



Analiza și Prognoza Seriilor de Timp

Capitolul 9: Prophet și TBATS



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

Obiective de învățare

La finalul acestui capitol, veți fi capabili să:

1. Gestionăți serii de timp cu **sezonalități multiple**
2. Utilizați **Facebook Prophet** pentru prognoză flexibilă cu sărbători
3. Aplicați modele **TBATS** pentru sezonialitate complexă
4. Comparați și selectați între metodele moderne de prognoză



Cuprins

Fundamente

- Sezonalități Multiple
- Modelul TBATS
- Facebook Prophet

Aplicații

- Comparație și Ghid de Selecție
- Studiu de Caz
- Rezumat și Quiz



Problema: tipare sezoniere complexe

Exemple din Lumea Reală

- Cerere de electricitate pe oră:** Tipare zilnice + săptămânaile + anuale
- Trafic web:** Zilnic + săptămânal + efecte de sărbători
- Vânzări retail:** Săptămânal + lunar + anual + sărbători
- Volum call center:** Pe oră + zilnic + săptămânal

Limitarea SARIMA

- SARIMA(p, d, q)(P, D, Q)_s standard gestionează doar o **singură** perioadă sezonieră s
- Pentru date orare cu tipare zilnice și săptămânaile, avem nevoie de $s_1 = 24$ și $s_2 = 168$



Soluții pentru sezonalități multiple

Abordări Tradiționale

- Termeni Fourier:** Adăugare regresori sin/cos
- Variabile dummy:** Mulți parametri
- Modele nested:** Specificare complexă

Abordări Moderne

- TBATS:** Automat, gestionează multe perioade
- Prophet:** Flexibil, interpretabil
- Metode neurale:** Deep learning

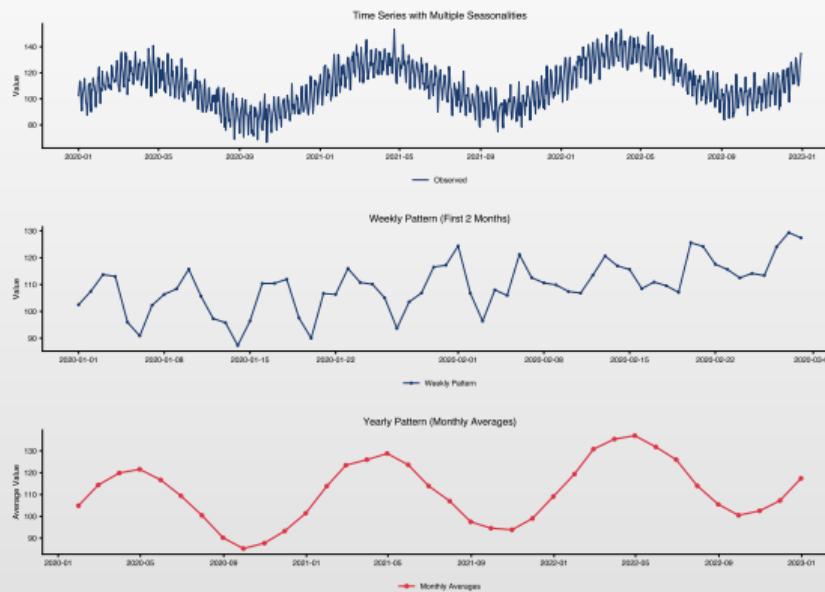
Comparație

- Rezumat comparativ:

Metodă	Nr. Max Sezonalități	Interpretabil
SARIMA	1	Da
Fourier + ARIMA	Multiple	Moderat
TBATS	Multiple	Moderat
Prophet	Multiple	Da



Exemplu: date orare cu sezonalități multiple



Q TSA_ch9_multiple_seasonality

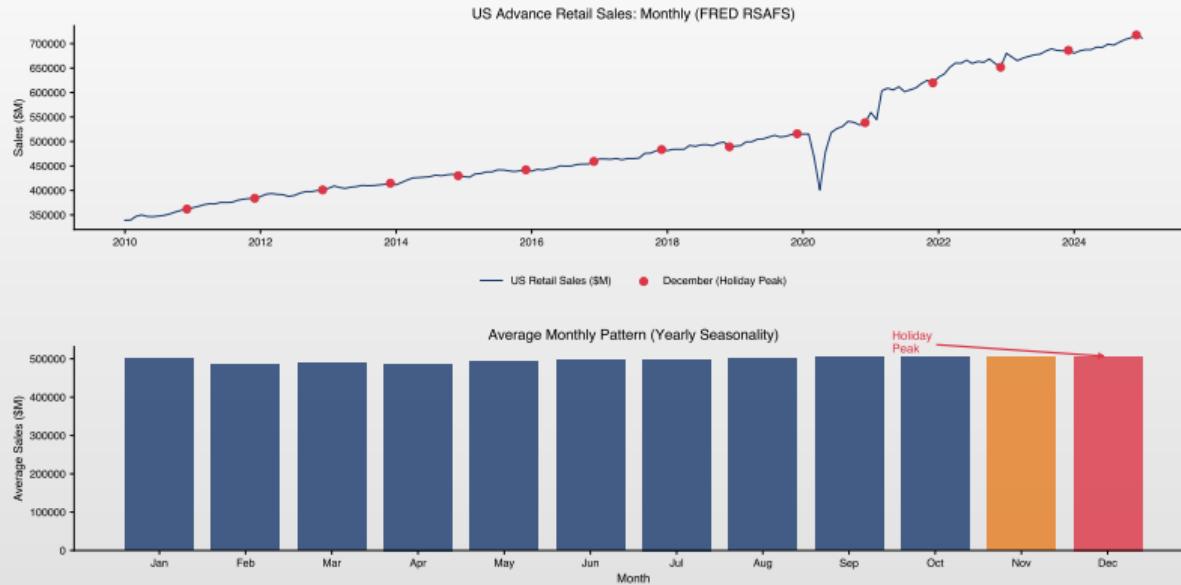


Exemplu real: cerere de electricitate



Q TSA_ch9_electricity_demand

Exemplu real: vânzări retail cu sărbători



Q TSA_ch9_retail_sales



Portret de cercetător: Rob J. Hyndman



*1967

W Wikipedia (en)

Biografie

- Statistician australian, profesor la Monash University
- Unul dintre cei mai influenți cercetători în prognoza seriilor de timp
- Creatorul pachetului **forecast** pentru R, utilizat pe scară largă
- Redactor-șef al *International Journal of Forecasting* (2005–2018)

Contribuții principale

- **Modelul TBATS** (2011) — Box-Cox trigonometric ARMA cu perioade sezoniere multiple
- **Cadrul ETS** — modele spațiu-stare de netezire exponențială cu selecție automată
- **Pachetul forecast** pentru R — setul standard de instrumente pentru prognoza seriilor de timp
- **Prognoza ierarhică** și metode de reconciliere a prognozelor

TBATS: ce înseamnă?

