

Rețele neuronale și deep learning

Cursul 3: Regularizare și generalizare





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 diagnosticați supraînvățarea (overfitting) din curbele de antrenare și validare și să explicați compromisul deplasare-varianță și dubla coborîre.
- Să scrieți penalizarea ponderilor, penalizarea L1, dropout, oprirea timpurie și normalizarea pe loturi cu formulele lor și să spuneți ce controlează fiecare.
- Să proiectați o schemă de validare fără scurgere de informație: setul păstrat deoparte, validarea încrucișată, împărțirea cronologică și reeșantionarea doar în setul de antrenare.
- Să evaluați un detector de fraudă pe date dezechilibrate cu precizia, sensibilitatea, aria de sub curbă (AUC) a caracteristicii de operare a receptorului (ROC), aria de sub curba precizie-sensibilitate (PR-AUC) și costul erorilor.

Abrevieri

AdamW	Adam cu penalizarea ponderilor decuplată
AI	inteligență artificială
AUC	aria de sub curba caracteristică de operare a receptorului
BN	normalizarea pe loturi (batch normalisation)
EUR	euro
GNN	rețea neuronală pe grafuri
GraphSAGE	eșantionare și agregare pe graf (o arhitectură GNN)
IEEE-CIS	Institute of Electrical and Electronics Engineers, Computational Intelligence Society
MAP	maximum a posteriori
MLP	perceptron multistrat
MSE	eroarea pătratică medie
OpenML	platforma Open Machine Learning
PCA	analiza componentelor principale
PR-AUC	aria de sub curba precizie-sensibilitate
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)
ULB	Université libre de Bruxelles
XAI	inteligență artificială explicabilă
XGBoost	extreme gradient boosting (arbori de decizie cu amplificare pe gradient)

Cuprins

- Generalizarea și supraînvățarea
- Penalizarea ponderilor
- Dropout, oprirea timpurie și normalizarea pe loturi
- Schema de validare
- Studiu de caz: detectarea fraudei cu carduri
- Sinteză

Un model este judecat pe solicitanți pe care nu i-a văzut niciodată

- **Pe scurt:** eroarea de antrenare este optimistă prin construcție; diferența de generalizare este ceea ce regularizarea încearcă să reducă.

$$\underbrace{R(\hat{f}) - \hat{R}_n(\hat{f})}_{\text{diferența de generalizare}}, \qquad \hat{f} = \arg\min_{f \in \mathcal{F}} \hat{R}_n(f) + \lambda\,\Omega(f)$$

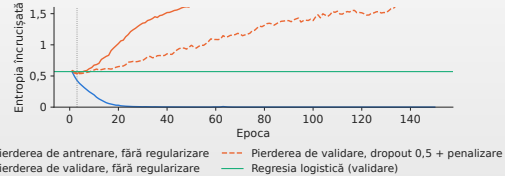
- **unde**
  - R pierderea așteptată pe date noi, $\hat{R}_n$ pierderea de antrenare, $\mathcal{F}$ clasa de modele
  - Ω o penalizare a complexității, $\lambda \geq 0$ ponderea ei
- Regularizarea = orice reduce R cu prețul unei $\hat{R}_n$ mai mari: penalizări, zgomot, oprire timpurie, mai multe date.
- Date: German credit, din colecția Universității California din Irvine (UCI), California housing și 28 924 de plăți cu cardul de la Université libre de Bruxelles (ULB), cu 492 de fraude
  - Modele: perceptroni multistrat (MLP).

- **În practică:** raportați întotdeauna pierderea de validare alături de cea de antrenare.

(Bishop & Bishop, 2024, ch. 9)

Fără regularizare, un MLP mare memorează 600 de solicitanți în cîteva epoci

- **Pe scurt:** MLP 256-256 (78 593 de parametri) pe German credit; linia punctată: epoca cea mai bună pe validare.



Sursa: UCI Machine Learning Repository, Statlog German credit. Date: 600 de solicitanți de antrenare și 200 de validare; MLP 256-256.

- Pierderea de validare 0,536 la epoca 3, 2,187 după 150
  - Cu dropout și penalizare 1,767
  - Logistic 0,570.

- **În practică:** o pierdere de antrenare aproape de zero pe date de credit este un semnal de alarmă.

(calcul propriu)

NN_L3_overfit_german

Eroarea pe test se descompune în deplasare, variață și zgomot

- **Pe scurt:** un model rigid ratează tiparul (deplasare), unul flexibil urmărește zgomotul eșantionului său (variață).

$$\mathbb{E}_{\mathcal{D},\varepsilon}\left[(y-\hat{f}_{\mathcal{D}}(x))^2\right] = \underbrace{\left(f(x)-\mathbb{E}_{\mathcal{D}}\hat{f}_{\mathcal{D}}(x)\right)^2}_{\text{deplasare}^2} + \underbrace{\mathrm{Var}_{\mathcal{D}}\left(\hat{f}_{\mathcal{D}}(x)\right)}_{\text{variață}} + \underbrace{\sigma^2}_{\text{zgomot}}$$

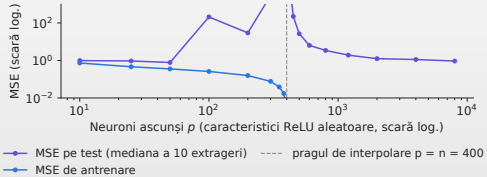
- **unde**
  - $y = f(x) + \varepsilon$, $\mathrm{Var}(\varepsilon) = \sigma^2$, $\hat{f}_{\mathcal{D}}$ modelul estimat pe un eșantion aleator de antrenare $\mathcal{D}$, derivarea în anexă
- Regularizarea acceptă puțină deplasare în schimbul unei reduceri mari a variației
  - Zgomotul σ^2 este o limită pe care niciun model nu o poate depăși.
- Curba clasică în U: eroarea pe test întîi scade, apoi crește cu capacitatea
  - Rețelele foarte mari pot avea o a doua coborîre (slide-ul următor).

