








Serii de timp

Capitolul 9: Învățare automată pentru serii de timp



Daniel Traian PELE

Academia de Studii Economice din București
IDA Institute Digital Assets
Blockchain Research Center
AI4EFin Artificial Intelligence for Energy Finance
Academia Română, Institutul de Prognoză Economică
MSCA Digital Finance

Acronime folosite în acest material (1/3)

- **AI**: Artificial Intelligence (inteligență artificială)
- **AR**: AutoRegressive (model) — model autoregresiv
- **ARIMA**: AutoRegressive Integrated Moving Average (model) — model autoregresiv integrat cu medie mobilă
- **AUC**: Area Under the (ROC) Curve (aria de sub curba ROC)
- **BET**: Bucharest Exchange Trading (index) — indicele de referință al Bursei de Valori București
- **CPU**: Central Processing Unit (procesorul central)
- **DHR**: Dynamic Harmonic Regression (regresie armonică dinamică)
- **DM**: Diebold–Mariano (test) — testul Diebold–Mariano de comparare a prognozelor
- **ENTSO-E**: European Network of Transmission System Operators for Electricity (rețeaua europeană a operatorilor de transport și de sistem pentru energie electrică)
- **ES-RNN**: Exponential Smoothing – Recurrent Neural Network (hybrid model) — model hibrid de netezire exponențială și rețea neuronală recurentă
- **ETS**: Error, Trend, Seasonality (exponential smoothing state space models) — eroare, trend, sezonalitate (modele de netezire exponențială)
- **GB**: Gradient Boosting (ensemble of regression trees) — gradient boosting (ansamblu de arbori de regresie)
- **GBRT**: Gradient Boosted Regression Trees (arbori de regresie cu gradient boosting)

Acronime folosite în acest material (2/3)

- **GLM**: Generalised Linear Model (model liniar generalizat)
- **HAR**: Heterogeneous AutoRegressive (model) — model autoregresiv heterogen
- **IAPC**: Indicele armonizat al prețurilor de consum
- **LSTM**: Long Short-Term Memory (network) — rețea cu memorie pe termen lung și scurt
- **MAE**: Mean Absolute Error (eroarea absolută medie)
- **MASE**: Mean Absolute Scaled Error (eroarea absolută medie scalată)
- **MDI**: Mean Decrease Impurity (scăderea medie a impurității)
- **MIMO**: Multiple-Input Multiple-Output (forecasting strategy) — strategia cu mai multe ieșiri: un model dă toate orizonturile deodată
- **ML**: Machine Learning (învățare automată)
- **MLP**: MultiLayer Perceptron (perceptron multistrat)
- **MSE**: Mean Squared Error (eroarea pătratică medie)
- **OLS**: Ordinary Least Squares (metoda celor mai mici pătrate)
- **OLS-3**: OLS with three predictors: size, book-to-market and momentum (Gu, Kelly and Xiu, 2020) — OLS cu trei predictori: capitalizarea, raportul valoare contabilă/valoare de piață și momentum
- **OWA**: Overall Weighted Average (of the relative sMAPE and MASE, M4 competition) — media ponderată a sMAPE și MASE relative (competiția M4)
- **PCR**: Principal Component Regression (regresia pe componente principale)

Acronime folosite în acest material (3/3)

- **PIB**: Produsul intern brut
- **PLS**: Partial Least Squares (regresia prin cele mai mici pătrate parțiale)
- **QLIKE**: Quasi-LIKElihood (loss function) — funcția de pierdere de tip cvasi-verosimilitate
- **RF**: Random Forest (pădure aleatoare)
- **RMSE**: Root Mean Squared Error (rădăcina erorii pătratice medii)
- **RMSSE**: Root Mean Squared Scaled Error (rădăcina erorii pătratice medii scalate)
- **RNN**: Recurrent Neural Network (rețea neuronală recurentă)
- **ROC**: Receiver Operating Characteristic (curve) — curba caracteristică de operare
- **RW**: Random Walk (forecast) — prognoza de tip mers aleator
- **SARIMA**: Seasonal ARIMA (model) — model ARIMA sezonier
- **SES**: Simple Exponential Smoothing (netezire exponențială simplă)
- **SGD**: Stochastic Gradient Descent (coborîrea stochastică pe gradient)
- **SUA**: Statele Unite ale Americii
- **UE**: Uniunea Europeană
- **WRMSSE**: Weighted Root Mean Squared Scaled Error (M5 competition) — rădăcina erorii pătratice medii scalate, ponderată (competiția M5)

Întrebarea de azi și traseul

- **Întrebarea**: pot random forest, gradient boosting și rețelele neuronale prognoza o serie de timp mai bine decît ARIMA, ETS și prognoza naivă și cum verificăm corect acest lucru?
  - învățarea automată (machine learning, ML): algoritmi care învață o regulă de predicție din exemple; pentru serii de timp, exemplele se construiesc din trecutul seriei
- **Traseul** capitolului
  - prognoza ca învățare supervizată: laguri, ferestre mobile, variabile de calendar; strategiile recursivă, directă și cu ieșiri multiple
  - validarea walk-forward și leakage-ul; ridge și lasso; arbori, random forest, gradient boosting; rețele neuronale mici și LSTM
  - intervale de prognoză; modele locale și globale; patru aplicații și competițiile M4, M5
- Pornim de la Capitolul 0 (măsuri de acuratețe, metode de referință) și de la Capitolul 4 (validare încrucișată pentru serii de timp, Diebold–Mariano)

Rezultatele învățării

- Transformați o serie de timp într-un tabel de învățare supervizată fără să folosiți informație din viitor
- Construiți prognoze pe mai mulți pași cu strategiile recursivă, directă și cu ieșiri multiple
- Validați un model prin walk-forward și recunoașteți leakage-ul și look-ahead bias
- Explicați ridge, lasso, arborii de regresie, random forest, gradient boosting, MLP și LSTM, cu parametrii lor principali
- Construiți intervale de prognoză prin regresie cuantilică și predicție conformală
- Comparați prognozele ML cu metodele statistice de referință pe aceeași perioadă de test, cu MASE și testul Diebold–Mariano

Bibliografie și instrumente

- Manual: Huang și Petukhina (2022), cap. 10 (învățare automată pentru serii de timp)
  - manual însoțitor, gratuit online: Hyndman și Athanasopoulos (2021), secț. 12.4 (rețele neuronale); teorie: Hastie, Tibshirani și Friedman (2009), cap. 3, 9, 10, 11, 15
- Sinteze: Hewamalage, Bergmeir și Bandara (2021) (rețele recurente), Lim și Zohren (2021) (deep learning), Montero-Manso și Hyndman (2021) (modele globale)
- Quantlet-urile Python ale capitolului: Quantlets/Ch_09
  - `scikit-learn`: `RidgeCV`, `LassoCV`, `RandomForestRegressor`, `HistGradientBoostingRegressor`, `MLPRegressor`, `TimeSeriesSplit`; un LSTM mic în `PyTorch`
  - modele mici, semințe fixe, doar CPU: fiecare notebook rulează în Google Colab în cîteva minute
- Notebook-ul cursului: deschideți în Google Colab
- Curs video: Applied Time Series Analysis with Python

Învățarea supervizată: date, funcția de pierdere și antrenarea

- **Datele**: perechi $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$; $\mathbf{x}_i$: **variabilele explicative** (features, intrările), y_i: **ținta** (target)
  - scopul: o funcție $\hat{f}$ cu pierdere așteptată $E[L(y, \hat{f}(\mathbf{x}))]$ mică pe date **noi**
  - funcția de pierdere: eroarea pătratică $L = (y - \hat{y})^2$ (prognozează media), eroarea absolută $|y - \hat{y}|$ (mediana)
- **Antrenarea**: alegem $\hat{f}$ dintr-o clasă de funcții (liniare, arbori, rețele) minimizînd pierderea medie pe datele de antrenare
  - **hiperparametri**: setări fixate înaintea antrenării (adîncimea arborelui, numărul de arbori, penalizarea); se aleg prin validare
- ML standard presupune perechi independente; într-o serie de timp ele sînt ordonate și dependente, deci tabelul și validarea trebuie să respecte timpul

De la o serie la un tabel

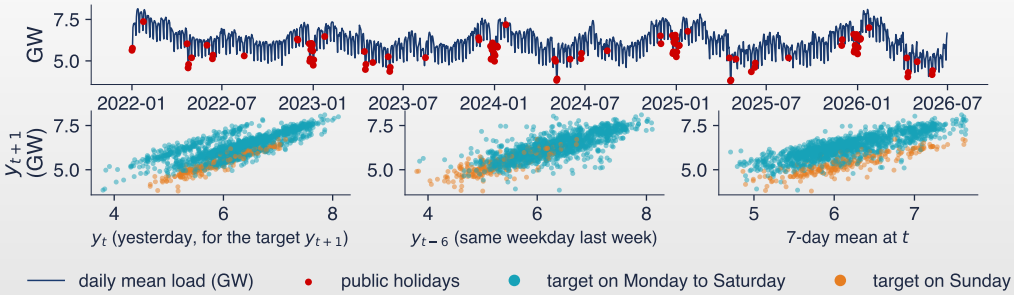
- **Laguri ca variabile explicative** (lag features): $y_t, y_{t-1}, \ldots, y_{t-p+1}$, cunoscute la originea prognozei t; ținta y_{t+h}
  - un AR(p) (Capitolul 2) este exact o regresie liniară pe acest tabel; ML înlocuiește funcția liniară cu una flexibilă
- **Variabile pe ferestre mobile**: medii, abateri standard, minime pe ultimele w valori
  - de exemplu media pe 7 zile $\bar{y}_t^{(7)} = \frac{1}{7}\sum_{j=0}^{6} y_{t-j}$
  - fereastra trebuie să se termine la t, originea: o fereastră care conține y_{t+1} introduce ținta în variabile (leakage)
- **Variabile de calendar** ale zilei-țintă: ziua săptămînii, luna, termeni Fourier $\sin(2\pi k\, d/365{,}25)$, sărbători
  - d: ziua din an; $k = 1, 2, \ldots$: armonica (un ciclu anual pentru $k = 1$)
  - se cunosc dinainte, deci sînt permise (Capitolul 4: termeni Fourier și Paștele ortodox); variabilele exogene (temperatura) doar dacă valorile lor sînt cunoscute sau prognozate la momentul t

Exemplu rezolvat: un tabel de laguri pentru consumul de energie electrică

Originea t	y_t	y_{t-1}	y_{t-2}	**ținta** y_{t+1}
lu	5,80	4,73	5,21	6,03
ma	6,03	5,80	4,73	6,38
mi	6,38	6,03	5,80	6,61
jo	6,61	6,38	6,03	6,59
vi	6,59	6,61	6,38	5,78
sî	5,78	6,59	6,61	5,09

- Consumul mediu zilnic al României (GW), ultimele zile dinaintea datei de 30 iunie 2025
  - fiecare rînd: ce știam la t și valoarea dorită pentru $t+1$
- Cu p laguri, primele $p-1$ zile nu au un rînd complet
  - o serie de lungime n dă $n-p-h+1$ rînduri pentru orizontul h
- Coloana zilei: scăderile de duminică se văd; modelul are nevoie de ziua săptămînii a zilei-**țintă**