Componentele TBATS

- T** \succ Sezonalitate **Trigonometrică** folosind termeni Fourier
- B** \succ Transformare **Box-Cox** pentru stabilizarea varianței
- A** \succ Erori **ARMA** pentru autocorelația reziduală
- T** \succ Componentă de **Trend** (posibil amortizat)
- S** \succ Componete **Sezoniere** (multiple permise)

Inovația Cheie

- TBATS folosește **reprzentare trigonometrică** pentru sezonalitate:

$$s_t^{(i)} = \sum_{j=1}^{k_i} \left[s_j^{(i)} \cos\left(\frac{2\pi j t}{m_i}\right) + s_j^{*(i)} \sin\left(\frac{2\pi j t}{m_i}\right) \right]$$

- m_i este perioadă sezonieră i și k_i este numărul de armonici



Transformarea Box-Cox

Definiție 1 (Transformarea Box-Cox)

Transformarea Box-Cox cu parametrul ω este definită astfel:

$$y_t^{(\omega)} = \begin{cases} \frac{y_t^\omega - 1}{\omega} & \text{dacă } \omega \neq 0 \\ \ln(y_t) & \text{dacă } \omega = 0 \end{cases}$$

Scop

- Stabilizarea varianței:** Face varianța constantă în timp
- Normalizare:** Reduce asimetria în date
- Valori uzuale:** $\omega = 0$ (log), $\omega = 0.5$ (rădăcină pătrată), $\omega = 1$ (fără transformare)



Structura modelului TBATS

Specificația Completă a modelului

- Ecuațiile modelului TBATS:

$$y_t^{(\omega)} = \ell_{t-1} + \phi b_{t-1} + \sum_{i=1}^M s_t^{(i)} + d_t \quad (1)$$

$$\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha d_t \quad (2)$$

$$b_t = \phi b_{t-1} + \beta d_t \quad (3)$$

$$d_t = \sum_{i=1}^p \varphi_i d_{t-i} + \sum_{j=1}^q \theta_j \varepsilon_{t-j} + \varepsilon_t \quad (4)$$

Notări

- $y_t^{(\omega)}$ ~ seria transformată Box-Cox (dacă $\omega \neq 1$)
- ℓ_t ~ nivelul local, b_t ~ trendul cu amortizare ϕ
- $s_t^{(i)}$ ~ M componente sezoniere cu perioade m_1, \dots, m_M
- d_t ~ procesul de eroare ARMA(p, q)



TBATS: evoluția stărilor sezonalității trigonometrice

Definiție 2 (Recursia trigonometrică în spațiul stărilor)

Pentru fiecare componentă sezonieră cu perioada m_i și k_i armonici, definim stările:

$$\begin{pmatrix} s_{j,t}^{(i)} \\ s_{j,t}^{*(i)} \end{pmatrix} = \begin{pmatrix} \cos(\lambda_j) & \sin(\lambda_j) \\ -\sin(\lambda_j) & \cos(\lambda_j) \end{pmatrix} \begin{pmatrix} s_{j,t-1}^{(i)} \\ s_{j,t-1}^{*(i)} \end{pmatrix} + \begin{pmatrix} \gamma_1^{(i)} \\ \gamma_2^{(i)} \end{pmatrix} d_t$$

unde $\lambda_j = \frac{2\pi j}{m_i}$ este frecvența armonică j .

Interpretare

- Matricea de rotație păstrează structura periodică
- Sezonalitatea totală: $s_t^{(i)} = \sum_{j=1}^{k_i} s_{j,t}^{(i)}$
- Parametri: $2k_i$ stări per perioadă sezonieră

TBATS: sezonalitate trigonometrică

De ce Termeni Fourier/Trigonometrici?

- Simplu:** Mai puțini parametri decât variabilele dummy
- Neted:** Captează natural tiparele sezoniere netede
- Flexibil:** Numărul de armonici k controlează complexitatea
- Perioade non-întregi:** Poate gestiona $s = 365.25$ pentru date zilnice

k mic (puține armonici)

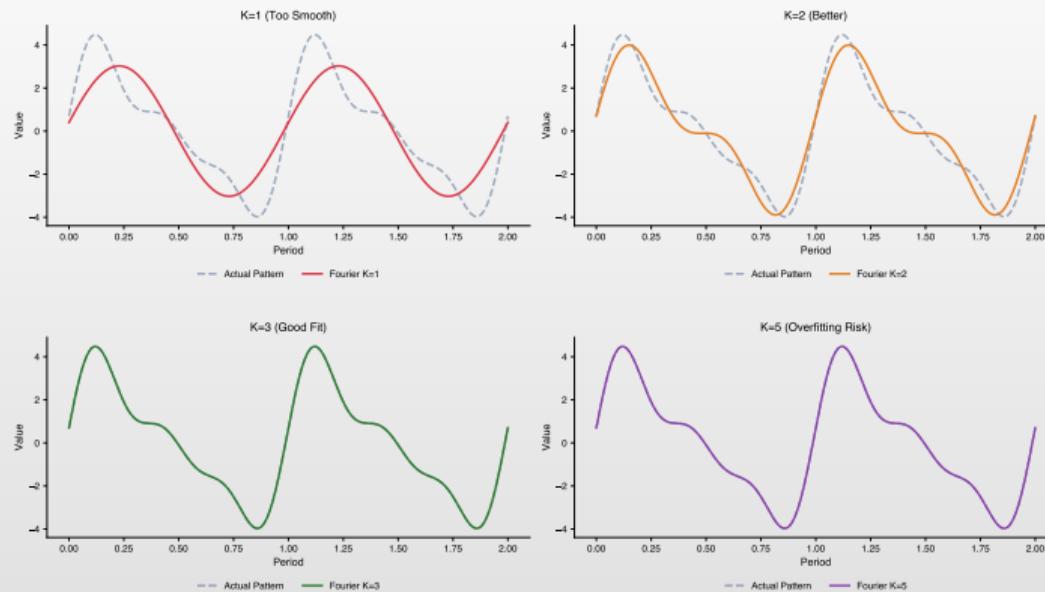
- Tipar neted
- Mai puțini parametri
- Poate rata vârfuri abrupte

k mare (multe armonici)