- **În practică:** cu puține observații domină variația: preferați modele simple sau regularizare puternică.

(Hastie, Tibshirani & Friedman, 2009, ch. 7)

După pragul de interpolare, mai mulți neuroni ascunși reduc din nou eroarea

- **Pe scurt:** p unități liniare rectificate (ReLU) aleatoare și fixe, ieșire de normă minimă, $n = 400$ de grupuri.



Sursa: StatLib, locuințe din California, recensământul din 1990 (Pace și Barry, 1997). Date: 400 de grupuri de antrenare și 5 000 de testare la fiecare extragere, 10 extrageri.

- Eroarea pătratică medie (MSE) pe test: 0,781 la $p = 50$, $1{,}0 \times 10^6$ la $p = n$, 0,932 la $p = 8\,000$ (mediana a 10 extrageri).

- **În practică:** lîngă $p = n$, un model mai mare poate fi mult mai slab, mult după, mărimea ajută din nou.

(Belkin et al., 2019; Nakkiran et al., 2021)

NN_L3_double_descent

Cuprins

- Generalizarea și supraînvățarea
- Penalizarea ponderilor
- Dropout, oprirea timpurie și normalizarea pe loturi
- Schema de validare
- Studiu de caz: detectarea fraudei cu carduri
- Sinteză

Penalizarea ponderilor ca penalizare L2 și ca distribuție a priori gaussiană a ponderilor

- **Pe scurt:** ponderile mari fac funcția abruptă și sensibilă la zgomot; micșorarea lor o face mai netedă.

$$\tilde{\mathcal{L}}(\mathbf{w}) = \mathcal{L}(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 \;\Rightarrow\; \mathbf{w}_{t+1} = (1-\eta\lambda)\,\mathbf{w}_t - \eta\nabla\mathcal{L}(\mathbf{w}_t); \qquad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \lambda^{-1}\mathbf{I}) \Rightarrow \hat{\mathbf{w}}_{\text{MAP}} = \arg\min \tilde{\mathcal{L}}$$

- **unde**
  - $\lambda > 0$ intensitatea penalizării, $(1-\eta\lambda)$ micșorarea la fiecare pas de coborîre stohastică pe gradient (SGD), estimarea maximum a posteriori (MAP)
- În modelul liniar este regresia ridge: pe direcția proprie k, coeficienții sînt micșorați cu $\frac{d_k}{d_k+\lambda}$ (Hoerl & Kennard, 1970).
- Cu Adam, termenul L2 este rescalat pe fiecare pondere
  - Adam cu penalizare decuplată (AdamW) micșorează toate ponderile la fel (Loshchilov & Hutter, 2019).
- **În practică:** penalizarea ponderilor este primul regularizator de ajustat
  - Căutați λ pe o grilă logaritmică, cu validare.

(Bishop & Bishop, 2024, ch. 9)

Exerciţiu practic: notebook §3

O penalizare L1 aduce ponderile exact la zero

- **Pe scurt:** bila L1 are colțuri pe axe, deci optimul penalizat cade adesea într-unul: rezultă un model rar (sparse).

$$\tilde{\mathcal{L}}(\mathbf{w}) = \mathcal{L}(\mathbf{w}) + \lambda\|\mathbf{w}\|_1, \qquad \text{pasul proximal:} \quad w_j \leftarrow \operatorname{sign}(u_j)\max\big(|u_j| - \eta\lambda, 0\big), \;\; u_j = w_j - \eta\,\partial_j\mathcal{L}$$

- **unde**
  - $\|\mathbf{w}\|_1 = \sum_j |w_j|$, pasul de prag moale aduce ponderile mici exact la zero (lasso, Tibshirani, 1996)
- L1 pe primul strat al unui MLP acționează ca selecție de variabile: o intrare ale cărei ponderi spre stratul ascuns sînt toate zero este oprită.
- Elastic net le combină: $\lambda_1\|\mathbf{w}\|_1 + \frac{\lambda_2}{2}\|\mathbf{w}\|_2^2$.

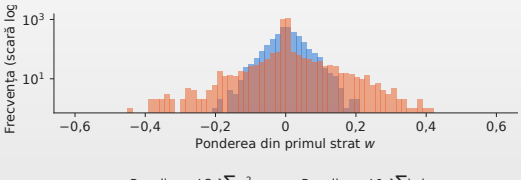
- **În practică:** folosiți L1 cînd documentația modelului cere o listă scurtă de intrări.

(Tibshirani, 1996)

Exercițiu practic: notebook §3

Ponderile primului strat cu penalizare L1 și cu penalizare L2

- **Pe scurt:** MLP cu 64 de neuroni ascunși pe German credit, Adam pe tot setul, penalizare doar pe primul strat.



Sursa: UCI Machine Learning Repository, Statlog German credit. Date: 600 de solicitanți de antrenare; MLP cu 64 de neuroni ascunși 3 072 de ponderi în primul strat.

- Ponderi cu $|w| < 0{,}01$: 67,2 % cu L1 față de 26,3 % cu L2 (3 072 de ponderi)
  - AUC pe test 0,768 cu L1, față de 0,761 cu L2.

