Laborator 7

În laboratorul precedent, funcția ***algoritmRosenblattOnline.m*** implementa algoritmul de învățare al lui Rosenblatt. Toolbox-ul ***nnet*** din Matlab conține funcții care realizează acest lucru pentru obiecte de tip rețea (network).

Funcția ***train*** implementează algoritmul de învățare al lui Rosenblatt, atât varianta online (setare implicită) cât și varianta offline. În varianta online sau incrementală vectorul de ponderi **w** și biasul b se modifică după fiecare exemplu misclasificat. În Matlab, pentru configurarea opțiunii de antrenare online se setează pentru o rețea câmpul „trainFcn" cu valoarea „trainc" (valoare implicită). În varianta offline sau batch vectorul de ponderi **w** și biasul b se modifică pe baza tuturor exemplelor misclasificate. În Matlab, pentru configurarea opțiunii de antrenare offline se setează pentru o rețea câmpul „trainFcn" cu valoarea „trainb".

Pentru punctele *a* și *b* din laboratorul trecut codul Matlab care realizează antrenarea online a perceptronului (rețea cu un neuron) este următorul:

```
%datele: exemplele + etichetele
X = [0 0 0 0.5 0.5 0.5 1 1;0 0.5 1 0 0.5 1 0 0.5];
T = [1 1 1 1 -1 -1 -1 -1];
%reprezentare grafica a datelor
figure(1), hold on;
eticheta1 = find(T==1);
etichetaMinus1 = find(T==-1);
plot(X(1,eticheta1),X(2,eticheta1),'or');
plot(X(1,etichetaMinus1),X(2,etichetaMinus1),'xg');
axis([-2 2 -2 2]);
%pune pauza 2 secunde
pause(2);
%creaza perceptron
net = newp([-1 1;-1 +1],1,'hardlims');
view(net);
pause(5);
%afiseaza proprietatile perceptronului legate de vectorul de ponderi + bias
disp('Proprietati legate de vectorul de ponderi:');
disp(net.inputWeights{1});
pause(3);
disp('Proprietati legate de bias:');
disp(net.biases{1});
pause(3);
%initializeaza parametri retelei, implicit ponderile + bias sunt nule
```

```
net = init(net);
%seteaza numarul de epoci pentru antrenare
net.trainParam.epochs = 100;
%antreneaza reteaua
[net,antrenare] = train(net,X,T);
figure(1);
%ploteaza dreapta de separare
plotpc(net.IW{1},net.b{1})
title('Reprezentarea grafica a exemplelor de antrenare si a dreptei de separare');
%simuleaza reteaua pentru datele de intrare
eticheteprezise = sim(net,X);
isequal(eticheteprezise,T)
```

Pornind de la codul Matlab anterior, realizați următoarele:

1. inițializați cu valori aleatoare vectorul de ponderi (net.IW{1}) și bias-ul (net.b{1}) modificând valoarea câmpului ”initFcn” (pentru net.inputWeights{1} și net.biases{1} din „initZero” în „rands”). Antrenați rețeaua în acest caz. Cum se modifică numărul de epoci necesare convergenței antrenării?
2. modificați codul Matlab astfel încât să plotați după fiecare iterație hiperplanul de separare.
3. adăugați punctul (-50,40) cu eticheta 1 mulțimii de antrenare. Cum se modifică timpul de convergență (=numărul de epoci) al algoritmului?
4. modificați funcțiile de învățare ale ponderilor și ale bias-ului din „learnp” în „learnpn” (folosiți help-ul și citiți despre deosebirea dintre ele). Cum se modifică timpul de convergență (=numărul de epoci) al algoritmului?
5. rezolvați punctele *c* și *d* din laboratorul 6 folosind o rețea antrenată online.
6. definiți o rețea cu un perceptron care să învețe funcțiile logice AND și OR. Reprezentați grafic punctele mulțimii de antrenare și dreapta de separare.
7. rulați punctele anterioare cu antrenarea de tip offline.