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

Θεμελιώδη Θέματα Επιστήμης Υπολογιστών

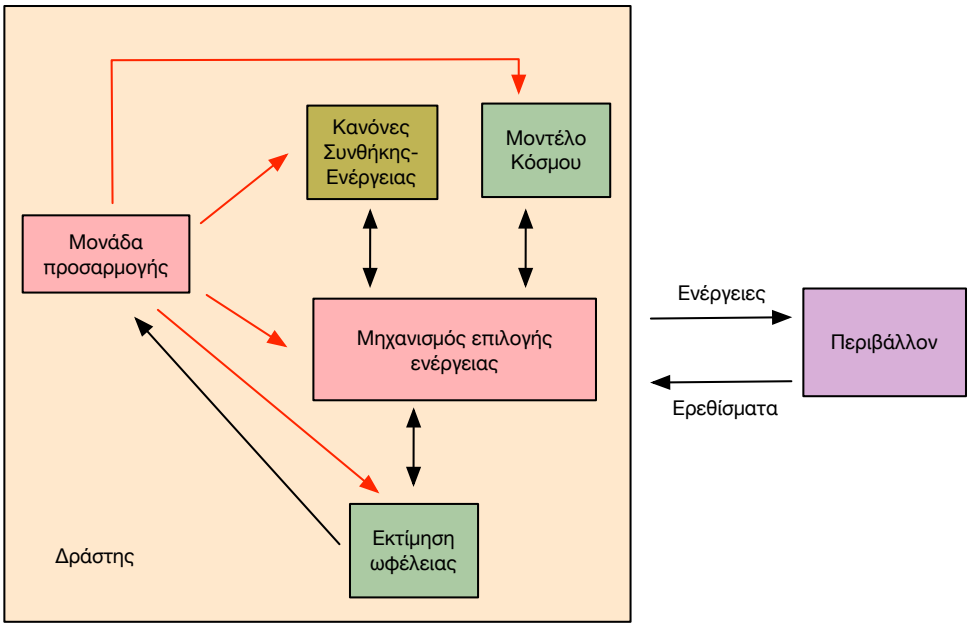
Τεχνητή Νοημοσύνη

Μηχανική μάθηση

Γιώργος Στάμου

ΤΝ και ανάπτυξη ευφυών δραστών

Ορθολογικός δράστης με μάθηση



Ευριστικά κριτήρια και αποφάσεις

Πρόβλημα

Φτάνεις σε ένα εστιατόριο για φαγητό.
Αποφάσισε αν θα περιμένεις για τραπέζι ή θα φύγεις.

Παράγοντες που επηρεάζουν την απόφαση

Εναλλακτικές: υπάρχει κάποιο εστιατόριο κοντά;
Ποιότητα αναμονής: υπάρχει άνετος χώρος αναμονής (πχ. μπαρ);
Ημέρα: Είναι σήμερα ΣΚ ή κάποια γιορτή;
Πείνα: Είσαι πολύ πεινασμένος;
Πληρότητα: Πόσοι τρώνε στο εστιατόριο (κανείς, αρκετοί, είναι γεμάτο);
Τιμή: πόσο ακριβό είναι το εστιατόριο (φτηνό, μέσο, ακριβό);
Καιρικές συνθήκες: Βρέχει έξω;
Κράτηση: Έχεις κάνει κράτηση;
Είδος εστιατορίου: Ινδικό, πιτσαρία, γκουρμέ;
Εκτίμηση αναμονής: εκτίμηση σε λεπτά (0-10, 10-30, 30-60, >60);

Μηχανική μάθηση

Ορισμός

Θα λέμε ότι ένα πρόγραμμα υπολογιστή μαθαίνει από μία εμπειρία *E* ως προς μια εργασία *T* και ένα μέτρο επίδοσης *P*, αν η επίδοσή του στην εργασία *T*, όπως αποτιμάται από το μέτρο *P*, βελτιώνεται με την εμπειρία *E*.

