Journal of Information and Data Management, 2025, 16:1, doi: 10.5753/jidm.2025.4668

Evaluating Window Size Effects on Univariate Time Series Forecasting with Machine Learning

João David Freitas [**Universidade de Fortaleza** | ***joaodavidfreitasc@edu.unifor.br***]
Caio Ponte [**Universidade de Fortaleza** | ***caioponte@unifor.br***]
Rafael Bomfim [**Universidade de Fortaleza** | ***bomfim@unifor.br***]
Carlos Caminha [**Universidade Federal do Ceará** | ***caminha@ufc.br***]

Universidade Federal do Ceará (UFC), Av. da Universidade, 2486 - Benfica, Fortaleza, Brazil 60020180.

Received: 5 July 2024 • **Published:** 23 August 2025

Abstract In the realm of time series prediction modeling, the window size (w) is a critical hyperparameter that determines the number of time units included in each example provided to a learning model. This hyperparameter is crucial because it allows the learning model to recognize both long-term and short-term trends, as well as seasonal patterns, while reducing sensitivity to random noise. This study aims to elucidate the impact of window size on the performance of machine learning algorithms in univariate time series forecasting tasks, specifically addressing the more challenging scenario of larger forecast horizons. To achieve this, we employed 40 time series from two different domains, conducting experiments with varying window sizes using four types of machine learning algorithms: Bagging (Random Forest), Boosting (AdaBoost), Stacking, and a Recurrent Neural Network (RNN) architecture, more specifically the Long Short-Term Memory (LSTM). The results reveal that increasing the window size generally enhances the evaluation metric values up to a stabilization point, beyond which further increases do not significantly improve predictive accuracy. This stabilization effect was observed in both domains when w values exceeded 100 time steps. Moreover, the study found that LSTM architectures do not consistently outperform ensemble models in various univariate time series forecasting scenarios.

Keywords: Time Series, Machine Learning, Window Size, Ensembles, Neural Networks.

1 Introduction

The forecasting of time series is an important task with applications in many areas, including economics, finance, health, engineering, social sciences, and climatology [De Gooijer and Hyndman, 2006]. Time series can be defined as sequences of terms ordered over time, usually at regular intervals, and can provide valuable information about trends, patterns, and behaviors of dynamic systems over time. By studying time series, one can uncover hidden information and predict future behaviors, enabling informed decision-making and the improvement of efficiency and effectiveness. Analyzing these series can also help identify changes in seasonal patterns or extreme events, such as peaks or drops in a series, which can be valuable for risk prevention or mitigation.

Despite all these applications, modeling a time series forecasting problem can present challenges in understanding how hyperparameters affect predictions. In addition to considering the definition of the window size (w), the number of past observations used as input to predict future values, it is necessary to explore a variety of hyperparameters of the adopted model, such as learning rate, number of hidden layers, filter sizes, among others. Moreover, it is important to properly define the multi-step forecasting strategies [Taieb *et al*., 2012] and the forecasting horizon size [Hamzaçebi *et al*., 2009]. Another crucial consideration is choosing appropriate transformations to deal with possible nonlinearities, seasonality, trends, and other characteristics of the time series [Salles *et al*., 2019]. In other words, time series modeling requires careful analysis of several factors to ensure accurate and reliable forecasts.

Specifically regarding the window size (w), this hyperparameter is important in time series forecasting because it defines the number of time units that are present in each example during the training process of a model, and it influences the accuracy and stability of the forecast [Azlan *et al*., 2019]. Choosing an appropriate window size is crucial for capturing long-term and short-term trends and seasonal patterns without making the model sensitive to random fluctuations or leading to overfitting. Often, the ideal choice of window size needs to take into account the complexity of the time series, the size of the dataset, the model used, and the available computational capacity.

Despite the importance of defining the window size, there are not many articles that address the study of the real impact of window size on time series forecasting using different models and a large forecast horizon. In this context, this article aims to study the impact of window size on time series forecasting. For this purpose, two real datasets were used, one from retail and one from urban mobility, with a total of 20 time series each, totaling 120,280 time units to be modeled in a forecasting problem. The evaluation was conducted using four classes of machine learning algorithms: a Bagging algorithm (Random Forest); a Boosting algorithm (AdaBoost); a Stacking algorithm; and a Recurrent Neural Network architecture (LSTM).

