

Rețele neuronale și deep learning

Cursul 1: Deep learning pentru afaceri: de la modelele liniare la perceptron și la perceptronul multistrat (MLP)

Prof. dr. Daniel Traian PELE

Academia de Studii Economice din București (ASE), Departamentul de Statistică și Econometrie
Academia Română, Institutul de Prognoză Economică (IPE)
IDA Institute Digital Assets ▪ AI4EFin ▪ MSCA Digital Finance

Obiectivele cursului

- Să situați deep learning printre instrumentele unui analist de afaceri: unde creează valoare și unde un model liniar este suficient.
- Să scrieți regresia liniară și regresia logistică drept rețele cu un singur neuron, cu funcția de pierdere ca verosimilitate.
- Să explicați perceptronul lui Rosenblatt, regula lui de învățare, teorema de convergență și limita lui (problema sau-ului exclusiv (XOR)).
- Să construiți un perceptron multistrat (MLP): propagarea înainte, funcțiile de activare, numărul de parametri, aproximarea universală.
- Să comparați un MLP cu regresia logistică și cu gradient boosting pentru scoringul de credit pe date reale, cu costuri și incertitudine.

Abrevieri

AI	inteligență artificială
AUC	aria de sub curba caracteristică de operare a receptorului
CPU	procesor central
DM	marca germană
ES	pierderea așteptată în coada distribuției (expected shortfall)
GELU	unitate liniară cu eroare gaussiană
GNN	rețea neuronală pe grafuri
GPU	procesor grafic
GraphSAGE	eșantionare și agregare pe graf (o arhitectură GNN)
HMEQ	setul de date despre creditele garantate cu locuința (home equity)
ICBE	International Conference on Business Excellence (conferința internațională de excelență în afaceri)
IEEE-CIS	Institute of Electrical and Electronics Engineers, Computational Intelligence Society
LLM	model lingvistic mare
MLP	perceptron multistrat
MSE	eroarea pătratică medie
PD	probabilitatea de neplată
ReLU	unitate liniară rectificată
ROC	caracteristica de operare a receptorului
SGD	coborîrea stohastică pe gradient
SHAP	explicații aditive Shapley
UCI	Universitatea California din Irvine (Machine Learning Repository)
USD	dolar american
VaR	valoarea la risc
VLM	model vizual-lingvistic
XAI	inteligență artificială explicabilă
XGBoost	extreme gradient boosting (arbori de decizie cu amplificare pe gradient)
XOR	sau exclusiv

Cuprins

- Deep learning pentru afaceri
- Modelele liniare sînt rețele cu un singur neuron
- Perceptronul
- Perceptronul multistrat
- Studiu de caz: scoringul de credit pe credite reale
- Sinteză

Deep learning învață singur variabilele pe care analiștii le construiau manual

- **Pe scurt:** deep learning, ramura inteligenței artificiale (AI) din spatele sistemelor actuale pentru imagini, text și vorbire, estimează din date un lanț de funcții simple.

$$f(\mathbf{x};\theta) = f^{(L)}\big(f^{(L-1)}(\cdots f^{(1)}(\mathbf{x}))\big), \qquad \hat{\theta} = \arg\min_{\theta} \frac{1}{n}\sum_{i=1}^{n} \ell\big(y_i, f(\mathbf{x}_i;\theta)\big)$$

- **unde**
  - $\mathbf{x}$ intrarea (solicitant, tranzacție, text), $f^{(l)}$ stratul l din L, θ toate ponderile
  - ℓ pierderea, n volumul eșantionului
- De ce acum: datele, hardware-ul și software-ul au apărut în același timp
  - Seturi mari de date etichetate, procesoare grafice (GPU) și diferențiere automată în PyTorch (Paszke et al., 2019).
- Ce s-a schimbat: învățarea reprezentărilor
  - Rețeaua își construiește singură variabilele din datele brute (LeCun, Bengio & Hinton, 2015).

- **În practică:** întrebați ce ar învăța rețeaua în plus față de ce poate scrie manual un analist de scoring.

(Bishop & Bishop, 2024, ch. 1, 6)

Unde creează valoare deep learning în afaceri și finanțe

Domeniu	Decizie	Rețea	Datele cursului
Credit	acceptarea sau respingerea unui credit; prețul riscului	MLP (cursurile 1, 3)	German credit, credite garantate cu locuința (HMEQ)
Plăți	blocarea sau aprobarea unei tranzacții cu cardul	MLP, autoencoder	fraude cu carduri (cursurile 2, 3, 8)
Piețe	dimensionarea unei poziții; o limită de risc	recurentă, transformer	indici, acțiuni de la Bursa de Valori București
Energie	oferta pe piața pentru ziua următoare	recurentă, modele fundație	prețurile energiei din România
Text	citirea știrilor, rapoartelor, discursurilor	embedding, transformer	PhraseBank, tweet-uri, discursuri
Rețele	semnalarea spălării banilor într-un graf de plăți	rețele pe grafuri	Elliptic Bitcoin

- Cursul urmează acest tabel: cursurile 1–3 date tabelare, 4 imagini, 5 și 13 serii de timp, 6, 7 și 12 text, 9 grafuri, 10 decizii, 11 explicații.

- **În practică:** porniți de la decizie și de la costul ei, apoi alegeți rețeaua potrivită tipului de date.

Pentru date tabelare puține, un model liniar sau boosting este adesea suficient

- **Pe scurt:** deep learning cîștigă pe date brute, de dimensiune mare; pe cîteva zeci de coloane construite trebuie să-și justifice locul.

- Comparația metodelor de scoring: din 41 de clasificatori pe 8 seturi de date de credit, regresia logistică rămîne o referință solidă
  - Ansamblurile de modele conduc (Lessmann et al., 2015).
- Comparația pe date tabelare: ansamblurile de arbori încă depășesc rețelele adînci pe date tabelare obișnuite, de mărime medie (Grinsztajn, Oyallon & Varoquaux, 2022).
- Reglementare: decizia de creditare trebuie explicată solicitantului.

- Folosiți o rețea cînd: intrările sînt imagini, text, secvențe sau grafuri
  - Datele sînt multe
  - Relația este puternic neliniară.
- Păstrați modelul de referință cînd: avem cîteva mii de rînduri, variabile construite, nevoie de un scorecard.
- Acest curs testează ambele situații pe credite reale, după un protocol fix.

- **În practică:** fiecare rețea din acest curs este raportată alături de un model de referință liniar sau de boosting.

Datele: trei seturi reale de date de credit

Setul de date	Rînduri	Evenimente	Sursa
German credit	1 000	300 riscuri proaste	Universitatea California din Irvine (UCI), Machine Learning Repository (Hofmann, 1994)
Credite garantate cu locuința (HMEQ)	5 960	1 189 neplăți	Baesens, Rösch & Scheule (2016)
California housing	20 640	(regresie)	grupuri de locuințe, recensămîntul din 1990 (Pace & Barry, 1997)

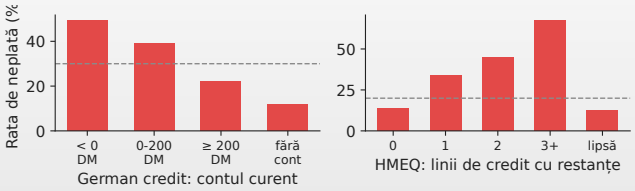
- German credit: 20 de atribute, sume în mărci germane (DM); rata de neplată 30,0 %.
- HMEQ: 12 variabile, rata de neplată 19,9 %; valorile lipsă sînt păstrate și modelate.
- California: valoarea mediană a locuințelor în sute de mii de dolari americani (USD), plafonată de sursă la 5.