Παράδειγμα

Στο πρόβλημα του εστιατορίου:

- Εργασία *T* είναι η εξαγωγή της απόφασης για το αν θα καθίσω στο εστιατόριο, δεδομένης της τιμής των παραγόντων.
- Επίδοση *P* είναι η μέτρηση του πόσες φορές είναι σωστή η απόφαση αυτή.
- Εμπειρία *E* είναι ένα σύνολο καταστάσεων για τις οποίες γνωρίζω τις τιμές των παραγόντων και τη σωστή απόφαση.

Ταξινόμηση

Τυπική διατύπωση προβλήματος ταξινόμησης

Δίνεται ένα σύνολο ζευγών (δεδομένα)

$$Z = \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{R}^p \times \{1,...,c\}$$

$x_i \in \mathbb{R}_p$ ένα διάνυσμα τιμών χαρακτηριστικών για κάποιο στοιχείο *x* (υποθέτουμε γενικά ότι τα χαρακτηριστικά παίρνουν πραγματικές τιμές).

$c_i \in \{1,2,...,c\}$ η κατηγορία (κλάση) στην οποία ανήκει το στοιχείο *x*.

Καλούμαστε να σχεδιάσουμε μία συνάρτηση

$$f : \mathbb{R}^p \rightarrow \{1,...,c\}$$

που ταξινομεί σωστά τα δεδομένα στις κλάσεις.

Ταξινόμηση

Ορολογία

- Η συνάρτηση *f* ονομάζεται ταξινομητής και δίνει στην έξοδό της την κλάση στην οποία ανήκει ένα στοιχείο *x*, όταν στην είσοδό της δέχεται το σύνολο τιμών των χαρακτηριστικών του *x*.
  - Οι ταξινομητές προβλέπουν την κλάση στην οποία ανήκει οποιοδήποτε στοιχείο *x*, όχι μόνο τα δεδομένα.
- Η διαδικασία ανάπτυξης της συνάρτησης *f* από τα δεδομένα ονομάζεται σχεδιασμός του ταξινομητή.
- Η διαδικασία της εκτέλεσης του *f* για κάποιο στοιχείο *x* ονομάζεται ταξινόμηση του *x*.
- Θα λέμε ότι το στοιχείο *x* ταξινομήθηκε σωστά αν και μόνο αν

$$f(x) = y$$

όπου *y* η κλάση στην οποία ανήκει το στοιχείο *x*.

Ταξινομητής κοντινότερου γείτονα

Πλάνο σχεδιασμού

- Αποθηκεύουμε όλα τα δεδομένα στη μνήμη.
- Συγκρίνουμε την είσοδο *x* με όλα τα στοιχεία των δεδομένων και βρίσκουμε το κοντινότερο, με βάση κάποια απόσταση.
- Δίνουμε στην έξοδο την κλάση στην οποία ανήκει το κοντινότερο στοιχείο των δεδομένων.

Τυπικά

- Η έξοδος του ταξινομητή δίνεται από τη σχέση

$$f(x) = y_\kappa \quad \text{με} \quad \|x - x_k\| = \min_{j=1,\ldots,n} \|x - x_j\|$$

$\|\cdot\|$ μία απόσταση στο χώρο $\mathbb{R}^q$

Ταξινομητής κοντινότερου γείτονα

Παρατηρήσεις

- Στην ουσία, δεν σχεδιάζουμε κάποιο ταξινομητή που κατά κάποιο τρόπο κωδικοποιεί τα δεδομένα, αλλά απλά αποθηκεύουμε τα δεδομένα και συγκρίνουμε την είσοδο με αυτά, για να ταξινομήσουμε την είσοδο σε μία κλάση.
- Η στρατηγική αυτή ονομάζεται οκνηρή μάθηση και έχει κάποια πλεονεκτήματα, όπως ότι μπορούν να ενσωματωθούν εύκολα νέα δεδομένα ή μπορούν να προσομοιωθούν πολύ εύκολα μη-γραμμικές συναρτήσεις.
- Από την άλλη πλευρά, οι ταξινομητές κοντινότερου γείτονα είναι πολύ ευαίσθητοι στα σφάλματα και μπορεί να γίνουν υπολογιστικά ακριβοί στην εκτέλεσή τους, αν έχουμε πολλά δεδομένα.

