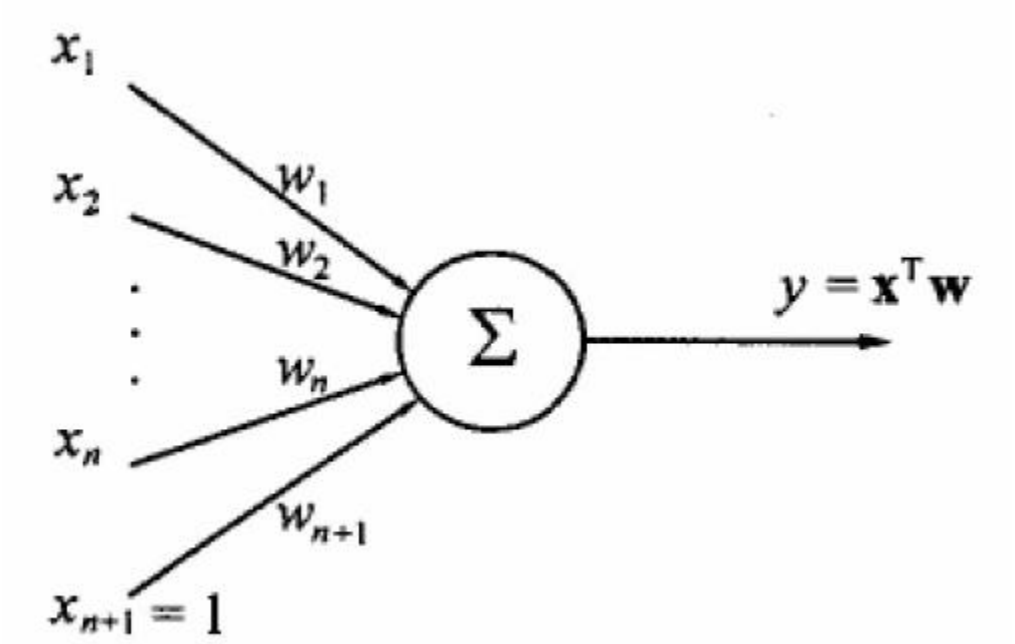
Laborator 8:

Algoritmul de învăţare Widrow Hoff bazat pe regula $\mu - LMS$

Această regulă a fost iniţial utilizată pentru a antrena unităţi liniare de forma:



Algoritmul de învăţare bazat pe regula $\mu - LMS$ constă în determinarea vectorului *w**, folosind metoda gradientului descendent, care minimizează suma pătratelor erorilor (SSE – sum squared error).

Fie mulţimea de antrenare $\left\{(x^1, d^1), (x^2, d^2), \ldots, (x^m, d^m)\right\}$.

Funcţia criteriu a sumei pătratelor erorilor (SSE) este:

$J(w) = \frac{1}{2}\sum_i [d^i - (x^i)^T w - b]^2 = \frac{1}{2}\sum_i (d^i - y^i)^2$, unde $y^i = (x^i)^T w + b$

Regula gradientului descendent de tip batch care minimizează funcţia criteriu $J(w)$ este următoarea:

$$w^{k+1} = w^k + \mu \sum_{i=1}^{m} (d^i - y^i) x^i$$

$$b^{k+1} = b^k + \mu \sum_{i=1}^{m} (d^i - y^i)$$

Cum în problemele practice avem $m > n+1$ este imposibilă satisfacerea tuturor cerinţelor $(x^k)^T w = d^k \quad k = \overline{1,m}$, deci regula de învăţare batch LMS de mai sus nu duce la găsirea unei soluţii astfel încât eroarea să fie 0, însă pentru μ suficient de

mic (de exemplu, $\mu = 0.005$), w^k va converge către minimul global *w** al funcţiei criteriu *J*(*w*).

Varianta incrementală a regulii de învăţare de mai sus este dată prin:

$$w^{k+1} = w^k + \mu(d^i - y^i)x^i$$

$$b^{k+1} = b^k + \mu(d^i - y^i)$$

S-a demonstrat că pentru $0 < \mu < \frac{2}{\max\limits_{i=\overline{1,m}} \left\| x^i \right\|}$ acest algoritm converge la soluţia, *w**, care minimizează funcţia criteriu într-un număr finit de paşi.

Astfel, folosind aceşti algoritmi de învăţare este posibil să obţinem un vector *w** care să nu poată clasifica corect o mulţime de antrenare. Acest fapt nu ar trebui să ne surprindă, întrucât se doreşte minimizarea funcţiei criteriu şi nu a ratei erorii de clasificare.

Minimul global al funcţiei criteriu este dat de formula: $w^* = (XX^T)^{-1}Xd$, unde $X = \left[x^1 \; x^2 \ldots x^m \right]$ şi $d = \left[d^1 \; d^2 \ldots d^m \right]^T$.