- **În practică:** verificați sursa, unitatea de observație și frecvența evenimentului înaintea oricărui model.

Exercițiu practic: notebook §1

Ratele de neplată diferă puternic între grupuri simple de solicitanți

- **Pe scurt:** înaintea oricărui model, ratele brute de neplată arată ce variabile poartă risc.



Sursa: UCI Statlog German credit; Baesens, Roesch și Scheule (2016), HMEQ. Date: 1 000 de solicitanți (German credit); 5 960 de credite (HMEQ).

- Probabilitatea de neplată (PD): 49,3 % cu sold negativ în contul curent, 11,7 % fără cont
  - 14,0 % fără linii cu restanțe, 67,3 % cu trei sau mai multe.

- **În practică:** un model trebuie să reproducă cel puțin aceste rate pe setul de validare.

(calcul propriu)

NN_L1_default_rates

Învățarea ca minimizare a pierderii medii pe datele de antrenare, evaluată pe date noi

- **Pe scurt:** estimăm pe un eșantion și plătim pentru erorile făcute cu următorii solicitanți.

$$\underbrace{\hat{R}_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell\big(y_i, f(\mathbf{x}_i;\theta)\big)}_{\text{riscul empiric (antrenare)}} \qquad \text{vs.} \qquad \underbrace{R(\theta) = \mathbb{E}_{(\mathbf{x},y)\sim P}\, \ell\big(y, f(\mathbf{x};\theta)\big)}_{\text{riscul pe solicitanți noi}}$$

- **unde**
  - P distribuția necunoscută a viitorilor solicitanți, ℓ pierderea, minimizăm $\hat{R}_n$, banca plătește R
- Trei seturi: antrenare (estimăm θ), validare (alegem modelul), testare (raportăm o dată).
- HMEQ, stratificat: 3 576 de credite de antrenare, 1 192 de validare, 1 192 de testare.
- Fără scurgere de informație: medianele și standardizarea se estimează doar pe partea de antrenare.

- **În practică:** setul de testare se folosește o singură dată, la final (schema de validare: cursul 3).

(Bishop & Bishop, 2024, ch. 1)

Exercițiu practic: notebook §2

Cuprins

- Deep learning pentru afaceri
- Modelele liniare sînt rețele cu un singur neuron
- Perceptronul
- Perceptronul multistrat
- Studiu de caz: scoringul de credit pe credite reale
- Sinteză

Regresia liniară este un neuron fără funcție de activare

- **Pe scurt:** o sumă ponderată a intrărilor plus un termen liber; pierderea pătratică este log-verosimilitatea gaussiană.

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b, \qquad \mathrm{MSE}(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \mathbf{w}^\top \mathbf{x}_i - b \right)^2$$

- **unde**
  - $\mathbf{x} \in \mathbb{R}^d$ intrările, $\mathbf{w}$ ponderile, b termenul liber
  - Eroarea pătratică medie (MSE) pierderea

$$\varepsilon_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2): \quad -\log L = \tfrac{n}{2} \log 2\pi\sigma^2 + \tfrac{n}{2\sigma^2} \mathrm{MSE}$$

- Verosimilitate maximă = cele mai mici pătrate: același $\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$.
- O rețea nu are o formulă explicită: se antrenează prin coborîre pe gradient (cursul 2).
- California, doar latitudinea: MSE în afara eșantionului 0,896 pentru dreaptă.

- **În practică:** dreapta de regresie este reperul pe care orice rețea de regresie trebuie să-l depășească în afara eșantionului.

(Bishop & Bishop, 2024, ch. 4)

Regresia logistică este un neuron cu funcție sigmoidă

- **Pe scurt:** neuronul produce o probabilitate, iar pierderea este minus log-verosimilitatea Bernoulli (entropia încrucișată).

$$p_i = \sigma(\mathbf{w}^\top \mathbf{x}_i + b) = \frac{1}{1 + e^{-z_i}}, \qquad \nabla_{\mathbf{w}} \mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} (p_i - y_i)\, \mathbf{x}_i$$

$$\mathcal{L}(\mathbf{w}, b) = -\frac{1}{n} \sum_{i=1}^{n} \Big[y_i \log p_i + (1 - y_i) \log(1 - p_i) \Big]$$

- Gradientul este eroarea înmulțită cu intrarea: aceeași formă apare în fiecare strat la propagarea înapoi.
- e^{w_j} este raportul șanselor pentru o creștere cu o unitate a lui x_j.
- Pierdere convexă: un singur minim global.

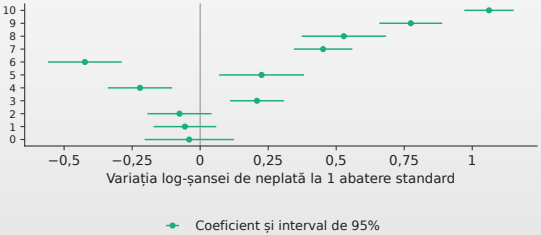
- **unde**
  - z_i logitul (logaritmul șansei), σ funcția sigmoidă, $y_i \in \{0, 1\}$ neplata
  - Derivarea în anexă

- **În practică:** scorul logistic este standardul industriei pentru modelele PD.

(Bishop & Bishop, 2024, ch. 5)

Restanțele și lipsa raportului datorii/venit cresc cel mai mult șansa de neplată

- **Pe scurt:** HMEQ, intrări standardizate: fiecare coeficient este variația log-șansei la o abatere standard.



Sursa: Baesens, Roesch și Scheule (2016), datele HMEQ. Date: 3 576 de credite de antrenare.

- Rapoartele șanselor e^{w_j}: 2,89 pentru raportul datorii/venit lipsă, 2,17 pentru liniile cu restanțe, 0,65 pentru un istoric mai vechi (pseudo-R^2 McFadden 0,374).

- **În practică:** un semn contrar intuiției (valoarea proprietății) este un motiv să analizați datele.

(calcul propriu)

NN_L1_logit_coef

Propagarea înainte pentru un solicitant real, pas cu pas

- **Pe scurt:** scorul este o sumă de contribuții $w_j x_j$; funcția sigmoidă îl transformă într-o probabilitate.

Intrare	x_j	w_j	$w_j x_j$
linii de credit cu restanțe	1,46	0,774	+1,13
numărul de linii de credit	2,81	−0,221	−0,62
raport datorii/venit lipsă	−0,51	1,062	−0,54
termenul liber b			−1,99
celelalte 8 intrări			+0,31
scor z			**−1,71**

- Solicitant din setul de testare care a intrat ulterior în neplată: credit de 15 400 USD, raport datorii/venit 36,4 %, 2 linii cu restanțe.
- Ieșirea: $p = \sigma(z) = 15{,}3\,\%$, sub pragul implicit de 50 %.
- Intrările x_j sînt standardizate: $x_j = (\text{brut}_j - \bar{x}_j)/s_j$, cu mediile și abaterile standard de antrenare.

- **În practică:** pragul este o decizie de afaceri: secțiunea 5 îl stabilește din costul unei neplăți neidentificate.