- **În practică:** o penalizare L1 mai puternică ar anula coloane întregi de intrări, care pot ieși apoi din fluxul de date.

(calcul propriu) NN_L3_l1_l2_weights

Cuprins

- Generalizarea și supraînvățarea
- Penalizarea ponderilor
- Dropout, oprirea timpurie și normalizarea pe loturi
- Schema de validare
- Studiu de caz: detectarea fraudei cu carduri
- Sinteză

Dropout antrenează un ansamblu de rețele subțiate care își împart ponderile

- **Pe scurt:** oprirea aleatoare a unor neuroni împiedică orice neuron să depindă de cîțiva alții: fiecare variabilă trebuie să fie utilă singură.

$$\text{antrenare:}\quad \tilde{h}_j = \frac{m_j}{1-p}\, h_j, \quad m_j \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(1-p); \qquad \text{test:}\quad \tilde{h}_j = h_j$$

- **unde**
  - p rata de dropout, m_j masca aleatoare, împărțirea la $1-p$ (dropout inversat) păstrează $\mathbb{E}[\tilde{h}_j] = h_j$, deci nimic nu se schimbă la testare
- Cu H neuroni care pot fi opriți, eșantionăm din 2^H subrețele
  - La testare, rețeaua completă aproximează media lor (Srivastava et al., 2014).
- Rate uzuale: 0,1–0,3 în MLP tabelare, 0,5 în straturile mari
  - În PyTorch, `model.eval()` îl dezactivează.

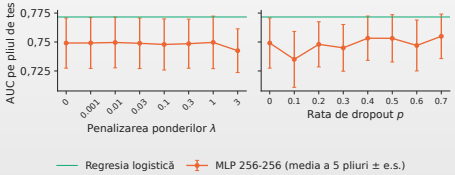
- **În practică:** dacă uitați `model.eval()` înainte de scorare, fiecare predicție devine aleatoare.

(Srivastava et al., 2014)

Exercițiu practic: notebook §4

Pe 600 de solicitanți, regularizarea nu egalează regresia logistică

- **Pe scurt:** MLP 256-256, 150 de epoci fără oprire timpurie; AUC pe pliul de test, 5 pliuri, pentru fiecare λ și p.



Sursa: UCI Machine Learning Repository, Statlog German credit. Date: 1 000 de solicitanți, validare încrucișată cu 5 pliuri, fără oprire timpurie (150 de epoci).

- AUC 0,749 fără regularizare, 0,750 la $\lambda = 1$, 0,755 la $p = 0{,}7$
  - Logistic 0,772
  - Norma ponderilor 18,0 față de 10,6 la $\lambda = 3$.

- **În practică:** pe date foarte puține, un model mai mic depășește unul mare regularizat.

(calcul propriu)

NN_L3_wd_dropout

Oprirea timpurie: păstrăm ponderile epocii celei mai bune pe validare

- **Pe scurt:** o antrenare mai lungă potrivește mai mult zgomot; ne oprim cînd pierderea de validare nu s-a mai îmbunătățit de ceva timp.

$$t^* = \arg\min_{t \le T} \mathcal{L}_{\text{val}}(\mathbf{w}_t), \quad \text{oprire cînd } t - t^* \ge P; \qquad \text{pierdere pătratică: } \eta\, t \approx \frac{1}{\lambda}$$

- **unde**
  - P răbdarea, în epoci, ultima relație spune că t pași de mărime η acționează ca o penalizare L2 de intensitate $\lambda \approx 1/(\eta t)$ (anexa)
- Setul de validare este folosit pentru decizia de oprire: nu mai este o estimare nedeplasată, deci este nevoie de un set de test separat.
- Este gratuită, ieftină și compatibilă cu orice alt regularizator (Prechelt, 1998).

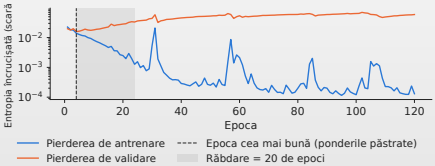
- **În practică:** salvați cea mai bună stare a modelului, nu ultima.

(Prechelt, 1998; Bishop & Bishop, 2024, ch. 9)

Exercițiu practic: notebook §5

La fraudele cu carduri, pierderea pe validare atinge minimul devreme, în timp ce pierderea pe antrenare continuă să scadă

- **Pe scurt:** MLP 256-256, împărțire cronologică; punctat: epoca cea mai bună pe validare; gri: fereastra de răbdare.



Sursa: ULB Machine Learning Group și Worldline, tranzacții cu carduri (OpenML 1597). Date: împărțire cronologică: 17 354 de tranzacții de antrenare 5 785 de validare.

- Epoca cea mai bună 4, oprire la 24 (răbdare 20)
  - Pierderea de validare 0,0159 atunci, 0,0593 după 120 de epoci.

- **În practică:** la evenimente rare, rețeaua începe să memoreze puținele fraude de antrenare după cîteva epoci.