This work is an extended version of [Freitas *et al*., 2023]. In this extension, we have expanded the related works sec-

tion with a more comprehensive review of the literature on the influence of window size on time series forecasting. Additionally, we included a detailed analysis of the Partial Autocorrelation Function (PACF) to help estimate the appropriate window size. We show that, although useful, the PACF does not provide a precise view of the exact window size to be used, making it necessary to conduct additional experiments to determine the ideal value.

This article is organized as follows: in Section 2, the state of the art related to this research is presented; in Section 3, the research methodology is detailed; in Section 4, the results of the tests conducted are presented; and finally, in Section 5, the conclusions of this research are outlined, along with a discussion of future work.

2 Related Works

Time series forecasting is a critical task with applications in various domains, such as finance, healthcare, and engineering. It involves predicting future values based on historical data, and its accuracy depends on multiple factors, including data characteristics and model hyperparameters. While there are limited studies directly addressing the impact of window size on time series forecasting, it is important to understand how various hyperparameters influence forecasting models. This section begins by discussing works that, like ours, explore the effects of hyperparameter variation in time series prediction, progressively narrowing the focus to the specific role of window size.

Several studies have analyzed the impact of hyperparameter tuning and feature selection on forecasting performance. Feature selection often involves determining which past observations (lags) are most relevant for forecasting. Huber and Stuckenschmidt [2020] analyzed the forecasting of retail products by adding special dates and holidays to the models, and also examined the effect of increasing the forecasting horizon in both single-step and multi-step approaches. Abolghasemi *et al*. [2020] developed a model to forecast product demand that takes into account systematic events, such as a product promotion or a change in weather. At the end of the tests, the model presented in the article performed better when the new features were added.

Similarly, the forecast horizon can influence the optimal window size, as longer horizons may require capturing longer-term dependencies in the data. Nikolopoulos and Fildes [2013] studied the incorporation of climate data to see how it impacts the sales forecast of a beer company using an econometric model. At the end of the study, the climate adjustment mechanism improved the company's forecast. Teixeira and Fernandes [2011] studied the impact of adding the risk of insolation, total monthly hours of sunshine, on the forecast of hotel occupancy in Portugal. The mean relative error was reduced by about 0.5

A common challenge in time series forecasting is the length of the forecast horizon, which is often limited to just one step ahead. This limits the application of models in situations requiring long-term planning. According to Cheng *et al*. [2006], long-term forecasts tend to have higher errors because the bias and variance of past forecasts affect future forecasts, resulting in accumulated error.

Directly related to window size, Chandra *et al*. [2021] evaluated the performance of some deep learning models with multi-step forecasting of seven time series. The models used were *LSTM*, bidirectional *LSTM*, *encoder-decoder LSTM*, and a convolutional neural network. The steps (1 to 10) were varied, and after the tests, the best results were from the bidirectional *LSTM* and the *encoder-decoder LSTM*. Their study indicates that model architecture and the amount of past information (window size) can significantly impact forecasting accuracy, especially in multi-step forecasting.

In a study on stock price forecasting, Shynkevich *et al*. [2017] tested the combination of forecast horizon and window size using three models. The forecast was treated as a classification problem to determine whether stocks would rise or fall. The results indicated that the optimal window size is close to the size of the forecast horizon.

According to Kil *et al*. [1997], determining the window size is crucial for time series forecasting as it significantly influences model performance. Frank *et al*. [2000] analyzed the impact of window size on two time series using two models: the Multilayer Perceptron (*MLP*) and the Radial Basis Function (*RBF*). The study demonstrated that small windows do not produce as significant results as larger windows. However, an optimal window size was identified, and increasing the size beyond this value can result in a decrease in forecast accuracy. Nonetheless, the study is limited by using only two time series and focusing more on neural networks, without using ensemble models, for example.

