Boosting algorithms for inflation forecasts

Matheus Vizzotto dos Santos - Universidade Federal do Rio Grande do Sul

Abstract

Boosting-based techniques have proved to attain high performance in classification tasks. A similar approach in the field of time series also offers greater accuracy and robustness compared to traditional models in areas such as healthcare, environmental sciences, finance and economics. This study evaluates the application of the AdaBoost, XGBoost, LightGBM, and CatBoost algorithms in forecasting Brazilian inflation. For this purpose, lags of the IPCA series and other monthly macroeconomic variables were used. The optimal hyperparameters were obtained using the Bayesian method in cross-validation. The results indicate a better forecasting ability for the CatBoost and XGBoost models compared to the others.

Keywords: forecast; time series; inflation; Brazil; boosting.

1 Introduction

Time series forecasting plays a fundamental role in the analysis and projection of economic variables, being especially relevant for the monitoring and forecasting of inflation in Brazil. Traditional models, such as ARIMA (Box and Jenkins, 1970) and Exponential Smoothing (Holt, 1957), although widely used, present limitations when it comes to capturing nonlinear patterns and complex interactions among the variables that influence the series of interest. However, machine learning-based methods have emerged as effective alternatives, offering greater flexibility and accuracy in forecasts. Among these methods, boosting algorithms such as AdaBoost, XGBoost, LightGBM, and CatBoost have established themselves as promising approaches for forecasting economic time series, including inflation. These models employ ensemble learning techniques, combining multiple decision trees to reduce errors and improve generalization. Unlike conventional approaches, boosting methods are capable of capturing nonlinear dynamics and subtle interactions among explanatory variables, providing more accurate forecasts even in highly volatile scenarios, such as those often observed in the Brazilian economy.

Beyond predictive robustness, boosting algorithms offer advantages such as the ability to handle missing data, computational efficiency when processing large volumes of data, and the possibility of incorporating advanced hyperparameter optimization techniques. In the context of inflation, these features are crucial, given that the data often exhibit structural trends, seasonality, and exogenous shocks that affect projections. The results obtained in this study will contribute to a better understanding of the potential of these techniques in macroeconomic variables.

2 Related Work

The use of machine learning for time series forecasting has become well established among researchers and practitioners. An example is the work of Zhang et al. (1998), which discussed the results and pitfalls associated with the use of artificial neural networks available

at the time, such as the Multi-Layer Perceptron (MLP) and its variants. Among the reasons for their attractiveness were the absence of initial assumptions and the fact that they are data-driven, meaning they are capable of capturing underlying relationships without further specifications. However, the results were mixed when nonlinear models were compared to traditional ones, with nonlinear models outperforming in some contexts and traditional models in others.

Another important, more recent work is that of Voyant et al. (2017), which explored the use of machine learning methods for solar radiation forecasting. Among the techniques analyzed were those classified under supervised learning (Support Vector Regression, decision trees, K-nearest neighbor) and unsupervised learning (K-means and hierarchical clustering). The study also addressed the growing use of ensemble learning methods (boosting, bagging, and random subspace) for time series forecasting, showing positive results.

Makridakis et al. (2018), in turn, compared the performance of Holt-Winters and ARIMA models with that of nonlinear models such as MLP, Bayesian Neural Networks, Radial Basis Functions, K-Nearest Neighbor, Support Vector Regression, and regression trees. Surprisingly (or not), the results pointed to the superiority of the ARIMA model over the others. The authors highlight this result due to a frequent issue with machine learning: the overfitting of models to the training data reduces their ability to generalize to unseen horizons.

In the latest competition in the series of tournaments organized by the same authors (the M5 Competition), held in 2020, important results emerged from the forecasting of over 42,000 time series. Makridakis et al. (2022) compared different approaches, including statistical models and machine learning methods, highlighting the widespread use of the Light Gradient Boosting Machine (LightGBM) by the top competitors among more than five thousand participants, especially when combined with appropriate feature engineering. Contrary to the studies referenced above, it was concluded that nonlinear models outperformed traditional ones.

Sousa et al. (2023) also sought to evaluate the performance of boosting algorithms in time series forecasting, specifically the XGBoost, LightGBM, and CatBoost models in the retail context. After considerable effort in engineering explanatory variables for the dataset and optimizing hyperparameters based on splitting the observations into four validation sets, XGBoost achieved the lowest Weighted Absolute Percentage Error (WAPE), although LightGBM showed greater stability when forecasting over different future cycles.

