BlueYonder
Fulfill your potential™

# Cyclic Boosting

A Pure-Python, Explainable, and Efficient ML Method

Felix Wick, April 2023

# This is old stuff.

# But it just works.

# Cyclic Boosting ML Methods

**family of off-the-shelf, general-purpose supervised machine learning methods for both regression and classification tasks (focus on structured data)**

**closest relatives: Generalized Additive Models (not a deep learning approach)**

**main difference: estimation of factors for each bin of the different features (instead of estimation of parameters like coefficients in linear regression or weights in neural networks)**

**→ individual explainability**

**scientific papers describing the methods: Cyclic Boosting, Demand Forecasting with Cyclic Boosting**

# Cyclic Boosting Library

**scikit-learn-like usage of library**

**open source: https://github.com/Blue-Yonder-OSS/cyclic-boosting**

**Python package: pypi**

**documentation: readthedocs**

# Cyclic Boosting | Different Modes/Scenarios


(conditional mean)

$Y \in [0, \infty)$

Poisson / Negative Binomial distribution (link function *ln*)

*example*

demand forecasts (mean)


(conditional mean)

$Y \in (-\infty, \infty)$

Gaussian distribution (link function *identity*)

*example*

profit predictions

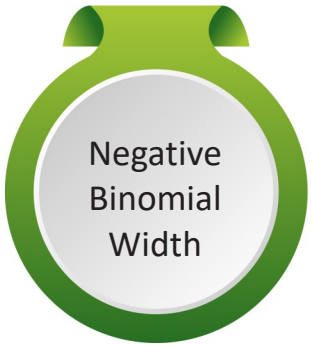

(probability)

$Y \in [0, 1]$

Bernoulli distribution (link function *logit*)

*example*

churn probability




(dispersion parameter)

$Y \in [0, 1]$

Negative Binomial distribution (link function *logit*)

*example*

demand forecasts as full probability distributions

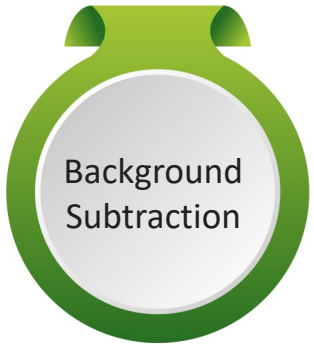

(elasticity parameter)

$Y \in [0, \infty)$

exponential distribution (link function *ln*)

*example*

individual price-demand elasticities




(conditional mean)

$Y \in (-\infty, \infty)$

Gaussian distribution (link function *identity*)

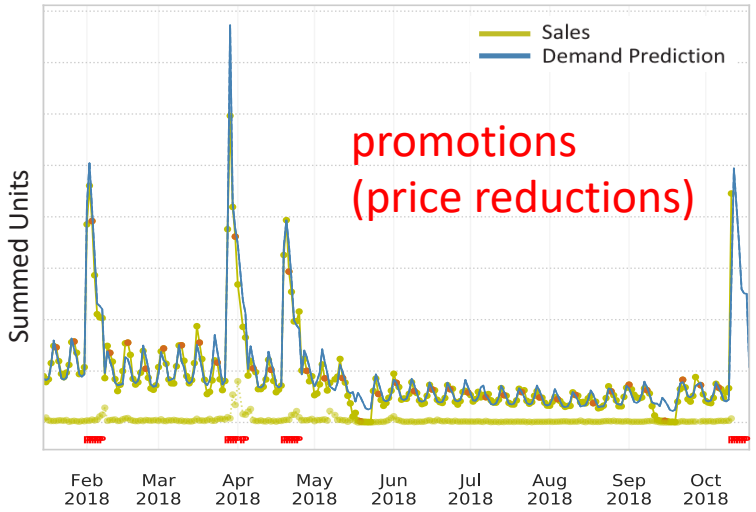
*example*

individual causal effects, e.g., customer targeting

# Example Use Case: Demand Forecasting

**many individual time series to consider**

typical retail grocery chain:
- products (items): ~20k
- locations (stores): ~500
- daily/hourly aggregated sales




**advantages of machine learning over traditional univariate time series forecasting**

**combined learning on all time series** of product-location combinations (rather than separately optimizing individual time series)
→ **reduces variance** by exploiting commonalities

**natural consideration of many exogenous variables** (prices, promotions, holidays, weather, ...)
→ **reduces bias**

to be noted:
- categorical features important (products and locations → high cardinality)
- mainly multiplicative effects
- demand (approximately) following Poisson (or rather negative binomial) distribution