Braga *et al*. [2019] conducted an extensive comparison between machine learning models, such as Random Forest and a Boosting algorithm (AdaBoost), and deep learning models (LSTM), highlighting the efficiency and accuracy of traditional models in scenarios with limited data. However, the article does not focus on window size nor clearly define whether the forecasting is done in a multi-step or one-step manner.

Some recent studies have specifically investigated the impact of window size on time series forecasting. Bergström and Hjelm [2019] investigated the impact of the window size on the forecasting of the American stock index S&P500 using *LSTM* networks. By evaluating five different window sizes and employing the Root Mean Squared Error (RMSE) as a performance metric, the authors found that a window size of ten days yielded the most accurate predictions. However, the study was limited to the S&P 500, LSTM and a specific time period, and further research is needed to generalize these findings to other datasets and time frames. On the other hand, Liu *et al*. [2022], used the SWLHT (Short Window Long Horizon Transformer) highlights the importance of the window size and how this choice directly impacts the model's ability to capture patterns in time series data and make accurate predictions. They used the *Transformer* model to test forecasts with small windows and larger forecast horizons on four time series of transformer temperature, and their experiments indicated that smaller windows combined with larger forecast horizons can lead to better predictions. Additionally, Ughi *et al*. [2023] compared the performance of *Transformer* models with simpler models (such as MLP) by testing four variations of window size across six

series from different domains.

While these studies contribute to understanding the role of window size, they are often limited to a small number of time series or specific models. The present article advances time series research by conducting experiments with more than one time series, in two different domains, and with varying amounts of annotations. It also progresses by studying the impact of window size in a more challenging scenario, where the forecast horizon is larger. This is particularly important as a significant portion of the articles in the literature conduct experiments with only one-step ahead predictions.

3 Methodology

This section details the datasets used and explains the data modeling conducted for this research. Two datasets were utilized: one for bus boarding counts and another for retail product sales. These datasets exhibit distinct patterns and were modeled as input for several machine learning models to predict future boarding counts and sales quantities.

3.1 Datasets

The first dataset concerns time series of retail store product sales in the supermarket sector, located in Fortaleza (Ceará). The data were obtained from a major retailer through a research project in partnership with the University of Fortaleza. Sales information for twenty products from the A curve (items with the highest contribution to store revenue) was collected from January 2, 2017, to April 30, 2019, totaling 849 days. Sales data for products, measured in units or kilograms, were aggregated daily for each analyzed product, forming the final time series. Product identifiers were anonymized. The months from January 2017 to December 2018 were set aside for training, and January 2019 to April 2019 for testing. Figure 1A illustrates a daily sales time series of an A curve product over a sample of 240 days."

The other dataset concerns time series of passenger boardings on the twenty most heavily used bus lines within the public transportation system in the city of Fortaleza (Ceará). Bus system passengers use a smart card with a user identifier, and each time this card is used, a boarding record is logged. The data were provided by the Fortaleza City Hall and have been used in other articles [Caminha *et al*., 2016, 2017, 2018; Ponte *et al*., 2018; Bomfim *et al*., 2020; Ponte *et al*., 2021; Araújo *et al*., 2023]. These data cover the period from January 1st, 2018, to July 31st, 2018, and the passenger boarding counts were aggregated hourly for each of the top twenty bus lines in the city. Thus, twenty time series were generated, each comprising over 5000 hours of boarding data for each bus line. The months from January to June were used for training, and July was used for testing. Figure 1B illustrates a time series of hourly boardings for a bus line, detailing the seasonal patterns observed in vehicle usage over a sample period of about 10 days.

For more details regarding access to the data used in this research, see the "Availability of data and materials section".

Before applying the forecasting methods, appropriate preparation of the time series data was conducted. Following an exploratory analysis on both datasets, it was identified that there were non-contiguous data, meaning missing data entries. These missing data indicate times when no passengers boarded at a certain hour of the day or when no sales occurred for a particular product on a given day. Therefore, these instances were filled with zero values to maintain the temporal structure of the series.