- Poate capta orice tipar
- Mai mulți parametri
- Risc de supraajustare



Aproximarea Fourier a sezonalității



Q TSA_ch9_fourier_approximation



TBATS în practică

Implementare Python

- **Pachet tbats:** Oferă selecție automată a modelului
 - ▶ Selectează automat parametrul Box-Cox ω
 - ▶ Alege numărul de armonici k_i pentru fiecare perioadă sezonieră
 - ▶ Selectează ordinele ARMA (p, q)
 - ▶ Testează trend amortizat vs neamortizat

Exemplu de Cod

- **Cod Python:**

```
from tbats import TBATS
estimator = TBATS(seasonal_periods=[7, 365.25])
model = estimator.fit(y)
forecast = model.forecast(steps=30)
```

Notă

- BATS este versiunea mai simplă fără termeni trigonometrici (folosește stări sezoniere tradiționale)



TBATS: avantaje și limitări

Avantaje

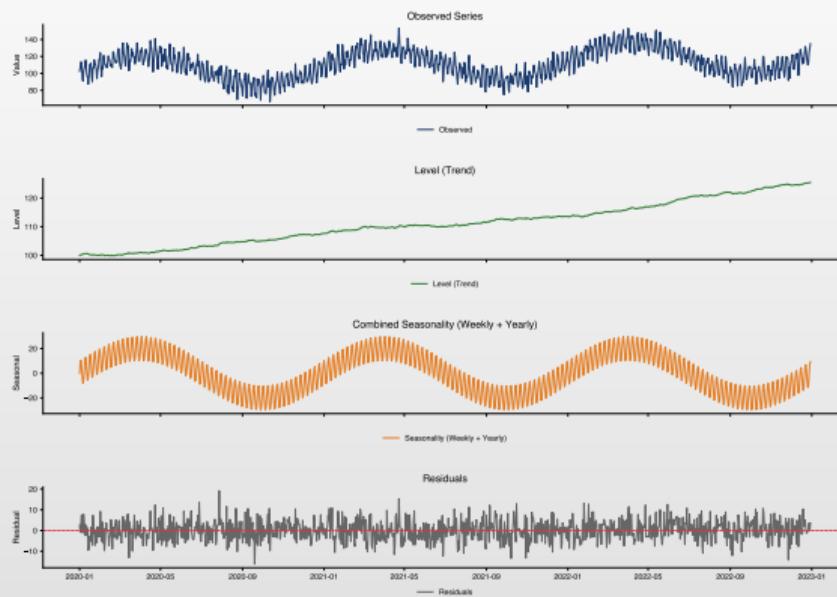
- Gestioneză **multiple** perioade sezoniere
- Selectie **automată** a modelului
- Gestioneză perioade **non-întregi** (365.25)
- Box-Cox** pentru heteroscedasticitate
- Bun pentru date de **frecvență înaltă**

Limitări

- Intensiv computațional**
- Fără **regresori externi**
- Mai puțin **interpretabil** decât Prophet
- Poate fi **lent** pentru serii foarte lungi
- Necesită **suficiente date** per sezon



Exemplu descompunere TBATS



 TSA_ch9_tbats_decomposition



Prophet: prezentare generală

Ce este Prophet?

- **Origine:** Procedură de prognoză dezvoltată de Facebook (Meta) în 2017
- **Proiectat pentru serii de timp de business cu:**
 - ▶ Efecte sezoniere puternice (zilnice, săptămânale, anuale)
 - ▶ Efecte de sărbători
 - ▶ Schimbări de trend (changepoints)
 - ▶ Date lipsă și outlieri

Filosofia Cheie

- *“Analyst-in-the-loop” forecasting*
- Prophet este proiectat pentru a fi ajustat de analiști cu cunoștințe de domeniu, dar care nu sunt neapărat experți în serii de timp



Structura modelului Prophet

Abordare prin Descompunere

- Prophet folosește o **descompunere aditivă**:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t$$

$g(t)$: Trend

- Liniar sau logistic
- Changepoints automate
- Saturație de creștere

$s(t)$: Sezonalitate

- Serii Fourier
- Perioade multiple
- Sezonialitate custom

$h(t)$: Sărbători

- Sărbători pe țară
- Evenimente custom
- Efecte de fereastră



Prophet: componenta de trend

Trend Liniar cu Changepoints

- **Ecuăția:** $g(t) = (k + \mathbf{a}(t)^T \boldsymbol{\delta}) \cdot t + (m + \mathbf{a}(t)^T \boldsymbol{\gamma})$
- **Parametri:**
 - ▶ $k \succ$ rata de creștere de bază
 - ▶ $\boldsymbol{\delta} \succ$ vector de ajustări de rată la changepoints
 - ▶ $\mathbf{a}(t) \succ$ indică ce changepoints sunt active la momentul t
 - ▶ $m \succ$ offset-ul, $\boldsymbol{\gamma}$ asigură continuitatea

Creștere Logistică (pentru trenduri cu saturatie)

- Ecuăția de creștere logistică:

$$g(t) = \frac{C(t)}{1 + \exp(-(k + \mathbf{a}(t)^T \boldsymbol{\delta})(t - (m + \mathbf{a}(t)^T \boldsymbol{\gamma})))}$$

- $C(t)$ este capacitatea maximă (posibil variabilă în timp)



Prophet: componenta de sezonalitate

Reprezentare prin Serii Fourier

- Pentru o perioadă sezonieră P , Prophet folosește:

$$s(t) = \sum_{n=1}^N \left[a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right]$$

Setări Implicite

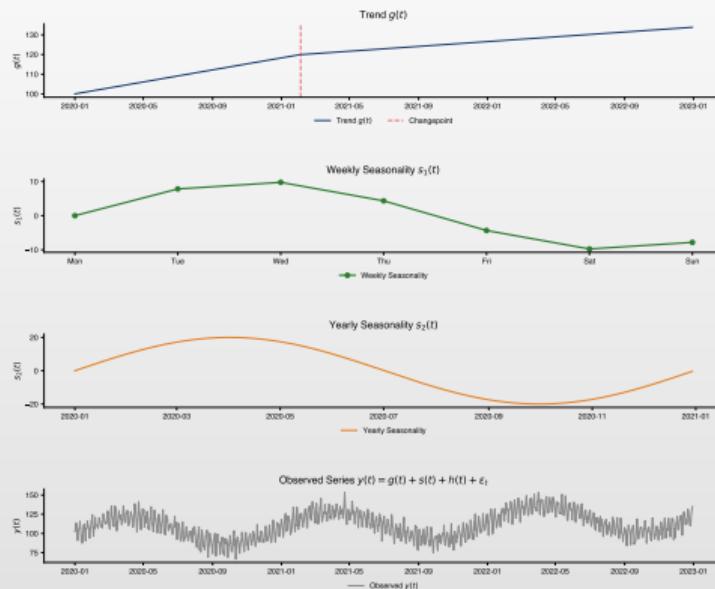
- **Anuală**: perioadă 365.25 zile, ordin Fourier 10
- **Săptămânală**: perioadă 7 zile, ordin Fourier 3
- **Zilnică**: perioadă 1 zi, ordin Fourier 4