(calcul propriu)

NN_L3_early_stopping

Normalizarea pe loturi standardizează fiecare neuron pe mini-lot

- **Pe scurt:** păstrarea preactivărilor centrate și scalate permite rețelelor adînci să se antreneze cu pași mai mari.

$$\hat{z}_{ij} = \frac{z_{ij} - \mu_{\mathcal{B},j}}{\sqrt{\sigma^2_{\mathcal{B},j} + \epsilon}}, \qquad y_{ij} = \gamma_j \hat{z}_{ij} + \beta_j, \qquad \mu_{\mathcal{B},j} = \frac{1}{B} \sum_{i \in \mathcal{B}} z_{ij}$$

- **unde**
  - $\sigma^2_{\mathcal{B},j}$ varianța pe lot a neuronului j, γ_j, β_j scara și deplasarea învățate, la testare, mediile mobile înlocuiesc statisticile lotului
- Normalizarea pe loturi (BN) netezește suprafața pierderii, ceea ce permite rate de învățare mai mari (Ioffe & Szegedy, 2015).
- Statisticile lotului adaugă zgomot la ieșirea fiecărui exemplu: un regularizator slab
  - Normalizarea pe strat (cursul 7) evită dependența de lot.

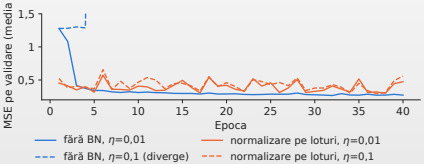
- **În practică:** cu normalizare pe loturi, `model.eval()` este obligatoriu, iar loturile foarte mici trebuie evitate.

(Ioffe & Szegedy, 2015)

Exercițiu practic: notebook §6

Normalizarea pe loturi permite antrenarea la o rată la care altfel rețeaua diverge

- **Pe scurt:** 6 straturi ascunse de 64 de neuroni pe California housing, SGD cu moment; mediane a 3 inițializări.



Sursa: StatLib, locuințe din California, recensământul din 1990 (Pace și Barry, 1997). Date: 6 straturi ascunse de 64 de neuroni, SGD cu moment 0,9, 3 inițializări.

- La $\eta = 0{,}1$: 3 din 3 inițializări diverg fără BN, cea mai bună MSE pe validare 0,283 cu BN
  - La $\eta = 0{,}01$: 0,265 fără, 0,277 cu BN.

- **În practică:** normalizarea pe loturi ajută mai ales optimizarea
  - Păstrați oprirea timpurie, eroarea ei fluctuează.

(calcul propriu)

NN_L3_batchnorm

Alți regularizatori: mai multe date, zgomot, ansambluri

- **Augmentarea datelor**: transformări care păstrează eticheta (decupări și oglindiri pentru imagini, cursul 4)
  - Pentru date tabelare, rareori sigură.
- **Injectarea de zgomot**: zgomotul gaussian pe intrări este echivalent, în prima aproximare, cu o penalizare L2 a gradientului în raport cu intrările.
- **Netezirea etichetelor**: antrenăm pe $(1-\epsilon)y + \epsilon/2$ în loc de ținte 0/1
  - Limitează probabilitățile prea sigure.
- **Ansamblurile**: media mai multor inițializări sau modele
  - Varianța scade aproximativ ca $1/M$ pentru M membri slab corelați.
- **Un model mai mic sau mai puține intrări**: cel mai vechi regularizator.
- **Mai multe date**: singurul fără efecte secundare
  - Încă un an de tranzacții valorează mai mult decît orice penalizare.

- **În practică:** studiul de caz despre fraudă face media a 5 inițializări: un ansamblu ieftin care raportează și incertitudinea.

(Bishop & Bishop, 2024, ch. 9)

Cuprins

- Generalizarea și supraînvățarea
- Penalizarea ponderilor
- Dropout, oprirea timpurie și normalizarea pe loturi
- Schema de validare
- Studiu de caz: detectarea fraudei cu carduri
- Sinteză

Setul păstrat deoparte, validarea încrucișată și validarea încrucișată imbricată

- **Pe scurt:** orice alegere făcută privind datele trebuie plătită cu date pe care alegerea nu le-a văzut.

$$\widehat{\mathrm{CV}}_K = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{|\mathcal{D}_k|}\sum_{i\in\mathcal{D}_k}\ell\big(y_i, \hat{f}^{(-k)}(\mathbf{x}_i)\big)$$

- **unde**
  - $\mathcal{D}_k$ pliul k, $\hat{f}^{(-k)}$ modelul estimat pe celelalte $K-1$ pliuri, de obicei $K=5$ sau 10, stratificat după etichetă
- Setul păstrat deoparte: o singură împărțire antrenare/validare/test
  - Ieftin, dar zgomotos pe date puține (slide-ul următor).
- Validarea încrucișată imbricată: o buclă interioară ajustează hiperparametrii, una exterioară măsoară întreaga procedură de ajustare.
- Raportați variația între pliuri, nu doar media
  - Comparați modelele cu testul t corectat din cursul 1.

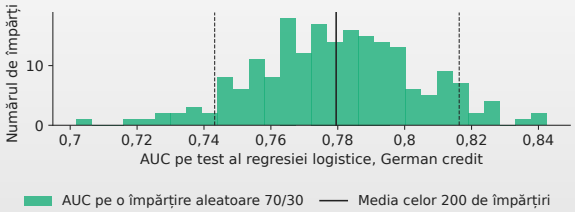
- **În practică:** performanța raportată trebuie să provină din date folosite nici la estimare, nici la alegere.

(Hastie, Tibshirani & Friedman, 2009, ch. 7)

Exerciții practice: notebook §7

O singură împărțire aleatoare poate muta AUC cu cîteva puncte

- **Pe scurt:** regresia logistică pe German credit, 200 de împărțiri aleatoare stratificate 70/30 ale acelorași solicitanți.



Sursa: UCI Machine Learning Repository, Statlog German credit. Date: 1 000 de solicitanți, 200 de împărțiri aleatoare stratificate.