Consumul de energie electrică: seria și trei variabile explicative



⊡ Consumul mediu zilnic al României, ENTSO-E (rețeaua europeană a operatorilor de transport și de sistem pentru energie electrică), 1 ianuarie 2022–30 iunie 2026, 1 642 de zile; jos: consumul de mîine față de trei variabile explicative

TSA_ch9_supervised_learning

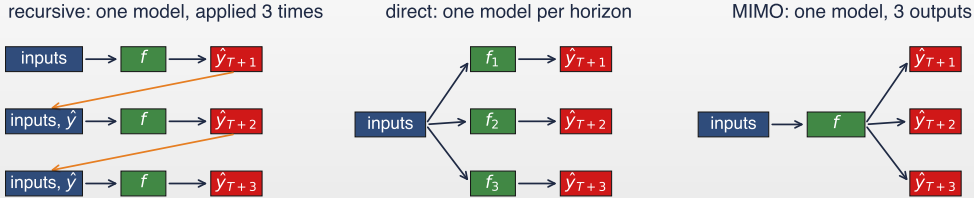
Interpretarea variabilelor pentru consum

- Media 6,18 GW; minimul 3,79 GW pe 20 aprilie 2025 (Paștele ortodox), maximul 8,15 GW pe 13 ianuarie 2022 (o săptămînă de iarnă)
  - ciclul săptămînal, ciclul anual (încălzire iarna, răcire vara) și scăderi mari de sărbători: sezonalitatea multiplă din Capitolul 4
- Corelația cu consumul de mîine: y_t 0,77; aceeași zi a săptămînii trecute 0,82; media pe 7 zile 0,74
  - doar cu y_t, duminicile formează un nor separat: un singur lag liniar nu poate reprezenta tiparul săptămînal
  - tabelul folosit mai departe are 1 614 rînduri și 30 de variabile explicative (14 laguri, două medii, aceeași zi a săptămînii, calendar)

Prognoze pe mai mulți pași: trei strategii

- **Recursivă** (iterată): un model pentru $h = 1$; prognoza lui înlocuiește valoarea necunoscută y_{T+1} și modelul se aplică din nou
  - așa prognozează ARIMA (Capitolul 3); erorile se acumulează cînd modelul pe un pas este greșit
- **Directă**: un model separat $\hat{f}_h$ pentru fiecare orizont $h = 1, \ldots, H$, antrenat pe ținta y_{t+h}
  - erorile nu se acumulează, dar avem H modele și mai puține rînduri pentru orizonturi lungi; prognozele pentru h diferite pot fi incoerente
- **Cu ieșiri multiple** (MIMO, multiple-input multiple-output): un model întoarce vectorul $(\hat{y}_{T+1}, \ldots, \hat{y}_{T+H})$
  - natural pentru rețelele neuronale; sinteză și comparație: Ben Taieb et al. (2012)

Prognoze recursive, directe și cu ieșiri multiple



- Albastru: intrările cunoscute la originea T; verde: modelele estimate; roșu: prognozele; portocaliu: o prognoză folosită din nou ca intrare

TSA_ch9_supervised_learning

Exemplu rezolvat: prognoze recursive și directe pe doi pași

- Modelul pe un pas estimat pe tabel: $\hat{y}_{t+1} = 1{,}2 + 0{,}8\, y_t$; ultima valoare $y_T = 5$
  - recursiv: $\hat{y}_{T+1} = 1{,}2 + 0{,}8 \cdot 5 = 5{,}20$; $\hat{y}_{T+2} = 1{,}2 + 0{,}8 \cdot 5{,}20 = 5{,}36$
  - regula implicită pe doi pași: $\hat{y}_{T+2} = 2{,}16 + 0{,}64\, y_T$
- Modelul direct estimat pe ținta y_{t+2}: $\hat{y}_{t+2} = 2{,}3 + 0{,}6\, y_t$; $\hat{y}_{T+2} = 5{,}30$
  - dacă AR(1) ar fi modelul adevărat, coeficienții direcți ar estima aceleași valori, 2,16 și 0,64
  - diferă: modelul pe un pas este greșit specificat la doi pași (de exemplu, un tipar săptămînal), iar modelul direct se adaptează orizontului
- Nicio strategie nu este întotdeauna superioară: le încercăm pe amîndouă pe datele de validare

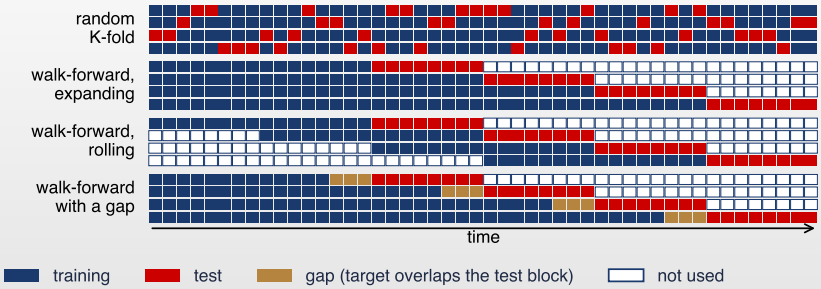
Recapitulare: prognoza ca învățare supervizată

- O prognoză este o predicție a lui y_{t+h} din variabile cunoscute la t: laguri, statistici pe ferestre mobile, calendar
- Un AR(p) este o regresie liniară pe laguri; ML schimbă funcția, nu tabelul
- Pe mai mulți pași: recursiv (un model, iterat), direct (un model pentru fiecare h), MIMO (un model, H ieșiri)

Eșantioanele de antrenare, validare și test

- Eșantionul de **antrenare**: estimează parametrii; de **validare**: alege hiperparametrii; de **test**: măsoară acuratețea finală, folosit o singură dată
  - pentru serii de timp cele trei eșantioane sînt blocuri consecutive, în această ordine
- Validarea **walk-forward** (validarea încrucișată pentru serii de timp, Capitolul 4): antrenăm pe datele de pînă la o origine, prognozăm blocul următor, mutăm originea înainte
  - fereastră **extinsă**: toate datele trecute; fereastră **mobilă**: ultimele W observații (se adaptează schimbărilor, folosește mai puține date)
  - erorile tuturor originilor se combină în MAE, RMSE, MASE (Capitolul 0) și se compară cu testul Diebold–Mariano
- Validarea încrucișată cu K grupuri aleatoare este validă doar pentru autoregresii pure cu erori de tip zgomot alb Bergmeir, Hyndman și Koo (2018)

Patru scheme de validare



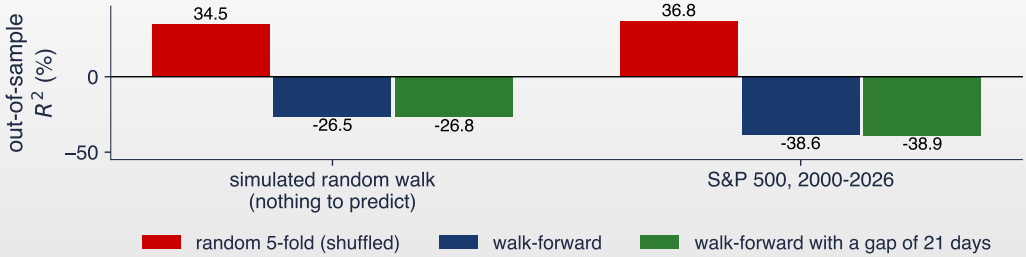
- *K*-fold aleator amestecă trecutul și viitorul; walk-forward păstrează ordinea; un spațiu (gap) elimină rîndurile de antrenare ale căror ținte se suprapun cu blocul de test

TSA_ch9_validation

Look-ahead bias și leakage-ul

- **Leakage-ul** (scurgerea de informație): informație care nu ar fi disponibilă la originea prognozei intră în antrenare sau în variabile Kaufman et al. (2012)
  - rezultat: scoruri excelente la validare, care dispar la utilizarea reală; în finanțe se numește **look-ahead bias**
- Surse tipice
  - o medie mobilă sau o diferență care include y_{t+1}; mediile mobile centrate și filtrele bilaterale (Capitolul 0)
  - scalarea, completarea valorilor lipsă sau selecția variabilelor estimate pe tot eșantionul înainte de împărțire
  - ținte suprapuse (sume pe următoarele h zile) cu grupuri aleatoare sau fără spațiu (gap)
  - date macroeconomice revizuite: versiunea de azi a PIB-ului nu era cunoscută la origine; vremea efectivă în locul prognozei meteo
- Regula: scrieți codul variabilelor ca funcție doar de datele de pînă la t și reestimați fiecare pas de preprocesare în fiecare fereastră de antrenare

Leakage-ul în practică: grupuri aleatoare pe ținte suprapuse



- Random forest pentru suma următoarelor 21 de randamente zilnice; variabile: sume trecute pe 5, 21, 63 de zile și volatilități; stînga: un mers aleator simulat; dreapta: S&P 500, 6 631 de zile

TSA_ch9_validation

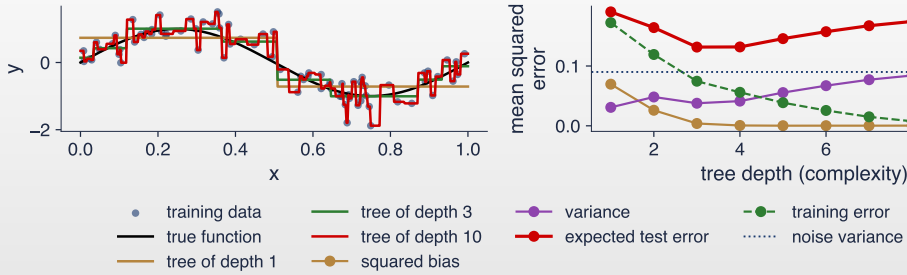
Interpretarea experimentului de leakage

- 5-fold aleator raportează $R^2 = 34{,}5\%$ pentru un mers aleator, unde nimic nu este previzibil, și 36,8% pentru S&P 500
  - țintele vecine au în comun 20 din 21 de randamente: un rînd de test are aproape-copii în grupurile de antrenare
- Walk-forward: $-26{,}5\%$ și $-38{,}6\%$; cu un spațiu de 21 de zile: $-26{,}8\%$ și $-38{,}9\%$
  - R^2 negativ în afara eșantionului: random forest este mai slab decît media istorică, cum ne așteptăm pentru randamente (Capitolul 1)
- Un model este atît de bun cît este validarea lui; un scor spectaculos este un motiv să căutăm un leakage