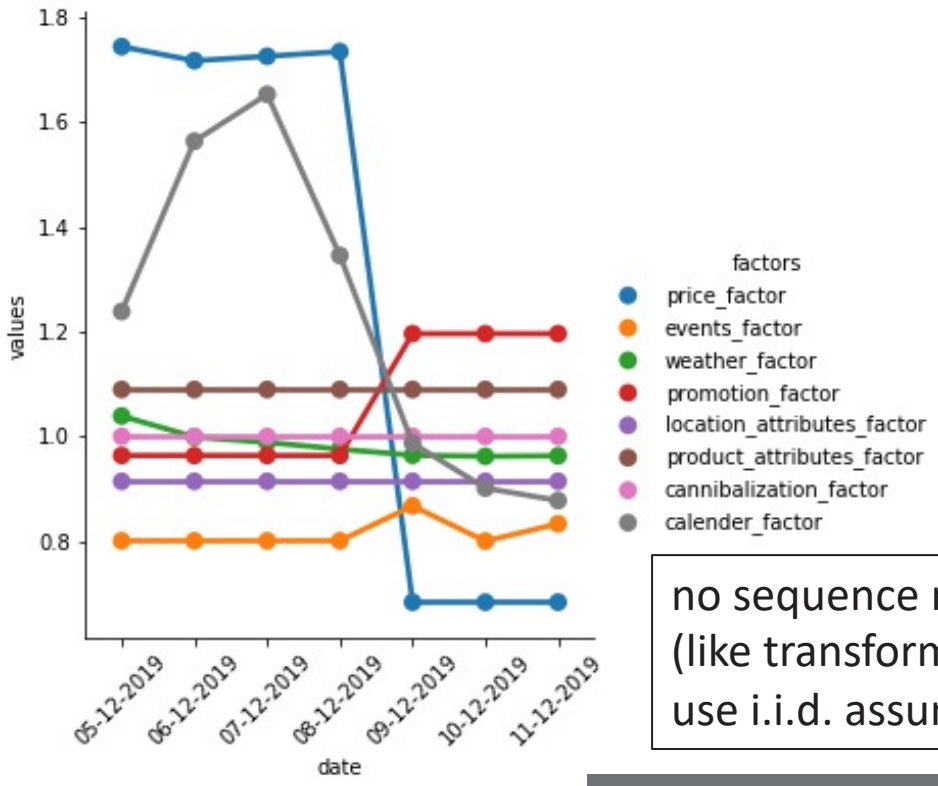
# Cyclic Boosting - Prediction View | Individual Explainability

Cyclic Boosting in multiplicative regression mode

**multiplicative model**

variation proportional to level

**individual item-store-day predictions**




no sequence model (like transformers), use i.i.d. assumption

Cyclic Boosting allows for detailed explanation of each individual prediction by means of contributions (in form of factors) of each feature in the model.

prediction: look up learned factors of relevant bin for each feature

$$\hat{y}_i = \mu \cdot \prod_{j=1}^{p} f_j^k \quad \text{with} \quad k = \{x_{j,i} \in b_j^k\}$$

- $\mu$: global target average
- $\prod_{j=1}^{p} f_j^k$: product over factors for all $p$ features in corresponding bins of sample $i$
- $k = \{x_{j,i} \in b_j^k\}$: bin $k$ of feature $j$

data binning of features (think of histograms)

do not confuse explainability with causality though
→ need for causal assumptions (e.g., specific smoothing)

# Cyclic Boosting Training

## Coordinate Descent: Boosting-like Update of Factors

Cyclic Boosting fitting procedure with three loops

```
while (stop criteria)        iterations
…
    for (features)           sequential
    …
        for (samples)        parallel
```

1. calculate global average μ, initialize all factors to 1

2. cyclically iterate through features and calculate factors for each feature bin (corresponding to minimization of quadratic loss)

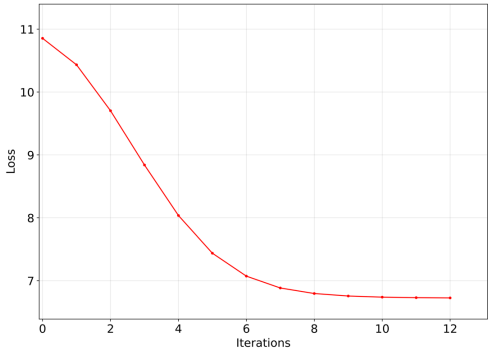
multiplicative regression mode (other modes work accordingly)

for simplicity: show only non-aggregated mode

$$f_j^k = \sum_{x_{j,i} \in b_j^k} y_i \Big/ \sum_{x_{j,i} \in b_j^k} \hat{y}_{j,i} \qquad \hat{y}_{j,i} = \mu \cdot \prod_{l \neq j} f_l^k$$

- $f_j^k$: bin $k$ of feature $j$
- $\sum_{x_{j,i} \in b_j^k}$: sum over all samples $i$ in bin $k$ of feature $j$
- $y_i$: numerator: target values
- $\hat{y}_{j,i}$: denominator: predictions excluding factor from current feature
- $\prod_{l \neq j} f_l^k$: product over all features excluding $j$; factors for corresponding feature bins of sample $i$

3. stop according to MAD or MSE criteria at end of iterations (full feature cycles) or when reaching given maximal number of iterations