- AUC pe test între 0,702 și 0,842
  - 90 % dintre împărțiri în [0,743; 0,816], abaterea standard 0,024.

- **În practică:** pe un set mic, o singură împărțire nu poate separa două modele: folosiți validarea încrucișată repetată.

(calcul propriu)

NN_L3_holdout_noise

Scurgerea de informație în setul de validare

- **Pe scurt:** orice informație din rîndurile de evaluare care ajunge în antrenare umflă scorul și dispare în producție.

Scurgerea	Exemplu	Remediu
Preprocesare pe toate rîndurile	standardizare sau completare estimate înainte de împărțire	estimare doar pe antrenare
Reeșantionare înainte de împărțire	copii ale unei fraude în antrenare și în test	supraeșantionare doar în pliul de antrenare
Ignorarea timpului	împărțire aleatoare a unei serii de timp sau a plăților	antrenare pe trecut, testare pe viitor
Ținta în intrări	„zile de întîrziere" folosite pentru a prezice neplata	doar informații cunoscute la momentul deciziei

- Serii de timp: validare cu fereastră extinsă sau pe blocuri; nu amestecați niciodată ordinea (Bergmeir & Benítez, 2012).

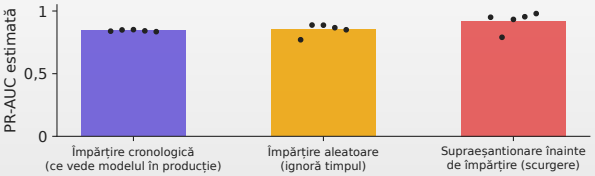
- **În practică:** notați, pentru fiecare intrare, momentul în care devine cunoscută.

(Kaufman et al., 2012)

Exerciții practic: notebook §8

Supraeșantionarea înainte de împărțire supraestimează performanța

- **Pe scurt:** același MLP 128-128 și aceleași tranzacții, evaluate în trei moduri; cîte 5 inițializări.



Sursa: ULB Machine Learning Group și Worldline, tranzacții cu carduri (OpenML 1597). Date: aceleași 28 924 de tranzacții, MLP 128-128 cu oprire timpurie, 5 inițializări.

- PR-AUC 0,843 cu împărțire cronologică, 0,853 cu împărțire aleatoare, 0,921 cu fraudele copiate de 20 de ori înainte de împărțire.

- **În practică:** doar estimarea cronologică descrie ce va vedea banca după punerea în producție.

(calcul propriu)

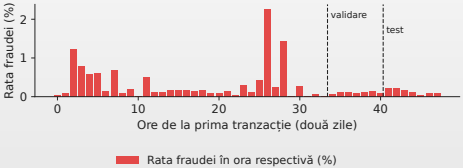
NN_L3_split_leakage

Cuprins

- Generalizarea și supraînvățarea
- Penalizarea ponderilor
- Dropout, oprirea timpurie și normalizarea pe loturi
- Schema de validare
- Studiu de caz: detectarea fraudei cu carduri
- Sinteză

Frauda este rară și grupată în timp

- **Pe scurt:** datele Université libre de Bruxelles (ULB) și Worldline, platforma Open Machine Learning (OpenML): două zile de plăți cu cardul.



Sursa: ULB Machine Learning Group și Worldline, tranzacții cu carduri (OpenML 1597). Date: septembrie 2013, 2 zile, 284 807 tranzacții (eșantion ponderat).

- 284 807 de plăți, rata fraudei 0,173 %
  - Păstrăm toate cele 492 de fraude și 10 % din plățile legitime (ponderea 10)
  - Intrări: 28 de scoruri de analiză a componentelor principale (PCA) și logaritmul sumei.

- **În practică:** rata orară variază puternic: o împărțire aleatoare ar amesteca orele liniștite și cele de atac.

(Dal Pozzolo et al., 2015)

NN_L3_fraud_by_hour

Măsuri de performanță pentru evenimente rare: precizia, sensibilitatea și PR-AUC

- **Pe scurt:** dacă nu semnalăm nimic avem o acuratețe de 99,8 %; banca vrea să știe cîte alerte sînt fraude și cîte fraude declanșează o alertă.

$$\text{precizia} = \frac{TP}{TP + FP}, \quad \text{sensibilitatea} = \frac{TP}{TP + FN}, \quad \text{PR-AUC} = \sum_k (R_k - R_{k-1})\, P_k$$

- **unde**
  - TP, FP, FN adevărat pozitive, fals pozitive, fals negative la un prag, P_k, R_k precizia și sensibilitatea la al k-lea prag
- PR-AUC depinde de rata fraudei (un scor aleator are PR-AUC = rata fraudei)
  - AUC nu depinde, ceea ce ascunde alarmele false (Saito & Rehmsmeier, 2015).
- Date subeșantionate: fiecare plată legitimă contează cu ponderea 10,00 în FP, deci precizia este cea a populației complete.

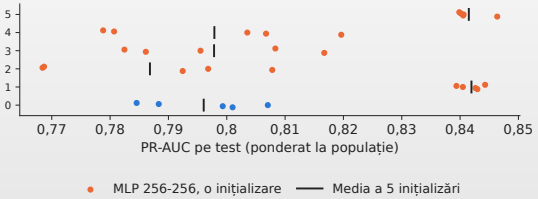
- **În practică:** raportați precizia la sensibilitatea pe care o poate gestiona echipa de investigații.