Overfitting și compromisul deplasare–variață

- Modelul $y = f(x) + \varepsilon$, $\mathrm{Var}(\varepsilon) = \sigma^2$; $\hat{f}$ estimat pe un eșantion de antrenare aleator
  - eroarea de test așteptată într-un punct x_0:
  - $E[(y_0 - \hat{f}(x_0))^2] = \underbrace{(E[\hat{f}(x_0)] - f(x_0))^2}_{\text{deplasare}^2} + \underbrace{\mathrm{Var}(\hat{f}(x_0))}_{\text{variață}} + \sigma^2$
  - f: funcția adevărată; y_0: o observație nouă în x_0; media se ia după eșantioanele de antrenare și zgomot; σ^2: zgomotul pe care niciun model nu îl poate elimina
- **Overfitting**: un model flexibil învață zgomotul; eroare de antrenare mică, eroare de test mare
  - **underfitting**: un model rigid ratează semnalul (deplasare mare)
- Fiecare metodă ML are un parametru de complexitate (adîncime, număr de arbori, penalizare, neuroni ascunși); validarea îl fixează

Deplasarea și varianța arborilor de regresie



⊡ Simulare: $y = \sin(2\pi x) + \varepsilon$, $\sigma = 0{,}3$, 80 de puncte, 300 de eșantioane de antrenare; arbori de adîncime 1 pînă la 10

TSA_ch9_validation

Interpretarea erorii de test în formă de U

- Adîncimea 1: deplasare^2 0,070, varianță 0,031; adîncimea 10: deplasare^2 0,000, varianță 0,090
  - eroarea de test așteptată este minimă la adîncimea 3 (0,132), peste varianța zgomotului, 0,09
- Eroarea de antrenare scade la 0,002 la adîncimea 10: nu poate alege complexitatea
- Ansamblurile (random forest, boosting) și penalizările (ridge, lasso) reduc varianța fără să piardă mult în deplasare

Recapitulare: validare fără leakage

- Walk-forward: antrenăm pe trecut, testăm pe blocul următor, mergem mai departe; grupuri aleatoare doar pentru autoregresii pure
- Leakage-ul: valori viitoare în variabile, preprocesare pe tot eșantionul, ținte suprapuse, date revizuite
- Eroarea de test = deplasare^2 + varianță + zgomot; validarea fixează complexitatea

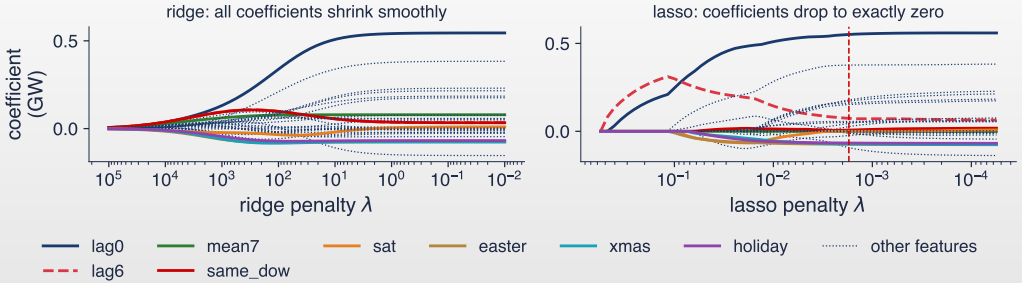
Multe laguri, metoda celor mai mici pătrate instabilă

- Model liniar pe p variabile standardizate: $\hat{y} = \beta_0 + \sum_j \beta_j x_j$; OLS (ordinary least squares, metoda celor mai mici pătrate) minimizează $\sum_t (y_t - \hat{y}_t)^2$
  - lagurile vecine sînt puternic corelate: coeficienții OLS devin mari, de semne opuse și instabili de la o fereastră la alta
- **Standardizăm** întîi: $x_j \to (x_j - \bar{x}_j)/s_j$, cu $\bar{x}_j$, s_j doar din fereastra de antrenare
  - altfel penalizarea tratează inegal variabilele măsurate în GW și variabilele binare
- Ideea regularizării: acceptăm puțină deplasare în schimbul unei scăderi mari a variației

Ridge și lasso

- **Ridge** Hoerl și Kennard (1970): $\min_\beta \sum_t (y_t - \beta_\mathbf{0} - \mathbf{x}_t'\beta)^\mathbf{2} + \lambda \sum_j \beta_j^\mathbf{2}$
  - $\lambda \geq 0$: penalizarea ($\lambda = 0$: OLS); $\mathbf{X}$: matricea variabilelor standardizate; $\mathbf{y}$: țintele; $\mathbf{I}$: matricea identitate
  - formă închisă $\hat{\beta} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-\mathbf{1}}\mathbf{X}'\mathbf{y}$; toți coeficienții se contractă spre 0, niciunul nu devine exact 0
- **Lasso** Tibshirani (1996) (least absolute shrinkage and selection operator): penalizarea $\lambda \sum_j |\beta_j|$
  - face unii coeficienți exact 0: selectează lagurile
- **Exemplu rezolvat**, o singură variabilă standardizată, $S_{xy} = \sum x_t y_t = 90$, $S_{xx} = \sum x_t^\mathbf{2} = 100$
  - OLS $90/100 = 0{,}90$; ridge, $\lambda = 50$: $90/150 = 0{,}60$; lasso, $\lambda = 40$: $(90 - 20)/100 = 0{,}70$; lasso, $\lambda \geq 180$: exact 0
  - lasso într-o dimensiune: $\hat{\beta} = \text{sign}(S_{xy}) \max(|S_{xy}| - \lambda/2, 0)/S_{xx}$ (prag moale, soft thresholding)
- λ se alege prin validare walk-forward (`TimeSeriesSplit`), niciodată prin grupuri aleatoare

Traiectoriile ridge și lasso pentru variabilele consumului



- Ținta: consumul de mîine; 30 de variabile standardizate; date pînă la 30 iunie 2025 (1 249 de rînduri); linia întreruptă: penalizarea lasso aleasă prin validare walk-forward

TSA_ch9_regularisation_trees

Interpretarea traiectoriilor coeficienților

- Cu o penalizare mare toți coeficienții sînt aproape de 0; cînd λ scade, intră primele lag6 (y_{t-6}, aceeași zi a săptămînii ca ținta), lag0 (y_t) și lag13; la penalizarea aleasă domină y_t, cu coeficientul 0,55 GW la o abatere standard
  - următorii coeficienți ca mărime sînt cei ai variabilelor binare ale zilelor (lunea 0,38): calendarul poartă cea mai mare parte din semnalul rămas
- La penalizarea aleasă, lasso păstrează 22 din 30 de variabile: multe laguri sînt redundante cînd y_t, ziua săptămînii și sărbătorile sînt incluse
- Ridge păstrează totul, dar contractă lin; pe originile de test din secțiunea 6, ridge este cel mai bun model ML

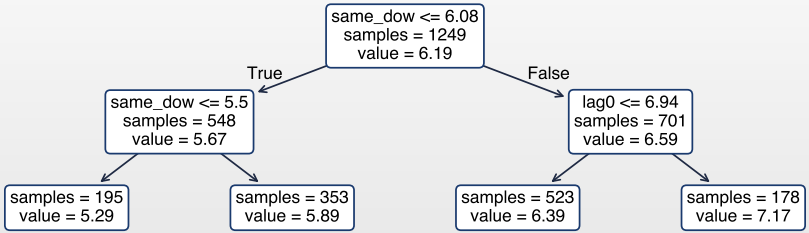
Recapitulare: ridge și lasso

- Cele mai mici pătrate penalizate: ridge $\lambda \sum \beta_j^2$ contractă; lasso $\lambda \sum |\beta_j|$ contractă și selectează
- Standardizăm în fiecare fereastră de antrenare; alegem λ prin walk-forward
- Un AR regularizat cu variabile de calendar este un reper ML puternic

Arborii de regresie

- Un **arbore de regresie** împarte spațiul variabilelor prin întrebări „$x_j \leq c$?" și prognozează, în fiecare regiune finală (frunză), media țintei din rîndurile de antrenare ale acesteia
  - împărțirea se alege lacom (greedy): perechea (j, c) cu cea mai mică sumă a pătratelor erorilor celor doi descendenți
- Puncte tari: neliniaritate și interacțiuni (ziua săptămînii $\times$ sezon) fără să le specificăm; nu necesită scalare
  - puncte slabe: un singur arbore este instabil (variață mare) și constant pe porțiuni
- **Un arbore nu poate extrapola**: prognoza lui este întotdeauna o medie a unor ținte din antrenare
  - pentru serii cu trend: modelăm diferențe, rate de creștere sau abaterea de la trend, nu niveluri

Un arbore de adîncime 2 pentru consumul de mîine



- Variabile: y_t (lag0), aceeași zi a săptămînii trecute (same_dow), indicatori de weekend și de sărbătoare; date pînă la 30 iunie 2025; cel puțin 20 de zile în fiecare frunză

TSA_ch9_regularisation_trees

Interpretarea arborelui

- Prima întrebare: a fost aceeași zi a săptămînii trecute sub 6,08 GW? Separă weekendurile și sărbătorile de zilele lucrătoare
  - patru frunze: 5,29; 5,89; 6,39 și 7,17 GW, cu 195, 353, 523 și 178 de zile
- Patru numere explică deja $R^2 = 0{,}64$ din varianță la antrenare; arborii mai adînci adaugă detalii și varianță
- Prognoza pentru orice zi este una dintre aceste patru medii

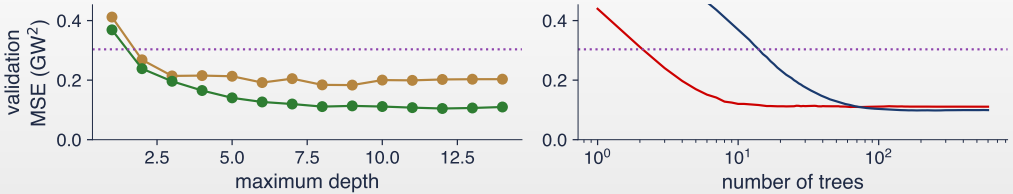
Random forest

- **Bagging** (bootstrap aggregation): creștem B arbori adînci pe eșantioane bootstrap ale rîndurilor și facem media lor
  - media a B arbori cu corelația ρ și varianța σ^2 are varianța $\rho\sigma^2 + (1-\rho)\sigma^2/B$
- **Random forest** Breiman (2001): la fiecare împărțire se încearcă doar o submulțime aleatoare de variabile (`max_features`)
  - astfel arborii se decorelează (ρ scade), iar media are o varianță mai mică
- Hiperparametri: numărul de arbori (un număr mai mare nu înrăutățește rezultatul, doar mărește timpul de calcul), `max_features`, dimensiunea minimă a frunzei
  - bootstrap-ul ignoră ordinea în timp în fereastra de antrenare; nu este o problemă, deoarece validarea este walk-forward

Gradient boosting

- **Boosting** Friedman (2001): adăugăm cîte un arbore mic, fiecare estimat pe reziduurile (gradientul negativ al funcției de pierdere) modelului curent
  - $F_m(\mathbf{x}) = F_{m-\mathbf{1}}(\mathbf{x}) + \nu\, g_m(\mathbf{x})$; F_m: modelul după m arbori; g_m: al m-lea arbore mic; ν: **rata de învățare** (0,01–0,3)
- **Exemplu rezolvat**: țintele $5, 6, 8$; pornim de la medie, $F_{\mathbf{0}} = 6{,}33$; reziduurile $-1{,}33$; $-0{,}33$; $+1{,}67$
  - un arbore cu o singură împărțire care izolează al treilea punct prognozează acolo 1,67; cu $\nu = 0{,}1$: $F_{\mathbf{1}} = 6{,}33 + 0{,}1 \cdot 1{,}67 = 6{,}500$
- Implementările rapide grupează variabilele în histograme: XGBoost Chen și Guestrin (2016), LightGBM Ke et al. (2017), `HistGradientBoostingRegressor` în scikit-learn
  - mai departe notăm GB pentru gradient boosting; cîștigătorii M5 (secțiunea 10) au folosit LightGBM
  - hiperparametri: numărul de arbori cu **oprire timpurie** pe datele de validare, dimensiunea arborilor, ν, dimensiunea minimă a frunzei

