b. soluții:

b.1. procedură bazată pe gradient: $a_n = a_{n-1} + \gamma_n \nabla_a Q(y_n, f(x_n, a_{n-1}))$, unde a_n = vectorul de parametri la pasul n, γ_n = rata de învățare, ∇_a = gradientul (în ce direcție și cu ce intensitate trebuie modificați parametrii), $Q(y_n, f(x_n, a_{n-1}))$ - o funcție care măsoară eroarea (sau pierderea) între valoarea estimată y_n și predicția modelului $f(x_n, a_{n-1})$.

b.2. ERM cu condiții convergență/ unicitate;

c. Demonstrații:

c.1. teoria aproximării stochastice pt procedurile bazate pe gradient

c.2 teoria convergenței uniforme pt principiul ERM.

Perceptronul - primește toate inputurile, cu ponderile lor și, printr-o funcție ce utilizează un prag, returnează una din 2 valori (peste / sub prag). Ultimul input are, de obicei, valoarea +1 sau -1, pornind de la input și vectorul comun ponderi w. Rezultatul (± 1) e comparat cu datele din setul de antrenament, numite generic, d: $\{x^1, d^1\}, \{x^2, d^2\}, \ldots$ Scopul perceptronului este găsirea vectorului ponderi w optim. Genul, putem considera că perceptronul împarte un spațiu în 2 părți.

Regula de învățare a perceptronilor (Rosenblatt, 1962): $\{w^1$ arbitrar
$w^{k+1} = w^k + \rho(d^k - y^k)$ (*)
unde ρ constantă denumită rată de învățare;
d^k, y^k din $\{-1, +1\}$. Există soluție (∃) una → (∃) o infinitate) doar dacă d^k, $k = 1, 2, \ldots, n$ (setul de antrenament este separabil liniar (subplanul dat cu valori +1 și cel dat cu -1).

Dacă $w^1 = 0 \Rightarrow$ limite de corecții pe care le poate face perceptronul în procesul de învățare k_0: $k_0 = \frac{\alpha^2 \|w^*\|^2}{\beta^2} = \frac{\beta^2 \|w^*\|^2}{\rho^2}$ sau $k_0 = \frac{\max_i \|x^i\|^2 \|w^*\|^2}{[\min_i (x^i)^T w^*]^2}$

- unde w^* e vectorul de greutăți ideal
- x_i sunt exemplele de antrenament
- k_0 = numărul maxim de corecții, pe care le face în procesul de învățare
- derivăm și obținem formule alternative, unde:

$\beta = \max_i \|x^i\|$ - maximul numerelor (lungimilor) exemplelor de antrenament

$\rho = \min_i x_i^T w^*$ - margin: distanța min a punctelor față de hiperplanul ideal definit de w^*.

Regula de învățare Widrow-Hoff (μ-LMS): cea implementează principiul perturbării minime: modifică actualizează ponderile cât mai puțin în direcția erorii → învățare stabilă și lină, nu haotică, ce duce ponderile la w^*. Acest vector minimizează media pătratelor erorilor dintre rezultatele rețelei și valorile corecte:

$J(w) = \frac{1}{2} \sum (d^i - y^i)^2$. Formula: $w^{k+1} = w^k - \mu \nabla J(w) = w^k + \mu \sum_{i=1}^{m} (d^i - y^i) x^i$,

- w = ponderile, μ = rata de învățare, d = etichete, y = ieșirea din rețea, $d^i - y^i$ = eroarea, m = pasul. Putem folosi J pentru a regla gradientul descendent pe panta cea mai abruptă a graficului $J(w)$ în punctul w^k, făcând variația Δw^k proporțională cu gradientul lui J în w^k. Obținem regula învățării gradientului descendent cel mai abrupt, sau a gradientului descendent. $J(w)$ definește o suprafață hiperparabolidală convexă cu un singur minimum: w^* (cel global).