(Saito & Rehmsmeier, 2015)

Exercițiu practic: notebook §9

Pe orele de test, oprirea timpurie este regularizatorul cu cel mai clar cîștig

- **Pe scurt:** MLP 256-256, împărțire cronologică, 5 inițializări pe configurație; precizia ponderată la populație.



Sursa: ULB Machine Learning Group și Worldline, tranzacții cu carduri (OpenML 1597). Date: setul de test cronologic: ultimele 5 785 de tranzacții (75 de fraude); 5 inițializări.

- PR-AUC pe test: nimic 0,796, oprire timpurie 0,842, penalizare 0,787, dropout 0,798, normalizare 0,798, toate 0,841.

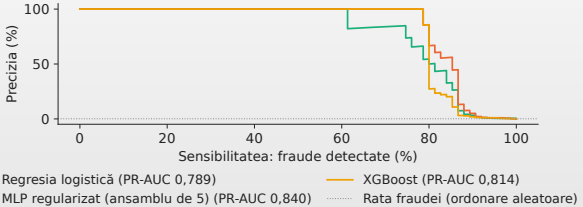
- **În practică:** cu 60 de epoci fixe, ceilalți regularizatori rămîn în zgomotul inițializărilor.

(calcul propriu)

NN_L3_reg_ablation

MLP regularizat comparat cu regresia logistică și cu gradient boosting

- **Pe scurt:** orele de test (ultimele 20 %); MLP mediază 5 inițializări cu oprire timpurie, penalizare și dropout.



Sursa: ULB Machine Learning Group și Worldline, tranzacții cu carduri (OpenML 1597). Date: setul de test cronologic: 5 785 de tranzacții 75 de fraude ponderat la populație.

- PR-AUC: logistic 0,789, MLP 0,840, extreme gradient boosting (XGBoost) 0,814
  - Sensibilitatea la precizie de 80 %: 74,7, 80,0, 80,0 %.

- **În practică:** cu 75 de fraude în test, cîteva puncte de diferență sînt în interiorul zgomotului.

(calcul propriu)

NN_L3_fraud_pr

Ponderile claselor și corecția pentru datele subeșantionate

- **Pe scurt:** ponderarea clasei rare schimbă nivelul scorurilor, nu doar ordinea lor; după subeșantionare, probabilitățile trebuie corectate.

$$\mathcal{L}_w = -\frac{1}{n}\sum_i \left[w_+ \, y_i \log p_i + w_- \, (1-y_i)\log(1-p_i)\right], \qquad p = \frac{\beta \, p_s}{\beta \, p_s - p_s + 1}$$

- **unde**
  - $w_\pm$ ponderile claselor, p_s probabilitatea învățată pe date în care plățile legitime au fost păstrate cu rata β (aici 0,1), p probabilitatea corectată
- Ponderile claselor sau reeșantionarea schimbă pragul efectiv
  - Ordonarea, deci și PR-AUC, se schimbă puțin.
- Orice am face, pragul trebuie să provină din costuri (slide-ul următor), iar probabilitățile trebuie corectate înainte de a fi folosite ca atare.

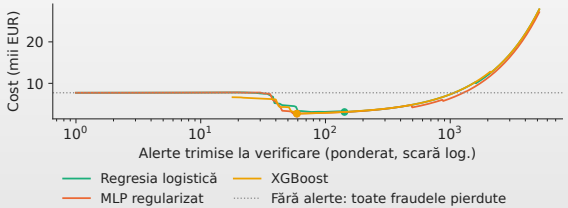
- **În practică:** nu raportați niciodată probabilități dintr-un model reeșantionat fără corecția pentru distribuția a priori.

(Dal Pozzolo et al., 2015)

Exercițiu practic: notebook §10

Numărul de alerte și ponderea pierderilor din fraudă evitate

- **Pe scurt:** o fraudă ratată costă suma ei; fiecare alertă costă o verificare de 5 euro (EUR), o constantă aleasă a priori.



Sursa: ULB Machine Learning Group și Worldline, tranzacții cu carduri (OpenML 1597). Date: perioada de test (ultimele 20 % din cele două zile); costul verificării 5 EUR pe alertă.

- Pierderi fără alerte 7 729 EUR
  - La numărul optim de alerte: logistic 3 093, MLP 2 694, XGBoost 2 695 EUR.

- **În practică:** bugetul de alerte este decizia de afaceri, modelul doar ordonează.

(calcul propriu)

NN_L3_fraud_cost

Sinteza metodelor de regularizare

Metoda	Ce face	Parametrul principal	Pe datele noastre
Penalizarea ponderilor	micșorează toate ponderile	λ	ponderi mai mici, aceeași AUC (German)
L1	aduce ponderi la zero	λ	majoritatea ponderilor primului strat aproape de zero
Dropout	antrenează un ansamblu de subrețele	p	în zgomotul inițializărilor
Oprirea timpurie	se oprește înainte de memorare	răbdarea P	cîștig clar la fraudă
Normalizarea pe loturi	stabilizează antrenarea	mărimea lotului	învață la η la care SGD simplu diverge
Model mai mic	mai puțini parametri	lățime, adîncime	regresia logistică depășește MLP pe German credit

- **În practică:** porniți cu oprirea timpurie și un model modest, apoi ajustați penalizarea
  - Adăugați dropout și normalizare pe loturi la rețelele adînci.