Un arbore, un random forest și boosting pe datele de consum



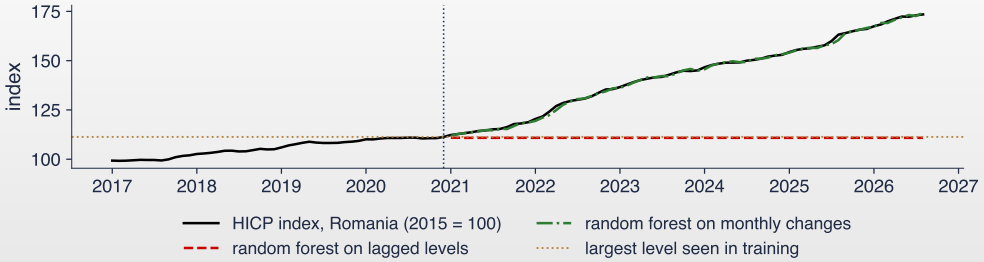
- Consumul de mîine; antrenare pînă la 31 decembrie 2024 (1069 de zile), validare ianuarie–iunie 2025 (180 de zile); linia punctată: prognoza naivă sezonieră

TSA_ch9_regularisation_trees

Interpretarea ansamblurilor

- Un singur arbore: cea mai bună MSE (mean squared error, eroarea pătratică medie) la validare, 0,183, la adîncimea 9; random forest: 0,105 la adîncimea 12
  - media elimină cea mai mare parte a variației; prognoza naivă sezonieră are 0,303
- Boosting cu $\nu = 0{,}3$: minimul 0,110 după 72 de arbori; cu $\nu = 0{,}05$: 0,098 după 256 de arbori
  - o rată de învățare mică are nevoie de mai mulți arbori și ajunge de obicei puțin mai jos; oprirea timpurie alege numărul de arbori

Arborii nu pot extrapola: nivelul prețurilor în România



- IAPC (indicele armonizat al prețurilor de consum) al României, Eurostat, 2015 = 100; prognoze cu o lună înainte din random forest antrenate pînă în decembrie 2020, pe laguri ale nivelurilor sau pe variațiile lunare logaritmice

TSA_ch9_regularisation_trees

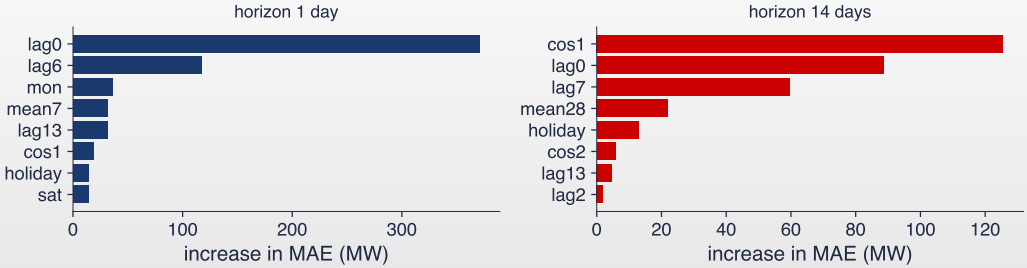
Interpretarea eșecului de extrapolare

- Cel mai mare indice din antrenare a fost 111,3; în august 2026 indicele a ajuns la 173,4
  - random forest pe niveluri rămîne în jurul valorii 110,8 timp de cinci ani: RMSE 36,9 puncte de indice
- Același random forest pe variațiile lunare, apoi cumulate, urmărește indicele: RMSE 0,71
- Din nou Capitolul 3: diferențiem sau transformăm seriile nestaționare înainte ca un arbore să le vadă

Variabilele folosite de model

- **Importanța prin permutare**: amestecăm o variabilă în datele de test și măsurăm cît crește eroarea
  - nu depinde de tipul modelului; se măsoară în afara eșantionului; variabilele corelate își împart (și își ascund) importanța
- Importanța calculată din impuritatea arborilor (MDI, mean decrease impurity) se obține la antrenare și favorizează variabilele continue: preferăm permutarea
- Importanța nu înseamnă cauzalitate: descrie modelul, nu sistemul energetic

Importanța prin permutare pentru gradient boosting



- Modele GB directe pentru orizonturile de 1 și 14 zile; antrenate pînă la 31 decembrie 2024, evaluate pe ianuarie–iunie 2025; 20 de permutări aleatoare pentru fiecare variabilă

TSA_ch9_load_forecasting

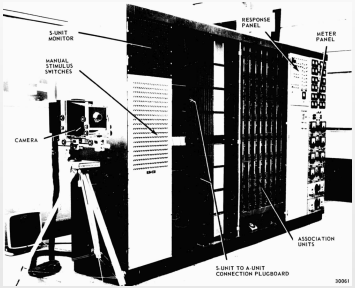
Interpretarea importanțelor

- Cu o zi înainte: lag0 (+371 MW de MAE cînd este amestecată) și lag6 (+117 MW) domină
  - ziua de ieri și aceeași zi a săptămînii trecute: persistența și ciclul săptămînal
- Cu paisprezece zile înainte: cos1 (+126 MW) și lag0 (+89 MW)
  - ciclul anual (cos1, termenul Fourier) cîștigă importanță pe măsură ce valorile recente devin mai puțin informative

Recapitulare: arbori și ansambluri

- Un arbore este o funcție constantă pe porțiuni, găsită prin împărțiri lacome; nu poate extrapola
- Random forest face media unor arbori adînci decorelați (mai puțină variantă); boosting adaugă arbori mici pe reziduuri (mai puțină deplasare)
- GB (gradient boosting, LightGBM, XGBoost, `HistGradientBoosting`): rata de învățare, numărul de arbori, oprirea timpurie
- Importanța prin permutare măsoară ce folosește modelul, în afara eșantionului

Perceptronul multistrat

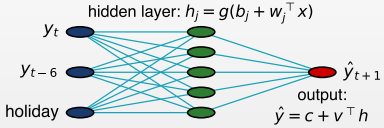


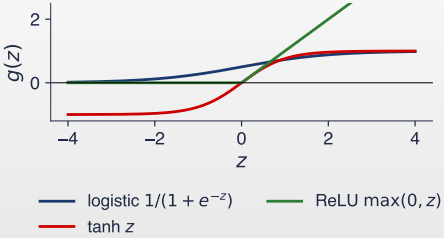
Mark I Perceptron, Cornell (1960): prima rețea neuronală antrenabilă, construită fizic

Imagine: J. C. Hay, A. E. Murray (1960), Mark I Perceptron operator's manual; public domain; Wikimedia Commons

- **MLP** (multilayer perceptron, perceptron multistrat), un strat ascuns cu q neuroni:
  - $h_j = g(b_j + \mathbf{w}_j'\mathbf{x})$, $j = 1, \ldots, q$; $\quad \hat{y} = c + \sum_j v_j h_j$
  - $\mathbf{x}$: intrările; $\mathbf{w}_j$, b_j: ponderile și termenul liber ale neuronului j; h_j: ieșirea lui; v_j, c: ponderile și termenul liber ale ieșirii
  - g: funcția de activare (ReLU $\max(0, z)$, tanh, logistică); cu $g(z) = z$ rețeaua redevine liniară
- Aproximarea universală Hornik, Stinchcombe și White (1989): suficienți neuroni ascunși aproximează orice funcție continuă
  - parametri: 3 intrări și 5 neuroni dau deja $3 \cdot 5 + 5 + 5 + 1 = 26$ de ponderi

O rețea mică și funcțiile ei de activare





- Stînga: 3 intrări (ziua de ieri, aceeași zi a săptămînii trecute, indicatorul de sărbătoare), 5 neuroni ascunși, o ieșire; dreapta: trei funcții de activare

TSA_ch9_neural_networks

Antrenarea unei rețele

- Minimizăm pierderea la antrenare prin **coborîre pe gradient**; gradienții se obțin prin retropropagare (back-propagation) Rumelhart, Hinton și Williams (1986)
  - SGD (stochastic gradient descent) și Adam folosesc loturi mici aleatoare; o trecere prin date este o epocă
- Suprafața pierderii are multe minime locale: rezultatele depind de ponderile inițiale aleatoare
  - fixăm sămînța și facem media cîtorva rețele cu semințe diferite (un ansamblu)
- Regularizare: o penalizare L2 pe ponderi, oprirea timpurie, dropout (dezactivarea aleatoare a unor neuroni)
  - intrările și țintele trebuie standardizate cu statisticile ferestrei de antrenare
- Datele puține (cîteva mii de rînduri) favorizează rețelele mici; rețelele adînci au nevoie de multe serii sau de date lungi, de frecvență înaltă

Rețele recurente și LSTM (1/2): rețeaua recurentă

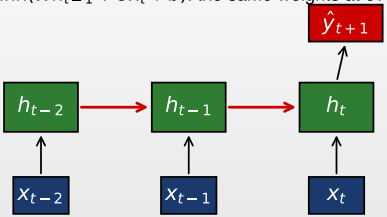
- **RNN** (recurrent neural network, rețea neuronală recurentă): o stare ascunsă duce trecutul mai departe, cu aceleași ponderi la fiecare pas
  - $h_t = \tanh(Wh_{t-\mathbf{1}} + Ux_t + b)$: noua stare combină starea anterioară și noua intrare
  - x_t: intrarea la momentul t; h_t: starea ascunsă; W, U, b: ponderile și termenul liber, comune tuturor pașilor; tanh ține fiecare componentă în $(-1, 1)$
  - prognoza se citește din ultima stare, de exemplu $\hat{y}_{t+\mathbf{1}} = v' h_t + c$
- Dificultatea: antrenarea prin mulți pași înmulțește multe derivate
  - gradienții dispar sau explodează: un RNN simplu uită ce s-a întîmplat cu mai mult de cîțiva pași în urmă
- Concluzia sintezei Hewamalage, Bergmeir și Bandara (2021)
  - pe serii unice și scurte, rețelele recurente sînt rareori mai precise decît ETS sau ARIMA
  - sînt utile mai ales cu multe serii înrudite și suficiente date (modele globale, Salinas et al. (2020))

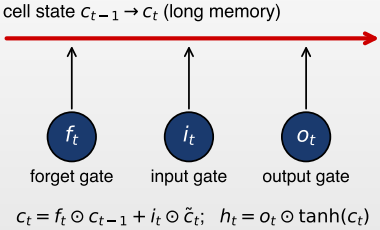
Rețele recurente și LSTM (2/2): celula LSTM