(calcul propriu)

Exercițiu practic: notebook §2

Valorile lipsă ca predictori ai neplății

- **Pe scurt:** solicitanții care nu își declară raportul datorii/venit intră mult mai des în neplată: lipsa datelor este o informație.

$$\tilde{x}_{ij} = \begin{cases} x_{ij} & \text{observat} \\ \mathrm{med}_{\text{train}}(x_j) & \text{lipsă} \end{cases}, \qquad m_{ij} = \mathbb{1}\{x_{ij} \text{ lipsă}\}$$

- Raportul datorii/venit lipsește pentru 21,3 % dintre creditele HMEQ.
- Rata de neplată este 62,0 % cînd lipsește, față de 8,6 % cînd este declarat.
- Fiecare dintre cele 10 intrări numerice primește o completare cu mediana și un indicator m_{ij}
  - Cu 8 variabile binare pentru scop și ocupație, rețeaua vede $d = 28$ intrări.

- **În practică:** nu eliminați rîndurile cu valori lipsă din datele de credit: ați elimina solicitanții cei mai riscanți.

(Baesens, Rösch & Scheule, 2016)

Exercițiu practic: notebook §2

Cuprins

- Deep learning pentru afaceri
- Modelele liniare sînt rețele cu un singur neuron
- Perceptronul
- Perceptronul multistrat
- Studiu de caz: scoringul de credit pe credite reale
- Sinteză

De la neuronul McCulloch–Pitts la perceptron

- **Pe scurt:** McCulloch și Pitts au scris neuronul ca unitate cu prag; Rosenblatt a făcut ca ponderile lui să învețe din exemple.

$$\hat{y} = \text{step}(\mathbf{w}^\top \mathbf{x} + b) = \begin{cases} +1 & \mathbf{w}^\top \mathbf{x} + b > 0 \\ -1 & \text{altfel} \end{cases}$$

- **unde**
  - $\mathbf{x}$ intrările (dendritele), $\mathbf{w}$ ponderile sinaptice, $-b$ pragul de activare
  - $\hat{y}$ clasa (axonul)

- 1943: calculul logic al neuronilor cu prag (McCulloch & Pitts, 1943).
- 1958: perceptronul învață $\mathbf{w}$ din exemple etichetate (Rosenblatt, 1958).
- 1969: un singur strat nu poate reprezenta XOR (Minsky & Papert, 1969).
- 1986: propagarea înapoi a erorii antrenează mai multe straturi (Rumelhart, Hinton & Williams, 1986).

- **În practică:** perceptronul este un clasificator liniar: frontiera lui $\mathbf{w}^\top \mathbf{x} + b = 0$ este un hiperplan.

(Bishop & Bishop, 2024, ch. 5)

Perceptronul își actualizează ponderile numai pe exemplele clasificate greșit

- **Pe scurt:** cînd un exemplu este clasificat greșit, ponderile se mută spre el; altfel nu se schimbă nimic.

dacă $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \le 0$: $\quad \mathbf{w} \leftarrow \mathbf{w} + \eta\, y_i \mathbf{x}_i, \quad b \leftarrow b + \eta\, y_i$

$$\mathcal{L}_P(\mathbf{w}, b) = -\sum_{i \in \mathcal{M}} y_i(\mathbf{w}^\top \mathbf{x}_i + b)$$

- **unde**
  - $y_i \in \{-1, +1\}$, $\eta > 0$ rata de învățare, $\mathcal{M}$ mulțimea exemplelor clasificate greșit
  - $\mathcal{L}_P$ criteriul perceptronului

- Actualizarea este un pas de coborîre stohastică pe gradient (SGD) pe $\mathcal{L}_P$, pentru un exemplu clasificat greșit.
- Spre deosebire de regresia logistică, exemplele corecte nu dau gradient: perceptronul ignoră cît de sigur este.
- Nu rezultă nicio probabilitate, ci doar o clasă.

- **În practică:** pentru credit, o clasă fără probabilitate nu poate fi evaluată și nici comparată cu un cost.

(Rosenblatt, 1958; Bishop & Bishop, 2024, ch. 5)

Dacă clasele pot fi separate, perceptronul se oprește după un număr finit de greșeli

- **Pe scurt:** o margine largă între clase înseamnă că sînt necesare puține corecții.

$$\exists \mathbf{w}^*,\ \|\mathbf{w}^*\| = 1 :\ y_i\, {\mathbf{w}^*}^\top \mathbf{x}_i \geq \gamma > 0\ \ \forall i, \quad \|\mathbf{x}_i\| \leq R \implies \#\text{greșeli} \leq \left(\frac{R}{\gamma}\right)^2$$

- **unde**
  - γ marginea celui mai bun hiperplan separator, R raza datelor, termenul liber este inclus în $\mathbf{x}$ (o constantă 1)
- Teorema lui Novikoff (1962): limita nu depinde nici de numărul de exemple, nici de dimensiune.
- Ideea demonstrației: $\mathbf{w}^\top \mathbf{w}^*$ crește cu cel puțin γ la fiecare greșeală, iar $\|\mathbf{w}\|^2$ cu cel mult R^2 (anexa).
- Dacă nu există un hiperplan separator, teorema nu spune nimic, iar ponderile se rotesc la nesfîrșit.

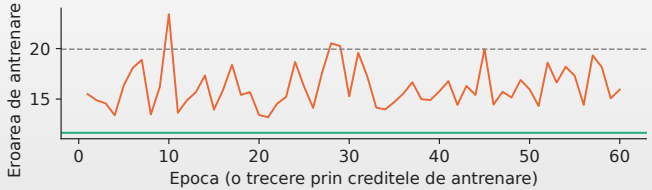
- **În practică:** datele reale de neplată nu sînt niciodată separabile: solicitanți cu același dosar pot rambursa sau nu.

(Novikoff, 1962)

Exercițiu practic: notebook §3

Perceptronul nu converge pe date de credit neseparabile

- **Pe scurt:** clasele se suprapun, așa că fiecare epocă corectează unele credite și le strică pe altele.



Sursa: Baesens, Roesch și Scheule (2016), datele HMEQ. Date: 3 576 de credite de antrenare.

- Eroarea după fiecare dintre cele 60 de epoci: 13,2–23,4 %, față de 11,7 % pentru regresia logistică și 19,9 % pentru „mereu fără neplată”.

- **În practică:** o pierdere netedă și o ieșire probabilistă rezolvă problema: rețelele folosesc sigmoide, nu trepte.

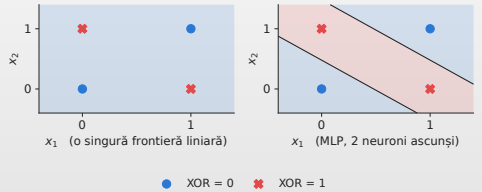
(calcul propriu)

NN_L1_perceptron_errors

Problema XOR și stratul ascuns

- **Pe scurt:** XOR este adevărat cînd exact o intrare este activă; cele două puncte pozitive stau pe o diagonală.

$$h_k = \tanh(\mathbf{w}_k^\top \mathbf{x} + b_k), \; k = 1, 2; \qquad p = \sigma(v_1 h_1 + v_2 h_2 + c)$$



Sursa: tabelul funcției logice XOR (fără date). Date: cele patru intrări XOR.