In the context of forecasting economic variables, Yoon (2021) proposed implementing boosting models and Random Forest to forecast Japan's real GDP growth using quarterly data with 147 observations per variable. As in the study above, hyperparameters were selected through a cross-validation process, allowing the construction of a model with a Mean Absolute Percentage Error (MAPE) of 19.86%, also supported by the Root Mean Squared Error (RMSE) metric.

Xiang et al. (2022) explored the use of a two-layer CatBoost model for medium and long-term electricity load forecasting in China, highlighting its computational efficiency and ability to handle heterogeneous datasets. Compared to the results from XGBoost, AdaBoost, Random Forest, Bagging, and MLP models, the proposed algorithm achieved the best forecasting performance, with a MAPE of 1.58%.

Another relevant study is that of Bentéjac et al. (2021), who conducted a comprehensive comparison of the main boosting algorithms, including XGBoost, LightGBM, and CatBoost, evaluating their performance across various applications, such as macroeconomic

data forecasting. Their results indicated that LightGBM offers better scalability for large datasets, while CatBoost is more effective for modeling time series with complex categorical features. XGBoost, meanwhile, ranked second in both accuracy and model training speed.

In addition to decision tree-based models, some studies explored the use of AdaBoost in the time series context. Khan et al. (2021) applied this method to forecast energy sector variables, comparing its performance with the other models discussed in this article. However, XGBoost and CatBoost proved to be the best algorithms for the task, with accuracy ranging from 80% to 92%. It was observed that although AdaBoost performed well in certain scenarios, it can be less efficient compared to techniques like XGBoost and LightGBM, especially when there is high variability in data.

Given this evidence, it is clear that boosting algorithms have gained prominence by offering significant improvements over traditional forecasting models. However, challenges remain, such as the proper selection of hyperparameters, the choice of predictive variables, and integration with hybrid techniques, such as neural networks and statistical models.

3 Methodology

3.1 Data

The object of this study consists of the monthly percentage variation of Brazilian inflation between January 2004 and December 2024, as published by IBGE. The choice of this period is due to potential issues related to the stabilization phase of Plano Real and the implementation of different monetary regimes prior to the initial cut-off date.

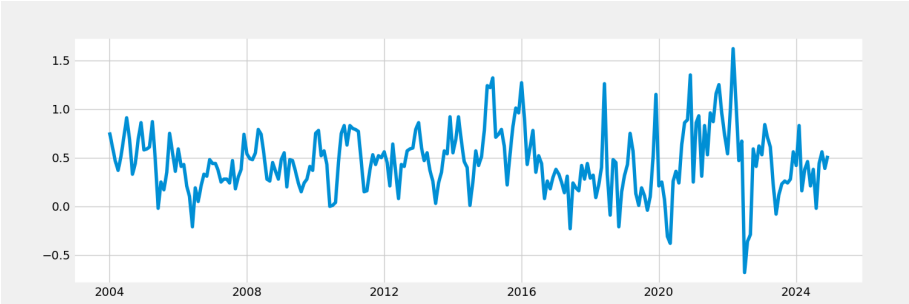

Figure 1: IPCA - Monthly variation.

As exogenous variables, time series commonly associated with price level forecasting were used. These include the monetary aggregates M1, M2, M3, and M4; the Selic rate; the exchange rate; and the Central Bank's Economic Activity Index (IBC-BR). All series have a monthly frequency, and 12 lags of each were used to predict inflation — which also included 12 lags of itself. In other words, the models were estimated with a maximum of 96 attributes.

To evaluate the performance of the algorithms, the dataset was divided into three partitions. The first, used to train each of the models (the *training* segment), consisted of 228 observations from January 2004 to December 2022. The second, used for hyperparameter tuning (the *validation* segment), contained 12 observations for 2023. The last was used to assess the out-of-sample accuracy of the models (the *test* segment), with 12 records for 2024.

3.2 Models

Boosting is a technique originally developed for classification problems but successfully adapted for regression tasks. The core idea is to combine several weak models, often referred to as *weak learners*, to create a more robust and accurate predictive model. Unlike methods such as *bagging*, which train models independently, boosting trains models sequentially, with each new model attempting to correct the errors made by its predecessors.