Ταξινομητής *k* κοντινότερων γειτόνων

Πλάνο σχεδιασμού

- Αποθηκεύουμε όλα τα δεδομένα στη μνήμη.
- Συγκρίνουμε την είσοδο με τα δεδομένα και βρίσκουμε τα *k* κοντινότερα (*k*<*n*), με βάση κάποια απόσταση.
- Δίνουμε στην έξοδο την κλάση στην οποία ανήκει η πλειοψηφία των *k* κοντινότερων δεδομένων.

Παρατηρήσεις

- Η ιδέα του να διαλέγουμε τα *k* κοντινότερα δεδομένα, αυξάνει ακόμα περισσότερο το υπολογιστικό κόστος.
- Από την άλλη πλευρά, αυξάνει την ανοχή στο θόρυβο.
- Η επιλογή του *k* επηρεάζει σημαντικά την επίδοση του αλγορίθμου.

LVQ (linear vector quantization)

Πλάνο σχεδιασμού

- Αντί να αποθηκεύουμε όλα τα δεδομένα στη μνήμη, αποθηκεύουμε ένα σύνολο από αντιπροσωπευτικά παραδείγματα (πρότυπα).
- Στην περίπτωση του αλγόριθμου LVQ αποθηκεύουμε ένα πρότυπο για κάθε κλάση εξόδου. Υπολογίζουμε το πρότυπο αυτό, με μία ευριστική διαδικασία (αλγόριθμος LVQ).
- Στη συνέχεια, ακολουθούμε τη λογική του κοντινότερου γείτονα (αλλά αυτή τη φορά μόνο για τα πρότυπα), δηλαδή:
  - Συγκρίνουμε την είσοδο με τα πρότυπα και βρίσκουμε το κοντινότερο, με βάση κάποια απόσταση.
  - Δίνουμε στην έξοδο την κλάση στην οποία ανήκει το κοντινότερο πρότυπο.

LVQ (linear vector quantization)

Αλγόριθμος υπολογισμού προτύπων

Είσοδος $Z = \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{R}^p \times \{1,...,c\}$

Αρχικοποίηση $V = \{v_1, v_2, \ldots, v_c\} \subset \mathbb{R}^p, t = 1, \alpha(t)$ (βήμα μάθησης)

1. For $k = 1,...,n$

   Βήμα 1. Βρες το πρότυπο νικητή $v_i \in V$

   $$\|x_k - v_i\| \leq \|x_k - v\|, \forall v \in V$$

   Βήμα 2. Μετακίνησε το πρότυπο νικητή

   $$v_i = v_i + \alpha(t)(x_k - v_i), \text{ if } y_k = i$$
   $$v_i = v_i - \alpha(t)(x_k - v_i), \text{ otherwise}$$

2. $t = t + 1$
3. Επανέλαβε από το Βήμα 1, αν δεν ισχύει το κριτήριο τερματισμού
4. Επέστρεψε $V = \{v_1, v_2, \ldots, v_n\}$

Ταξινομητής naive Bayes

Υποθέσεις

- Τα χαρακτηριστικά είναι boolean, δηλαδή παίρνουν δύο τιμές.
- Συνεπώς, μπορούμε να τα θεωρήσουμε σαν ενδεχόμενα που είναι αληθή ή ψευδή, ώστε να ακολουθήσουμε πιθανοτική προσέγγιση.