(CURS)

$w^{k+1} = w^k - \rho \nabla J(w)\Big|_{w=w^k} = w^k - \rho\left[\frac{\partial J}{\partial w_1}\ \frac{\partial J}{\partial w_2} \dots \frac{\partial J}{\partial w_{m+1}}\right]^T\Big|_{w=w^k}$, ρ = rata de învățare (step-size) - cât de mare e pasul, iar $\nabla J(w)$ este vectorul derivatelor parțiale - direcția în care $J(w)$ crește cel mai rapid ⇒ luăm opusul ei pentru a coborî. Aceasta este regula batch LMS (Least Mean Squares) - se actualizează după toate exemplele, o dată/epocă.

- Regula μ-LMS sau LMS (se incrementează după fiecare exemplu/pas): $w^1 = 0$ sau arbitrar, $w^{k+1} = w^k + \mu(d^k - y^k)x^k$, unde $0 < \mu < 2/(\max_i \|x^i\|^2)$. Se poate demonstra că: $w^* = X^+ d$, unde $X = [x^1 x^2 \dots x^m]$ matricea de intrare, $d = [d^1 d^2 \dots d^m]^T$ vectorul de etichete, iar $X^+ = (XX^T)^{-1}X$ - e inversa generalizată ⇒ putem obține direct (fără iterații) soluția pt w^* dacă e o problemă liniar-separabilă și avem toate datele, dar: cost computațional mare pt inversă + nu se poate aplica dacă datele vin în flux. Regula batch e mai eficientă, dacă poate fi aplicată, însă diferența devine din ce în ce mai neglijabilă, cu cât se folosesc achiziții de epoci consecutive mai mare (ex. 20) pt. învățare.

Regula de învățare Delta - Extensie a regulii LMS, care permite învățarea și când funcția de activare este neliniară, dar derivabilă. Aplicăm gradientul descendent ⇒ $\nabla J(w) =$ $= -(d-y)f'(net)x$, unde $(d-y)$ = eroarea între răspuns dorit și răspuns actual; $f'(net)$ = derivata funcției de activare față de suma ponderată de intrări, x = vectorul de intrare. ⇒ regula Delta: $\begin{cases} w^1 \text{ arbitrar} \\ w^{k+1} = w^k + \rho[d^k - f(net^k)]f'(net^k)x^k = w^k + \rho\delta^k x^k, \end{cases}$

unde $net^k = (x^k)^T w^k$ și $f' = \frac{df}{dnet}$, iar $(x^k)^T$ e vectorul de intrare de la pasul k, transpus.

Dacă $f(net) = \tanh(\beta net) \Rightarrow f'(net) = \beta[1 - f^2(net)]$. Dacă $f(net) = \frac{1}{(1+e^{-\beta net})} \Rightarrow$

$\Rightarrow f'(net) = \beta f(net)[1 - f(net)]$.

- Funcția de activare este o funcție matematică aplicată individual pe fiecare perceptron (neuron), care primește ca argument suma ponderată a valorilor de la toți perceptronii din stratul anterior.

- Pentru a preveni stagnarea când $f'(net) \to 0$, se adaugă un bias pozitiv ε:
$\{ w^{k+1} = w^k + \rho[d^k - f(net\,k)][f'(net\,k) + \varepsilon]x^k$,

Capabilitățile computaționale ale perceptronilor: Dacă avem un set de antrenament $T = \{(x_i, t_i), i = 1:n\}$ și seturile de inputuri $S^+ = \{x : (\exists i)\ t_i = 1, x = x_i\}$ și $S^- = \{x : (\exists i)\ t_i = -1, x = x_i\}$ - seturi cu eticheta 1, respectiv -1.

Def. Separabilitatea liniară - spunem că S^+ și S^- sunt liniar separabile, dacă $(\exists)$ $\{S^+ \subset H^+, S^- \subset H^-$ (H - unde sunt semiplanuri/semispații) sau $(\forall)x \in S^+$, $w^T x - b \geq 0$, $(\forall x \in S^-)$ $w^T x - b < 0$, iar b = pragul (echivalent cu biasul - b în rețele neuronale)

(1) Un set de antrenament poate fi învățat fără eroare de un perceptron, dacă acesta este liniar separabil.