3.2.1 AdaBoost

Adaptive Boosting (AdaBoost), introduced by Freund and Schapire (1997), was one of the first boosting algorithms developed. A modification proposed by Drucker (1997) for regression problems assigns weights to weak models based on their performance in minimizing a loss function.
The model can be understood as follows: given a training set $\{(x_i, y_i)\}_{i=1}^n$, where x_i represents the samples and $y_i \in \mathbb{R}$ are the real values of the response variable, AdaBoost for regression combines multiple weak estimators to form a strong model. Initially, each sample is assigned a uniform weight:

$$w_i^{(1)} = \frac{1}{n}, \quad \forall \quad i = 1, \ldots, n.$$

Then, for each boosting iteration:

1. A weak model $h_t(x)$ is trained using the weights $w_i^{(t)}$.
2. The weighted absolute error is calculated:

$$\epsilon_t = \frac{\sum_{i=1}^n w_i^{(t)} |y_i - h_t(x_i)|}{\sum_{i=1}^n w_i^{(t)}}$$

3. The importance of the estimator is determined:

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right)$$

4. The sample weights are updated:

$$w_i^{(t+1)} = w_i^{(t)} \exp \left(\alpha_t |y_i - h_t(x_i)| \right)$$

5. The weights are normalized so that $\sum w_i^{(t+1)} = 1$.

The final model is obtained as a weighted combination of the weak estimators:

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x).$$

3.2.2 XGBoost

Extreme Gradient Boosting (XGBoost) is a machine learning algorithm developed by Chen and Guestrin (2016), who introduced significant improvements to the traditional *Gradient*

Boosting Decision Trees (GBDT), making the model faster and more scalable. Its main advantages include support for parallelization, advanced regularization, and optimization of the objective function, which justifies its widespread use in recent years for both classification and regression tasks.
The general objective function of XGBoost can be defined as:

$$\mathcal{L}(\Theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{1}$$

where $l(y_i, \hat{y}_i)$ is the loss function (for example, mean squared error); $\Omega(f_k)$ is the regularization term; f_k represents the individual decision trees; and K is the total number of trees. The term $\Omega(f_k)$ is used to penalize the number of leaves and the magnitude of the leaf weights to prevent model overfitting, such that

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{2}$$

where T is the number of leaves in the tree; γ controls the penalty for the number of leaves; w_j are the weights of the leaves; and λ is the L2 regularization factor. XGBoost uses a second-order approximation to optimize the objective function, given by the Taylor expansion of the loss function:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{3}$$

where $g_i = \frac{\partial l(y_i, \hat{y}_i)}{\partial \hat{y}_i}$ is the gradient of the loss function and $h_i = \frac{\partial^2 l(y_i, \hat{y}_i)}{\partial \hat{y}_i^2}$ is its Hessian. The score for the best split in a tree under the algorithm is given by

$$\text{Gain} = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma \tag{4}$$

where G_L, G_R and H_L, H_R are the sums of the gradients and Hessians in the left and right regions of the split, and γ penalizes splits that do not bring significant gain. Since the gradient guides the tree splitting in the region of greatest error reduction, XGBoost becomes more efficient than its predecessors.

3.2.3 LightGBM

The *Light Gradient Boosting Machine* (LightGBM) was developed by Ke et al. (2017) as an even more efficient alternative to XGBoost, focusing on optimizing training time and memory usage. While XGBoost uses a level-wise growth approach, where all nodes at a given depth are expanded before moving on to the next, LightGBM uses a depth-wise (leaf-wise) growth strategy, where the leaf with the highest loss reduction is expanded first. This procedure was enabled by two innovative techniques at the time, which allowed LightGBM to handle large volumes of data and high-dimensionality more efficiently: the *Gradient-based One-Side Sampling* (GOSS) and *Exclusive Feature Bundling* (EFB) strategies.
The GOSS technique reduces the number of samples used in training by selecting all instances with large gradients (indicating high errors) and randomly sampling among those

with small gradients. Mathematically, the method works as follows: let O be the training dataset at a given node of a decision tree. The variance gain by splitting attribute j at point d at this node is defined as