Θεώρημα Bayes

Για δύο ανεξάρτητα γεγονότα *A* και *B*, ισχύει

$$p(A \mid B) \cdot p(B) = p(B \mid A) \cdot p(A)$$

Επομένως, αν σπάσουμε το *A* σε ένα σύνολο από ανεξάρτητα γεγονότα $A_1, A_2, \ldots, A_c$ ισχύει

$$p(A_i \mid B) = \frac{p(A_i) \cdot p(B \mid A_i)}{\sum_{j=1}^{c} P(A_j) \cdot P(B \mid A_j)}$$

Ταξινομητής naive Bayes

Εφαρμογή του κανόνα Bayes στην ταξινόμηση

- Τα σχετικά γεγονότα είναι:
  - Το στοιχείο ανήκει στην κλάση *i* (ή απλά *i*)
  - Το στοιχείο έχει το χαρακτηριστικό *x* (ή απλά *x*)
- Επομένως έχουμε:

$$p(i \mid x) = \frac{p(i) \cdot p(x \mid i)}{\sum_{j=1}^{c} p(j) \cdot p(x \mid j)}$$

- Αν τα *p* χαρακτηριστικά είναι ανεξάρτητα: $p(x \mid i) = \prod_{k=1}^{p} p(x^{(k)} \mid i)$
- Επομένως τελικά:

$$p(i \mid x) = \frac{p(i) \cdot \prod_{k=1}^{p} p(x^{(k)} \mid i)}{\sum_{j=1}^{c} p(j) \cdot \prod_{k=1}^{p} p(x^{(k)} \mid j)}$$

Ταξινομητής naive Bayes

Πλάνο σχεδιασμού

- Ξεκινάμε από το σύνολο δεδομένων

$$Z = \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{R}^p \times \{1,\ldots,c\}$$

- Υπολογίζουμε τις πιθανότητες

$p(i)$ σχετική συχνότητα της κλάσης *i*

$p(x^{(k)} \mid i)$ σχετική συχνότητα του χαρακτηριστικού $x^{(k)}$ στην κλάση $y = i, i = 1,\ldots,c, k = 1,\ldots,p$

- Όταν δίνεται ένα στοιχείο *x* με συγκεκριμένα χαρακτηριστικά, από τη σχέση

$$p(i \mid x) = \frac{p(i) \cdot \prod_{k=1}^{p} p(x^{(k)} \mid i)}{\sum_{j=1}^{c} p(j) \cdot \prod_{k=1}^{p} p(x^{(k)} \mid j)}$$

υπολογίζουμε την πιθανότητα να ανήκει σε κάθε κλάση *i*

Ταξινομητής perceptron

Υπόθεση

- Τα στοιχεία ταξινομούνται σε δύο κλάσεις (δηλαδή c=2).

Το πρόβλημα ταξινόμησης διατυπώνεται ως εξής:

Δίνονται τα δεδομένα

$$Z = \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathbb{R}^p \times \{-1,1\}$$

Ζητείται ο ταξινομητής

$$f : \mathbb{R}^p \to \{-1,1\}$$

ο οποίος μηδενίζει το σφάλμα

$$J = \sum_{i=1}^{n} |\, y_i - f(x_i) \,|$$

Ταξινομητής perceptron

Πλάνο σχεδιασμού

- Σχεδιάζουμε το γραμμικό ταξινομητή:

$$f(x) = \text{sign}(w^\top x - w_0)$$

- Προσπαθούμε να υπολογίσουμε τις παραμέτρους w, w_0 ώστε να ταξινομούνται σωστά τα δεδομένα, δηλαδή

