Laborator 6

Două mulțimi de puncte dintr-un spațiu *d*-dimensional se numesc liniar separabile dacă ele pot fi separate de un hiperplan. În acest caz identificăm cele două mulțimi de puncte cu două clase (vom folosi în cele ce urmează clasele -1 și 1). Problemele de clasificare binară în care cele două clase sunt liniar separabile pot fi rezolvate cu succes de un perceptron.

Un perceptron cu vectorul de ponderi $\mathbf{w} = (w_1,w_2,...,w_d)$ și bias-ul b implementează pentru vectorul de intrare $\mathbf{x} = (x_1,x_2,...,x_d)$ ecuația hiperplanului dat de: $\mathbf{w}^T\mathbf{x} + b = 0$. Pentru funcția de transfer *hardlims*, vectorul **x** este clasificat în clasa y = hardlims($\mathbf{w}^T\mathbf{x}$ + b).

Programul demonstrativ **nnd4db** ilustrează o problemă de clasificare binară în $\mathbf{R}^2$ folosind un perceptron. Rulați acest program și încercați să realizați ce impact au vectorul de ponderi **w** și bias-ul b asupra hiperplanului (= dreaptă în $\mathbf{R}^2$) de separare $\mathbf{w}^T\mathbf{x} + b = 0$ a celor două clase.

Pentru problemele de clasificare binare liniar separabile, *algoritmul de învățare al lui Rosenblatt* arată că într-un număr finit de pași, putem găsi (*învăța*, *antrena*) un vector de ponderi $\mathbf{w}^*$ și un bias b^* care definesc un hiperplan de separare al celor două clase. Programul demonstrativ **nnd4pr** ilustrează acest algoritm de învățare. Rulați acest program și încercați să realizați ce se întâmplă cu vectorul de ponderi curent **w** și biasul curent b după fiecare clasificare corectă sau incorectă a unui punct din mulțimea de antrenare.

Pentru mulțimea de antrenare $S = \{(\mathbf{x}^1,t^1), (\mathbf{x}^2,t^2), ..., (\mathbf{x}^m,t^m)\}$ vrem să găsim $\mathbf{w}^*$ și b^* astfel încât pentru fiecare exemplu $\mathbf{x}^i$ ieșirea y^i a perceptronului să fie aceeași cu eticheta t^i, adică y^i = hardlims($\mathbf{w}^{*T}\mathbf{x}^i + b^*$) să fie egală cu t^i. O variantă a algoritmului de învățare a lui Rosenblatt este cea *incrementală* (sau *online*), în care vectorul de ponderi **w** și bias-ul b curenți sunt modificați după fiecare exemplu misclasificat. În general, pentru a obține soluția ($\mathbf{w}^*$, b^*) ce definește hiperplanul de separare, algoritmul necesită să parcurgă mulțimea de antrenare a exemplelor de mai multe ori. O asemenea parcurgere se numește *epocă*. Algoritmul se oprește în momentul în care de-a lungul unei epoci nu se modifică vectorul de ponderi **w** și bias-ul b curenți. În cele ce urmează prezentăm algoritmul lui Rosenblatt, varianta online.

Algoritmul de învățare al lui Rosenblatt, varianta online
Input: mulțimea de antrenare S = $\{(\mathbf{x}^1,t^1), (\mathbf{x}^2,t^2), ..., (\mathbf{x}^m,t^m)\}$;
Output: vectorul de ponderi w^* și bias-ul b^* (pentru cazul de liniar separabilitate aceștia determină hiperplanul de separare);
Iterația 0: Inițializează w și b: $\mathbf{w} = \mathbf{w}^0$, $b = b^0$, k = 0 iterația curentă.
Iterația k+1:

1. se selectează exemplul de antrenare $\mathbf{x}^i$ cu i = k mod (m+1);
2. se calculează y^i, eticheta exemplului x^i, $y^i = \text{hardlims}((\mathbf{w}^k)^T\mathbf{x}^i + b^k)$;
3. dacă $y^i = t^i$, se păstrează parametri curenți, adică $\mathbf{w}^{k+1} = \mathbf{w}^k$, $b^{k+1} = b^k$, altfel se modifică parametri astfel: $\mathbf{w}^{k+1} = \mathbf{w}^k + t^i\mathbf{x}^i$, $b^{k+1} = b^k + t^i$
4. dacă k e multiplu lui m (suntem la sfârșitul epocii k/m)
    a. dacă în epoca k/m nu s-au modificat parametri w și b atunci $w^* = w$, $b^* = b$. Algoritmul se termină.
5. k = k+1;

Realizați următoarele:

a. scrieți funcția ***algoritmRosenblattOnline.m*** care implementează algoritmul descris mai sus, punând condiția ca numărul de epoci de antrenare să nu depășească o valoare *nrMaximEpoci* dată. Funcția primește ca parametri mulțimea de antrenare S reprezentată prin matricea de example X și vectorul T de clase (cu valori 1 sau -1), valoarea nrMaximEpoci și returnează vectorul de ponderi w^*, biasul b^* precum și eroarea de antrenare (procentul de exemple misclasate) corespunzătoare fiecărei epoci.
b. apelați funcția ***algoritmRosenblattOnline.m*** pentru mulțimea de antrenare S = $\{([0,0]^T,1), ([0,0.5]^T,1), ([0,1]^T,1), ([0.5,0]^T,1), ([0.5,0.5]^T,-1), ([0.5,1]^T,-1), ([1,0]^T,-1), ([1,0.5]^T,-1)\}$, nrMaximEpoci = 10, reprezentând grafic punctele din mulțimea de antrenare (folosiți markere/culori diferite pentru puncte din clase diferite) și hiperplanul (dreapta) de separare (folosiți funcția plotpc.m) dat de w^* și b^*. Reprezentați evoluția erorii de antrenare.
c. scrieți funcția ***genereazaPuncteDeplasateFataDePrimaBisectoare.m*** care generează *m* puncte (x_i,y_i) în pătratul de dimensiuni [-1 1] x [-1 1] asociindu-le etichetele 1 sau -1 corespunzător situării deasupra/dedesubtul primei bisectoare. Ordonatele punctelor generate sunt apoi deplasate cu o valoare *deplasare* dată astfel: la ordonatele punctelor cu eticheta 1 se adună valoarea deplasare iar la ordonatele punctelor cu eticheta -1 se scade valoare deplasare.
d. apelați funcția ***genereazaPuncteDeplasateFataDePrimaBisectoare.m*** pentru m = 10, 50, 100, 250, 500 și pentru deplasare = 0.5, 0.3, 0.1,

0.01, -0.1, -0.3 obținând mulțimile de antrenare $S_{m,deplasare}$. Apelați apoi funcția ***algoritmRosenblattOnline.m*** pentru mulțimile de antrenare $S_{m,deplasare}$ plotând de fiecare dată hiperplanul de separare. Pentru fiecare valoare a parametrului deplasare, plotați numărul de epoci necesare algoritmului să conveargă la soluția (w^*,b^*) pentru cele 5 valori ale lui m. Ce observați?