$$V_{j|O}(d) = \frac{1}{n_O}\left(\frac{(\sum_{\{x_i \in O: x_{ij} \leq d\}} g_i)^2}{n^j_{l|O}(d)} + \frac{(\sum_{\{x_i \in O: x_{ij} > d\}} g_i)^2}{n^j_{r|O}(d)}\right) \tag{5}$$

where $n_O = \sum I[x_i \in O]$, $n^j_{l|O}(d) = \sum I[x_i \in O : x_{ij} \leq d]$, and $n^j_{r|O}(d) = \sum I[x_i \in O : x_{ij} > d]$. Initially, the training instances are ranked by the absolute value of their gradients in descending order. Then, the top α fraction of instances with the largest gradients is retained, forming a subset A. The remaining instances, A^c, consisting of $(1-\alpha) \times 100\%$ of the samples with smaller gradients, are randomly sampled to create another subset B with size $b \times |A^c|$. Finally, the instances are split according to the estimated variance gain $\tilde{V}_j(d)$ over the subset $A \cup B$:

$$\tilde{V}_j(d) = \frac{1}{n}\left(\frac{(\sum_{x_i \in A_l} g_i + \frac{1-\alpha}{b}\sum_{x_i \in B_l} g_i)^2}{n^j_l(d)} + \frac{(\sum_{x_i \in A_r} g_i + \frac{1-\alpha}{b}\sum_{x_i \in B_r} g_i)^2}{n^j_r(d)}\right) \tag{6}$$

where $A_l = \{x_i \in A : x_{ij} \leq d\}$, $A_r = \{x_i \in A : x_{ij} > d\}$, $B_l = \{x_i \in B : x_{ij} \leq d\}$, and $B_r = \{x_i \in B : x_{ij} > d\}$. Therefore, the estimated variance gain is applied to a smaller subset rather than the full dataset to determine the best split point, making the algorithm computationally less expensive.

EFB, on the other hand, addresses high-dimensional data by combining features that rarely take nonzero values simultaneously, particularly for categorical attributes. Both techniques maintain model accuracy while significantly reducing processing time.

3.2.4 CatBoost

Prokhorenkova et al. (2018) proposed the CatBoost algorithm to efficiently handle categorical data and address the discrepancy between the accuracy of boosting models on training and test data.

Unlike the previously cited models, CatBoost can handle categorical variables without requiring preprocessing, which typically involves *one-hot* or *label encoding*. These common approaches can lead to an excessive increase in data dimensionality, while the *target statistics* method used by CatBoost simply replaces the categories with statistics derived from the target variable.

Another innovation introduced by the study is the *ordered boosting* technique, which reduces overfitting caused by the *target leakage* problem — that is, when information from the future or information directly related to the target variable is incorrectly included in the training data, which would not be available during model evaluation on test data. This technique can be seen as a generalization of *gradient boosting*, where training is conducted in a specific order to avoid future information leakage during the model fitting process.

3.3 Hyperparameter Optimization

In traditional forecasting techniques, significant time could be spent selecting the best parameter values for a model, such as following the Box and Jenkins methodology to determine the orders of autoregressive and moving average components in an ARIMA

model. However, this is generally not the case for nonparametric models, especially those discussed in this study. Here, it is more useful to perform targeted testing of different values for specific parameters of each algorithm — a task that can be approached in several ways. The most popular of these methods are discussed next.

3.3.1 Grid Search

Grid Search is an exhaustive approach to hyperparameter optimization, where all possible combinations of hyperparameters are evaluated. The strategy involves initially defining a grid of values for each hyperparameter and evaluating every combination, ensuring that the optimal list of hyperparameters is found. Its minimization problem can be defined as

$$\theta^* = \arg\min_{\theta \in \mathcal{T}} \mathcal{L}(\theta), \tag{7}$$

where θ^* is the vector of optimal hyperparameters; $\mathcal{T}$ is the set of all possible hyperparameter combinations; and $\mathcal{L}(\theta)$ is the model's loss function with hyperparameters θ.
Although *Grid Search* is a simple and intuitive approach, it is not computationally efficient, especially when the number of hyperparameters or the range of possible values for each becomes significantly large.

3.3.2 Random Search

Random Search is an alternative to *Grid Search*, where instead of evaluating all possible hyperparameter combinations, random combinations are selected within the search space. This approach can be more efficient than Grid Search, especially when the number of hyperparameters is large, as it does not exhaustively explore all combinations in a grid and can therefore cover a broader space within a fixed time constraint.