Atenție

- Ordin Fourier N mai mare \succ mai multă flexibilitate (tipare mai complexe) dar risc mai mare de supraajustare



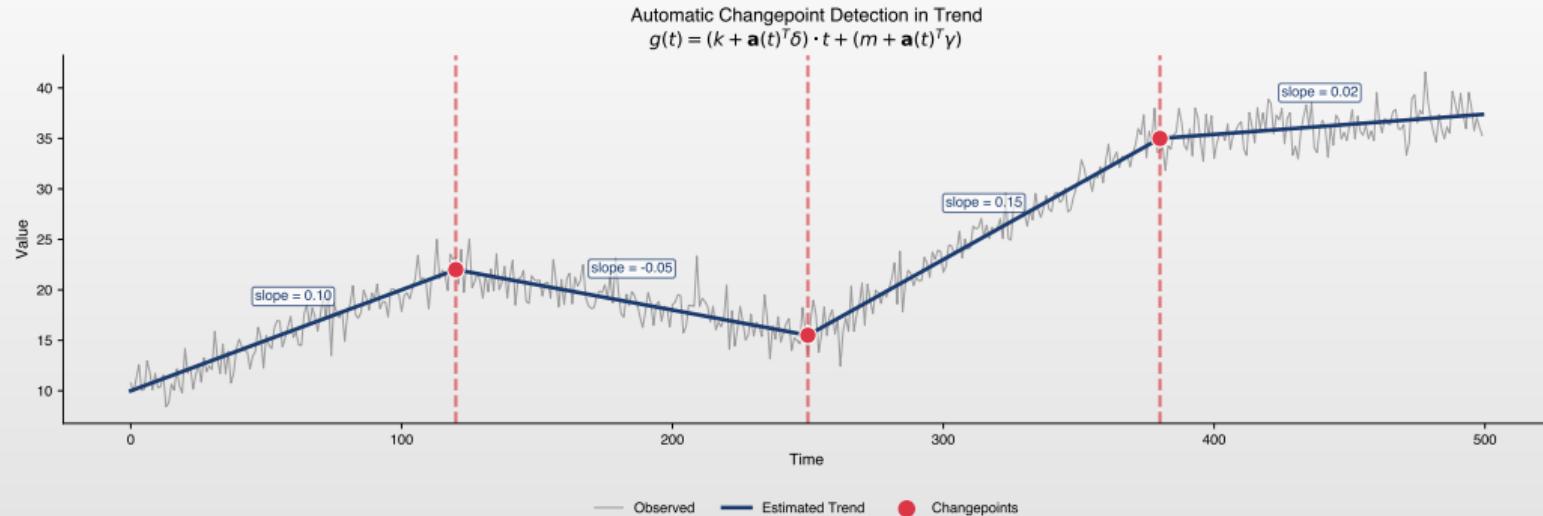
Descompunerea componentelor Prophet



 TSA_ch9_prophet_components



Detectarea changepoints în trend



Q TSA_ch9_changepoint_detection



Prophet: efecte de sărbători

Modelul de Sărbători

- Ecuatia efectelor de sărbătoare:

$$h(t) = Z(t) \cdot \kappa$$

- $Z(t)$ este o matrice indicător pentru sărbători și κ sunt efectele sărbătorilor

Caracteristici

- **Sărbători integrate:** 60+ țări suportate
- **Sărbători custom:** Adăugați propriile evenimente (Black Friday, evenimente companie)
- **Efecte de fereastră:** Sărbătorile pot afecta zilele înainte/după
- **Prior scale:** Controlează regularizarea efectelor de sărbătoare

Exemplu de Cod

- **Cod Python:**

```
holidays = pd.DataFrame({'holiday': 'black_friday', ....})  
model = Prophet(holidays=holidays)
```



Prophet în practică

Utilizare de Bază

- Cod Python:

```
from prophet import Prophet
import pandas as pd

# Datele trebuie să aibă coloane 'ds' (dată) și 'y' (valoare)
df = pd.DataFrame({'ds': dates, 'y': values})

model = Prophet()
model.fit(df)

future = model.make_future_dataframe(periods=365)
forecast = model.predict(future)
```

Adăugare Sezonalitate Custom

- Cod Python:

```
model = Prophet(weekly_seasonality=False)
model.add_seasonality(name='monthly', period=30.5, fourier_order=5)
model.add_seasonality(name='quarterly', period=91.25, fourier_order=3)
```



Prophet: cuantificarea incertitudinii

Trei Surse de Incertitudine

- Incertitudine de trend:** Changepoints viitoare sunt incerte
- Incertitudine de sezonalitate:** Incertitudine în estimarea parametrilor
- Zgomot de observație:** Aleatorietate inerentă

Intervale de Predicție

- Prophet oferă:**
 - ▶ Prognoză punctuală: `yhat`
 - ▶ Limita inferioară: `yhat_lower`
 - ▶ Limita superioară: `yhat_upper`
- Implicit:** interval de 80%, schimbați cu `interval_width=0.95`

Notă

- Incertitudinea crește cu orizontul de prognoză, în special pentru incertitudinea de trend



Prophet: parametri de ajustare

Parametri Cheie

- `changepoint_prior_scale`: Flexibilitate trend (implicit: 0.05)
- `seasonality_prior_scale`: Flexibilitate sezonalitate (implicit: 10)
- `holidays_prior_scale`: Mărime efect sărbători (implicit: 10)
- `seasonality_mode`: 'additive' sau 'multiplicative'
- `changepoint_range`: Portiune din istoric pentru changepoints

Sfaturi Practice

- Supraajustare pe trend?** Micșorați `changepoint_prior_scale`
- Subajustare pe sezonalitate?** Măriți `seasonality_prior_scale`
- Amplitudinea sezonieră variază?** Folosiți `seasonality_mode='multiplicative'`