- Acuratețea pe cele patru puncte: 50 % pentru orice frontieră liniară, 100 % cu doi neuroni ascunși.
- **În practică:** interacțiunile de tipul „datorii mari și istoric scurt” seamănă cu XOR.

(Minsky & Papert, 1969) **DATE SIMULATE (doar pentru ilustrare)** NN_L1_xor

Cuprins

- Deep learning pentru afaceri
- Modelele liniare sînt rețele cu un singur neuron
- Perceptronul
- Perceptronul multistrat
- Studiu de caz: scoringul de credit pe credite reale
- Sinteză

Un MLP alternează transformări afine și activări neliniare

- **Pe scurt:** fiecare strat calculează variabile noi din cele anterioare; ultimul strat este o regresie logistică pe ele.

$$\mathbf{h}^{(0)} = \mathbf{x}, \quad \mathbf{h}^{(l)} = a\big(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\big),\ l = 1, \dots, L-1, \quad p(\mathbf{x}) = \sigma\big(\mathbf{w}^{(L)\top}\mathbf{h}^{(L-1)} + b^{(L)}\big)$$

- **unde**
  - $\mathbf{W}^{(l)} \in \mathbb{R}^{H_l \times H_{l-1}}$, $\mathbf{b}^{(l)}$ ponderile și termenii liberi ai stratului l, de lățime H_l ($H_0 = d$), a funcția de activare, element cu element
- Fără a, straturile se reduc la unul singur: $\mathbf{W}^{(2)}\mathbf{W}^{(1)}\mathbf{x}$ este tot liniar (anexa).
- Rețeaua noastră de credit: $d = 28$ intrări, două straturi ascunse a cîte 64 de unități liniare rectificate (ReLU), o ieșire sigmoidă („MLP 64-64”).

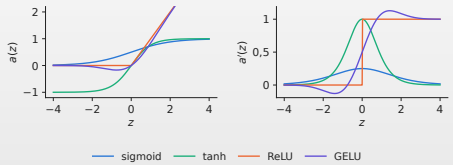
- **În practică:** lățimea și adîncimea sînt alegeri de proiectare
  - Secțiunea 5 le alege pe setul de validare.

(Bishop & Bishop, 2024, ch. 6)

Exercițiu practic: notebook §4

Funcțiile de activare și proprietățile lor

$$\sigma(z) = \frac{1}{1+e^{-z}}, \quad \tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}, \quad \mathrm{ReLU}(z) = \max(0, z), \quad \mathrm{GELU}(z) = z\,\Phi(z)$$



Sursa: funcții analitice (fără date). Date: funcții reprezentate pe [-4, 4].

- Sigmoida și tanh se saturează ($\sigma' \leq 0{,}25$)
  - Unitatea liniară rectificată (ReLU) transmite gradientul 1 pentru $z > 0$
  - Unitatea liniară cu eroare gaussiană (GELU, Φ funcția de repartiție normală) apare în arhitecturile transformer.

- **În practică:** folosiți ReLU în straturile ascunse
  - Activarea de ieșire decurge din variabila țintă.

(Bishop & Bishop, 2024, ch. 6) **DATE SIMULATE (doar pentru ilustrare)** NN_L1_activations

Stratul de ieșire și pierderea decurg din tipul variabilei țintă

- **Pe scurt:** fiecare pereche (activare de ieșire, pierdere) este un estimator de verosimilitate maximă pentru o distribuție a lui y.

Ținta	Ieșirea	Pierderea	Exemplu
număr real	$\hat{y} = z$	MSE: $\frac{1}{n}\sum(y_i - \hat{y}_i)^2$	valoarea locuinței, un preț
binară	$p = \sigma(z)$	entropia încrucișată: $-\frac{1}{n}\sum[y \log p + (1-y)\log(1-p)]$	neplată, fraudă
K clase	$p_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$	$-\frac{1}{n}\sum_i \log p_{i,y_i}$	clasa de rating, sentimentul știrilor
cuantila τ	$\hat{q} = z$	pinball: $\frac{1}{n}\sum \rho_\tau(y_i - \hat{q}_i)$	valoarea la risc (VaR), cursul 2

- $\rho_\tau(u) = u(\tau - \mathbb{1}\{u < 0\})$; funcția softmax cu $K = 2$ se reduce la sigmoidă.

- **În practică:** un model PD are nevoie de o ieșire sigmoidă și de entropia încrucișată, ca scorurile să fie probabilități.

(Bishop & Bishop, 2024, ch. 5–6)

Un MLP mic are deja de sute de ori mai mulți parametri decît scorul logistic

- **Pe scurt:** fiecare conexiune este o pondere, iar fiecare neuron are un termen liber.

$$|\theta| = \sum_{l=1}^{L} (H_{l-1} + 1)\, H_l, \qquad H_0 = d,\ H_L = 1$$

- Regresia logistică pe HMEQ: $(d+1) \cdot 1 = 29$ de parametri.
- MLP 64-64: $(d+1) \cdot 64 + 65 \cdot 64 + 65 \cdot 1 = 6\,081$ de parametri pentru 3 576 de credite de antrenare.
- Mai mulți parametri decît neplăți în datele de antrenare: fără oprire pe baza setului de validare, rețeaua le-ar memora (cursul 3).
- Aici fiecare MLP este antrenat cu Adam (cursul 2) și păstrează ponderile din epoca cu cea mai bună pierdere de validare.

- **În practică:** numărați parametrii înainte de antrenare: aflați astfel de cîte date și de cîtă regularizare veți avea nevoie.

Exercițiu practic: notebook §4

Un singur strat ascuns poate aproxima orice funcție continuă

- **Pe scurt:** suficiente „cocoașe" sigmoide, adunate, pot desena orice curbă netedă pe un domeniu mărginit.

$$\forall g \in C(K),\ \forall \varepsilon > 0\ \ \exists H,\ v_k, \mathbf{w}_k, b_k :\quad \sup_{\mathbf{x}\in K} \left| g(\mathbf{x}) - \sum_{k=1}^{H} v_k\, \sigma(\mathbf{w}_k^\top \mathbf{x} + b_k) \right| < \varepsilon$$

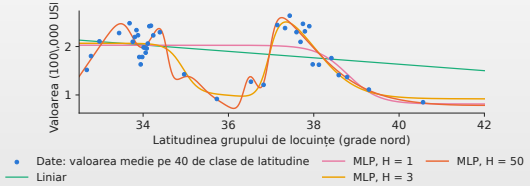
- **unde**
  - $K \subset \mathbb{R}^d$ compact, g continuă, σ sigmoidală (Cybenko, 1989) sau nepolinomială (Hornik, 1991)
- Ce spune: familia MLP este suficient de bogată.
- Ce nu spune: cît de mare trebuie să fie H, dacă coborîrea pe gradient găsește ponderile sau dacă ele generalizează.

- **În practică:** teorema justifică încercarea unui MLP
  - Doar testele în afara eșantionului justifică păstrarea lui.

(Cybenko, 1989; Hornik, 1991)

Mai mulți neuroni ascunși urmăresc geografia prețurilor din California

- **Pe scurt:** valorile au vîrfuri lîngă Los Angeles și San Francisco; $\hat{y}(x) = c + \sum_{k=1}^{H} v_k \sigma(w_k x + b_k)$, cu $H = 1, 3, 50$.



