.9443, 3.0557]$

# Τερματισμός της μεθόδου Gradient Descent

- Μέγιστος αριθμός επαναλήψεων.
- $\|x_{k+1} - x_k\| \leq e$
- $|f(x_{k+1}) - f(x_k)| \leq e$
- $\frac{\|x_{k+1} - x_k\|}{\|x_k\|} \leq e$  (δεν δουλεύει αν  $x_k \simeq 0$ )

# Momentum Gradient Descent

- Για τον υπολογισμό του νέου σημείου λαμβάνεται υπόψιν και το ιστορικό των μεταβολών
- Εισάγεται μια ακόμα παράμετρος με το όνομα momentum (παράμετρος  $m$ )
- Η ενημέρωση γίνεται σύμφωνα με τον τύπο:

$$x_{k+1} = x_k - h \nabla f(x_k) - m(x_k - x_{k-1})$$

- Οι μεταβολές είναι πιο ομαλές, αλλά απαιτείται η χρήση δύο παραμέτρων.



- 1 Στην μέθοδο αυτή το βήμα είναι μεταβλητό στην ενημέρωση των βαρών.
- 2 Επίσης για την ενημέρωση των βαρών χρησιμοποιεί τα πρόσημα των παραγώγων και όχι τις ίδιες τις παραγώγους.
- 3 Η ενημέρωση γίνεται

$$w^{(k+1)} = w^{(k)} - n^{(k)} \times \text{sign}\left(\frac{\partial E}{\partial w^{(k)}}\right)$$

# Το βήμα στον RPROP

- 1 Το βήμα δίνεται από τον τύπο

$$n^{(k)} = \begin{cases} \min(n^{(k-1)} \times a, n_{\max}) & , \frac{\partial E^{(k)}}{\partial w^{(k)}} \times \frac{\partial E^{(k-1)}}{\partial w^{(k-1)}} > 0 \\ \max(n^{(k-1)} \times b, n_{\min}) & , \frac{\partial E^{(k)}}{\partial w^{(k)}} \times \frac{\partial E^{(k-1)}}{\partial w^{(k-1)}} < 0 \\ n^{(k-1)} & , \text{otherwise} \end{cases}$$

- 2 Όπου  $a > 1 > b$
- 3 Το βήμα περιορίζεται μεταξύ των τιμών  $n_{\min}$  και  $n_{\max}$  για να μην γίνει πολύ μεγάλο ή πολύ μικρό.
- 4 Σε περίπτωση που η παράγωγος γίνει 0, τότε βρισκόμαστε σε τοπικό ελάχιστο της συνάρτησης σφάλματος και επομένως το βήμα δεν αλλάζει.

## Η μέθοδος ADAM

- Χρησιμοποιείται από το 2015 ευρύτατα στα βαθιά νευρωνικά δίκτυα.
- Δεν απαιτεί την χρήση δευτέρων παραγώγων.
- Είναι χρήσιμη σε περιπτώσεις που η παράγωγος έχει θόρυβο στον υπολογισμό της ή και υπάρχουν προβλήματα αριθμητικής ακρίβειας.
- Χρησιμοποιεί διανύσματα momentum και όχι απλώς έναν αριθμό.
- Παίρνει μέρος στον υπολογισμό τόσο η παράγωγος όσο και το τετράγωνό της.

## Ο αλγόριθμος της μεθόδου ADAM (παράμετροι)

- Παράμετρος  $\alpha$ , μέγεθος βήματος, συνήθως 0.001
- Παράμετροι  $\beta_1$ ,  $\beta_2$ , είναι οι παράγοντες momentum με συνήθεις τιμές 0.9 και 0.999

## Ο αλγόριθμος ADAM (βήματα)

- 1 Θέσε  $m_0 = 0$ , πρώτο διάνυσμα momentum
- 2 Θέσε  $u_0 = 0$ , δεύτερο διάνυσμα momentum
- 3 Θέσε  $k = 0$ , αριθμός επαναλήψεων
- 4 Έστω ένα νέο δείγμα από την συνάρτηση  $x_0$
- 5 Μέχρι τερματισμό κάνε
  - 1  $g_k = \nabla f(x_k)$
  - 2  $m_{k+1} = \beta_1 m_k + (1 - \beta_1) g_k$
  - 3  $u_{k+1} = \beta_2 u_k + (1 - \beta_2) g_k^2$
  - 4  $\hat{m}_{k+1} = \frac{1}{1 - \beta_1^{k+1}} m_{k+1}$ , παίρνει δυνάμεις για τα  $\beta_1, \beta_2$  ώστε να μειώνεται η επίδρασή τους όσο περνάνε οι επαναλήψεις.
  - 5  $\hat{u}_{k+1} = \frac{1}{1 - \beta_2^{k+1}} u_{k+1}$
  - 6  $x_{k+1} = x_k - \alpha \frac{\hat{m}_{k+1}}{(\sqrt{\hat{u}_{k+1} + \epsilon})}$
  - 7  $k = k + 1$