Prophet: avantaje și limitări

Avantaje

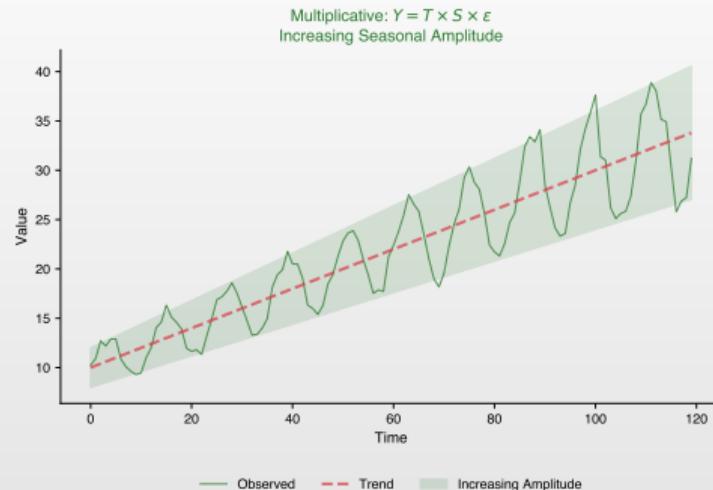
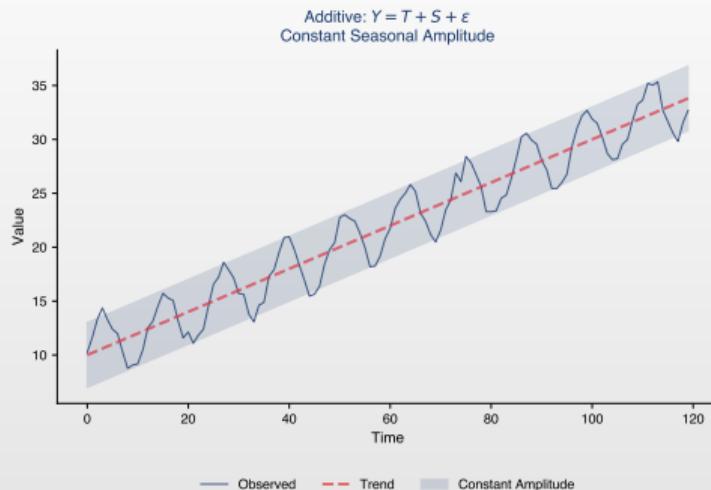
- Ușor de folosit:** Ajustare minimă necesară
- Interpretabil:** Descompunere clară
- Gestionează date lipsă bine**
- Efecte sărbători integrate**
- Sezonalități multiple**
- Regresori externi suportați**
- Ajustare rapidă**

Limitări

- Nu bazat pe ARIMA:** Fără modelare autocorelație
- Focus pe date zilnice:** Mai puțin potrivit pentru frecvență foarte înaltă
- Ipoteze de trend:** Liniar/logistic poate să nu se potrivească
- CV integrat:** `cross_validation()` disponibil, dar necesită configurare atentă
- Risc supraajustare cu multe sezonalități**



Sezonalitate aditivă vs multiplicativă



Q TSA_ch9_additive_vs_multiplicative

TBATS vs Prophet: comparație directă

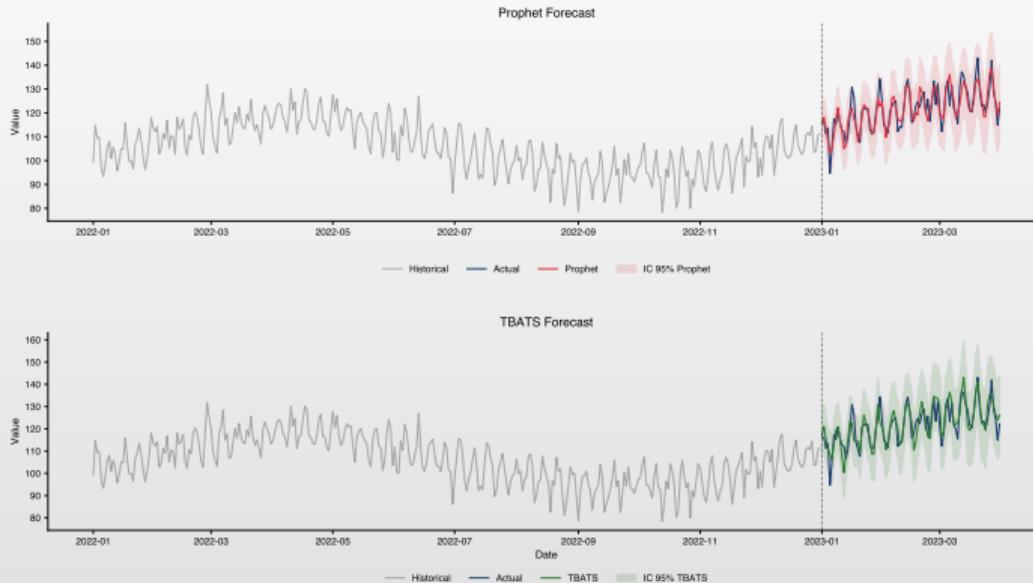
Comparație detaliată

Rezumat al diferențelor cheie:

Caracteristică	TBATS	Prophet
Sezonalități multiple	Da (automat)	Da (manual sau auto)
Efecte sărbători	Nu	Da (integrat)
Regresori externi	Nu	Da
Changepoints trend	Nu (neted)	Da (automat)
Date lipsă	Necesită interpolare	Gestioneză nativ
Interpretabilitate	Moderată	Înaltă
Viteză calcul	Lent	Rapid
Date frecvență înaltă	Bun	Moderat
Perioade non-întregi	Da (ex: 365.25)	Da
Intervale incertitudine	Da	Da



Comparație Prophet vs TBATS: programe



Q TSA_ch9_prophet_vs_tbats



Când să folosim fiecare model

Folosiți TBATS când:

- Date de frecvență înaltă (orare, sub-zilnice)
- Multiple perioade sezoniere complexe
- Nu sunt necesari regresori externi
- Se preferă selecție automată a modelului
- Se dorește framework tradițional state-space

Folosiți Prophet când:

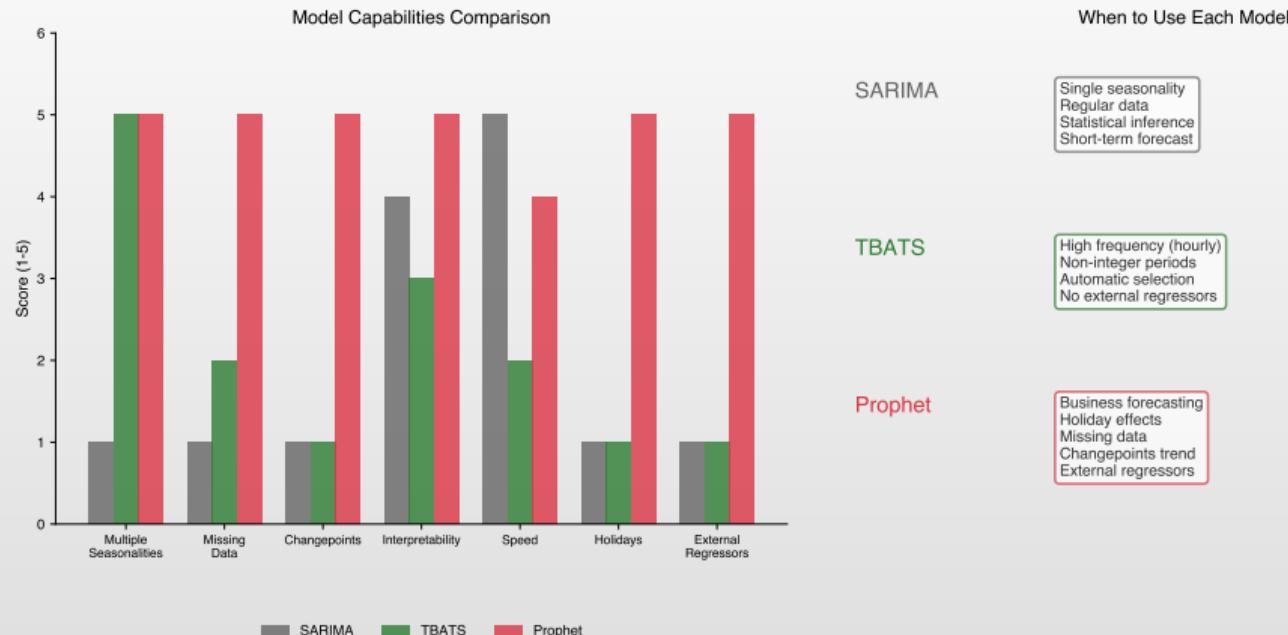
- Prognoză de business (zilnic/săptămânal)
- Efectele sărbătorilor sunt importante
- Trendul are rupturi structurale
- Sunt prezente date lipsă
- Interpretabilitatea este cheie
- Sunt disponibili regresori externi

Ghid General

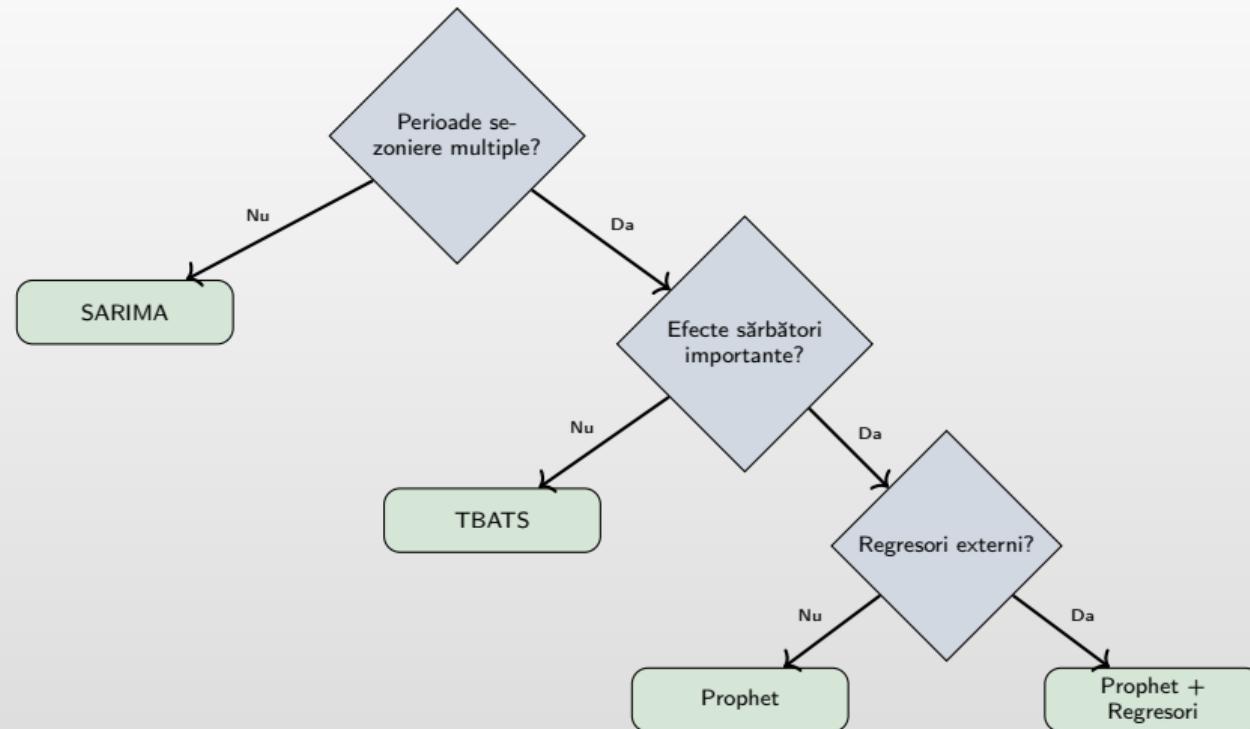
- Prophet:** pentru aplicații de business cu date zilnice
- TBATS:** pentru aplicații tehnice cu date de frecvență înaltă



Ghid selecție model



Diagramă de decizie



Metrici de evaluare

Definiție 3 (Metrici de acuratețe a prognozei)

Fie y_t valorile reale, \hat{y}_t prognozele și n orizontul de prognoză:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2} \quad (\text{penalizează erorile mari})$$

$$\text{MAE} = \frac{1}{n} \sum_{t=1}^n |y_t - \hat{y}_t| \quad (\text{robust la outlieri})$$

$$\text{MAPE} = \frac{100\%}{n} \sum_{t=1}^n \left| \frac{y_t - \hat{y}_t}{y_t} \right| \quad (\text{independent de scară})$$

Acoperire

Pentru intervalele de predicție $[\hat{y}_t^L, \hat{y}_t^U]$, rata de acoperire este proporția valorilor reale care se încadrează în interval. Tintă: să corespundă nivelului nominal (ex: 80%).