- **LSTM** (long short-term memory) Hochreiter și Schmidhuber (1997): o stare a celulei c_t actualizată aditiv, controlată de trei porți
  - porțile: $f_t = \sigma(W_f[h_{t-\mathbf{1}}, x_t] + b_f)$ (uitare), $i_t = \sigma(W_i[h_{t-\mathbf{1}}, x_t] + b_i)$ (intrare), $o_t = \sigma(W_o[h_{t-\mathbf{1}}, x_t] + b_o)$ (ieșire)
  - conținutul nou propus: $\tilde{c}_t = \tanh(W_c[h_{t-\mathbf{1}}, x_t] + b_c)$
  - $\sigma(z) = 1/(1 + e^{-z}) \in (0, 1)$: funcția logistică; $[h_{t-\mathbf{1}}, x_t]$: cei doi vectori puși unul sub altul; W_f, W_i, W_o, W_c și b_f, b_i, b_o, b_c: ponderile și termenii liberi
- Actualizarea celulei și a ieșirii
  - $c_t = f_t \odot c_{t-\mathbf{1}} + i_t \odot \tilde{c}_t$: cu f_t apropiat de 1 memoria veche se păstrează pe mulți pași; i_t stabilește cît conținut nou intră
  - $h_t = o_t \odot \tanh(c_t)$: poarta de ieșire decide ce parte a memoriei este transmisă mai departe; $\odot$: produsul element cu element
- Fiecare poartă este un mic strat logistic: un LSTM cu 32 de unități și 8 intrări are circa 5 400 de ponderi

O rețea recurentă și celula LSTM

$h_t = \tanh(Wh_{t-1} + Ux_t + b)$: the same weights at every step





- Stînga: RNN desfășurată pe trei pași; dreapta: starea celulei LSTM (roșu) și cele trei porți; $\odot$: produsul element cu element

TSA_ch9_neural_networks

Sepp Hochreiter și LSTM



Sepp Hochreiter (2015), Universitatea Johannes Kepler din Linz

Foto: Eulenreich (2015); CC BY-SA 4.0; Wikimedia Commons

- În lucrarea de diplomă din 1991, la München, Hochreiter a analizat de ce gradienții dispar în rețelele recurente
- Împreună cu Jürgen Schmidhuber a propus LSTM Hochreiter și Schmidhuber (1997)
  - una dintre cele mai citate lucrări despre rețele neuronale
- LSTM a stat la baza recunoașterii vorbirii și a traducerii automate înainte de transformer (Capitolul 11)

Recapitulare: rețele neuronale

- MLP: straturi de $g(b + \mathbf{w}'\mathbf{x})$; antrenat prin coborîre pe gradient; sămînța, scalarea, penalizarea L2, oprirea timpurie
- RNN și LSTM procesează secvențe; starea celulei LSTM evită dispariția gradienților
- Datele puține favorizează modelele mici; rețelele neuronale au nevoie de multe serii sau de date lungi

Exercițiul de prognoză



Linia de 750 kV de la Mărășești: rețeaua de transport a României

Foto: TrainSimFan (2022); CC BY-SA 4.0; Wikimedia Commons

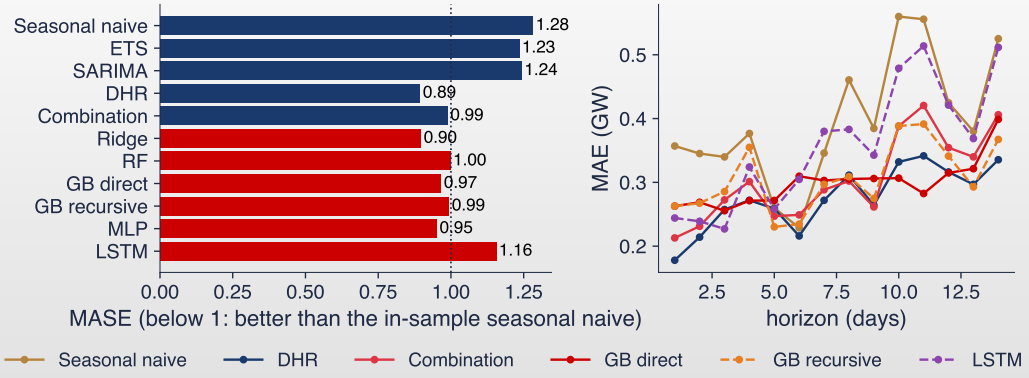
- Schema de evaluare din Capitolul 4: 26 de origini ale prognozei, la fiecare 14 zile, de la 30 iunie 2025 la 15 iunie 2026
  - orizontul 1–14 zile; fereastră extinsă de la 1 ianuarie 2022
  - modele statistice (Capitolul 4): naivă sezonieră, ETS, SARIMA, DHR (regresie armonică dinamică), combinația lor
- Modele ML, reestimate la fiecare origine:
  - directe: ridge, random forest (200 de arbori), GB (300 de arbori, $\nu = 0{,}05$), un model pentru fiecare orizont
  - recursiv: GB pe tabelul pe un pas; MIMO: un MLP (3 rețele, 32 de neuroni) și un LSTM (56 de zile la intrare, 14 la ieșire)
- Scala MASE: MAE în eșantion a metodei naive sezoniere săptămînale, 309 MW

Rezultatele: 26 de origini, 14 orizonturi

Model	MAE (GW)	RMSE (GW)	MASE	DM p, față de DHR
Naivă sezonieră	0,396	0,529	1,28	$< 0{,}001$
ETS	0,381	0,492	1,23	0,015
SARIMA	0,384	0,477	1,24	0,003
DHR	0,276	0,350	0,89	–
Combinația	0,305	0,380	0,99	0,065
Ridge	0,277	0,337	0,90	0,942
RF	0,308	0,388	1,00	0,068
GB direct	0,298	0,369	0,97	0,163
GB recursiv	0,307	0,388	0,99	0,113
MLP	0,294	0,362	0,95	0,546
LSTM	0,357	0,447	1,16	0,016

- DM: testul Diebold–Mariano pe erorile absolute medii pe origine (26 de valori), cu corecția Harvey–Leybourne–Newbold Harvey, Leybourne și Newbold (1997)

MASE și acuratețea pe orizonturi



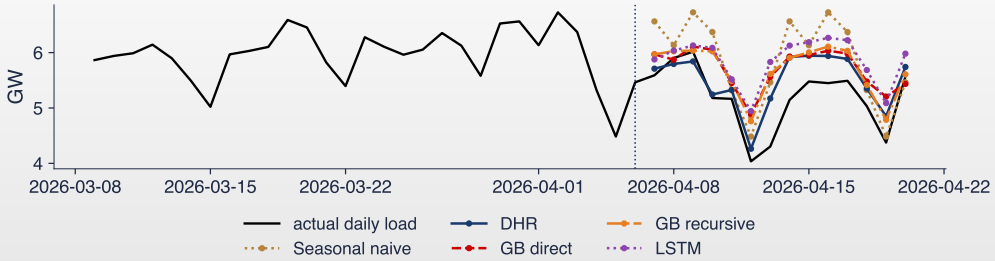
⊡ Stînga: MASE pentru cele unsprezece modele (albastru: Capitolul 4; roșu: ML); dreapta: MAE pe orizonturi pentru șase modele

TSA_ch9_load_forecasting

Interpretarea rezultatelor pentru consum

- DHR (MASE 0,89) și ridge (0,90) sînt cele mai precise; diferența nu este semnificativă (p = 0,942)
  - ambele sînt modele liniare cu variabilele de calendar potrivite; arborii și rețelele nu le depășesc
- GB direct (0,97), MLP (0,95), random forest (1,00) depășesc clar metoda naivă sezonieră (DM p = 0,0015; 0,0074; 0,0015) și au un MASE mai mic decît SARIMA și ETS (diferență netestată cu DM)
  - GB direct și recursiv sînt apropiate (0,97 și 0,99); LSTM (1,16) este mai slab decît toate modelele, cu excepția celui naiv, SARIMA și ETS
- Lecția: variabilele contează mai mult decît algoritmul; cu patru ani de date zilnice, un model liniar bine specificat este greu de depășit

Prognozele în jurul Paștelui ortodox din 2026



- Originea 6 aprilie 2026; prognoze pe 14 zile; MAE în această fereastră: DHR 344 MW, GB direct 536 MW, GB recursiv 503 MW, LSTM 661 MW

TSA_ch9_load_forecasting

Interpretarea ferestrei de Paște

- Toate modelele, cu excepția LSTM, văd indicatorul de Paște, dar arborii au doar patru sărbători de Paște la antrenare: prognozează o scădere prea mică
  - DHR estimează un singur coeficient pentru Paște din aceleași patru episoade și folosește informația mai eficient
- LSTM nu vede indicatorul de sărbătoare: repetă tiparul săptămînal și nu surprinde deloc sărbătoarea
- Evenimentele rare cer fie structură (un coeficient de regresie), fie multe serii care au în comun evenimentul (un model global)

Recapitulare: consumul de energie electrică

- Aceleași origini, același orizont, aceleași erori: modelele ML intră în comparația din Capitolul 4
- Ridge și DHR: nicio diferență semnificativă ($p = 0{,}942$); GB, random forest și MLP au un MASE mai mic decît SARIMA și ETS; LSTM este cel mai slab model ML
- Variabilele de calendar și sărbătorile decid clasamentul

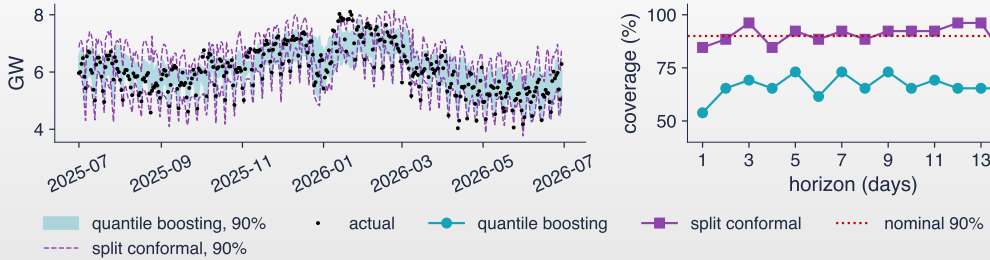
Regresia cuantilică și funcția de pierdere pinball

- O prognoză punctuală nu ajunge: operatorul rețelei are nevoie de consumul pe care nu îl va depăși cu probabilitatea de 95%
  - cuantila τ, q_τ, a lui y_{t+h}: $P(y_{t+h} \leq q_\tau) = \tau$; un interval de 90%: $[q_{\mathbf{0{,}05}}, q_{\mathbf{0{,}95}}]$
- **Funcția de pierdere pinball**: $L_\tau(y, q) = \tau(y - q)$ dacă $y \geq q$ și $(1 - \tau)(q - y)$ dacă $y < q$ Koenker și Bassett (1978)
  - valoarea ei așteptată este minimă în cuantila adevărată; pentru $\tau = 0{,}5$ este jumătate din eroarea absolută
  - exemplu, $\tau = 0{,}95$, $q = 7{,}0$ GW: $y = 7{,}4$ costă $0{,}95 \cdot 0{,}4 = 0{,}380$; $y = 6{,}5$ costă $0{,}05 \cdot 0{,}5 = 0{,}025$
- Orice algoritm poate minimiza funcția pinball: regresia cuantilică liniară, GB cuantilic (`loss='quantile'`), rețelele