Sursa: StatLib, locuințe din California, recensămîntul din 1990 (Pace și Barry, 1997). Date: 19 648 de grupuri de locuințe sub plafonul sursei.

- MSE pe datele păstrate deoparte: liniar 0,896, $H = 1$ 0,855, $H = 3$ 0,757, $H = 50$ 0,731 (fără cele 992 de grupuri de la plafonul de 500 000 USD).

- **În practică:** efectele de localizare sînt neliniare: un termen liniar în latitudine nu le poate surprinde.

(calcul propriu)

NN_L1_uat_california

Adîncimea și lățimea unei rețele

- **Pe scurt:** fiecare strat compune variabilele stratului anterior, astfel încît numărul de bucăți liniare poate crește cu adîncimea.

  ReLU MLP: $f(\mathbf{x})$ este liniară pe porțiuni; numărul de regiuni liniare pînă la $O\left(H^{dL}\right)$

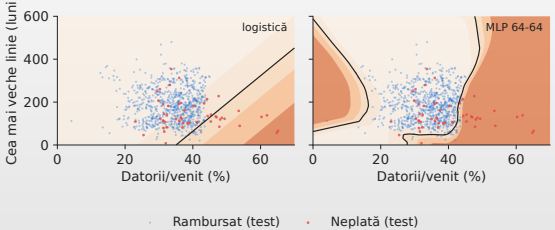
- **unde**
  - H lățimea fiecărui strat, L numărul de straturi ascunse, d dimensiunea intrării (limita lui Montúfar et al., 2014)
- Pentru unele funcții, o rețea puțin adîncă are nevoie de exponențial mai mulți neuroni decît una adîncă.
- Pe datele tabelare de credit, două sau trei straturi ascunse sînt de obicei suficiente
  - Adîncimea contează la imagini, text și secvențe.

- **În practică:** începeți cu 2 straturi ascunse pe date tabelare
  - Adăugați straturi doar dacă pierderea de validare continuă să scadă.

(Bishop & Bishop, 2024, ch. 6)

Frontiera de decizie neliniară a unui MLP cu două intrări

- **Pe scurt:** PD prezisă pentru două variabile HMEQ: benzi drepte pentru modelul logistic, regiuni curbe pentru MLP.



Sursa: Baesens, Roesch și Scheule (2016), datele HMEQ. Date: 2 708 de credite de antrenare și 871 de testare cu ambele variabile.

- Aria de sub curbă (AUC) a caracteristicii de operare a receptorului (ROC) = $\Pr(p_{\text{neplată}} > p_{\text{bun}})$
  - Cu aceste două intrări: logistic 0,744, MLP 64-64 0,736.

- **În practică:** o frontieră flexibilă merită doar dacă intrările conțin semnalul.

(calcul propriu)

NN_L1_boundary_hmeq

Cuprins

- Deep learning pentru afaceri
- Modelele liniare sînt rețele cu un singur neuron
- Perceptronul
- Perceptronul multistrat
- Studiu de caz: scoringul de credit pe credite reale
- Sinteză

Protocolul experimental: aceeași împărțire și aceleași intrări pentru trei modele

- **Pe scurt:** o comparație corectă schimbă doar modelul; datele, împărțirea și metrica sînt fixate înainte de orice antrenare.

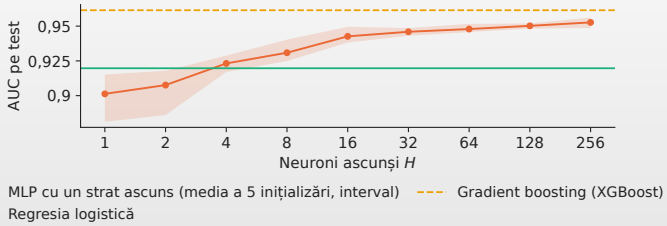
- Regresia logistică: verosimilitate maximă, fără penalizare pe HMEQ
  - O penalizare ridge ușoară ($C = 1$) pe German credit.
- MLP: straturi ReLU, Adam cu rata de învățare 10^{-3}, loturi de 128 (64 pe German credit), epoca cea mai bună pe validare, 5 inițializări.
- Extreme gradient boosting (XGBoost): adîncime 4, rata de învățare 0,03, oprire timpurie pe validare (Chen & Guestrin, 2016).
- HMEQ: o împărțire stratificată 60/20/20
  - AUC, curba ROC, calibrarea, intervalul bootstrap al cîștigului de AUC.
- German credit: validare încrucișată cu 5 pliuri, repetată de 5 ori (25 de pliuri de test), test t corectat.
- Valoarea inițială a generatorului 20261108 (+ indicele rulării)
  - Antrenare pe procesorul central (CPU), sub un minut pe familie de modele.

- **În practică:** scrieți protocolul înainte de antrenare: este primul lucru pe care îl cere cel care validează modelul.

Exercițiu practic: notebook §5

Pe HMEQ, un strat ascuns mai lat crește AUC pe test peste scorul logistic

- **Pe scurt:** un strat ascuns de lățime H; fiecare punct este media a 5 inițializări, banda arată intervalul lor.



Sursa: Baesens, Roesch și Scheule (2016), datele HMEQ. Date: 3 576 de credite de antrenare 1 192 de validare 1 192 de testare.

- AUC 0,901 la $H = 1$, 0,953 la $H = 256$
  - Logistic 0,920, XGBoost 0,961
  - Cîștigul se aplatizează după circa 16 neuroni.

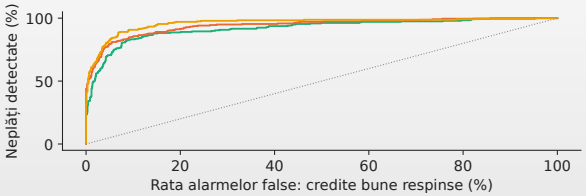
- **În practică:** aici interacțiunile contează: rețeaua depășește scorul liniar, iar boosting le depășește pe amîndouă.

(calcul propriu)

NN_L1_width_auc

La 10 % alarme false, MLP detectează mai multe neplăți decît scorul logistic

- **Pe scurt:** curba ROC reprezintă Pr($p \geq c$ | neplată) în funcție de Pr($p \geq c$ | bun), pentru fiecare prag c.



Sursa: Baesens, Roesch și Scheule (2016), datele HMEQ. Date: 1 192 de credite de testare (238 de neplăți).

- Neplăți detectate cînd respingem 10 % dintre creditele bune: logistic 83,2 %, MLP 85,3 %, XGBoost 90,8 % (238 de neplăți în test).

- **În practică:** citiți curba la rata de respingere acceptată de afacere, nu doar aria de sub ea.

(calcul propriu) NN_L1_roc_hmeq

Cîștigul MLP pe HMEQ depășește zgomotul inițializării și al setului de testare

- **Pe scurt:** o singură rulare nu dovedește nimic: repetăm antrenarea și reeșantionăm setul de testare.

$$\Delta = \mathrm{AUC}_{\mathrm{MLP}} - \mathrm{AUC}_{\mathrm{logit}}, \qquad \Delta_b^* = \mathrm{AUC}_{\mathrm{MLP}}(\mathcal{T}_b^*) - \mathrm{AUC}_{\mathrm{logit}}(\mathcal{T}_b^*),\ b = 1, \ldots, 2000$$

- **unde**
  - $\mathcal{T}_b^*$ o reeșantionare bootstrap a celor 1 192 de credite de testare, aceeași pentru ambele modele (perechi)
- Inițializări: MLP 64-64 are în medie AUC pe test 0,945 pe 5 inițializări (abaterea standard 0,004; interval 0,942–0,951).
- Zgomotul setului de testare: $\Delta = 0{,}023$, intervalul bootstrap de 95 % [0,008; 0,037]: zero este exclus.
- Variația între inițializări a modelului logistic este zero: pierderea lui este convexă.