Notăm nr. maxim de dihotomii care poate fi introdus de un perceptron asupra unui set S de n elemente din R^d. $\mathcal{L}(n,d) = \max_{S : |S| = n,\ S \subset R^d} \mathcal{L}(S)$

Introducem sist de ecuații a diferențelor finite: $\Delta(n,d) = \Delta(n-1,d) + \Delta(n-1,d-1)$, unde $\Delta(n,d)$ = nr. maxim de dihotomii (etichetări cu ± 1) pe un set de n puncte, este calculat dacă adăugăm un pct nou (creștem n) sau creștem spațiul (d) și satisfacem condițiile: $\Delta(1,d) = \hat{2}(1,d) = 2$

$\Delta(n,1) = \hat{2}(n,1) = 2(n+1) - 2 = 2n$

1. Al un pct pe care-l poți etichetă în 2 feluri: ± 1.
2. Pentru spațiul cu 1 dimensiune (o axă), perceptronul împarte axa în intervale prin n puncte $\Rightarrow$ se pot genera doar $2n$ separări diferite ($\ll 2^n$), între fiecare 2 puncte separate: $+1$ la stg și -1 la dreapta sau invers.

Teorema limitei superioare (pt dihotomii / Teorema lui Cover): Pt $(\forall)$ set de n puncte din spațiul R^d, numărul maxim de etichetări ± 1 (adică dihotomii), pe care le poate realiza un perceptron este: $\Delta(n,d) = \begin{cases} 2\sum_{i=0}^{d} \binom{n-1}{i}, & \text{if } n > d+1 \\ 2^n, & \text{if } n \le d+1 \end{cases}$, unde $\binom{n-1}{i}$ sunt combinări de $n-1$ luate câte i = coeficientul binomial $\frac{(n-1)!}{i!(n-1-i)!}$

Deci, un perceptron poate învăța corect $(\forall)$ set, care are $\Delta(n,d) < 2^n$.

Deci, probabilitatea ca un perceptron Rosenblatt să fie capabil să distingă Telemințe e:

$P(n,d \ge \Delta(n,d)) / 2^n = \left(\frac{1}{2}\right)^{n-1} \sum_{k=0}^{d} \binom{n-1}{k}$.

Imputuri binare: primesc cuvânt de k biți și returnează 0 sau 1 pt fiecare. Numărul total de combinații (imputuri) posibile $= 2^d$.

Numărul total de funcții booleene posibile cu d intrări: $\hat{2}(2^d)$ - pentru că fiecare combinație poate returna 2 valori: ± 1. Numărul maxim de funcții booleene implementate de un perceptron $B(d) \le \Delta(2^d, d) < 2^{d^2}/(d-1)!$, unde $B(d)$ = nr. exact de funcții booleene implementabile de 1 perceptron

$\Delta(2^d, d)$ = limita superioară teoretică (dar nu exactă) pt. câte dihotomii se pot realiza în R^d cu 2^d puncte de un perceptron. $\hat{2}(d^2)/(d-1)!$ = o limită mai jos, dar care scade rapid ca proporție din $\hat{2}(2^d)$.

Regula de învățare adaptivă Ho-Kashyap (AHK): 3 reguli (porunite de la LMS): AHK I (probleme separabile liniar), AHK II (probleme separabile liniar), AHK III (probleme non-separabile liniar).

Considerăm o problemă de distingere în 2 clase c_1 și c_2 a unor seturi de n numere, notate fiecare generic cu x. Avem un preche (set de numere, etichetă) pt. antrenament $\{x^i, d^i\}$, $i = 1, 2, \dots, m$, $x^i \in R^{n+1}$, iar ultimul x^i este un bias constant cu valoarea 1. Considerăm sgn - funcția de activare a perceptronului. Atunci un perceptron poate fi învățat să distingă corect prechile de antrenament, dacă sunt satisfăcute următoarele inegalități, pt $i = 1, \dots, m$:

w = un singur vector de ponderi, care se aplică fiecărui x^i: $(x^i)^T w \begin{cases} > 0, & \text{if } d^i = +1 \\ < 0, & \text{if } d^i = -1 \end{cases}$

Dacă notăm vectorii z: $z^i = \begin{cases} +x^i, & \text{if } d^i = +1 \\ -x^i, & \text{if } d^i = -1 \end{cases}$ (sau $z^i = x^i \cdot d^i$) și avem pseudoinversa $Z^+ = [z^1 z^2 \dots z^m]$, atunci $Z^T w > 0$.