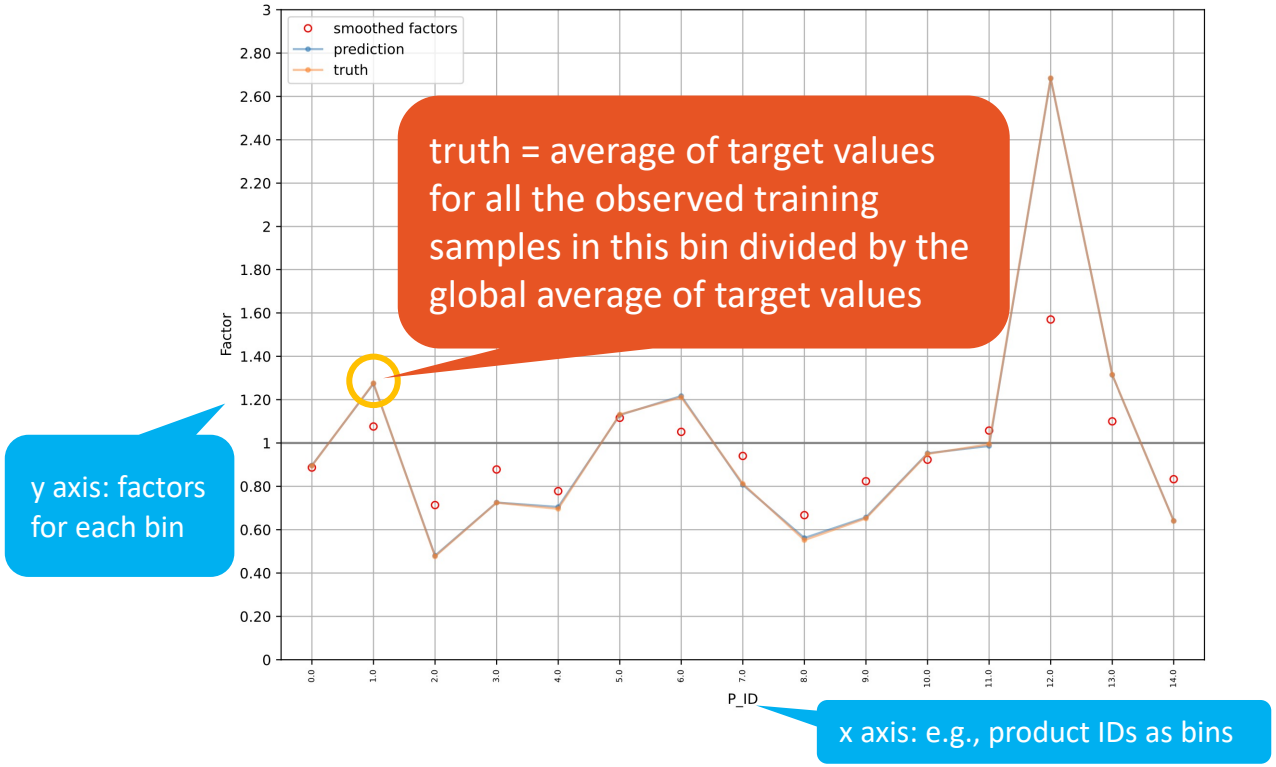


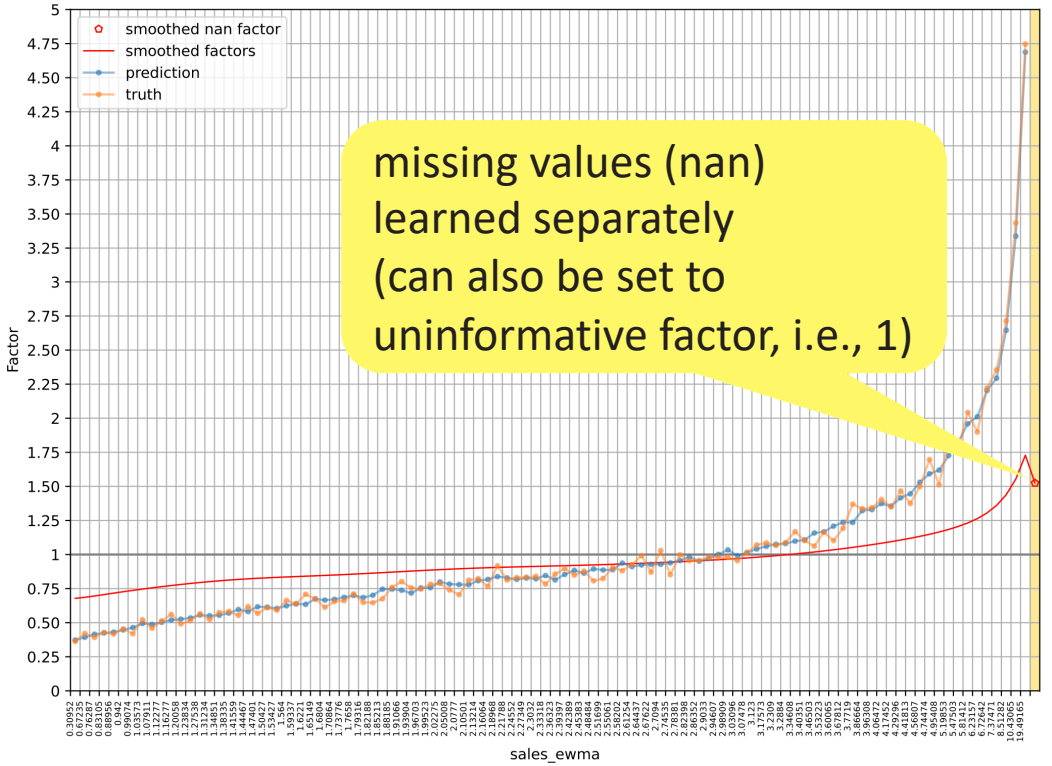
# Binning

categorical features retain original categories (learning of specific factor for each of the bins)
→ supporting categorical features with high cardinality

continuous features discretized to:

- either having same bin width (equidistant binning)
- or containing approximately same number of observations (equistatistics binning) with different bin widths




local optimization in each bin: allows learning of rare effects with low bias

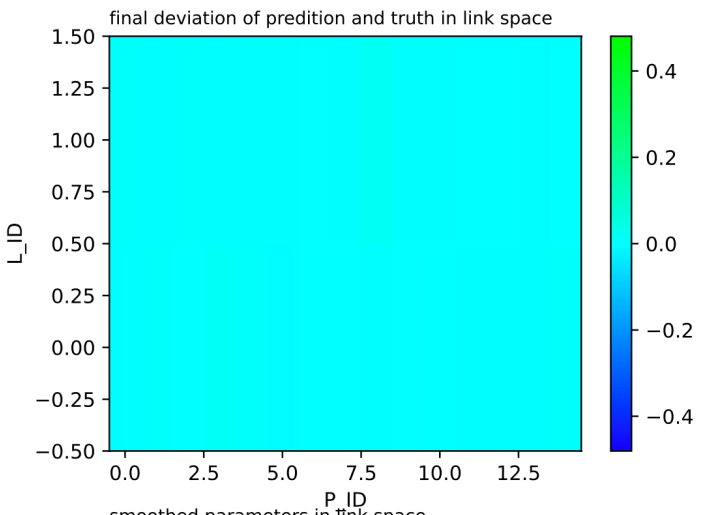
# Interaction Terms

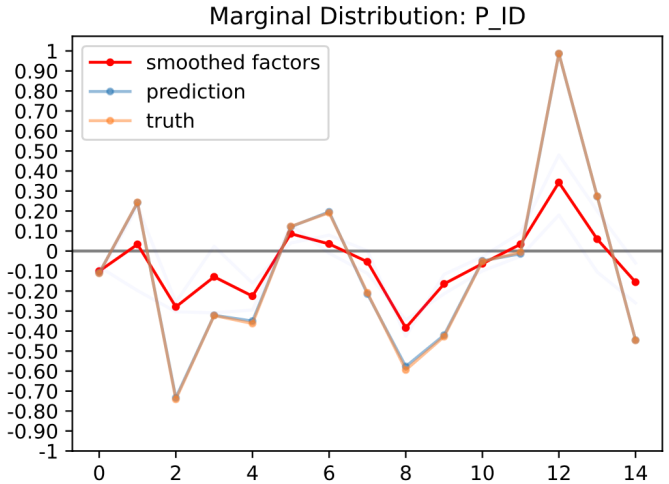
e.g., different holiday effects for different products

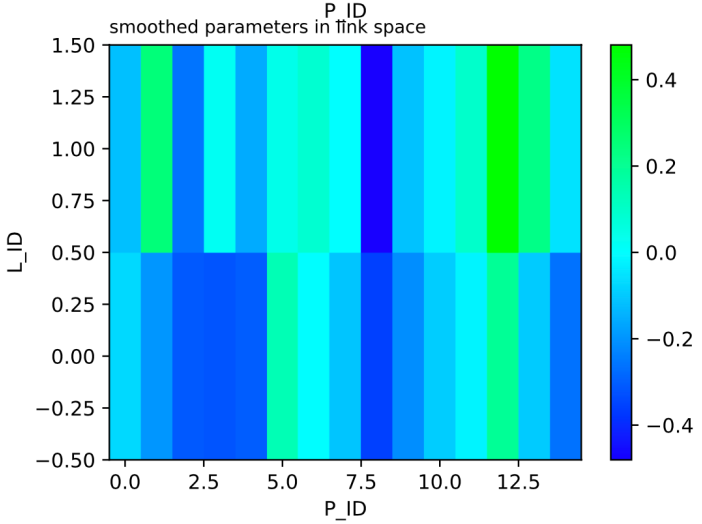
require even more local optimization (rare effects)

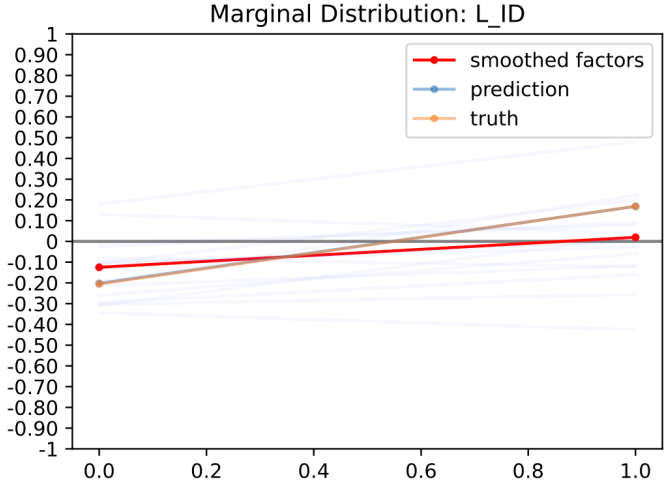
→ include binned interaction terms (e.g., 2D or 3D)

can (partly) enable hierarchical model structure (in combination with coordinate descent): interaction terms with product groups, products, locations