Cuprins

- Generalizarea și supraînvățarea
- Penalizarea ponderilor
- Dropout, oprirea timpurie și normalizarea pe loturi
- Schema de validare
- Studiu de caz: detectarea fraudei cu carduri
- Sinteză

Idei principale

- Supraînvățarea apare ca o pierdere de antrenare care scade și o pierdere de validare care crește
  - Pe date puține începe după cîteva epoci.
- Penalizarea ponderilor (distribuție a priori gaussiană), L1 (raritate), dropout (ansamblu), oprirea timpurie (L2 implicit) și normalizarea pe loturi (ajutor la optimizare) controlează capacitatea în moduri diferite.
- Modelele foarte mari pot prezenta dubla coborîre: curba clasică în U nu spune totul.
- Validarea trebuie să imite producția: împărțire cronologică, preprocesare și reeșantionare doar în partea de antrenare (PR-AUC 0,843 corect față de 0,921 cu scurgere).
- Pentru evenimente rare, folosiți PR-AUC, precizia la o sensibilitate gestionabilă și costul erorilor, cu mai multe inițializări.

Ecuațiile principale

$$\mathbb{E}(y - \hat{f})^2 = \text{bias}^2 + \text{Var}(\hat{f}) + \sigma^2$$

$$\mathbf{w}_{t+1} = (1 - \eta\lambda)\mathbf{w}_t - \eta\nabla\mathcal{L}$$

$$w_j \leftarrow \text{sign}(u_j)(|u_j| - \eta\lambda)_+$$

$$\tilde{h}_j = m_j h_j/(1 - p)$$

$$\hat{z} = (z - \mu_{\mathcal{B}})/\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}$$

$$\eta\, t^* \approx 1/\lambda$$

$$P = \frac{TP}{TP+FP},\ R = \frac{TP}{TP+FN}$$

$$p = \beta p_s/(\beta p_s - p_s + 1)$$

Cercetările grupului nostru: detectarea fraudei și avertizarea timpurie

- **Inteligența artificială (AI) explicabilă pentru fraudă** (Diaconu & Pele, 2026): pe datele de fraudă Institute of Electrical and Electronics Engineers, Computational Intelligence Society (IEEE-CIS), XGBoost și CatBoost au avut AUC mai mare, în timp ce o rețea neuronală pe grafuri (GNN), eșantionare și agregare pe graf (GraphSAGE), a surprins tipare relaționale ale fraudei pe care modelele tabelare nu le pot descrie.
- **Explicații pe mai multe niveluri**: cadrul lor de AI explicabilă (XAI) combină explicațiile aditive Shapley (SHAP) pentru variabile cu eliminarea de noduri și muchii pentru graf
  - Lucrarea conchide că explicarea fraudei trebuie să combine mai multe tipuri de modele — replicat în studiul de caz 1 (site-ul cursului).
- **Avertizarea timpurie pentru criptoactivele „zombie”** (Będowska-Sójka, Wójcik & Pele, 2026): sisteme de avertizare timpurie bazate pe învățare automată, care prezic ce criptoactive devin active „zombie”
  - Un model de avertizare timpurie trebuie validat înainte în timp, ca în secțiunea 4.
- 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 spirală, zgomot 50 și 10 % date de antrenare, și o rețea cu mai multe straturi ascunse: pierderea pe test rămîne peste cea de antrenare, adică supraînvățare.
  - Setați regularizarea la L2, apoi la L1, cu rata 0,01: L1 aduce multe ponderi la zero, desenate ca legături subțiri, iar L2 le micșorează pe toate.

Lecturi și resurse suplimentare

- **Bishop & Bishop (2024)**, capitolul 9: regularizarea: bishopbook.com.
- **Hastie, Tibshirani & Friedman (2009)**, capitolul 7: evaluarea și selecția modelelor: hastie.su.domains.
- **Dive into Deep Learning**, secțiunile 3.7, 5.6, 8.5: d2l.ai.
- **Manualul de detectare a fraudei** (Le Borgne et al., ULB): fraud-detection-handbook.github.io.
- **Acest curs**: cod, date, notebook-uri, Quantlets — github.com/danpele/NEURAL_BIZ.
- **Scurgerea de informație**: Kaufman et al. (2012).

Test 1: citirea curbelor

- Pierderea de antrenare continuă să scadă, cea de validare crește după epoca 3.
  - Ce faceți?

Răspuns

- Rețeaua supra-ajustează: păstrați ponderile din epoca 3 (oprire timpurie) și adăugați penalizare sau dropout
  - Nu antrenați mai mult.

Test 2: L1 sau L2

- Ce penalizare poate opri complet o intrare și de ce?

Răspuns

- L1: pasul de prag moale aduce ponderile mici exact la zero
  - Aici 67,2 % dintre ponderile primului strat sînt sub 0,01 (L2: 26,3 %).

Test 3: dropout la testare

- Cu dropout inversat și $p = 0{,}3$, ce trebuie făcut cu activările la testare?

Răspuns

- Nimic: împărțirea la $1 - p$ în antrenare păstrează deja media
  - Apelați doar `model.eval()`.

Test 4: supraeșantionarea

- De ce supraeșantionarea fraudelor înainte de împărțire crește PR-AUC de la 0,843 la 0,921?

Răspuns

- Copii ale aceleiași fraude ajung în antrenare și în test, deci modelul este testat pe fraude memorate
  - O parte din diferență vine și din ignorarea ordinii în timp.

Test 5: metrica potrivită

- Un model de fraudă are AUC 0,977.
  - Este suficient pentru a-l pune în producție?

Răspuns

- Nu: la o rată a fraudei de 0,173 %, o AUC mare poate ascunde multe alarme false
  - Verificați PR-AUC (0,840), precizia la sensibilitatea țintă și costul alertelor.