Dacă definim un vector m-dimensional pozitiv, $b = [b^1 b^2 \dots b^m]$ (fiecare b^i să fie poate o anumită valoare), obținem $Z^T w = b$. Facem acest pentru a putea folosi la ce prog de pregătire a distingerii, ce trebuie egalat sau depășit de fiecare set w. La alegerea arbitrară e o alegere proastă, el trebuie lăsat să fie dinamic găsit.

În algoritmul Ho-Kashyap se încearcă ajungerea la o valoare a lui b iterativ, pornind de la un b mic. Apoi, se încearcă minimizarea funcției de cost:

$J(w, b) = \frac{1}{2} \| Z^T w - b \|^2$.

CURS

Norma euclidiană a vectorului: $\|v\| = \sqrt{v_1^2 + \dots + v_n^2}$

Dacă $w = Z^+ b$, unde $Z^+ = (ZZ^T)^{-1} Z$, pt. $m > n+1$. În continuare, recalculăm succesiv $b^{k+1} > 0$ cu formula de gradient descendent: $b^{k+1} = b^k + \frac{1}{2}[\varepsilon + |\varepsilon|^k]$, unde $\varepsilon^k = Z^T \cdot w^k - b^k$. Apoi recalculăm w. Așadar, până când $\varepsilon = 0$ sau foarte aproape și pozitiv (indică separabilitate liniară) sau până când $\varepsilon < 0$ (indică lipsa separabilității liniare). Calculare directă a Z^+ poate fi evitată cu un algoritm gradient-descendent: procedura adaptivă mod batch Ho-Kashyap. Pornim de la $J(w, b) = \frac{1}{2}\|Z^T w - b\|^2$. Apoi, gradientul lui J în funcție de w și b este dat de: $\nabla_b J(w,b)|_{w^k, b^k} = -(Z^T w^k - b^k)$ respectiv $\nabla_w J(w,b)|_{w^k, b^{k+1}} = -Z(Z^T w^k - b^{k+1})$. Dacă folosim în locul ecuației de gradient descendent, $-0{,}5(\varepsilon + |\varepsilon^k|)$, cu $\varepsilon^k = Z^T w^k - b^k$. atunci: $b^{k+1} = b^k + \frac{\rho_1}{2}(|\varepsilon^k| + \varepsilon)$, cu $\varepsilon^k = Z^T w^k - b^k$

$w^{k+1} = w^k - \rho_2 Z(Z^T w^k - b^{k+1}) = w^k + \frac{\rho_1 \rho_2}{2} Z\left[|\varepsilon^k| + \varepsilon^k\left(1 - \frac{2}{\rho_1}\right)\right]$, unde:

ρ_1 și ρ_2 = constante de învățare strict pozitive.

Dacă $\rho_1 = 0$ și $b^1 = 1$, ecuația se reduce la regula de învățare μ-LMS. for dacă $0 < \rho_2 < 2$ și $0 < \rho_2 < \frac{2}{\lambda_{max}}$, unde λ_{max} e cea mai mare valoare proprie a matricei de finite ZZ^T.