# Smoothing

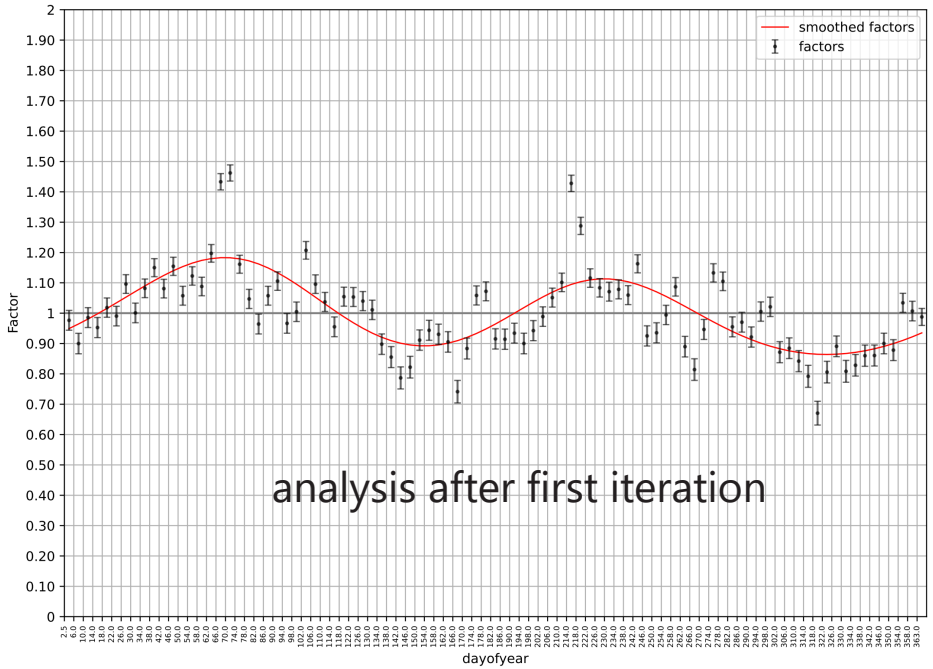
to avoid overfitting:
regularization (smoothing) across bins
→ **drastic reduction of variance** by ignoring fluctuations

separate smoothings in each iteration

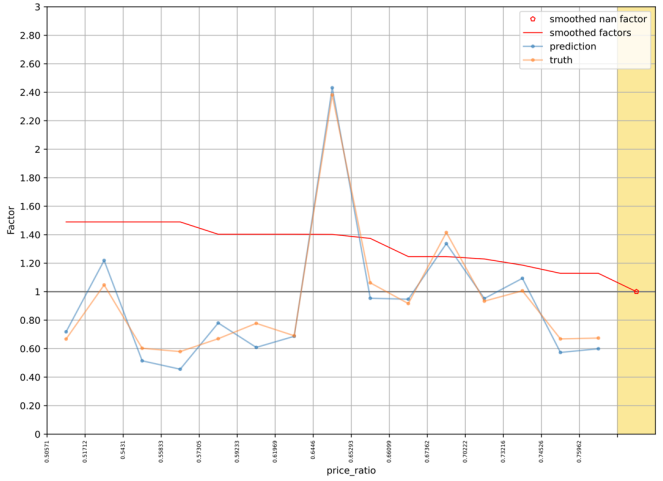
in general: orthogonal polynomials

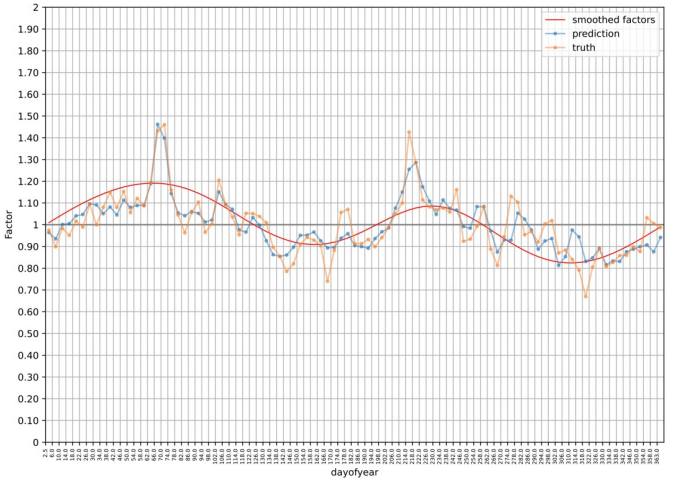
another way to include *prior knowledge* via
- monotonic requirements
- sinusoidal functions
- (piecewise) linear

use fitted functions (smoothed factors) instead of original factors




analysis after last iteration:



# Analysis Plots

**support EDA and modeling**

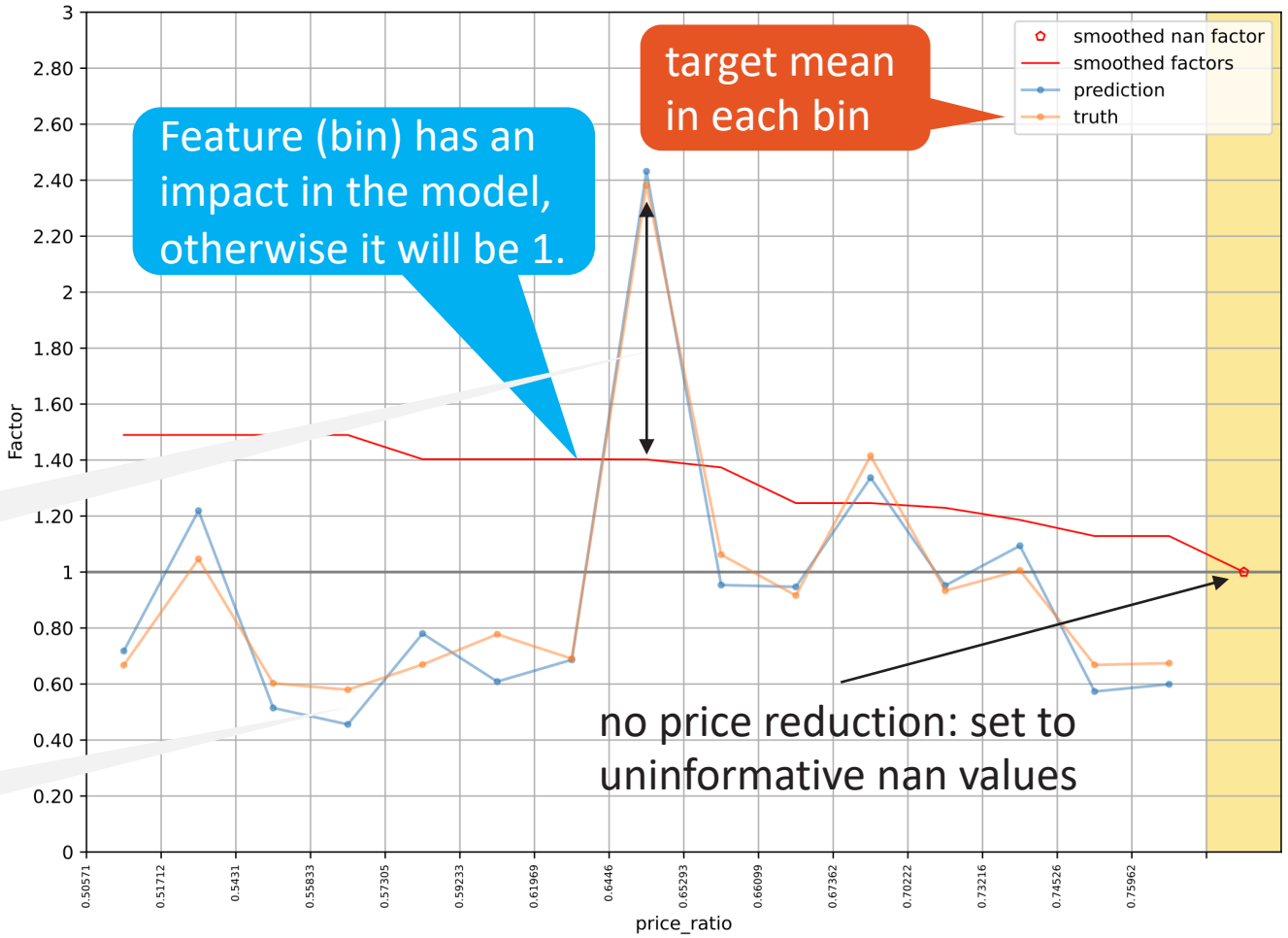
**model transparency → ease of development**

**automatically generated**

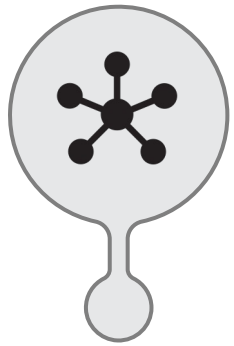
Deviations between smoothed factors and predictions come from correlations with other features.

Deviations of predictions from truths show potential model weaknesses (biases) in different bins.

analysis after last iteration:

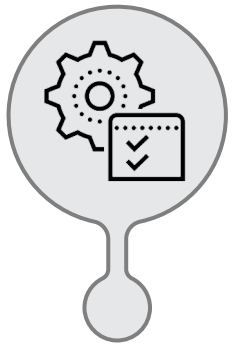


# Cyclic Boosting | Characteristics



**TOPOLOGY**

Generalized Additive Model

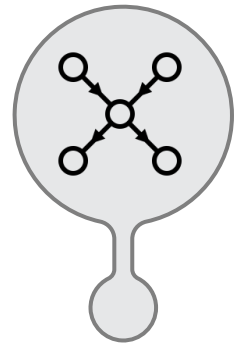


**OPTIMIZATION**

cyclic coordinate descent

**PREREQUISITE**

binning of features



**REGULARIZATION**

Bayesian updates and smoothing of factor distributions

| | | |
|---|---|---|
| similarities to: | backfitting, forward-stagewise modeling (aka boosting) | LightGBM |