- 6 Τέλος - Μέχρι

# Παραλλαγές μεθόδου ADAM και Gradient Descent

- AdaGrad
- AdaDelta
- AdaMax
- AMSGrad
- RMSProp
- NADam

- Αναπτύχθηκε το 1983 από τον Kirkpatrick κυρίως για συνδυαστικά προβλήματα.
- Μιμείται την φυσική διαδικασία της Ανόπτυσης.
- Στην ανόπτυση ζεσταίνεται ένα μέταλλο σε υψηλή θερμοκρασία και εν συνεχεία μειώνεται η θερμοκρασία μέχρι να φτάσει στο 0.
- Στην άνοδο τα μόρια του μετάλλου κινούνται γρήγορα (αναζήτηση λύσεων) αλλά στην συνέχεια όσο αυτό ψύχεται μειώνεται η ταχύτητά τους.
- Ο τρόπος που πέφτει η θερμοκρασία ονομάζεται cooling schedule
- Στα νευρωνικά δίκτυα απαιτείται συνδυασμός της μεθόδου αυτής και υπάρχουσων τεχνικών τοπικής αναζήτησης.

# Ενδεικτικός αλγόριθμος

1 Θέτουμε  $k = 0$ ,  $T_0 > 0$

- 1 Έστω  $x_0$  το αρχικό σημείο.
- 2 Έστω  $N_{\text{eps}} > 0$  ένας θετικός ακέραιος.
- 3 Έστω  $e > 0$ , ένας μικρός θετικός δεκαδικός αριθμός.
- 4 Για  $i = 1, \dots, N_{\text{eps}}$  επανέλαβε

- 1 Έστω  $y$  ένα νέο δείγμα.
- 2 Αν  $f(y) \leq f(x_k)$   $x_{k+1} = y$
- 3 Διαφορετικά  $x_{k+1} = y$  με πιθανότητα

$$\min \left\{ 1, \exp \left( -\frac{f(y) - f(x_k)}{T_k} \right) \right\}$$

5 Τέλος Για

- 6 Ενημέρωση θερμοκρασίας  $T_k$  σύμφωνα με τον μηχανισμό ψύξης.
- 7  $k = k + 1$
- 8 Αν  $T_k \leq e$  τερματισμός
- 9 Αλλιώς μετάβαση στο βήμα 5.



# Τεχνικές μείωσης Θερμοκρασίας

- Θεωρούμε πως ξεκινάμε πάντα από μια μεγάλη θερμοκρασία  $T_0$ 
  - $k$  είναι η επανάληψη του αλγορίθμου.
  - Εκθετική μείωση:

$$T_k = T_0 a^k, \quad 0.8 \leq a \leq 0.9$$

- Logarithmical multiplicative cooling:

$$T_k = \frac{T_0}{1 + a \log(1 + k)}$$

- Linear multiplicative cooling:

$$T_k = \frac{T_0}{1 + ak}$$

- Quadratic multiplicative cooling:

$$T_k = \frac{T_0}{1 + ak^2}$$

## Ορισμοί

- 1 Στις περισσότερες περιπτώσεις τα δεδομένα υποβάλλονται σε κανονικοποίηση πριν την παρουσίαση στο ΤΝΔ
- 2 Έχει σαν αποτέλεσμα και μικρότερες τιμές στις παραγώγους.
- 3 Μέθοδοι κανονικοποίησης:
  - 1 Ελαχίστου - μεγίστου
  - 2 z-score
  - 3 Δεκαδική κλιμάκωση (διαίρεση με δυνάμεις του 10)
- 4 Δεν είναι δίκαιος τρόπος, καθώς πρέπει να γνωρίζουμε εκ των προτέρων πληροφορία τόσο από το σύνολο εκπαίδευσης όσο και από το σύνολο ελέγχου.
- 5 Η κανονικοποίηση της εξόδου στο διάστημα  $[0,1]$  είναι πιο δίκαιη και βοηθάει περισσότερο.