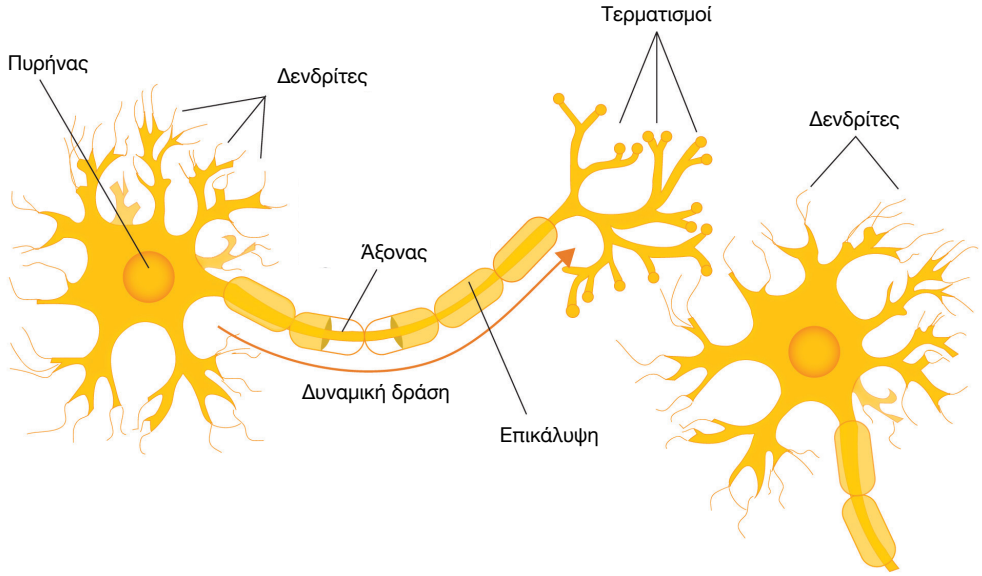
$$\text{sign}(w^\top x_i - w_0) = y_i$$

Παρατηρήσεις

- Η διαδικασία υπολογισμού των παραμέτρων (τις ονομάζουμε βάρη) λέγεται αλγόριθμος εκμάθησης του perceptron.
- Δεν υπάρχει πάντα λύση στις γραμμικές εξισώσεις.
- Όταν υπάρχει λύση, λέμε ότι τα δεδομένα είναι γραμμικά διαχωρίσιμα

