Adaptacyjny neuron liniowy

Jacek Kluska

Politechnika Rzeszowska

Adaline = Adaptive Linear Neuron - postawienie problemu

1. Rozszerzone wektory:

$$\mathbf{w} = [w_1, \ldots, w_N, b]^T, \; \mathbf{x} = [x_1, \ldots, x_N, 1]^T$$

2. Liniowa funkcja aktywacji:

$$f(n) = n.$$

Postawienie problemu: Dla danych uczących

$$\{(\mathbf{x}_1, d_1), \ldots, (\mathbf{x}_Q, d_Q)\} \subset \mathbb{R}^{N+1} \times \mathbb{R}$$

i neuronu

$$y = \mathbf{w}^T \mathbf{x}$$

należy podać taki sposób uczenia, aby

$$E = \sum_{k=1}^{Q} (d_k - y_k)^2 \to \min$$

Metoda 1: Równanie Wienera-Hopfa

1. Liczność zbioru danych

$$Q \geqslant N + 1$$

2. "Zakłócenia"

$$d_k = \mathbf{w}^T \mathbf{x}_k + e_k$$

Definiujemy

$$X = [\mathbf{x}_1, \ldots, \mathbf{x}_Q], \qquad D = [d_1, \ldots, d_Q]^T$$

Stąd

$$E = \sum_{k=1}^{Q} \left(d_k - \mathbf{w}^T \mathbf{x}_k \right)^2$$

$$= \sum_{k=1}^{Q} d_k^2 - 2\mathbf{w}^T \sum_{k=1}^{Q} d_k \mathbf{x}_k + \mathbf{w}^T \sum_{k=1}^{Q} \mathbf{x}_k \mathbf{x}_k^T \mathbf{w}$$

$$= D^T D - 2\mathbf{w}^T X D + \mathbf{w}^T X X^T \mathbf{w}$$

$$\frac{\partial E}{\partial \mathbf{w}} = \frac{\partial \left(D^T D - 2\mathbf{w}^T X D + \mathbf{w}^T X X^T \mathbf{w} \right)}{\partial \mathbf{w}}$$
$$= \left(XX^T + \left(XX^T \right)^T \right) \mathbf{w} - 2XD = \mathbf{0}$$

1. Dla $\det\left(XX^T\right) \neq 0$ otrzymujemy **równanie Wienera-Hopfa**

$$\mathbf{w}^* = \left(XX^T\right)^{-1} XD$$

2. Dla $\det\left(XX^T\right) = 0$ poszukuje się pseudoinwersji

$$X^+ = \left(XX^T\right)^{-1} X$$

   Wada: Odwracanie macierzy o dużych rozmiarach.

3. Minimum E

$$E\left(\mathbf{w}^*\right) = D^T \left(D - X^T \mathbf{w}^* \right)$$

Metoda 2: Algorytm Newtona-Raphsona

Hesjan

$$H = \frac{\partial^2 E}{\partial \mathbf{w}^2} = \frac{\partial \left(2XX^T\mathbf{w} - 2XD\right)}{\partial \mathbf{w}} = 2XX^T$$

$$H^{-1} = \frac{1}{2}\left(XX^T\right)^{-1}, \quad \text{dla} \quad \det XX^T \neq 0$$

Mnożymy obustronne przez H^{-1} równanie

$$\frac{\partial E}{\partial \mathbf{w}} = 2XX^T\mathbf{w} - 2XD$$

Metoda 2: Algorytm Newtona-Raphsona - c.d.

Dla $\mathbf{w}^* = \left(XX^T\right)^{-1} XD$ jest

$$H^{-1}\frac{\partial E}{\partial \mathbf{w}} = \frac{1}{2}\left(XX^T\right)^{-1}\left(2XX^T\mathbf{w} - 2XD\right)$$

$$H^{-1}\frac{\partial E}{\partial \mathbf{w}} = \mathbf{w} - \mathbf{w}^*$$

Algorytm Newtona-Raphsona

$$\mathbf{w}^* = \mathbf{w} - H^{-1}\frac{\partial E}{\partial \mathbf{w}}$$

f - liniowa: start z **dowolnego** $\mathbf{w} \Rightarrow$ rozwiązanie po pierwszej iteracji o ile $H > 0$.