# Off-The-Shelf Method for Structured Data

- “simple” algorithm, but robust and fast
- few hyperparameters to be tuned
- not much data pre-processing needed
- easily configurable for different data types
- supporting missing values in input data
- assisting model development with individual analysis plots for features
- allowing buiding of complex models by means of interaction terms

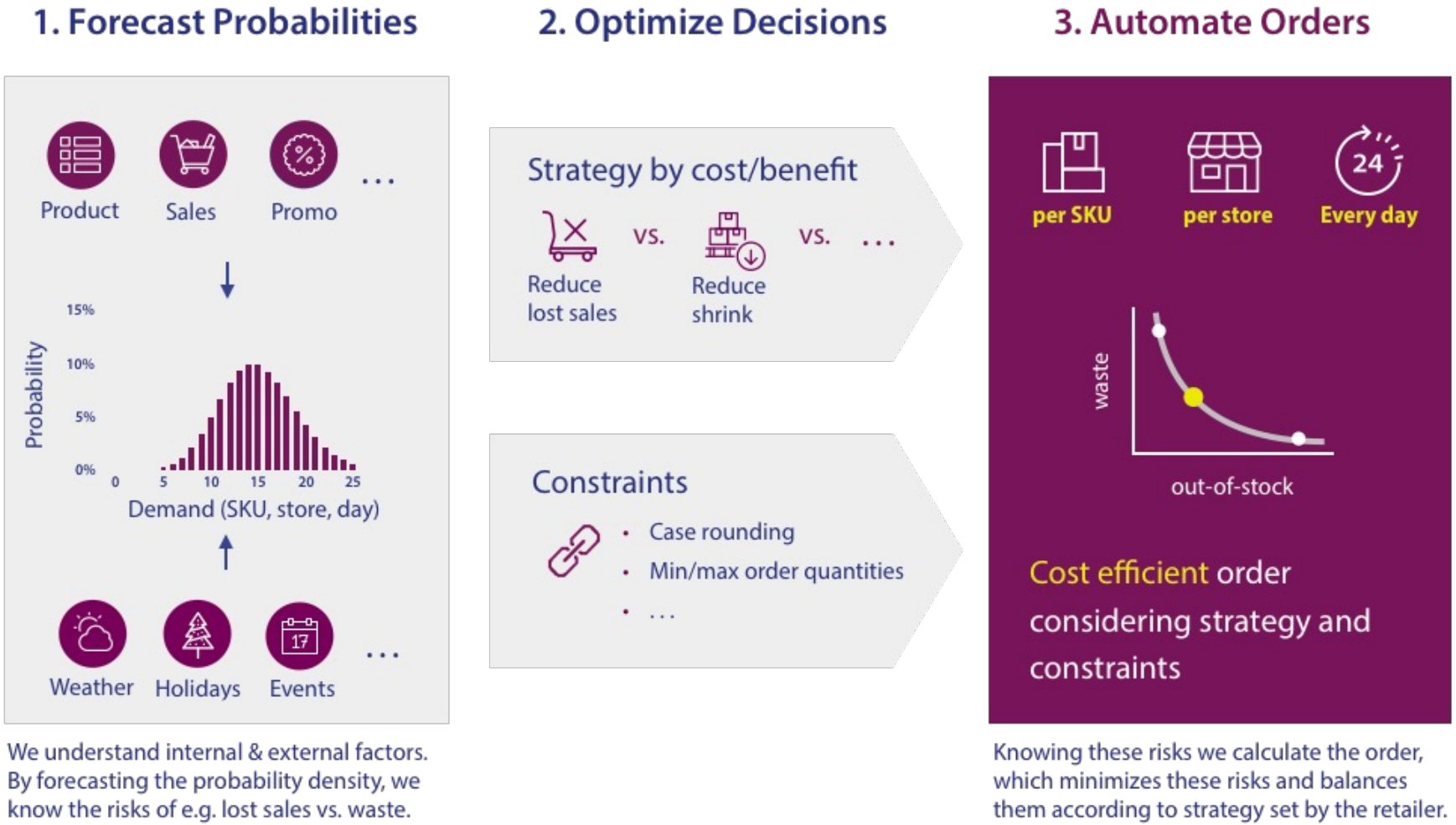
- (multiplicative or additive) regression (location parameter)
- classification

# Other Modes: Width Prediction (Scale)

important busines application: automated replenishment

**full, individual PDF predictions (e.g., probability distributions for each product-location-day combination)**

**by means of separate ML models for mean and variance (actually, indirect prediction of variance via dispersion parameter), assuming negative binomial distribution of target (e.g., demand) in maximum likelihood estimation**