3.2 Modeling Sliding Window

Sliding window concept was used in the process of modeling the time series forecasting problem. The sliding window technique involves an approach where the time series data is divided into smaller segments of constant size (w). The term sliding refers to the process of shifting the window along the series with a specified step (p) to the right, enabling the construction of a training dataset.

Sliding window are a technique used to transform a time series into a labeled dataset, where each window contains a set of past observations from the time series considered as input for forecasting models. The observation immediately following the end of the window is defined as the target value to be predicted given the past window. Therefore, this technique is highly suitable for supervised machine learning methods like regression to forecast time series data.

The Figure 2 illustrates the process of generating input and output examples for training AI models. A daily time series, shown in green as an example, is transformed into a labeled dataset. The first input sample, x_1, is generated and contains w values (window size), representing the features to be learned by the models. These features enable the models to capture the temporal dependencies of the series from patterns and trends present in past observations. The value y_1 indicates the day immediately following the x_1 window, representing the target variable to be predicted. The window is then shifted p steps to the right to obtain new samples for the dataset, repeating the process until the entire series is covered. Depending on w and p, there may be overlap in the data, reinforcing the discovery of data patterns and increasing the number of training samples. In Figure 2, the value of p is 1, indicating that the next input (x_2) and output (y_2) are shifted by only one day to the right.

3.3 Model Training

After the dataset is created, the process of training the Machine Learning models begins using labeled data generated by the sliding window technique. Models based on ensembles and Neural Networks were used to forecast the hourly boarding counts and daily product sales. The main objective of this research is to evaluate how w impacts the quality of predictions in univariate time series performed by these models. Thus, the value of w was varied, and an associated error was calculated for each w. For mobility data, w started at 24 hours and increased by 24 hours until reaching a maximum w of 360 hours. For retail data, w started with sequences of 7, 14, 21, and 30 days, and then increased by 30 days until reaching a maximum w of 365 days. Due to the seasonality of some retail products, which may exhibit characteristic behaviors weekly, biweekly, and monthly, these initial days

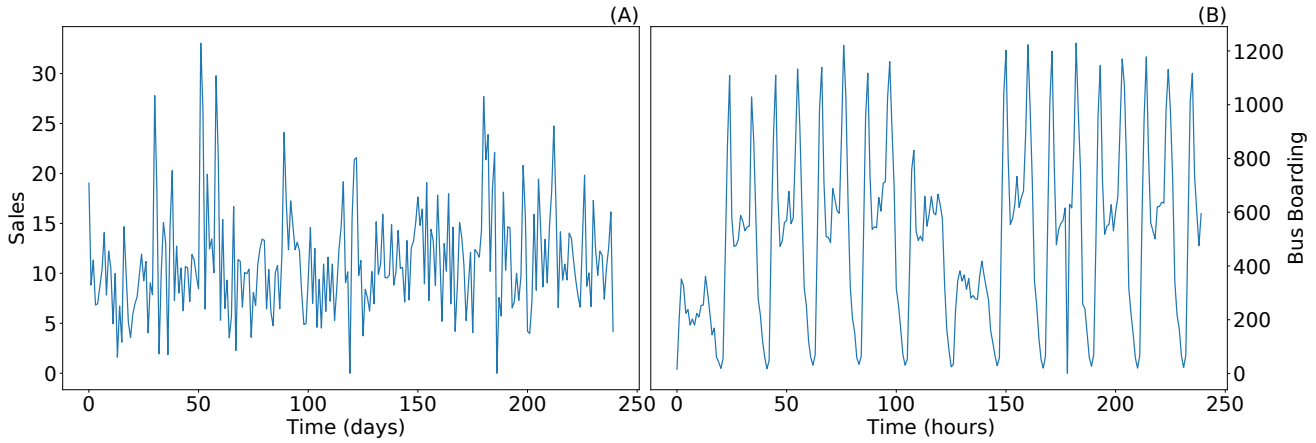

Figure 1. Examples of time series in the studied domains. In (A), 240 days of sales data for a product from Curve A are illustrated. The y-axis represents the quantity sold of this product, and the x-axis represents time in days. In (B), the first 240 hours of data from bus line 42 are shown. The y-axis represents the number of passengers boarding that specific bus line, and the x-axis represents time in hours.

were chosen more personalized. A value of $p = 1$ was chosen for both domains.