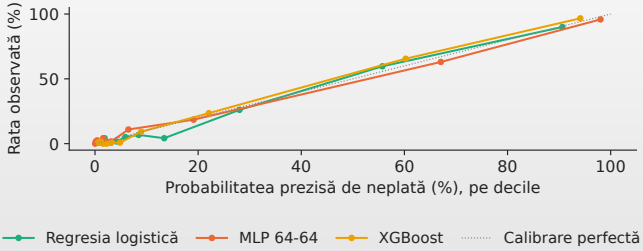
- **În practică:** raportați cîștigul unui model cu un interval
  - Un cîștig mai mic decît variația dintre inițializări nu este un cîștig.

(calcul propriu)

Exercițiu practic: notebook §5

Scorurile sînt apropiate de probabilități calibrate

- **Pe scurt:** în fiecare decilă a PD prezise, rata observată de neplată ar trebui să fie egală cu cea prezisă.



Sursa: Baesens, Roesch și Scheule (2016), datele HMEQ. Date: 1 192 de credite de testare.

- Scorul Brier $\frac{1}{n}\sum(p_i - y_i)^2$: logistic 0,0742, MLP 0,0625, XGBoost 0,0549
  - Rata constantă 0,1598.

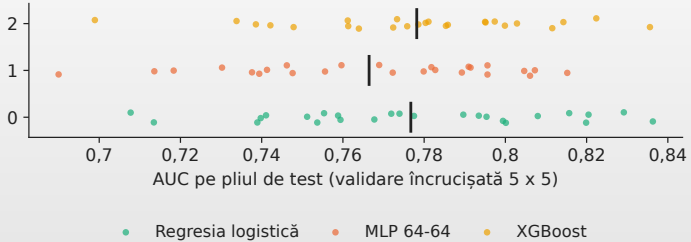
- **În practică:** provizioanele și prețurile folosesc PD ca atare: verificați calibrarea, nu doar ordonarea.

(calcul propriu)

NN_L1_calibration

Pe 1 000 de solicitanți germani, MLP nu depășește regresia logistică

- **Pe scurt:** fiecare punct este AUC pe un pliu de test; bara neagră este media celor 25 de pliuri.



Sursa: UCI Machine Learning Repository, Statlog German credit. Date: 1 000 de solicitanți, 25 de pliuri de test.

- AUC medie: logistic 0,777, MLP 0,766, XGBoost 0,778
  - MLP minus logistic: $t = -0{,}93$, $p = 0{,}362$ (slide-ul următor).

- **În practică:** cu 1 000 de rînduri, păstrați scorecard-ul logistic: rețeaua adaugă risc, nu acuratețe.

(calcul propriu)

NN_L1_german_cv

De ce testul t obișnuit exagerează diferențele în validarea încrucișată

- **Pe scurt:** seturile de antrenare ale pliurilor diferite se suprapun, deci scorurile pliurilor sînt corelate pozitiv.

$$t = \frac{\bar{d}}{\sqrt{\left(\frac{1}{J} + \frac{n_{\text{test}}}{n_{\text{train}}}\right) s_d^2}} \sim t_{J-1}, \qquad d_j = \text{AUC}_j^{\text{MLP}} - \text{AUC}_j^{\text{logit}}$$

- **unde**
  - $J = 25$ de pliuri, s_d^2 varianța de selecție a lui d_j, $n_{\text{test}}/n_{\text{train}} = 1/4$ pentru 5 pliuri
- Termenul $n_{\text{test}}/n_{\text{train}}$ este corecția Nadeau–Bengio: fără ea, t este mult prea mare.
- MLP față de logistic: $t = -0{,}93$ ($p = 0{,}362$)
  - XGBoost față de logistic: $t = 0{,}12$ ($p = 0{,}909$).

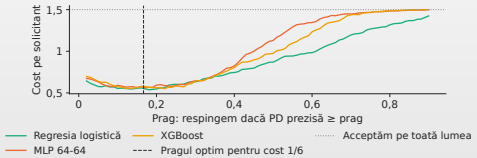
- **În practică:** schimbarea unui model într-o bancă are nevoie de dovezi care rezistă acestei corecții.

(Nadeau & Bengio, 2003)

Exercițiu practic: notebook §6

Pragul de 1/6, sensibil la costuri, cînd o neplată ratată costă de cinci ori cît un client bun respins

$$\text{respingem dacă } \; c_{\mathrm{FN}}\, p \;\geq\; c_{\mathrm{FP}}(1-p) \iff p \;\geq\; \frac{c_{\mathrm{FP}}}{c_{\mathrm{FP}}+c_{\mathrm{FN}}} = \frac{1}{6}$$



Sursa: UCI Machine Learning Repository, Statlog German credit. Date: 1 000 de solicitanți, predicții în afara pliului.

- Costul pe solicitant (matricea de cost UCI 5:1): la 1/6 logistic 0,554, MLP 0,568
  - La 1/2 0,870 și 1,130
  - Acceptarea tuturor 1,50.

- **În practică:** aici alegerea pragului contează mai mult decît alegerea modelului.

(calcul propriu)

NN_L1_cost_threshold

Implementarea minimală a unui MLP în PyTorch

```
import torch, torch.nn as nn
model = nn.Sequential(
    nn.Linear(d, 64), nn.ReLU(),
    nn.Linear(64, 64), nn.ReLU(),
    nn.Linear(64, 1))            # logit z
loss_fn = nn.BCEWithLogitsLoss()  # sigmoid + cross-entropy
opt = torch.optim.Adam(model.parameters(), lr=1e-3)
for epoch in range(300):
    for xb, yb in train_batches:
        opt.zero_grad()
        loss = loss_fn(model(xb), yb)
        loss.backward()           # backpropagation
        opt.step()                # Adam update
    # evaluate on validation, keep the best weights
p = torch.sigmoid(model(x_test))   # predicted PD
```

- `BCEWithLogitsLoss` combină sigmoida și entropia încrucișată într-o singură funcție, stabilă numeric.
- `backward()` calculează toate gradienții prin propagare înapoi
  - `step()` aplică optimizatorul: ambele sînt tema cursului 2.
- Notebook-ul adaugă pregătirea datelor, oprirea pe baza validării și valorile inițiale ale generatorului.

- **În practică:** aceleași cincisprezece rînduri antrenează toate rețelele tabelare din curs
  - Se schimbă doar datele și straturile.

(Paszke et al., 2019)

Exerciții practic: notebook §7

Cînd folosim un MLP pentru date tabelare de afaceri

Ce arată acest curs

- HMEQ, 3 576 de credite: AUC MLP 0,945 față de logistic 0,920
  - XGBoost 0,961.
- German credit, 1 000 de solicitanți: nicio diferență semnificativă ($p = 0{,}362$).
- În ambele cazuri: pragul din matricea de cost contează mai mult decît modelul.