- 1 Χρησιμοποιείται στα βαθιά νευρωνικά δίκτυα.
- 2 Η έξοδος κάθε επιπέδου κανονικοποιείται πριν την είσοδο στο επόμενο επίπεδο

## Αρχικοποίηση σε χαμηλές τιμές

- 1 Είναι ο πιο απλός τρόπος αρχικοποίησης.
- 2 Χρησιμοποιείται στην πράξη περισσότερο από όλους.
- 3 Τα βάρη αρχικοποιούνται ομοιόμορφα σε χαμηλό διάστημα τιμών πχ.  $[-0.1, 0.1]$

# Αρχικοποίηση Xavier

- 1 Τα βάρη αρχικοποιούνται ομοιόμορφα στο διάστημα  $\left[-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}\right]$
- 2  $d$  είναι η διάσταση των προτύπων εισόδου.
- 3 Υπάρχει και η κανονικοποιημένη εκδοχή:  $\left[-\frac{\sqrt{6}}{\sqrt{d+m}}, \frac{\sqrt{6}}{\sqrt{d+m}}\right]$ , όπου  $m$  είναι ο αριθμός των κόμβων στο τρέχον επίπεδο.

## Γενίκευση

- 1 Σκοπός της μάθησης είναι η εύρεση κρυμμένων συσχετίσεων
- 2 Γενίκευση είναι η μάθηση δεδομένων σε άγνωστα πρότυπα, τα οποία δεν έχουν χρησιμοποιηθεί κατά την εκπαίδευση
- 3 Χρήση πολλών νευρώνων: το δίκτυο μαθαίνει πολύπλοκες συναρτήσεις αλλά δεν μπορεί να μάθει σε άγνωστα δεδομένα το ίδιο καλά
- 4 Χρήση λίγων νευρώνων: το δίκτυο μαθαίνει απλές μόνο συναρτήσεις.

- 1 Διαίρεση του συνόλου εκπαίδευσης σε εκπαίδευσης και επικύρωσης.
- 2 Εκπαιδεύεται το δίκτυο στο μικρότερο πλέον σύνολο εκπαίδευσης
- 3 Η εκπαίδευση διακόπτεται όταν στο σύνολο επικύρωσης ξεκινά να ανεβαίνει στο σφάλμα.
- 4 Χρειάζεται προσεκτική επιλογή του ποσοστού για το σύνολο επικύρωσης
- 5 Απαραίτητα προϋπόθεση να έχουμε αρκετά δεδομένα στα χέρια μας.

# N-Fold

- 1 Θέτουμε την παράμετρο  $N$
- 2 Διαιρούμε το σύνολο των δεδομένων σε  $N$  σύνολο  $S_1, S_s, \dots, S_N$
- 3 Για  $i=1..N$ 
  - 1 Δημιουργία συνόλου εκπαίδευσης  $T$  με όλα τα  $S_j$  για  $j$  διαφορετικό του  $i$
  - 2 Εκπαίδευση στο σύνολο  $T$  και αποτίμηση του σφάλματος στο  $S_i$  με σφάλμα  $E_i$
- 4 Ο μέσος όρος των σφαλμάτων  $E_i$  είναι και το τελικό σφάλμα.

Στις πιο πολλές περιπτώσεις  $N=10$  αλλά και η τιμή  $N=2$  χρησιμοποιείται αρκετά συχνά.





Η μέθοδος Folding χρησιμοποιείται πάρα πολύ αλλά σε κάποιες περιπτώσεις τα δεδομένα δεν είναι αρκετά για να γίνει η παραπάνω διαδικασία. Σε αυτήν την περίπτωση χρησιμοποιείται η τεχνική leave one out, στην οποία το σύνολο ελέγχου αποτελείται μόνον από ένα πρότυπο.

# Σύνοψη

- Παρουσίαση δικτύων MLP
- Μέθοδοι εκπαίδευσης
- Ειδικά θέματα MLP

# Βιβλιογραφία I

-  Rosenblatt, Frank (1958), The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Cornell Aeronautical Laboratory, Psychological Review, v65, No. 6, pp. 386–408.
-  Freund, Y. and Schapire, R. E. 1998. Large margin classification using the perceptron algorithm. In Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT' 98). ACM Press.