3.3.3 Bayesian Optimization

Bayesian Optimization (Snoek et al., 2012) is a probabilistic approach to hyperparameter optimization, aiming to balance exploration of new hyperparameter combinations with exploitation of areas already known to produce good results. Instead of exhaustively evaluating all combinations or selecting random values, a probabilistic model is used to model the loss function and identify the most promising points in the search space.
The main idea is to model the loss function using a surrogate model (such as a Gaussian process) and, based on it, decide where to conduct new hyperparameter evaluations. The algorithm seeks to find the combination of hyperparameters that minimizes the loss function with the fewest number of evaluations possible. The procedure can be characterized as:

$$\theta^* = \arg\min_{\theta \in \mathcal{T}} \mathbb{E}[\mathcal{L}(\theta)], \tag{8}$$

where $\mathbb{E}[\mathcal{L}(\theta)]$ is the expected value of the loss function, estimated through the probabilistic model, and $\mathcal{T}$ is the hyperparameter search space. This will be the technique employed in the present study, due to its versatility; a description of the hyperparameters selected for each model can be found in the Appendix.

3.4 Evaluation Metrics

To establish an objective function for the optimization of the models' parameters and to compare the forecasting results, some performance metrics will be presented:

- Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2 \tag{9}$$

- Root Mean Squared Error (RMSE)

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2} \tag{10}$$

- Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t| \tag{11}$$

- Mean Absolute Percentage Error (MAPE)

$$\text{MAPE} = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \tag{12}$$

- Symmetric Mean Absolute Percentage Error (SMAPE)

$$\text{SMAPE} = \frac{100}{n}\sum_{t=1}^{n}\frac{|y_t - \hat{y}_t|}{\frac{|y_t|+|\hat{y}_t|}{2}}, \tag{13}$$

where y_t and $\hat{y}_t$ are the observed and predicted values at time t, respectively, and n is the total number of observations.
It is important to highlight some key characteristics of these metrics. The Mean Squared Error penalizes larger errors more heavily, since errors are squared — making it a metric more sensitive to *outliers*. The Mean Absolute Percentage Error is widely used because it expresses errors as a percentage, which facilitates interpretation and comparison across different datasets. However, it has limitations when the actual values (y_t) are very close to zero, as it can result in very large or undefined values Hyndman and Koehler (2006). Another issue with this metric is that it treats over-predictions and under-predictions differently. To correct for this, the symmetric version (SMAPE) is used, which treats errors in a more balanced way.

4 Results and Discussion

After obtaining the optimal parameters for each of the models, the latter were re-estimated using the training and validation partitions (that is, data from January 2004 to December

2023). For the in-sample forecasts, XGBoost was the best-performing model, followed by CatBoost. The former achieved an RMSE of 0.0005; the latter, 0.0146. This ranking was consistent across the other in-sample evaluation metrics as well.
To reproduce the out-of-sample results, a one-step-ahead forecast was performed for each month of 2024, yielding the results shown in Figure 2.

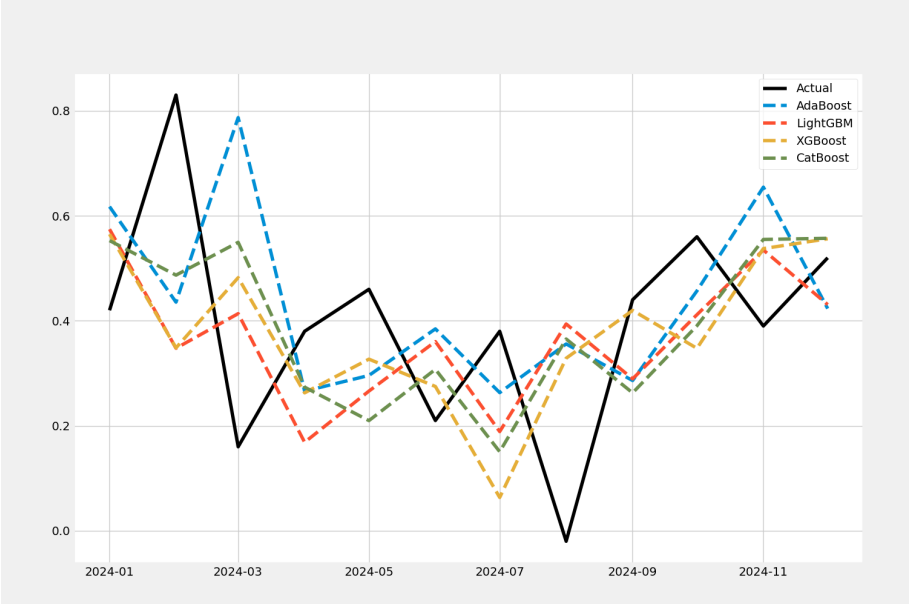

Figure 2: One-step-ahead forecast.