Regulă practică

- Puține rînduri, variabile construite, explicație obligatorie: scorecard logistic.
- Mii de rînduri, interacțiuni: testați boosting și un MLP față de el.
- Text brut, imagini, secvențe, grafuri sau o rețea folosită pentru mai multe sarcini: deep learning.

- **În practică:** alegeți cel mai simplu model al cărui cîștig în afara eșantionului rezistă la inițializări, reeșantionare și matricea de cost.

Cuprins

- Deep learning pentru afaceri
- Modelele liniare sînt rețele cu un singur neuron
- Perceptronul
- Perceptronul multistrat
- Studiu de caz: scoringul de credit pe credite reale
- Sinteză

Idei principale

- Regresia liniară și regresia logistică sînt rețele cu un neuron
  - Pierderile lor sînt verosimilități, gradienții lor sînt eroarea înmulțită cu intrarea.
- Perceptronul converge doar pe date separabile
  - Datele reale de credit nu sînt separabile, iar XOR arată limita unui singur strat.
- Un MLP suprapune transformări afine și activări
  - Un strat ascuns aproximează orice funcție continuă, adîncimea îl face eficient.
- Pe credite reale: MLP depășește regresia logistică pe HMEQ (AUC 0,945 față de 0,920), dar nu pe German credit
  - Boosting este ștacheta.
- Evaluați cu mai multe inițializări, intervale, calibrare și costul erorilor, niciodată cu o singură cifră de acuratețe.

Ecuațiile principale

$$p = \sigma(\mathbf{w}^\top \mathbf{x} + b)$$

$$\mathcal{L} = -\tfrac{1}{n} \sum [y \log p + (1-y) \log(1-p)]$$

$$\nabla_{\mathbf{w}} \mathcal{L} = \tfrac{1}{n} \sum (p_i - y_i) \mathbf{x}_i$$

$$\mathbf{w} \leftarrow \mathbf{w} + \eta\, y_i \mathbf{x}_i \quad \text{dacă e greșit}$$

$$\mathbf{h}^{(l)} = a\big(\mathbf{W}^{(l)} \mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\big)$$

$$|\boldsymbol{\theta}| = \sum\nolimits_l (H_{l-1} + 1) H_l$$

$$\#\text{greșeli} \leq (R/\gamma)^2$$

$$\text{respingem dacă } p \geq \frac{c_{\mathrm{FP}}}{c_{\mathrm{FP}} + c_{\mathrm{FN}}}$$

Cercetările grupului nostru: deep learning pentru decizii de afaceri

- **Detectarea explicabilă a fraudei** (Diaconu & Pele, 2026): a comparat XGBoost și CatBoost cu o rețea neuronală pe grafuri (GNN), eșantionare și agregare pe graf (GraphSAGE), pe datele de fraudă Institute of Electrical and Electronics Engineers, Computational Intelligence Society (IEEE-CIS)
  - Boosting a avut AUC mai mare, modelul pe graf a surprins tiparele relaționale ale fraudei
  - Un cadru multinivel de inteligență artificială explicabilă (XAI) combină explicațiile aditive Shapley (SHAP) cu eliminarea de noduri și muchii — replicat în studiul de caz 1 (site-ul cursului).
- **Criptoactivele „zombie”** (Będowska-Sójka, Wójcik & Pele, 2026): avertizare timpurie, prin învățare automată, pentru criptoactivele care devin „zombie” — cursul 3.
- **Măsuri de risc obținute din modele lingvistice** (Pele et al., 2026): valoarea la risc (VaR) și pierderea așteptată în coada distribuției (ES), obținute prin interogarea modelelor lingvistice mari (LLM) — cursul 12.
- **Modele fundație pentru riscul extrem** (Jheng, Tak, Găman, Mazurencu-Marinescu-Pele & Pele, 2026): VaR și ES fără antrenare (zero-shot), cu calibrare conformală, pe piețele din Europa Centrală și de Est — cursul 13.
- **Anomaliile pieței văzute ca imagini** (Jheng, Pele, Tak & Găman, 2026; premiul pentru cea mai bună lucrare la International Conference on Business Excellence (ICBE) 2026): indicatorii pieței, desenați ca imagini, sînt clasificați de un model vizual-lingvistic (VLM), cu justificări în limbaj natural, pe S&P 500 — cursul 14.

Rezultatele sînt cele declarate în lucrări; cifrele de pe celelalte slide-uri provin din calculele noastre.

Explorați vizual

- **TensorFlow Playground**
  - Alegeți setul de date în formă de cerc și păstrați doar intrările x_1, x_2. Fără strat ascuns, frontiera rămîne o dreaptă, ca la regresia logistică; adăugați un strat ascuns cu patru neuroni și frontiera se închide în jurul cercului.
  - Păstrați stratul ascuns și setați activarea la Linear: frontiera redevine dreaptă, pentru că o compunere de funcții liniare este liniară.
- **3Blue1Brown, capitolul 1**: ce calculează ponderile, termenii liberi și straturile unei rețele, pe cifre scrise de mînă.

Lecturi și resurse suplimentare

- **Bishop & Bishop (2024)**, capitolele 4–6: modele liniare, clasificare, rețele adînci: bishopbook.com.
- **Dive into Deep Learning**, capitolele 3–5, cu cod PyTorch: d2l.ai.
- **Goodfellow, Bengio & Courville**, capitolul 6: deeplearningbook.org.
- **Tutorialele PyTorch**: pytorch.org/tutorials.
- **Acest curs**: cod, date, notebook-uri, Quantlets — github.com/danpele/NEURAL_BIZ.
- **Scoringul de credit în detaliu**: Baesens, Rösch & Scheule (2016).

Test 1: regresia logistică ca neuron

- Ce pierdere face ca o rețea cu un neuron și ieșire sigmoidă să fie echivalentă cu regresia logistică?

Răspuns

- Entropia încrucișată, adică minus log-verosimilitatea Bernoulli
  - Gradientul ei este $\frac{1}{n}\sum(p_i - y_i)\mathbf{x}_i$.

Test 2: perceptronul pe date reale

- De ce eroarea de antrenare a perceptronului pe HMEQ sare între 13,2 % și 23,4 %?

Răspuns

- Clasele nu sînt separabile liniar, deci limita lui Novikoff nu se aplică: fiecare corecție a unui credit le strică pe altele.

Test 3: numărarea parametrilor

- Cîți parametri are un MLP pe intrările HMEQ ($d = 28$) cu un strat ascuns de 32 de neuroni și o ieșire?

Răspuns

- $(d + 1) \cdot 32 + (32 + 1) \cdot 1 = 961$.

Test 4: pragul optim pentru cost

- O neplată ratată costă 5, un client bun respins costă 1.
  - Peste ce PD ar trebui banca să respingă?

Răspuns

- $p \geq c_{\mathrm{FP}}/(c_{\mathrm{FP}} + c_{\mathrm{FN}}) = 1/6$
  - Pe German credit, acest prag reduce costul pe solicitant al modelului logistic de la 0,870 (pragul $1/2$) la 0,554.

Test 5: cînd merită un MLP?

- Pe German credit, MLP are AUC medie 0,766, iar regresia logistică 0,777.
  - Ce recomandați?