O procedură adaptivă Ho-Kashyap se obține de la funcția criteriu de instanță:

$J(w, b) = \frac{1}{2}[(z^i)^T w - b^i]$, care generează regulile de incrementare:

$b_i^{k+1} = b_i^k + \frac{\rho_1}{2}(|\varepsilon_i|^k + \varepsilon_i^k)$, cu $\varepsilon_i^k = (z^i)^T w^k - b_i^k$

$w^{k+1} = w^k - \rho_2 z^i[(z^i)^T w^k - b_i^{k+1}] = w^k + \frac{\rho_1 \rho_2}{2}\left[|\varepsilon_i^k| + \varepsilon_i^k\left(1 - \frac{2}{\rho_1}\right)\right] z^i$

Alternativ, scriem: $\Delta b_i = \rho_1 \varepsilon_i^k$ și $\Delta w = \rho_2(\rho_1 - 1)\varepsilon_i^k z_i$, dacă $\varepsilon_i^k < 0$

$\Delta b_i = 0$ și $\Delta w = -\rho_2 \varepsilon_i^k z_i$, dacă $\varepsilon_i^k \geq 0$, unde Δb și Δw = diferențele dintre valorile actualizate și curente ale lui b și w. Aceasta este regula de învățare AHK I.

Regula de învățare AHK II pornește de la AHK I, dar permite reducerea Δb și în spectrul valorilor negative, cât timp nu se obține o margine negativă:

$\Delta b_i = \rho_1 \varepsilon_i^k$ și $\Delta w = \rho_2(\rho_1 - 1)\varepsilon_i^k z^i$, dacă $b_i^k + \rho_1 \varepsilon_i^k > 0$

$\Delta b_i = 0$ și $\Delta w = -\rho_2 \varepsilon_i^k z^i$, dacă $b_i^k + \rho_1 \varepsilon_i^k \leq 0$, unde:

$0 < \rho_2 < \frac{2}{\max \|z^i\|^2}$ și $0 < \rho_1 < 2$

Regula de învățare AHK III se poate aplica și problemelor neseparabile liniar. Aici Δw e setat la 0 în regula AHK II pt. $\Delta b_i = 0$

Norma Manhattan

Versiune batch: $J(w) = \frac{1}{r}\sum_{i=1}^{m} |d^i - y^i|^r$ sau versiunea de instanță:

$J(w) = \frac{1}{r}|d^i - y^i|^r$. Forma generală a funcției gradient:

$\nabla J(w) = -\operatorname{sgn}(d-y)|d-y|^{r-1} f'(net) x$

Regula de învățare prin corelare. Pornește de la funcția criteriu de calculare a gradientului descendent: $J(x) = -\sum_{i=1}^{m} y^i d^i$. Calculăm w: $w^1 = 0$, $w^{k+1} = w^k + \rho d^k x^k$, unde $y^i = (x^i)^T w$

Dacă setăm $\rho = \eta$, atunci $\Rightarrow w^* = \sum_{i=1}^{m} d^i x^i = Xd$.

Regula de învățare prin corelație (varianta lotcea prin corelație)

Pornind de la funcția criteriu pentru gradient:

$J(w) = -\frac{1}{2}\sum_{i=1}^{m}(y^i - \langle y \rangle)(d^i - \langle d \rangle)$, unde $\langle y \rangle$ și $\langle d \rangle$ sunt medii calculate pentru toate perechile de antrenament, ptr. output și respectiv, etichete.

O altă funcție criteriu este măsurarea erorii relative de entropie instantanee:

$J(w) = \frac{1}{2}\left[(1+d)\ln\left(\frac{1+d}{1+y}\right) + (1-d)\ln\left(\frac{1-d}{1-y}\right)\right]$, unde $d \in (-1, 1)$, deci $y = f(net) = \tanh(\beta net)$, atunci $\nabla J(w) = -\beta(d-y)x$.

Considerăm următoarea funcție criteriu generală ptr. gradientul descendent:

$J(w) = \sum_{i=1}^{m} g(z^T w)$, unde $z = x$, deci $x \in$ clasei C_1 și $z = -x$, deci $x \in$ clasei C_2.

Fie $s = z^T w$, atunci $J(w)$ este bine formulat (ne va ajunge garantat un w^*), dacă (condițiile lui Wittner și Butler):

1. $g(s)$ este diferențiabilă;
2. ptr. $(\forall)$ s, $-dg(s)/ds \geq 0$
3. $(\exists)$ $\varepsilon > 0$ a.î. $-dg(s)/ds \geq \varepsilon$, ptr. $(\forall)$ $s \leq 0$
4. $g(s)$ este mărginită inferior