Bibliografie (1/2)

- 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
- Belkin, M., Hsu, D., Ma, S. & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32), 15849–15854. doi:10.1073/pnas.1903070116
- Bergmeir, C. & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, 191, 192–213. doi:10.1016/j.ins.2011.12.028
- Bishop, C. M. & Bishop, H. (2024). *Deep Learning: Foundations and Concepts*. Springer. doi:10.1007/978-3-031-45468-4
- Chen, T. & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of KDD 2016*, 785–794. doi:10.1145/2939672.2939785
- Dal Pozzolo, A., Caelen, O., Johnson, R. A. & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. *IEEE Symposium Series on Computational Intelligence*, 159–166. doi:10.1109/SSCI.2015.33
- 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
- Hastie, T., Tibshirani, R. & Friedman, J. (2009). *The Elements of Statistical Learning* (2nd ed.). Springer. doi:10.1007/978-0-387-84858-7
- Hoerl, A. E. & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55–67. doi:10.1080/00401706.1970.10488634
- Hofmann, H. (1994). Statlog (German Credit Data). UCI Machine Learning Repository. doi:10.24432/C5NC77

Bibliografie (2/2)

- Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of ICML*, 448–456. arXiv:1502.03167
- Kaufman, S., Rosset, S., Perlich, C. & Stitelman, O. (2012). Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data*, 6(4), 1–21. doi:10.1145/2382577.2382579
- Loshchilov, I. & Hutter, F. (2019). Decoupled weight decay regularization. *International Conference on Learning Representations*. arXiv:1711.05101
- Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B. & Sutskever, I. (2021). Deep double descent: where bigger models and more data hurt. *Journal of Statistical Mechanics*, 124003. doi:10.1088/1742-5468/ac3a74
- Pace, R. K. & Barry, R. (1997). Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3), 291–297. doi:10.1016/S0167-7152(96)00140-X
- Prechelt, L. (1998). Early stopping — but when? In *Neural Networks: Tricks of the Trade*, LNCS 1524, 55–69. Springer. doi:10.1007/3-540-49430-8_3
- Saito, T. & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3), e0118432. doi:10.1371/journal.pone.0118432
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 1929–1958.
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58(1), 267–288. doi:10.1111/j.2517-6161.1996.tb02080.x

Anexă: derivarea descompunerii deplasare-varianță

$$\mathbb{E}\big[(y-\hat{f})^2\big] = \mathbb{E}\big[(f+\varepsilon-\hat{f})^2\big] = \sigma^2 + \mathbb{E}\big[(f-\hat{f})^2\big]$$
$$\mathbb{E}\big[(f-\hat{f})^2\big] = \mathbb{E}\big[(f-\bar{f}+\bar{f}-\hat{f})^2\big] = (f-\bar{f})^2 + \mathbb{E}\big[(\hat{f}-\bar{f})^2\big], \qquad \bar{f} = \mathbb{E}_{\mathcal{D}}\hat{f}$$

- Termenii micști dispar pentru că $\mathbb{E}[\varepsilon] = 0$, ε este independent de $\mathcal{D}$, iar $\mathbb{E}[\hat{f} - \bar{f}] = 0$.
- Media după x dă descompunerea MSE așteptate pe test.

Anexă: de ce oprirea timpurie acționează ca o penalizare L2

$$\mathcal{L}(\mathbf{w}) \approx \tfrac{1}{2}(\mathbf{w}-\mathbf{w}^*)^\top \mathbf{H}(\mathbf{w}-\mathbf{w}^*),\ \mathbf{w}_0 = \mathbf{0}: \quad \mathbf{w}_t = \left[\mathbf{I} - (\mathbf{I}-\eta\mathbf{H})^t\right]\mathbf{w}^*$$

L2: $\tilde{\mathbf{w}} = (\mathbf{H}+\lambda\mathbf{I})^{-1}\mathbf{H}\mathbf{w}^*$; pe direcția proprie k: $1-(1-\eta h_k)^t \approx \dfrac{h_k}{h_k+\lambda}$ cînd $\lambda \approx \dfrac{1}{\eta t}$

- Ambele micșorează cel mai mult direcțiile cu curbură h_k mică
  - Oprirea după t pași este o penalizare de intensitate aproximativ $1/(\eta t)$.

(Bishop & Bishop, 2024, ch. 9)

Anexă: penalizarea ponderilor și penalizarea L2 pentru SGD și pentru Adam

$$\text{SGD: } \mathbf{w} - \eta \nabla \left(\mathcal{L} + \tfrac{\lambda}{2} \|\mathbf{w}\|^2 \right) = (1 - \eta\lambda)\mathbf{w} - \eta \nabla \mathcal{L}$$

$$\text{Adam+L2: } \mathbf{w} - \eta \frac{\hat{m}(\nabla \mathcal{L} + \lambda \mathbf{w})}{\sqrt{\hat{v}} + \epsilon} \qquad \text{vs.} \qquad \text{AdamW: } (1 - \eta\lambda)\mathbf{w} - \eta \frac{\hat{m}(\nabla \mathcal{L})}{\sqrt{\hat{v}} + \epsilon}$$

- Cu Adam, gradientul L2 este împărțit la $\sqrt{\hat{v}}$: ponderile cu gradienți mari aproape nu sînt micșorate.
- AdamW reface micșorarea uniformă
  - Este optimizatorul folosit în experimentele acestui curs.

(Loshchilov & Hutter, 2019)