using factors in exponential terms

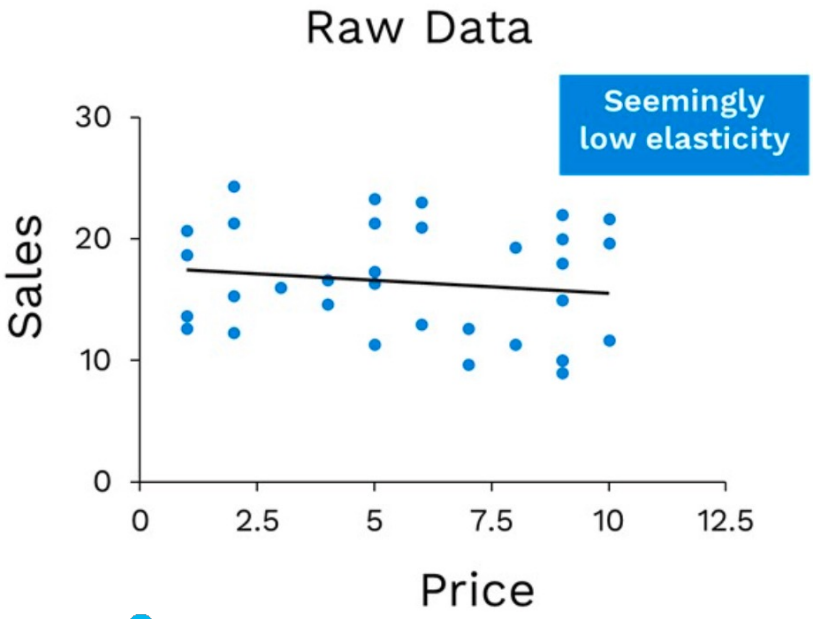
using negative sample weights

# Other Modes: Elasticity Prediction (Shape), Background Subtraction

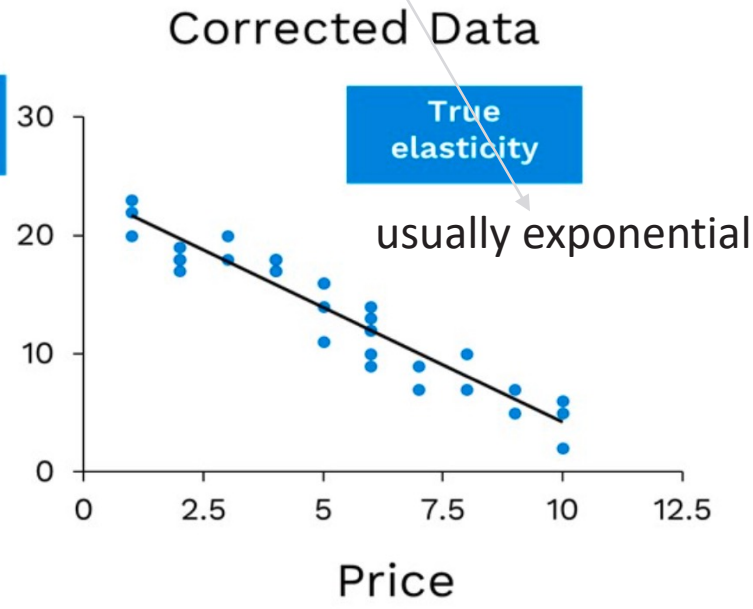
**other business application: demand shaping by causal inference (mainly beyond ML/CB), examples:**

- **dynamic pricing: influence demand of different products by price setting**
- **customer targeting: influence individual customer demand by couponing (subtract unaffected customers)**

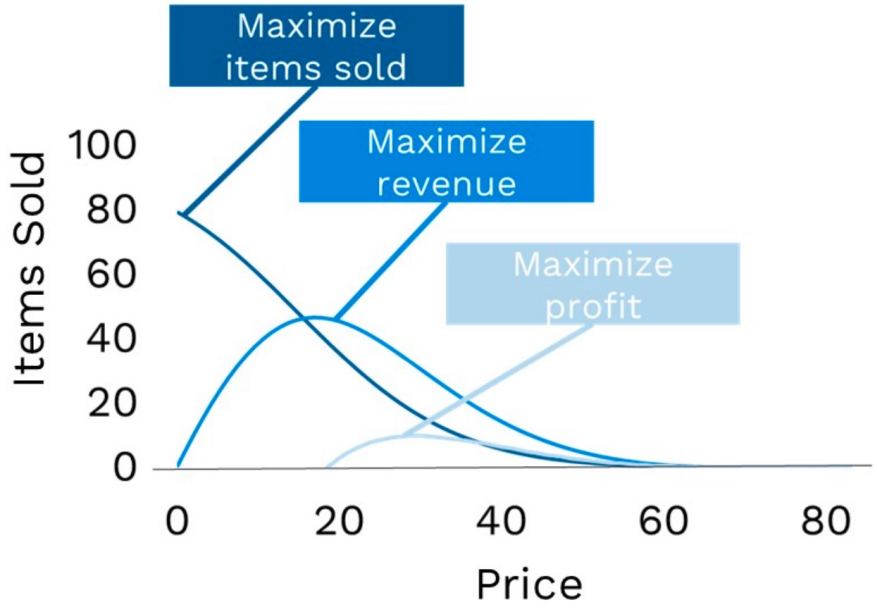
confounded effect:




after de-confounding:




use for pricing policies:



# Now Try It Yourself …

**hackathon:**

**https://www.kaggle.com/competitions/blueyonder-pyconpydata-2023**

**(Please feel free to come to our Blue Yonder booth if you have any questions or just want to chat.)**

**task: retail demand forecasting**

**You can also try to beat Cyclic Boosting with other methods like Gradient Boosting or a transformer.**


BlueYonder
Fulfill your potential™