Metoda 3: Idealna metoda spadku gradientu

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta_1 \frac{\partial E}{\partial \mathbf{w}}|_k, \qquad \eta_1 > 0$$

Uwaga

Metoda bazuje na **epoce** uczenia (epoch-based method). Podczas k - tej epoki prezentuje się **cały zbiór uczący** a nie pojedynczy element.
Niech $\eta = 2\eta_1 > 0$.

$$\begin{aligned} \mathbf{w}_{k+1} &= \mathbf{w}_k - \frac{\eta}{2}\left(2XX^T\mathbf{w}_k - 2XD\right) \\ &= \left(I - \eta XX^T\right)\mathbf{w}_k + \eta XD \\ &= A\mathbf{w}_k + B \end{aligned}$$

Metoda 3: Idealna metoda spadku gradientu - c.d.

$\mathbf{w}_1 = A\mathbf{w}_0 + B, \quad \mathbf{w}_2 = A^2\mathbf{w}_0 + AB + B$
$\mathbf{w}_3 = A^3\mathbf{w}_0 + A^2B + AB + B, \quad \ldots$

$$\begin{aligned}\mathbf{w}_k &= A^k\mathbf{w}_0 + \sum_{m=0}^{k-1} A^m B \\ &= \left(I - \eta XX^T\right)^k \mathbf{w}_0 + \eta \sum_{m=0}^{k-1} \left(I - \eta XX^T\right)^m XD\end{aligned}$$

Dla $\mathbf{w}_0 = \mathbf{0}$ przy $k \to \infty$ jest $\mathbf{w}_\infty = \mathbf{w}^*$:

$$\mathbf{w}^* = \underbrace{\sum_{m=0}^{\infty} \left(I - \eta XX^T\right)^m \eta XD}_{X^+}$$

Rozwiązanie jak w metodach 1 i 2.
Zbieżność procedury:

$$\underline{0 < \eta < \frac{2}{trace\left(XX^T\right)}}$$

Metoda 4: Metoda "delta" Widrowa-Hoffa

Nie bazuje ona na epoce uczenia, lecz na **pojedynczym** elemencie pary uczącej ze zbioru treningowego (pattern-based method).
Idea: $E_k = (d_k - y_k)^2/2$, $y = f(n) = f\left(\mathbf{w}^T\mathbf{x}\right)$.

$$
\begin{aligned}
\mathbf{w}_{k+1} &= \mathbf{w}_k - \eta \frac{\partial E}{\partial \mathbf{w}}\Big|_k \\
&= \mathbf{w}_k + \eta\,(d_k - y_k)\, f'(n_k)\, \mathbf{x}_k \\
&= \mathbf{w}_k + \eta \delta_k \mathbf{x}_k, \qquad \eta > 0
\end{aligned}
$$

$$
\underline{0 < \eta < \eta_c = \frac{2}{trace\left(XX^T\right)}}
$$

Dla $\eta < 0.1\eta_c$ rezultat jest taki sam, jak w metodzie 3 (Kecman 2001, s.236).

Metoda 5: Rekurencyjna metoda najmniejszych kwadratów (RLS=Recursive Least Squares)

$$d_k = \mathbf{x}_k^T \mathbf{w}_k + e_k$$
$$d_{k+1} = \mathbf{x}_{k+1}^T \mathbf{w}_{k+1} + e_{k+1}$$

Jednak dla wzorca $(\mathbf{x}_{k+1}, d_{k+1})$ używamy **tej samej wagi $\mathbf{w}_k$**

$$\underline{d_{k+1} = \mathbf{x}_{k+1}^T \mathbf{w}_k + e_{k+1}}$$

Korzystamy z równania Widrowa-Hoffa

$$\mathbf{w}_k = \left(X_k X_k^T \right)^{-1} X_k D_k$$

Cel: nie chcemy odwracać macierzy.