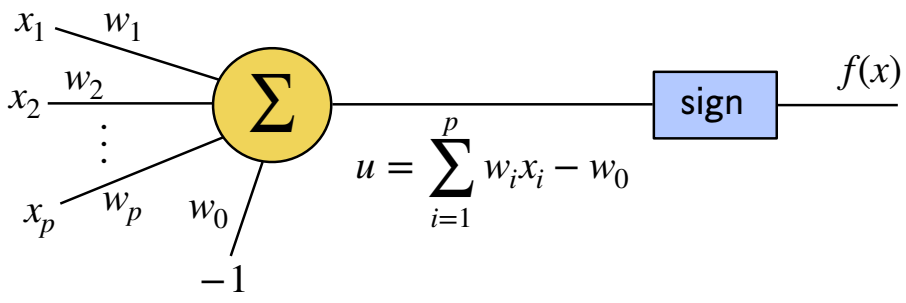
Ταξινομητής perceptron

Ιδέα βασισμένη στους βιολογικούς νευρώνες

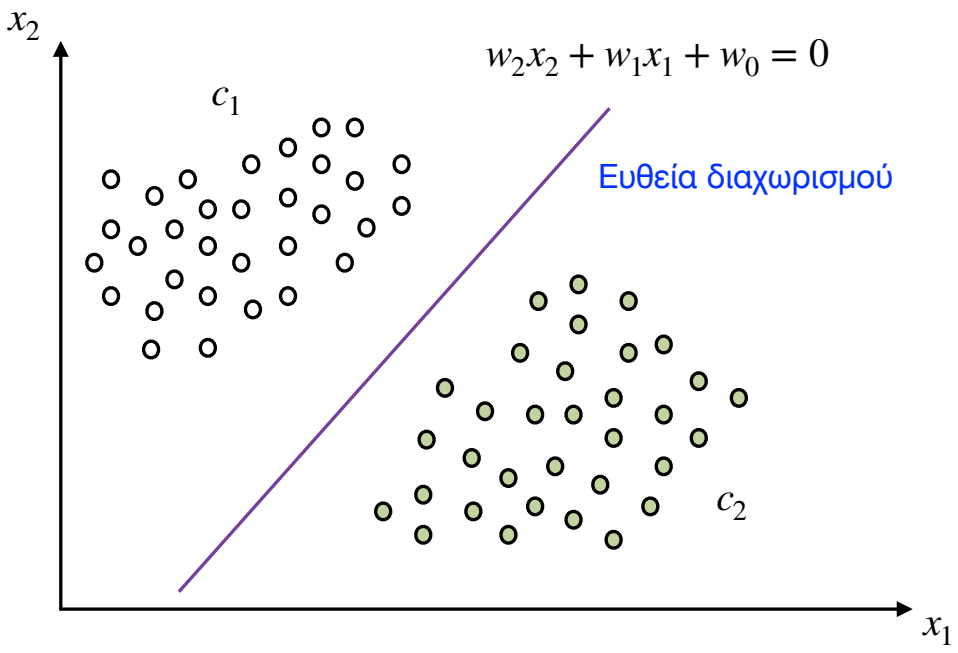


Ταξινομητής perceptron

Τεχνητός νευρώνας



Ταξινομητής perceptron



Ταξινομητής perceptron

Αλγόριθμος εκμάθησης

Στρατηγική ανάλυσης των δεδομένων

- 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 διάνυσμα βαρών, αν υπάρχει σφάλμα ταξινόμησης

Επανάληψη k	1	2	…	n	n+1	n+2	…	2n	2n+1	…
Δεδομένο x	1	2	…	n	1	2	…	n	1	…
	Εποχή 1				Εποχή 2				…	

Ταξινομητής perceptron

Αλγόριθμος εκμάθησης

Κανόνας σταθερής αύξησης βαρών

Κατά την επανάληψη *k* έχουμε:

Πρότυπο στην είσοδο: $(x(k), y(k))$

Έξοδος perceptron: $f(x(k)) = \text{sign}(w(k)^{\top} x(k) - w_0(k))$

Τότε, τα νέα βάρη δίνονται από τη σχέση:

$$w(k+1) = w(k) + \beta(y(k) - f(x(k))x(k)$$

βήμα εκπαίδευσης

Ταξινομητής 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 ανανέωσε τα βάρη:

$$w(k+1) = w(k) + \beta(y(k) - f(x(k))x(k)$$

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

Ταξινομητής 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.

$$
\begin{aligned}
u(k) &= w(k)^{\top} x(k) \\
&= w(k-1)x(k) + \beta(y(k-1) - f(x(k-1))x(k-1)x(k) \\
&= u(k-1) + \beta(y(k) - f(x(k))\|x\|^2 \qquad \text{if } x(k) = x(k-1)
\end{aligned}
$$

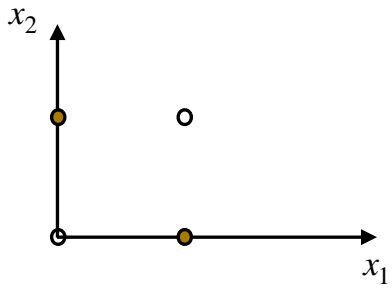
$$
\begin{cases}
-2, & \text{if } y(k) = -1, f(x(k)) = 1 \\
2, & \text{if } y(k) = 1, f(x(k)) = -1
\end{cases}
$$

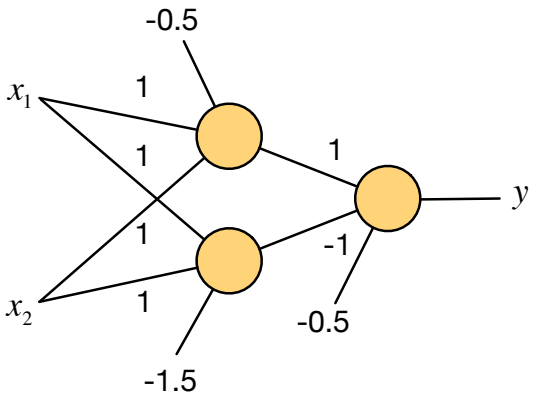
Σύγκλιση του κανόνα perceptron

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

Ταξινομητής perceptron

Πρόβλημα XOR





Σύγχρονες τεχνολογίες - Νευρωνικά δίκτυα