Predicția conformală prin împărțire

- **Ideea** Lei et al. (2018): folosim erorile modelului pe date pe care nu le-a văzut pentru a fixa lățimea intervalului
  - 1. estimăm modelul pe datele de antrenare fără ultimele n puncte (setul de calibrare)
  - 2. calculăm erorile absolute $|e_i|$ pe setul de calibrare și le ordonăm
  - 3. $\hat{q}$ = a $\lceil (n+1)(1-\alpha) \rceil$-a cea mai mică valoare; intervalul $\hat{y} \pm \hat{q}$
  - α: proporția admisă de valori în afara intervalului ($\alpha = 0{,}1$ pentru un interval de 90%); $\lceil \cdot \rceil$: rotunjirea în sus
- Garanția: acoperire de cel puțin $1 - \alpha$ dacă erorile sînt interschimbabile (ordinea lor nu contează)
  - seriile de timp nu sînt interschimbabile: garanția este aproximativă; folosim date de calibrare recente și reverificăm acoperirea prin walk-forward
- Aici: $n = 182$ de zile, $\alpha = 0{,}1$, $\hat{q}$ = a 165-a cea mai mică eroare absolută, cîte un set de calibrare pentru fiecare orizont și origine

Intervale de 90% pentru consum



- GB direct, cele 364 de prognoze din secțiunea 6 (toate originile și orizonturile); stînga: intervale și valori realizate; dreapta: acoperirea pe orizonturi

TSA_ch9_load_forecasting

Interpretarea intervalelor

- GB cuantilic acoperă 66,5%, cu lățimea medie de 884 MW; predicția conformală acoperă 90,4%, cu lățimea de 1310 MW
  - GB cuantilic: 19,0% dintre valori sub interval și 14,6% deasupra (nominal 5% și 5%)
- Cuantilele estimate prin arbori sînt zgomotoase în cozi și prea înguste; calibrarea conformală corectează lățimea cu erori reale din afara eșantionului
- Raportați întotdeauna acoperirea empirică alături de un interval

Recapitulare: intervale de prognoză

- Prognozele cuantilice minimizează funcția pinball; un interval este o pereche de cuantile
- Predicția conformală: $\hat{y} \pm$ o cuantilă empirică a erorilor de calibrare
- Verificăm acoperirea prin walk-forward

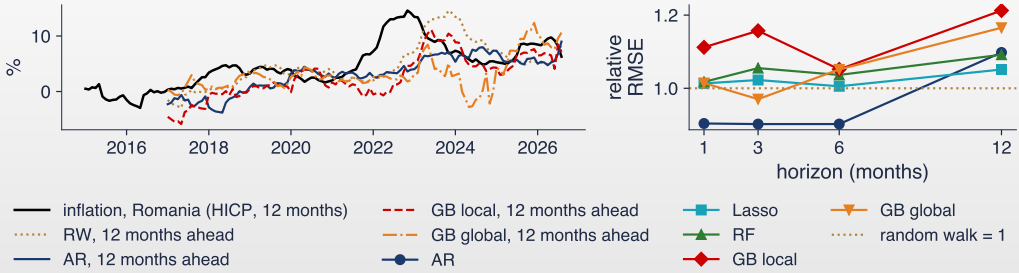
Modele locale și globale

- Model **local**: un model pentru fiecare serie, estimat doar pe acea serie (ARIMA, ETS, modelele de pînă acum)
  - puținele observații ale unei serii limitează complexitatea pe care și-o poate permite un model local
- Model **global**: un singur model estimat pe rîndurile combinate ale mai multor serii înrudite Montero-Manso și Hyndman (2021), Januschowski et al. (2020)
  - aceleași variabile pentru fiecare serie (laguri scalate, calendar); opțional, un identificator al seriei
  - mai multe date pentru fiecare parametru: algoritmii complecși (GB, rețele) devin utilizabili; cîștigătorii M4 și M5 au fost globali
- Riscul: seriile trebuie să aibă dinamici comune; un model global poate face să dispară prin mediere ceea ce este specific unei serii

Exercițiul pentru inflație

- Ținta: inflația anuală IAPC a României (Eurostat), $\pi_t = 100(P_t/P_{t-\mathbf{12}} - 1)$, P_t: indicele IAPC, la orizonturile $h = 1, 3, 6, 12$ luni
  - strategie directă; modelele prognozează variația $\pi_{t+h} - \pi_t$, deci arborii nu extrapolează niciodată nivelul
- Variabile la t: $\pi_t, \pi_{t-\mathbf{1}}, \pi_{t-\mathbf{2}}, \pi_{t-\mathbf{3}}, \pi_{t-\mathbf{6}}, \pi_{t-\mathbf{12}}$, ultimele trei variații lunare logaritmice și sumele lor pe 3 și 6 luni, luna calendaristică
  - modele: mers aleator (RW, $\hat{\pi}_{t+h} = \pi_t$), AR (OLS pe aceleași variabile), lasso, random forest, GB local (doar România), GB global (27 de țări UE combinate)
- Walk-forward: reestimare în fiecare ianuarie pe toate rîndurile a căror lună-țintă este deja observată (din 2006); originile de test ianuarie 2016–iulie 2026
- Rezultate înrudite: Medeiros et al. (2021) arată că random forest depășește metodele de referință pentru inflația SUA cu peste 100 de predictori

Inflația din România: prognoze și RMSE relativ



⊡ Stînga: inflația și prognozele cu 12 luni înainte, reprezentate la luna-țintă; dreapta: RMSE relativ la mersul aleator

TSA_ch9_inflation_global

Rezultatele pentru inflație

RMSE (pp), relativ la RW	$h = 1$	$h = 3$	$h = 6$	$h = 12$
RW (mers aleator)	0,63 (1,00)	1,41 (1,00)	2,25 (1,00)	3,90 (1,00)
AR	0,57 (0,90)	1,28 (0,90)	2,03 (0,90)	4,28 (1,10)
Lasso	0,64 (1,01)	1,45 (1,02)	2,26 (1,01)	4,10 (1,05)
RF	0,64 (1,02)	1,49 (1,06)	2,33 (1,04)	4,25 (1,09)
GB local	0,70 (1,11)	1,64 (1,16)	2,37 (1,05)	4,72 (1,21)
GB global	0,64 (1,01)	1,37 (0,97)	2,37 (1,05)	4,54 (1,17)

- DM p față de RW, AR: 0,019 ($h = 1$), 0,186 ($h = 3$), 0,388 ($h = 6$); GB global față de GB local: 0,117; 0,113; 0,999; 0,828
- Origini de test: de la 127 ($h = 1$) la 116 ($h = 12$) luni; DM cu erori pătratice și $h - 1$ autocovariațe

Interpretarea rezultatelor pentru inflație

- AR liniar este cel mai bun model pînă la 6 luni (cu circa 10% sub mersul aleator); niciun model ML nu depășește semnificativ mersul aleator
  - la 12 luni niciun model nu depășește mersul aleator: vîrful din 2022 (14,6% în noiembrie 2022) a venit din prețurile energiei, nu din trecutul inflației
- GB global este mai bun decît cel local la orizonturi scurte (RMSE 1,37 față de 1,64 la $h = 3$), dar nesemnificativ ($p = 0{,}113$)
  - combinarea a 27 de țări dă arborilor suficiente rînduri; GB local, cu 120–240 de rînduri la fiecare reestimare, face overfitting
- Circa 120 de erori lunare de test nu pot separa modele ale căror RMSE diferă cu 5%: concluzia onestă este „nu avem dovezi”

Recapitulare: modele locale și globale

- Un model global combină multe serii înrudite și permite algoritmilor flexibili să folosească mai multe date
- Inflația din România: AR este cel mai bun; GB global depășește GB local; niciun model nu depășește mersul aleator la 12 luni
- Eșantioane mici: raportăm p-value-urile DM, nu doar clasamentele RMSE

Volatilitatea realizată: HAR și ML (1/2)

- Ținta: logaritmul variației zilnice medii pe următoarele 5 zile; aproximarea variației zilnice din prețurile de deschidere, maxim, minim și închidere Garman și Klass (1980)
  - $\hat{\sigma}_t^{\mathbf{2}} = [\ln(O_t/C_{t-\mathbf{1}})]^{\mathbf{2}} + \frac{\mathbf{1}}{\mathbf{2}}[\ln(H_t/L_t)]^{\mathbf{2}} - (2\ln 2 - 1)[\ln(C_t/O_t)]^{\mathbf{2}}$: pătratul randamentului de peste noapte plus termenul Garman–Klass
  - O_t, H_t, L_t, C_t: prețurile de deschidere, maxim, minim și închidere din ziua t
  - volatilitatea este persistentă și previzibilă (Capitolul 5): terenul potrivit pentru ML
- Modelul **HAR** (heterogeneous autoregressive) Corsi (2009): OLS pe logaritmul variației din ultima zi, săptămînă și lună
  - $\ln \bar{\sigma}^{\mathbf{2}}_{t+\mathbf{1}:t+\mathbf{5}} = c + \beta_d \ln \hat{\sigma}_t^{\mathbf{2}} + \beta_w \ln \bar{\sigma}^{\mathbf{2}}_{t-\mathbf{4}:t} + \beta_m \ln \bar{\sigma}^{\mathbf{2}}_{t-\mathbf{21}:t} + u_t$; $\bar{\sigma}^{\mathbf{2}}_{a:b}$: media lui $\hat{\sigma}^{\mathbf{2}}$ în zilele de la a la b; u_t: eroarea
  - modelele ML adaugă 8 variabile: laguri ale variației zilnice, media trimestrială, randamentul, partea lui negativă și valoarea lui absolută

Volatilitatea realizată: HAR și ML (2/2)

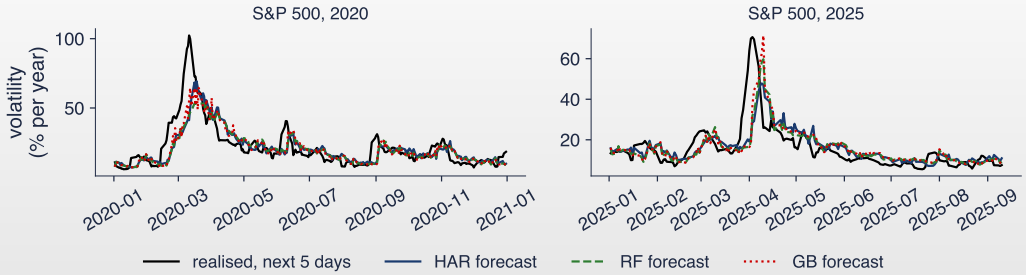
- Walk-forward anual din 2013, cu un spațiu de 5 zile între antrenare și test
- R^2 în afara eșantionului față de HAR: $R^2_{\mathbf{HAR}} = 1 - \sum_t (y_t - \hat{y}_t)^2 / \sum_t (y_t - \hat{y}_t^{\mathbf{HAR}})^2$
  - y_t: logaritmul varianței realizate; $\hat{y}_t$, $\hat{y}_t^{\mathbf{HAR}}$: prognozele modelului și ale HAR; pozitiv: mai precis decît HAR
- QLIKE Patton (2011): $L(\sigma^2, h) = \sigma^2/h + \ln h$, funcția de pierdere quasi-likelihood pentru varianță (Capitolul 5); o valoare mai mică este mai bună
  - σ^2: varianța realizată; h: prognoza varianței; pentru un σ^2 dat, pierderea este minimă la $h = \sigma^2$ și penalizează mai mult prognozele prea mici
  - robustă la zgomotul aproximării varianței: ordonează prognozele la fel ca varianța adevărată
- Testul Diebold–Mariano (Capitolul 4) pe diferențele zilnice ale QLIKE