De asemenea, acest algoritm poate fi folosit pentru a determina vectorul ponderilor unei reţele de tip perceptron care să conducă la clasificarea corectă a mulţimii de antrenare.

Aplicaţie 1:

Fie mulţimea de antrenare următoare:
S = {([-2,2]^T,-1), ([-2,3]^T,-1), ([-1,1]^T,-1), ([-1,4]^T,-1), ([0,0]^T,-1), ([0,1]^T,-1), ([0,2]^T,-1), ([0,3]^T,-1), ([1,0]^T,+1), ([1,1]^T,-1), ([2,1]^T,+1), ([2,2]^T,-1), ([3,-1]^T,+1), ([3,0]^T,+1), ([3,1]^T,+1), ([3,2]^T,+1), ([4,-2]^T,+1), ([4,1]^T,+1), ([5,-1]^T,+1), ([5,0]^T,+1)}.

1) Implementaţi în MATLAB algoritmul de învăţare Widrow-Hoff folosind regula de învăţare $\mu - LMS$ varianta batch (antrenaţi reţeaua folosind 100 de paşi) şi varianta incrementală alegând exemplele din mulţimea de antrenare aleator (antrenând reţeaua în acest caz în 2000 de epoci) şi plotaţi dreapta de separare obţinută pe baza lui *w* şi a lui *b* pentru mulţimea de antrenare S de mai sus, respectiv punctele din mulţimea de antrenare. Observaţie: Considerăm rata de învăţare μ = 0.005, vectorul ponderilor iniţial $w^1 = [0\ 0]^T$ şi utilizarea unei reţele fără bias.

2) Studiaţi convergenţa celor doi algoritmi de la punctul 1), plotând pe axa Ox paşii de antrenare versus valorile $\left\| w^k - w^* \right\|^2$ corespunzătoare pe axa Oy (adică pentru vectorii de ponderi obţinuţi la iteraţia *k*) pentru fiecare algoritm în parte, unde *w** reprezintă soluţia care minimizează funcţia criteriu SSE. Interpretaţi rezultatele. Indicaţie: se vor folosi axe logaritmice pe axa Ox (vezi semilogx).

3) Punctele a) şi b) considerând în procesul de antrenare exemplele din mulţimea de antrenare într-o ordine deterministă pentru algoritmului *LMS* varianta incrementală.

4) Reprezentaţi grafic suprafaţa de eroare asociată funcţiei criteriu *J*(*w*) în funcţie de ponderile w1 şi w2 şi studiaţi convergenţa algoritmilor către vectorul de ponderi care minimizează această funcţie criteriu, plotând pe această suprafaţă valoarea erorii date de vectorii de ponderi obţinuţi după fiecare epocă/iteraţie a algoritmilor. Indicaţie: Se construieşte o reţea de puncte în plan în pătratul $[-0.3, 0.3] \times [-0.3, 0.3]$, corespunzătoare vectorilor de ponderi *w* şi se calculează în fiecare punct *w* valoarea funcţiei criteriu (funcţia de eroare), *J*(*w*).

Aplicaţie 2:

Să se construiască o mulţime de antrenare având ca vectori de intrare puncte bidiminsionale şi etichete reale, astfel încât problema găsirii ponderilor să fie nedeterminată, adică să admită mai multe soluţii (vectori de ponderi care minimizează funcţia de eroare SSE). Realizaţi suprafaţa de eroare asociată funcţiei de eroare în raport cu ponderile w1 şi w2 şi reprezentaţi pe aceasta vectorul de ponderi care minimizează funcţia SSE obţinut folosind newlind şi de asemenea şirul vectorilor de ponderi obţinuţi în antrenarea perceptronului folosind funcţia train. Interpretaţi rezultatele.

Aplicaţie 3:

Aplicaţia 2 pentru o mulţime de date de antrenare neliniare.
De exemplu:

```
P = [ 1.0   2.0   3.0]
T = [ 4.0   5.0   6.0];
```

Studiaţi evoluţia funcţiei de performaţă (a funcţiei de eroare) după fiecare epocă şi eroarea minimă obţinută după antrenare.

Aplicaţie 4:

Definiţi o mulţime de antrenare ca în aplicaţia 2 şi construiţi un perceptron cu funcţie de transfer liniară pe care îl antrenaţi folosind regula de învăţare a lui Widrow Hoff pentru o rată de învăţare de două ori mai mare decât cea furnizată de funcţia *maxlinlr*. Realizaţi suprafaţa de eroare asociată funcţiei de eroare în raport cu ponderile w1 şi w2 şi reprezentaţi pe aceasta vectorul de ponderi care minimizează funcţia SSE obţinut folosind metoda celor mai mici pătrate (w*) şi vectorii de ponderi rezultaţi după fiecare aplicare a regulii de învăţare. Vizualizaţi rezultatele şi comentaţi-le.

Bibliografie:

D. Enăchescu (2003), *Elements of Statistical Learning. Applications in Data Mining*, Padova University Press