Răspuns

- Păstrați scorecard-ul logistic: diferența nu este semnificativă ($p = 0{,}362$), iar modelul mai simplu este mai ușor de explicat și de validat.

Bibliografie (1/2)

- Bishop, C. M. & Bishop, H. (2024). *Deep Learning: Foundations and Concepts*. Springer. doi:10.1007/978-3-031-45468-4
- Baesens, B., Rösch, D. & Scheule, H. (2016). *Credit Risk Analytics*. Wiley. doi:10.1002/9781119449560
- Będowska-Sójka, B., Wójcik, P. & Pele, D. T. (2026). Early warning systems for cryptocurrency markets: Predicting 'zombie' assets using machine learning. *The North American Journal of Economics and Finance*, 81, 102543. doi:10.1016/j.najef.2025.102543
- Chen, T. & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of KDD 2016*, 785–794. doi:10.1145/2939672.2939785
- Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4), 303–314. doi:10.1007/BF02551274
- Diaconu, D. & Pele, D. T. (2026). Explainable AI in financial fraud detection: Evidence from the IEEE-CIS fraud dataset. *Proceedings of the International Conference on Business Excellence*, 20(1), 118–132. doi:10.2478/picbe-2026-0015
- Grinsztajn, L., Oyallon, E. & Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? *Advances in Neural Information Processing Systems*, 35. arXiv:2207.08815
- Hofmann, H. (1994). Statlog (German Credit Data). UCI Machine Learning Repository. doi:10.24432/C5NC77
- Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2), 251–257. doi:10.1016/0893-6080(91)90009-T
- Jheng, S.-L., Pele, D. T., Tak, R. & Găman, Ș. (2026). A multimodal vision-language framework for financial anomaly detection. *Proceedings of the International Conference on Business Excellence*, 20(1), 191–201. doi:10.2478/picbe-2026-0020
- Jheng, S.-L., Tak, R., Găman, Ș., Mazurencu-Marinescu-Pele, M. & Pele, D. T. (2026). Can foundation models manage risk? Zero-shot VaR and ES forecasting with conformal calibration in CEE markets. *Economic Computation and Economic Cybernetics Studies and Research*, 60(2), 267–290. doi:10.24818/18423264/60.2.26.14
- LeCun, Y., Bengio, Y. & Hinton, G. (2015). Deep learning. *Nature*, 521, 436–444. doi:10.1038/nature14539

Bibliografie (2/2)

- Lessmann, S., Baesens, B., Seow, H.-V. & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, 247(1), 124–136. doi:10.1016/j.ejor.2015.05.030
- McCulloch, W. S. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4), 115–133. doi:10.1007/BF02478259
- Minsky, M. & Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. MIT Press.
- Nadeau, C. & Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, 52(3), 239–281. doi:10.1023/A:1024068626366
- Novikoff, A. B. J. (1962). On convergence proofs on perceptrons. *Proceedings of the Symposium on the Mathematical Theory of Automata*, 12, 615–622. Polytechnic Institute of Brooklyn.
- Pace, R. K. & Barry, R. (1997). Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3), 291–297. doi:10.1016/S0167-7152(96)00140-X
- Montúfar, G., Pascanu, R., Cho, K. & Bengio, Y. (2014). On the number of linear regions of deep neural networks. *Advances in Neural Information Processing Systems*, 27. arXiv:1402.1869
- Paszke, A. et al. (2019). PyTorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32. arXiv:1912.01703
- Pele, D. T., Bolovăneanu, V., Lin, M.-B., Ren, R., Ginavar, A. T., Spilak, B., Andrei, A.-V., Toma, F.-M., Lessmann, S. & Härdle, W. K. (2026). In the beginning was the Word: LLM-VaR and LLM-ES. *Expert Systems with Applications*, 295, 128676. doi:10.1016/j.eswa.2025.128676
- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386–408. doi:10.1037/h0042519
- Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536. doi:10.1038/323533a0

Anexă: gradientul pierderii logistice

$$\sigma'(z) = \sigma(z)\big(1-\sigma(z)\big), \qquad \ell_i = -y_i \log p_i - (1-y_i)\log(1-p_i), \quad p_i = \sigma(z_i)$$

$$\frac{\partial \ell_i}{\partial z_i} = -\frac{y_i}{p_i} p_i(1-p_i) + \frac{1-y_i}{1-p_i} p_i(1-p_i) = p_i - y_i$$

$$\frac{\partial \ell_i}{\partial \mathbf{w}} = \frac{\partial \ell_i}{\partial z_i}\,\frac{\partial z_i}{\partial \mathbf{w}} = (p_i - y_i)\,\mathbf{x}_i, \qquad \frac{\partial^2 \mathcal{L}}{\partial \mathbf{w}\,\partial \mathbf{w}^\top} = \frac{1}{n}\sum_i p_i(1-p_i)\mathbf{x}_i\mathbf{x}_i^\top \succeq 0$$

- Hessiana este pozitiv semidefinită: pierderea este convexă, deci coborîrea pe gradient atinge minimul global.
- Aceeași regulă de derivare a funcțiilor compuse, aplicată strat cu strat, este propagarea înapoi a erorii (cursul 2).

Anexă: demonstrația teoremei de convergență a perceptronului

- Pornim de la $\mathbf{w}_0 = \mathbf{0}$ și notăm cu $\mathbf{w}_k$ ponderile după a k-a greșeală, pe exemplul $(\mathbf{x}, y)$: $\mathbf{w}_k = \mathbf{w}_{k-1} + y\mathbf{x}$ ($\eta = 1$).
- Progresul: $\mathbf{w}_k^\top \mathbf{w}^* = \mathbf{w}_{k-1}^\top \mathbf{w}^* + y\mathbf{x}^\top \mathbf{w}^* \geq \mathbf{w}_{k-1}^\top \mathbf{w}^* + \gamma$, deci $\mathbf{w}_k^\top \mathbf{w}^* \geq k\gamma$.
- Creșterea: $\|\mathbf{w}_k\|^2 = \|\mathbf{w}_{k-1}\|^2 + 2y\mathbf{w}_{k-1}^\top \mathbf{x} + \|\mathbf{x}\|^2 \leq \|\mathbf{w}_{k-1}\|^2 + R^2$, pentru că exemplul era greșit
  - Deci $\|\mathbf{w}_k\|^2 \leq kR^2$.
- Cauchy–Schwarz: $k\gamma \leq \mathbf{w}_k^\top \mathbf{w}^* \leq \|\mathbf{w}_k\| \leq \sqrt{k}\, R$, de unde $k \leq (R/\gamma)^2$.

(Novikoff, 1962)

Anexă: straturile liniare suprapuse se reduc la unul singur

$$\mathbf{W}^{(2)}\left(\mathbf{W}^{(1)}\mathbf{x}+\mathbf{b}^{(1)}\right)+\mathbf{b}^{(2)}=\underbrace{\mathbf{W}^{(2)}\mathbf{W}^{(1)}}_{\mathbf{W}}\mathbf{x}+\underbrace{\mathbf{W}^{(2)}\mathbf{b}^{(1)}+\mathbf{b}^{(2)}}_{\mathbf{b}}$$

- Orice număr de straturi liniare formează o singură transformare liniară: adîncimea fără activare adaugă parametri, nu expresivitate.
- Cu o funcție ReLU între ele, f devine liniară pe porțiuni, cu un număr de porțiuni care crește cu adîncimea.