Rezultatele: S&P 500 și DAX, 2013–2026

Model	S&P: R^2 față de HAR (%)	QLIKE	DM t	DAX: R^2 față de HAR (%)	QLIKE	DM t
HAR	+0,0	0,414	–	+0,0	1,023	–
Lasso	+7,1	0,400	−5,12	+7,9	1,015	−2,42
RF	+1,0	0,416	0,32	+3,5	1,024	0,03
GB	−0,5	0,416	0,35	+1,1	1,028	0,75
MLP	+8,3	0,398	−3,96	+8,6	1,015	−1,69

- 3 444 și 3 473 de zile de test; DM $t < -1{,}96$: modelul are un QLIKE semnificativ mai mic decît HAR
- Lasso și MLP obțin un R^2 față de HAR de 7–9% și reduc semnificativ QLIKE pe S&P 500; random forest și GB nu depășesc HAR
- Cîștigul vine din variabilele suplimentare (randamentul negativ: efectul de levier din Capitolul 5), folosite lin; arborii taie o relație netedă în trepte

Prognozele volatilității în 2020 și 2025



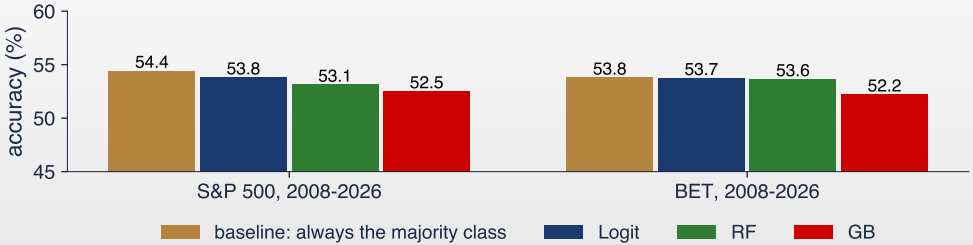
S&P 500: volatilitatea realizată anualizată a următoarelor 5 zile și prognozele HAR, random forest și GB, 2020 și 2025

TSA_ch9_finance

Prognoza semnului randamentului din ziua următoare: alegerea reperului

- Clasificare: ținta $1\{r_{t+\mathbf{1}} > 0\}$ (1 dacă randamentul de mîine este pozitiv, altfel 0); variabile: ultimele cinci randamente, sume pe 5, 21, 63 de zile, volatilitatea pe 21 și 63 de zile
  - modele: regresia logistică (logit), random forest, GB; walk-forward anual din 2008
- **Reperul corect** nu este 50%: piețele cresc în mai multe zile decît scad
  - „întotdeauna în creștere" (clasa majoritară din antrenare) are dreptate în 54,4% din zilele S&P 500 și în 53,8% din zilele BET
- Acuratețea, S&P 500: logit 53,8%, random forest 53,1%, GB 52,5%; BET: 53,7%, 53,6%, 52,2%
- Niciun model nu depășește reperul; AUC (aria de sub curba ROC) 0,505–0,533, aproape de 0,5: nici capacitate de ordonare
  - AUC: probabilitatea ca o zi de creștere aleasă la întîmplare să primească un scor mai mare decît o zi de scădere aleasă la întîmplare; 0,5: fără putere, 1: ordonare perfectă

Acuratețea față de reper



- Acuratețea în afara eșantionului, 2008–2026: 4 705 de zile S&P 500 și 4 686 de zile BET

TSA_ch9_finance

Interpretarea experimentului privind semnul

- Un model cu „53% acuratețe" pare performant; față de 54,4% pentru „întotdeauna în creștere", este mai slab decît a nu face nimic
  - GB pe S&P 500 este semnificativ mai slab decît reperul ($p = 0{,}008$): modelele flexibile ajustează zgomotul
- Eficiența în formă slabă (Capitolul 1): randamentele trecute nu conțin aproape nicio informație despre semnul următorului randament
- Volatilitatea este previzibilă, direcția nu: ML confirmă ce au arătat modelele statistice din Capitolele 1 și 5

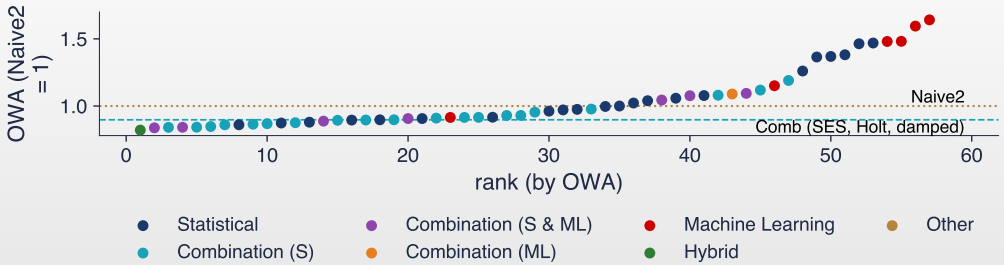
Recapitulare: volatilitatea și semnul randamentului

- Volatilitatea realizată: lasso și MLP cu variabile suplimentare sînt mai precise decît HAR; arborii nu
- Semnul randamentului: comparăm cu clasa majoritară, nu cu 50%; niciun model nu o depășește

Competițiile M

- Din 1982, Spyros Makridakis organizează competiții deschise de prognoză: toate metodele prognozează aceleași serii, evaluate de organizatori pe date ascunse
  - versiunea onestă a testului walk-forward: niciun participant nu își poate ajusta metoda pe datele de test
- Makridakis, Spiliotis și Assimakopoulos (2018a): pe 1 045 de serii lunare din M3, metodele ML populare au fost mai puțin precise decît opt metode statistice la toate orizonturile și mult mai lente
- M4 (2018), Makridakis, Spiliotis și Assimakopoulos (2020): 100 000 de serii cu șase frecvențe, 61 de metode, clasate după OWA (overall weighted average)
  - $\text{OWA} = \frac{1}{2}(\text{sMAPE}/\text{sMAPE}_{N2} + \text{MASE}/\text{MASE}_{N2})$; $N2$: Naive2, prognoza naivă a seriei desezonalizate; sub 1: mai precis decît Naive2
  - sMAPE (symmetric mean absolute percentage error, eroarea procentuală absolută medie simetrică) $= \frac{200}{H} \sum_{h=1}^{H} |y_{T+h} - \hat{y}_{T+h}|/(|y_{T+h}| + |\hat{y}_{T+h}|)$, în %
- M5 (2020), Makridakis, Spiliotis și Assimakopoulos (2022): 42 840 de serii zilnice ierarhice de vînzări Walmart, orizontul de 28 de zile

M4: acuratețea tuturor metodelor



- Fișierul oficial de evaluare M4 (github.com/Mcompetitions/M4-methods): OWA pentru cele 59 de metode clasate, propuneri și metode de referință, colorate după tip; 2 metode cu OWA peste 1,7 nu apar pe grafic

TSA_ch9_case_studies

Interpretarea M4

- Câștigătorul: modelul hibrid ES-RNN al lui Smyl (2020), netezire exponențială în interiorul unui LSTM antrenat pe toate seriile (un model global): OWA 0,821, sMAPE 11,37%
  - Comb, media metodelor SES, Holt și trend amortizat: OWA 0,898, sMAPE 12,55%; Naive2: sMAPE 13,56%
- Cele 6 metode ML pure: cea mai bună pe locul 23 (OWA 0,915); niciuna nu a depășit Comb, doar una a depășit Naive2 Makridakis, Spiliotis și Assimakopoulos (2018b)
  - 12 dintre cele mai bune 17 metode au fost combinații
- Lecția: pe serii scurte și eterogene, combinăm modele statistice sau lăsăm ML să învețe de la toate seriile; ML pur, aplicat local, este mai slab

M5: precizia gradient boosting pe datele din comerț



Un magazin Walmart: datele M5 sînt vînzările zilnice a 3 049 de produse în 10 magazine

Foto: Walmart Corporate; CC BY 2.0; Wikimedia Commons

- 5 507 echipe din 101 țări; măsura erorii WRMSSE (weighted root mean squared scaled error)
  - RMSSE: RMSE a prognozelor raportată la RMSE în eșantion a prognozei naive la un pas
  - WRMSSE: media RMSSE, ponderată cu valoarea vînzărilor
  - cîștigătorul (WRMSSE 0,520) a fost cu 22,4% mai precis decît cea mai bună metodă de referință, netezirea exponențială de jos în sus (0,671)
- Cele mai multe metode de top au folosit LightGBM Ke et al. (2017)
  - cîștigătorul a făcut media unor modele LightGBM combinate pe magazin, categorie și departament, recursive și directe
  - modele globale, variabile bogate de calendar și preț, multe serii înrudite: condițiile în care ML este superior

Gu, Kelly și Xiu (2020): învățare automată pentru randamentele acțiunilor

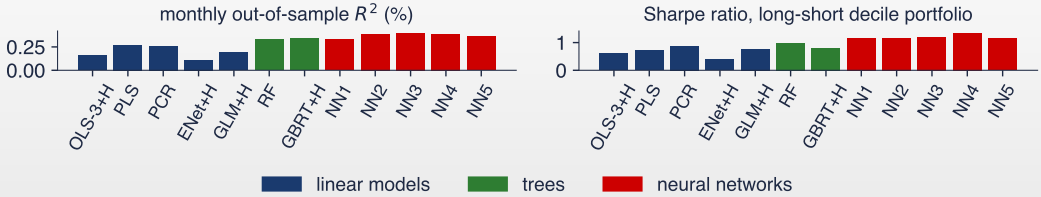


Charles M. Harper Center, University of Chicago Booth School of Business

Foto: Michael Barera (2013); CC BY-SA 4.0; Wikimedia Commons

- Gu, Kelly și Xiu (2020): randamentele lunare a circa 30 000 de acțiuni americane, 1957–2016, 920 de predictori
  - 12 metode, de la OLS la rețele neuronale cu cinci straturi
  - walk-forward: 18 ani de antrenare, 12 de validare, 30 de ani de test, reestimare anuală
- R^2 lunar în afara eșantionului: cel mult 0,40% (rețea cu trei straturi); OLS cu toți predictorii: $-3{,}46\%$
- R^2 mic, valoare mare: un portofoliu long-short ordonat după prognoze atinge un raport Sharpe anual de 1,35 (patru straturi), față de 0,61 pentru OLS-3
  - long-short: cumpărăm decila cu cele mai mari prognoze și vindem decila cu cele mai mici
  - raportul Sharpe: randamentul mediu anual împărțit la abaterea lui standard anuală