Metoda 5: RLS - c.d.

$$\mathbf{w}_k = \underbrace{\left(X_k X_k^T\right)^{-1}}_{P_k} \underbrace{X_k D_k}_{F_k} = P_k F_k$$

$$\mathbf{w}_{k+1} = \left(X_{k+1} X_{k+1}^T\right)^{-1} X_{k+1} D_{k+1} = P_{k+1} F_{k+1}$$

$$\begin{aligned} P_{k+1}^{-1} &= [X_k, \mathbf{x}_{k+1}] \begin{bmatrix} X_k^T \\ \mathbf{x}_{k+1}^T \end{bmatrix} \\ &= X_k X_k^T + \mathbf{x}_{k+1} \mathbf{x}_{k+1}^T = P_k^{-1} + \mathbf{x}_{k+1} \mathbf{x}_{k+1}^T \end{aligned}$$

$$\begin{aligned} F_{k+1} &= X_{k+1} D_{k+1} = [X_k, \mathbf{x}_{k+1}] \begin{bmatrix} D_k \\ d_{k+1} \end{bmatrix} \\ &= F_k + \mathbf{x}_{k+1} d_{k+1} \end{aligned}$$

Metoda 5: RLS - c.d.

Lemma

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B\left(C^{-1} + DA^{-1}B\right)^{-1} DA^{-1}$$

Stąd

$$P_{k+1} = \left(\underbrace{P_k^{-1}}_{A} + \underbrace{\mathbf{x}_{k+1}}_{B} \cdot 1 \cdot \underbrace{\mathbf{x}_{k+1}^T}_{D} \right)^{-1}$$

$$= \underline{P_k - P_k \mathbf{x}_{k+1} \left(1 + \mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1}\right)^{-1} \mathbf{x}_{k+1}^T P_k}$$

Metoda 5: RLS - c.d.

Fakty:

1. $\mathbf{w}_{k+1} = P_{k+1}F_{k+1} = P_{k+1}\left(F_k + \mathbf{x}_{k+1}d_{k+1}\right)$
2. $\mathbf{w}_k = P_k F_k \Rightarrow F_k = P_k^{-1}\mathbf{w}_k$
3. $d_{k+1} = e_{k+1} + \mathbf{x}_{k+1}^T \mathbf{w}_{k+1}$
4. $F_{k+1} = F_k + \mathbf{x}_{k+1}d_{k+1}$
5. $P_{k+1} = \left(P_k^{-1} + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^T\right)^{-1}$

$$
\begin{aligned}
\mathbf{w}_{k+1} &= P_{k+1}\left(F_k + \mathbf{x}_{k+1}d_{k+1}\right) \\
&= P_{k+1}\left(P_k^{-1}\mathbf{w}_k + \mathbf{x}_{k+1}\left(e_{k+1} + \mathbf{x}_{k+1}^T\mathbf{w}_k\right)\right) \\
&= P_{k+1}\left(\left(P_k^{-1} + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^T\right)\mathbf{w}_k + \mathbf{x}_{k+1}^T e_{k+1}\right) \\
&= \underline{\mathbf{w}_k + P_k \mathbf{x}_{k+1} e_{k+1}}
\end{aligned}
$$

Metoda 5: RLS - algorytm

Dane $\{(\mathbf{x}_1, d_1), \ldots, (\mathbf{x}_Q, d_Q)\} \in \mathbb{R}^{N+1} \times \mathbb{R}$

$$\mathbf{w} = [w_1, \ldots, w_N, b]^T, \quad \mathbf{x} = [x_1, \ldots, x_N, 1]^T$$

1. $w_1 = 0$, $P_1 = \mathrm{diag}(\alpha, \ldots, \alpha)_{(N+1)\times(N+1)}$, $\alpha = 10^8 ... 10^{15}$
2. Faza prezentacji: $k = 1$

$$e_k = d_k - \mathbf{x}_k^T \mathbf{w}_k$$

3. $e_{k+1} = d_{k+1} - \mathbf{x}_{k+1}^T \mathbf{w}_k$
4. Czynnik zapominania $\lambda \in [0.92,\ 0.99]$

$$P_{k+1} = \frac{1}{\lambda}\left(P_k - P_k \mathbf{x}_{k+1}\left(\lambda + \mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1}\right)^{-1} \mathbf{x}_{k+1}^T P_k\right)$$

5. $\mathbf{w}_{k+1} = \mathbf{w}_k + P_k \mathbf{x}_{k+1} e_{k+1}$
6. If $\sum_{k=1}^{Q} e_k^2 < \varepsilon$ then STOP else 2.