Studiu de caz: prognoza cererii de energie

Problema

- Obiectiv:** Prognozați cererea de electricitate pe oră
- Provocări:**
 - ▶ Tipar zilnic \succ vârf la prânz și seara
 - ▶ Tipar săptămânal \succ mai scăzut în weekend
 - ▶ Tipar anual \succ mai mare vara (AC) și iarna (încălzire)
 - ▶ Efecte sărbători \succ cerere mai mică în sărbători

Abordare

- Pas 1:** Încercați TBATS cu perioade [24, 168, 8766]
- Pas 2:** Încercați Prophet cu sezonalitate zilnică, săptămânală, anuală + sărbători
- Pas 3:** Comparați folosind cross-validation



Studiu de caz: interpretarea rezultatelor

Metrici de Evaluare

- MAPE**: Mean Absolute Percentage Error
- RMSE**: Root Mean Square Error
- Acoperire**: % din valori reale în intervalul de predicție

Rezultate Tipice

- Comparație performanță:

Model	MAPE	RMSE	Acoperire
SARIMA (doar zilnic)	8.5%	450 MW	75%
TBATS	4.2%	220 MW	82%
Prophet	4.8%	250 MW	85%
Prophet + sărbători	3.9%	200 MW	88%

Concluzie

- Modelele cu sezonalități multiple depășesc semnificativ SARIMA cu o singură sezonalitate



Exercițiu AI: Gândire critică

Prompt de testat în ChatGPT / Claude / Copilot

"Am 3 ani de date orare de consum de energie. Folosește Facebook Prophet pentru a prognoza săptămâna viitoare. Include sărbătorile și evenimentele speciale. Vreau cod Python complet."

Exercițiu:

1. Rulați prompt-ul într-un LLM la alegere și analizați critic răspunsul.
2. Prophet detectează automat sezonalitățile multiple (zilnică, săptămânală)?
3. Cum sunt specificate sărbătorile? Specifică țări sau evenimente personalizate?
4. Folosește cross-validation cu puncte de referință (performance _metrics)?
5. TBATS ar fi mai potrivit pentru această frecvență? De ce sau de ce nu?

Atenție: Codul generat de AI poate rula fără erori și arăta profesional. *Asta nu înseamnă că e corect.*



Concluzii cheie

Sezonalități Multiple

- Datele din lumea reală au adesea tipare sezoniere multiple
- SARIMA standard gestionează doar o perioadă sezonieră
- TBATS și Prophet sunt proiectate pentru această provocare

Selectia modelului

- **TBATS:** Automat, gestionează frecvență înaltă, fără regresori externi
- **Prophet:** Interpretabil, efecte sărbători, regresori externi
- Ambele folosesc termeni Fourier pentru reprezentare eficientă a sezonalității

De Reținut

- **Validare:** Folosiți întotdeauna cross-validation adecvat pentru serii de timp!



Întrebări?

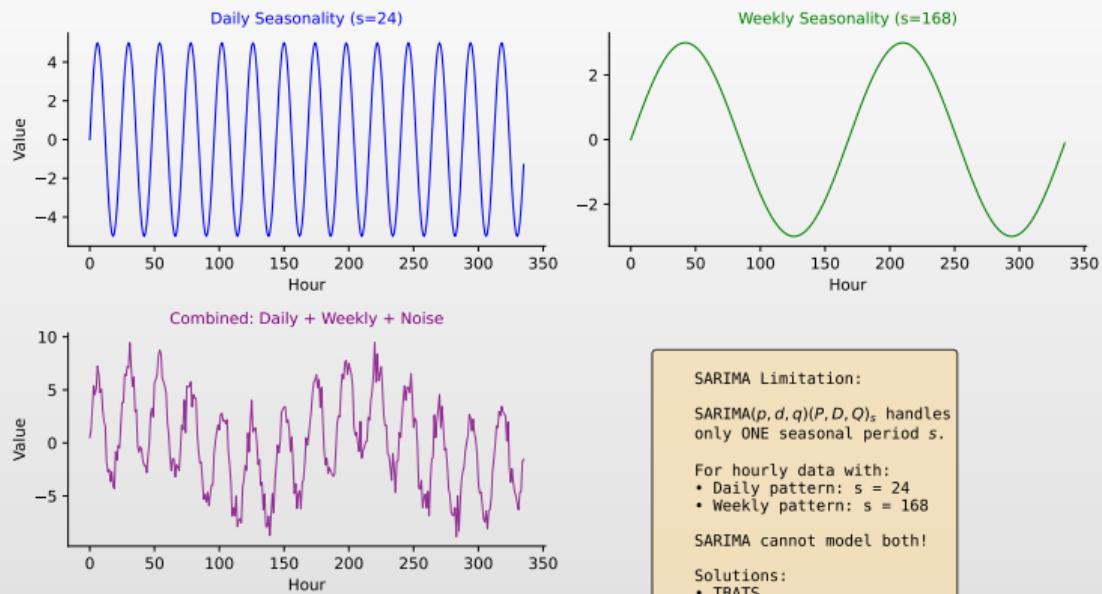
Întrebări?

Pași Următori

- Exersați cu notebook-ul Jupyter
- Încercați Prophet pe propriile date
- Explorați NeuralProphet pentru extensia deep learning



Quiz 1: Sezonalitate multiplă



Întrebare: De ce nu poate SARIMA(p, d, q)(P, D, Q) $_s$ standard să modeleze date orare de electricitate cu tipare zilnice și săptămânale?

Analiza și Prognoza Seriilor de Timp

Răspuns: SARIMA gestionează doar o singură perioadă sezonieră s . Nu se pot seta $s = 24$ și $s = 168$ simultan.

Quiz 2: Componentele TBATS

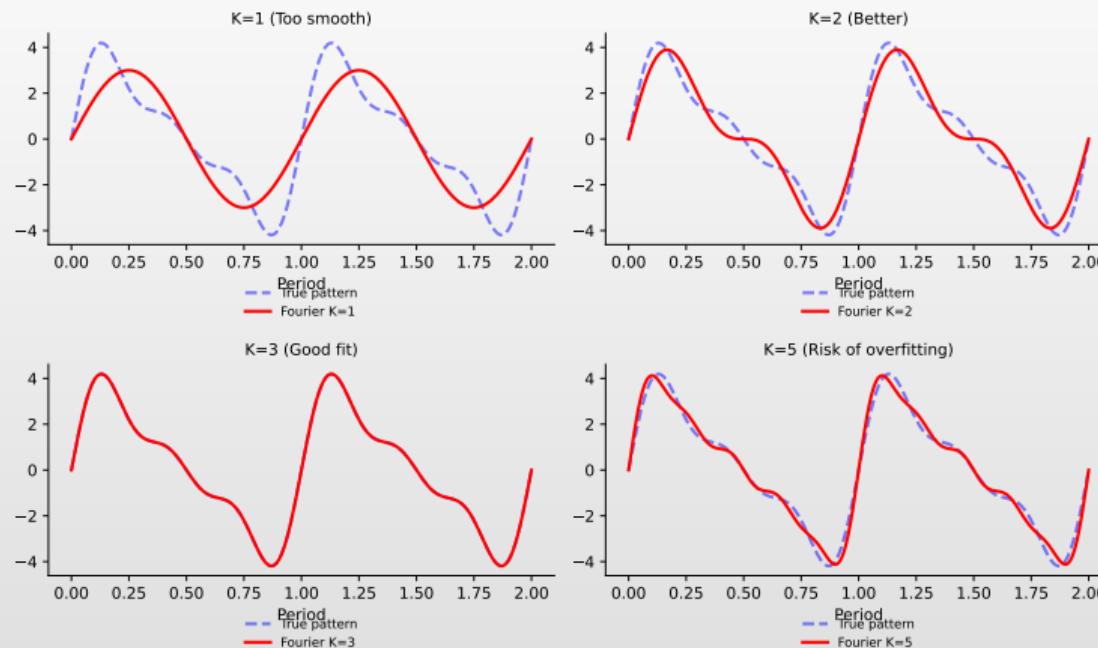
TBATS: What Does It Stand For?