Gu, Kelly și Xiu (2020): R^2 în afara eșantionului și rapoarte Sharpe



- Cifrele publicate: Tabelul 1 (R^2 lunar în afara eșantionului, toate acțiunile) și Tabelul 7 (raportul Sharpe anualizat al portofoliului long-short pe decile, ponderat cu valoarea); +H: funcția de pierdere Huber
- OLS-3: OLS pe mărime, raportul valoare contabilă/valoare de piață și momentum; PLS, PCR: partial least squares, regresie pe componente principale; ENet: elastic net
- GLM: model liniar generalizat cu spline; RF: random forest; GBRT: arbori de regresie cu gradient boosting; NN1–NN5: rețele cu 1–5 straturi ascunse

TSA_ch9_case_studies

Recapitulare: studii de caz

- M4: cele mai precise sînt combinațiile și un model hibrid global; ML pur local este mai slab decît metodele simple
- M5: modelele globale LightGBM cu variabile de calendar și preț sînt clar cele mai precise
- Gu, Kelly și Xiu: predictibilitate mică, dar reală, în secțiunea transversală a acțiunilor; modelele neliniare aduc valoare

Foundation models: trimitere la Capitolul 11

- Un **foundation model** pentru serii de timp este o rețea mare (un transformer) pre-antrenată pe milioane de serii și folosită fără reantrenare (zero-shot)
  - modelul global extrem: Chronos, TimesFM, Moirai, Lag-Llama (studiu individual, Capitolul 11)
- Se aplică aceleași reguli: walk-forward pe date pe care modelul nu le-a văzut la pre-antrenare, aceleași metode de referință, MASE și DM

Contribuția posibilă a AI

- **Cod**: o primă versiune a tabelului de variabile, a buclei walk-forward și a tabelului de evaluare
- **Explicații**: o a doua explicație a boosting-ului, a porților LSTM sau a predicției conformale
- **Explorare**: multe seturi de variabile și hiperparametri deodată (doar pe datele de validare)
- Exemplu de prompt
  - `Write Python code that builds lag, rolling-mean and calendar features for daily electricity load, trains HistGradientBoostingRegressor with the direct strategy for horizons 1-14, evaluates it walk-forward against the seasonal naive forecast with MASE and the Diebold-Mariano test, and refits every preprocessing step inside each training window.`

Verificări necesare

- Fiecare variabilă folosește doar date pînă la origine: verificați deplasarea lagurilor și a ferestrelor mobile pe un tabel tipărit
- Fără `KFold(shuffle=True)`, fără scalare sau selecție de variabile estimate pe tot eșantionul
- Metodele de referință sînt prezente: naivă sau naivă sezonieră, ETS sau ARIMA, pe aceleași origini
- Comparația are un test DM; clasificarea are reperul clasei majoritare
- Semințele sînt fixate, iar rezultatul se menține cu o altă sămînță
- Fiecare referință citată există: verificați DOI-ul

Idei de reținut

- Prognoza ML = învățare supervizată pe un tabel de laguri, statistici pe ferestre mobile și variabile de calendar cunoscute la origine
- Doar validare walk-forward; leakage-ul produce scoruri spectaculoase și false
- Ridge și lasso, random forest, GB, MLP, LSTM: fiecare are un parametru de complexitate fixat prin validare
- Arborii nu pot extrapola: modelăm variații, nu niveluri
- Datele noastre: modelele liniare cu variabile bune au cel puțin aceeași precizie ca ML pentru consum și inflație; ML ajută pentru volatilitate; niciun model nu prognozează semnul randamentelor
- ML este superior cînd există multe serii înrudite (modele globale, hibridul din M4, LightGBM în M5); comparăm întotdeauna cu metode de referință, MASE și DM

Formule de reținut

Mărimea	**Formula**
Prognoza directă	$\hat{y}_{T+h} = \hat{f}_h(y_T, \ldots, y_{T-p+\mathbf{1}}, \mathbf{z}_{T+h})$, $\mathbf{z}_{T+h}$: variabilele de calendar
Ridge, lasso	$\min \sum_t (y_t - \mathbf{x}_t'\beta)^{\mathbf{2}} + \lambda \sum_j \beta_j^{\mathbf{2}}$; $\quad +\lambda \sum_j \lvert\beta_j\rvert$
Deplasare–varianță	$E[(y_{\mathbf{0}} - \hat{f}(x_{\mathbf{0}}))^{\mathbf{2}}] = \text{deplasare}^{\mathbf{2}} + \text{Var}(\hat{f}(x_{\mathbf{0}})) + \sigma^{\mathbf{2}}$
Boosting	$F_m(\mathbf{x}) = F_{m-\mathbf{1}}(\mathbf{x}) + \nu\, g_m(\mathbf{x})$
MLP	$\hat{y} = c + \sum_j v_j\, g(b_j + \mathbf{w}_j'\mathbf{x})$
LSTM	$c_t = f_t \odot c_{t-\mathbf{1}} + i_t \odot \check{c}_t, \quad h_t = o_t \odot \tanh(c_t)$
Funcția pinball	$L_\tau(y, q) = \max(\tau(y-q), (\tau-1)(y-q))$
Predicția conformală	$\hat{y} \pm \hat{q}, \quad \hat{q} = \lvert e\rvert_{(\lceil (n+\mathbf{1})(\mathbf{1}-\alpha)\rceil)}$
MASE	$\text{MAE}/\frac{\mathbf{1}}{n-m}\sum_{t>m} \lvert y_t - y_{t-m}\rvert$

Autoevaluare

- **Întrebare**: o medie mobilă pe 7 zile centrată în t este folosită ca variabilă pentru y_{t+1}. Este permis?
  - **Răspuns**: nu: conține $y_{t+1}, y_{t+2}, y_{t+3}$; folosiți media $y_{t-6}, \ldots, y_t$
- **Întrebare**: un random forest antrenat pe nivelurile PIB-ului pînă în 2019 prognozează anul 2024. Ce nu funcționează?
  - **Răspuns**: prognoza lui nu poate depăși cel mai mare nivel văzut la antrenare; modelați ratele de creștere
- **Întrebare**: un clasificator prognozează semnul randamentelor zilnice cu acuratețea de 53%. Este util?
  - **Răspuns**: doar dacă depășește semnificativ ponderea zilelor de creștere în perioada de test (circa 54% pentru S&P 500)
- Urmează: Capitolul 10, modele în spațiul stărilor, filtrul Kalman și modele Markov switching

Bibliografie (1/4)

- Ben Taieb, S., Bontempi, G., Atiya, A. F., & Sorjamaa, A. (2012). A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Systems with Applications*, 39(8), 7067–7083. doi:10.1016/j.eswa.2012.01.039
- Bergmeir, C., Hyndman, R. J., & Koo, B. (2018). A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics & Data Analysis*, 120, 70–83. doi:10.1016/j.csda.2017.11.003
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. doi:10.1023/A:1010933404324
- Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. doi:10.1145/2939672.2939785
- Corsi, F. (2009). A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2), 174–196. doi:10.1093/jjfinec/nbp001
- Diebold, F. X., & Mariano, R. S. (1995). Comparing predictive accuracy. *Journal of Business & Economic Statistics*, 13(3), 253–263. doi:10.1080/07350015.1995.10524599
- Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5), 1189–1232. doi:10.1214/aos/1013203451
- Garman, M. B., & Klass, M. J. (1980). On the estimation of security price volatilities from historical data. *The Journal of Business*, 53(1), 67–78. doi:10.1086/296072
- Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5), 2223–2273. doi:10.1093/rfs/hhaa009

Bibliografie (2/4)

- Harvey, D., Leybourne, S., & Newbold, P. (1997). Testing the equality of prediction mean squared errors. *International Journal of Forecasting*, 13(2), 281–291. doi:10.1016/S0169-2070(96)00719-4
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning* (2nd ed.). Springer. doi:10.1007/978-0-387-84858-7
- Hewamalage, H., Bergmeir, C., & Bandara, K. (2021). Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1), 388–427. doi:10.1016/j.ijforecast.2020.06.008
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780. doi:10.1162/neco.1997.9.8.1735
- 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
- Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359–366. doi:10.1016/0893-6080(89)90020-8
- Huang, C., & Petukhina, A. (2022). *Applied Time Series Analysis and Forecasting with Python*. Springer. doi:10.1007/978-3-031-13584-2
- Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and Practice* (3rd ed.), Section 12.4, Neural network models. OTexts. otexts.com/fpp3/nnetar.html
- Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4), 679–688. doi:10.1016/j.ijforecast.2006.03.001

Bibliografie (3/4)

- Januschowski, T., Gasthaus, J., Wang, Y., Salinas, D., Flunkert, V., Bohlke-Schneider, M., & Callot, L. (2020). Criteria for classifying forecasting methods. *International Journal of Forecasting*, 36(1), 167–177. doi:10.1016/j.ijforecast.2019.05.008
- 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
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 30. proceedings.neurips.cc
- Koenker, R., & Bassett, G. (1978). Regression quantiles. *Econometrica*, 46(1), 33–50. doi:10.2307/1913643
- Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., & Wasserman, L. (2018). Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523), 1094–1111. doi:10.1080/01621459.2017.1307116
- Lim, B., & Zohren, S. (2021). Time-series forecasting with deep learning: A survey. *Philosophical Transactions of the Royal Society A*, 379(2194), 20200209. doi:10.1098/rsta.2020.0209
- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018a). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLOS ONE*, 13(3), e0194889. doi:10.1371/journal.pone.0194889
- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018b). The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34(4), 802–808. doi:10.1016/j.ijforecast.2018.06.001

Bibliografie (4/4)

- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1), 54–74. doi:10.1016/j.ijforecast.2019.04.014
- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, 38(4), 1346–1364. doi:10.1016/j.ijforecast.2021.11.013
- Medeiros, M. C., Vasconcelos, G. F. R., Veiga, Á., & Zilberman, E. (2021). Forecasting inflation in a data-rich environment: The benefits of machine learning methods. *Journal of Business & Economic Statistics*, 39(1), 98–119. doi:10.1080/07350015.2019.1637745
- Montero-Manso, P., & Hyndman, R. J. (2021). Principles and algorithms for forecasting groups of time series: Locality and globality. *International Journal of Forecasting*, 37(4), 1632–1653. doi:10.1016/j.ijforecast.2021.03.004
- Patton, A. J. (2011). Volatility forecast comparison using imperfect volatility proxies. *Journal of Econometrics*, 160(1), 246–256. doi:10.1016/j.jeconom.2010.03.034
- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536. doi:10.1038/323533a0
- Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. (2020). DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3), 1181–1191. doi:10.1016/j.ijforecast.2019.07.001
- Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1), 75–85. doi:10.1016/j.ijforecast.2019.03.017
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1), 267–288. doi:10.1111/j.2517-6161.1996.tb02080.x