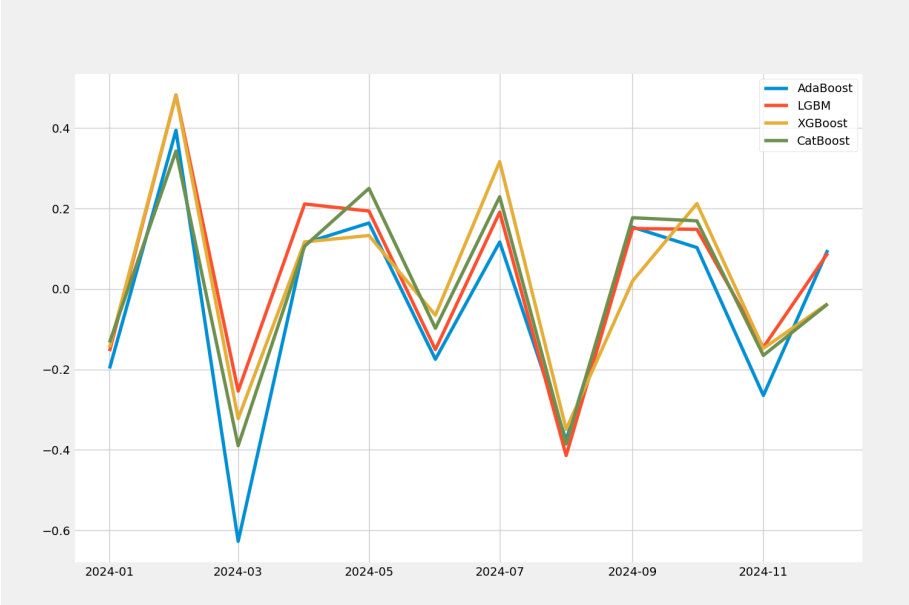

Figure 3: Forecast errors $(y_t - \hat{y}_t)$.

Figure 4 displays the corresponding accuracy metrics. As suggested by Hyndman and Athanasopoulos (2018), a baseline *naive* forecast was also performed, where predictions are

simply the last observed value (in this case, the IPCA value from December 2023 repeated across all months of 2024). However, the results from the naive model were omitted from the figures because they did not outperform the best models in this study. The XGBoost model showed the best performance according to the Mean Absolute Error and the Mean Absolute Percentage Error, with values of 0.1954 and 191.5450%, respectively. AdaBoost slightly outperformed XGBoost only in terms of the SMAPE metric, achieving a value of 0.6170. CatBoost, on the other hand, performed better out of sample when considering the Mean Squared Error and its root (0.0550 and 0.2346).
Given the small difference in accuracy according to the SMAPE, XGBoost demonstrated the best overall performance, ranking first in most of the error measures. CatBoost ranked second, followed by LightGBM, and finally AdaBoost, which ranked last in four out of five metrics.

Model/Metric	MAE	MAPE	SMAPE	MSE	RMSE
AdaBoost	0,2319	223,7532%	0,6171	0,0772	0,2779
CatBoost	0,2069	212,6666%	0,6238	0,0550	0,2346
LightGBM	0,2154	221,4818%	0,6448	0,0589	0,2428
XGBoost	0,1955	191,5450%	0,6179	0,0569	0,2385

Figure 4: Forecast metrics for the test partition.