T	Trigonometric	Fourier terms for seasonality $\sum[a_n \cos(\frac{2\pi n t}{m}) + b_n \sin(\frac{2\pi n t}{m})]$
B	Box-Cox	Variance stabilization $y^{(\omega)} = (y^\omega - 1)/\omega$
A	ARMA	Error autocorrelation $\phi(L)d_t = \theta(L)\varepsilon_t$
T	Trend	Level + slope (possibly damped) $\ell_t = \ell_{t-1} + \phi b_{t-1}$
S	Seasonal	Multiple seasonal periods m_1, m_2, \dots, m_T

Întrebare: Ce reprezintă fiecare literă din TBATS?

Răspuns: Sezonialitate Trigonometrică, transformare Box-Cox, erori ARMA, Trend, componente Sezoniere.

Quiz 3: Armonici Fourier

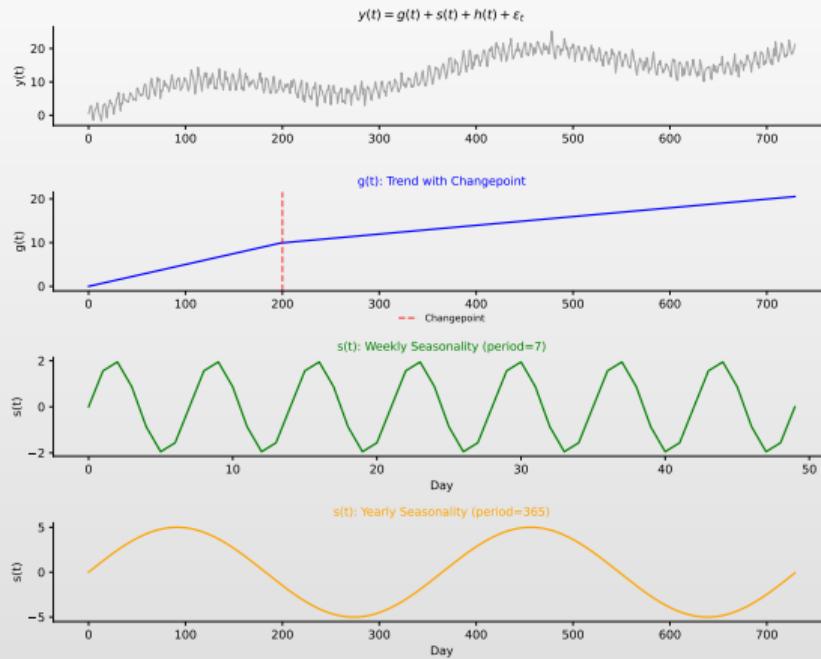


Întrebare: Ce se întâmplă când creștem numărul de armonici Fourier K ?

Răspuns: Un K mai mare captează tipare mai complexe, dar crește riscul de supraajustare.



Quiz 4: Descompunerea Prophet



Întrebare: Care sunt componentele principale în modelul Prophet $y(t) = g(t) + s(t) + h(t) + \varepsilon_t$?

Răspuns: $g(t)$ = trend cu changepoints, $s(t)$ = sezonalitate, $h(t)$ = efecte de sărbători.



Quiz 5: Comparație modele

TBATS vs Prophet: Head-to-Head Comparison

Feature	TBATS	Prophet
Multiple seasonalities	Yes (automatic)	Yes (manual/auto)
Holiday effects	No	Yes (built-in)
External regressors	No	Yes
Trend changepoints	No (smooth)	Yes (automatic)
Missing data	Needs interpolation	Handles natively
Interpretability	Moderate	High
Computation speed	Slow	Fast
High-frequency data	Good	Moderate
Non-integer periods	Yes (e.g., 365.25)	Yes
Best for	Technical/high-freq	Business/daily

Întrebare: Ce caracteristici cheie are Prophet pe care TBATS nu le are?

Răspuns: Efecte de sărbători, regresori externi, changepoints în trend, gestionare nativă a datelor lipsă.



Bibliografie I

Prophet

- Taylor, S.J., & Letham, B. (2018). Forecasting at Scale, *The American Statistician*, 72(1), 37–45.
- Harvey, A.C. (1989). *Forecasting, Structural Time Series Models and the Kalman Filter*, Cambridge University Press.

TBATS și netezire exponențială

- De Livera, A.M., Hyndman, R.J., & Snyder, R.D. (2011). Forecasting Time Series with Complex Seasonal Patterns Using Exponential Smoothing, *JASA*, 106(496), 1513–1527.
- Hyndman, R.J., Koehler, A.B., Ord, J.K., & Snyder, R.D. (2008). *Forecasting with Exponential Smoothing: The State Space Approach*, Springer.
- Taylor, J.W. (2003). Short-term Electricity Demand Forecasting Using Double Seasonal Exponential Smoothing, *Journal of the Operational Research Society*, 54(8), 799–805.

Bibliografie II

Comparații și competiții de prognoză

- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition, *International Journal of Forecasting*, 36(1), 54–74.
- Hyndman, R.J., & Athanasopoulos, G. (2021). *Forecasting: Principles and Practice*, 3rd ed., OTexts.
- Petropoulos, F., et al. (2022). Forecasting: Theory and Practice, *International Journal of Forecasting*, 38(3), 845–1054.

Resurse online și cod

- **Quantlet:** <https://quantlet.com> ➔ Depozit de cod pentru statistică
- **Quantinar:** <https://quantinar.com> ➔ Platformă de învățare metode cantitative
- **GitHub TSA:** <https://github.com/QuantLet/TSA> ➔ Cod Python pentru acest curs



Vă Mulțumim!

Întrebări?

Materialele cursului sunt disponibile la: <https://danpele.github.io/Time-Series-Analysis/>