The results are similar to those in previous studies to some extent. As seen in the related work section, LightGBM was the model that appeared most frequently among the winners of the M5 competition; Bentéjac et al. (2021) also obtained similar results. However, that was not the case in the present study, where LightGBM ranked third overall. On the other hand, CatBoost's performance corroborates the findings of Xiang et al. (2022) for a two-layer architecture, while XGBoost's performance aligns with that reported by Sousa et al. (2023).
One-step-ahead forecasting may not be very practical for monthly series, as operational planning or budgeting often requires a longer forecast horizon. In this regard, future research could extend the present study by evaluating model performance across different forecast horizons and cumulatively. Moreover, considering the context in which some of these algorithms were developed, further work could involve enhancing the feature engineering step by including interactions among lags, seasonal dummy variables, and rolling-window statistics such as moving averages and standard deviations.

References

Bentéjac, C., A. Csörgő, and G. Martínez-Muñoz (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review 54*, 1937–1967.

Box, G. E. and G. M. Jenkins (1970). *Time series analysis: forecasting and control*. Holden-Day.

Chen, T. and C. Guestrin (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794.

Drucker, H. (1997). Improving regressors using boosting techniques. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML)*, pp. 107–115. Morgan Kaufmann.

Freund, Y. and R. E. Schapire (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences 55*(1), 119–139.

Holt, C. C. (1957). Forecasting seasonals and trends by exponentially weighted moving averages.

Hyndman, R. J. and G. Athanasopoulos (2018). *Forecasting: principles and practice*. OTexts.

Hyndman, R. J. and A. B. Koehler (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting 22*(4), 679–688.

Ke, G., Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems 30*.

Khan, A. M., A. BinZiad, and A. A. Subaii (2021). Boosting algorithm choice in predictive machine learning models for fracturing applications. In *SPE Asia Pacific Oil and Gas Conference and Exhibition*, pp. D011S009R003. SPE.

Makridakis, S., E. Spiliotis, and V. Assimakopoulos (2018). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLoS ONE 13*(3), e0194889.

Makridakis, S., E. Spiliotis, and V. Assimakopoulos (2022). M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting 38*(4), 1346–1364. Special Issue: M5 competition.

Prokhorenkova, L., G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin (2018). Catboost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems 31*, 6638–6648.

Snoek, J., H. Larochelle, and R. P. Adams (2012). Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pp. 2951–2959.

Sousa, A. C. C., T. G. do Rêgo, Y. d. A. M. Barbosa, T. de Menezes, et al. (2023). Comparing gradient boosting algorithms to forecast sales in retail. In *Anais do XX Encontro Nacional de Inteligência Artificial e Computacional*, pp. 596–609. SBC.

Voyant, C., G. Notton, S. Kalogirou, M.-L. Nivet, C. Paoli, F. Motte, and A. Fouilloy (2017). Machine learning methods for solar radiation forecasting: A review. *Renewable energy 105*, 569–582.

Xiang, W., P. Xu, J. Fang, Q. Zhao, Z. Gu, and Q. Zhang (2022). Multi-dimensional data-based medium- and long-term power-load forecasting using double-layer catboost. *Energy Reports 8*, 8511–8522.

Yoon, J. (2021). Forecasting of real gdp growth using machine learning models: Gradient boosting and random forest approach. *Computational Economics 57*(1), 247–265.

Zhang, G., B. E. Patuwo, and M. Y. Hu (1998). Forecasting with artificial neural networks:: The state of the art. *International journal of forecasting 14*(1), 35–62.

Forecasting probability density functions in Hilbert Spaces

Matheus Vizzotto dos Santos - Universidade Federal do Rio Grande do Sul
Flávio A. Ziegelmann - Universidade Federal do Rio Grande do Sul
Eduardo de Oliveira Horta - Universidade Federal do Rio Grande do Sul

Summary

Functional Data Analysis (FDA) has emerged as a rapidly evolving field, extending classical statistical methods to data represented by functions. In this context, time-dependent analysis can also be generalized by treating each observation as a function rather than a scalar or vector, giving rise to functional time series. This work focuses on forecasting a specific class of functional objects: probability density functions (PDFs). A key challenge in this setting arises from the fact that PDFs do not form a vector space, but instead reside in a convex subset, rendering standard functional time series techniques a tricky endeavor. To address this, we propose a transformation approach that maps PDFs into a Hilbert space, enabling the application of established functional time series tools. The effectiveness of this method will be illustrated through an application to high-frequency financial data, with results highlighting its potential compared to other state-of-the-art tools for accurate and interpretable forecasting.

Keywords: Functional data analysis; functional time series; probability density functions; forecasting; Karhunen-Loève expansion.