The models used were a Bagging (Random Forest), a Boosting (AdaBoost), a Stacking, and a Recurrent Neural Network architecture (LSTM). In *Random Forest* [Breiman, 2001] and *AdaBoost* [Freund *et al*., 1996], the default parameters of *Scikit-learn* [Pedregosa *et al*., 2011] were used. In the neural network *LSTM* [Hochreiter and Schmidhuber, 1997], the implementation of *Tensorflow* [Abadi *et al*., 2016] was used, with the parameters *neurons* = 200, *batch_size* = 32, and with activation function *ReLu*, *epochs* = 200, validation of 20% and optimizer "Adam" with a learning rate of 0.0001. The *Stacking* [Wolpert, 1992] has as $estimators$: *Ridge Regressor* [Hoerl and Kennard, 1970], *SVR* [Drucker *et al*., 1996], *Random Forest*, *Gradient Boosting* [Friedman, 2001], *KNN* [Cover and Hart, 1967], and Linear Regression [Galton, 1889] as the *final estimator*, all in their *default* versions.

3.4 Multiple Steps Ahead Forecast

The process of inference in time series multiple steps ahead, or long-term forecasting, consists of predicting the next h values in the future. Some forecasting approaches can be performed as Iterative, Direct, MIMO (*Multiple Input Multiple Output*), or combinations of these strategies [Taieb *et al*., 2012]. In this research, the Iterative approach was used, where to estimate future values for a time horizon h, one prediction is made at a time. Figure 3 illustrates the forecasting process, where the model M receives as input a window of size w (initially indexed from 1 to t) and produces $\hat{y}_{t+1}$ as output. Then, the process is repeated by adding the most recent prediction, $\hat{y}_{t+1}$, to the new input to produce the second inference, $\hat{y}_{t+2}$ (disregarding the oldest value of the previous input, always maintaining windows of size w). Depending on the number h (a sufficiently large number) and w, there are a moment when only estimated values are used as inputs, instead of the real data.

The evaluation of the results was done using the Symmetric Mean Absolute Percentage Error (*SMAPE*) [Makridakis, 1993], chosen because it is a percentage error, as the retail *dataset* contains different units (e.g., products sold by unit and products sold by weight). The *SMAPE* is also a geometric mean error, making it ideal for comparing the performance of multiple models and in a high number of predictions, according to [Kreinovich *et al*., 2014].

4 Results and Discussion

4.1 PACF Analysis

The Partial Autocorrelation Function (PACF) measures the correlation between a time series value and its lags, while adjusting for the effects of all intermediate lags. In other words, unlike simple autocorrelation, which considers all previous lags, the PACF isolates the direct effect of each specific lag on the current value by removing indirect influences [Box and Jenkins, 1976]. That function is widely used to determine the appropriate window size in time series forecasting problems [Leites *et al*., 2024]. Figure 4 illustrates the impact of the window size (w) on the forecast of a retail sector time series, highlighting how the lags with higher autocorrelation observed in the PACF influence the choice of w.

In Figure 4, in (A), we present the PACF plot of the time series, which helps identify the most significant lags. It is important to note that lag 0, which represents the autocorrelation of a value with itself, is always 1 and therefore should be disregarded in the analysis.

Items from (B) to (G) in Figure 4 show line graphs comparing predicted and actual values and for different window sizes (w) of 7, 14, 21, 30 and 90. In these figures, the blue line represents units sold of the product (y-axis), and the x-axis represents the forecast horizon from 0 to 120. The 10 light gray lines represent 10 predictions made for the respective w. Additionally, the sMAPE error metric is displayed above each plot, indicating the accuracy of the model for each window size.

The analysis of these figures reveals that predictions vary considerably for smaller window sizes (such as 7, 14, and 21), showing greater dispersion in the prediction lines. As w increases, predictions become more consistent, with the gray lines overlapping more. This indicates that larger window sizes may better capture underlying patterns, reducing

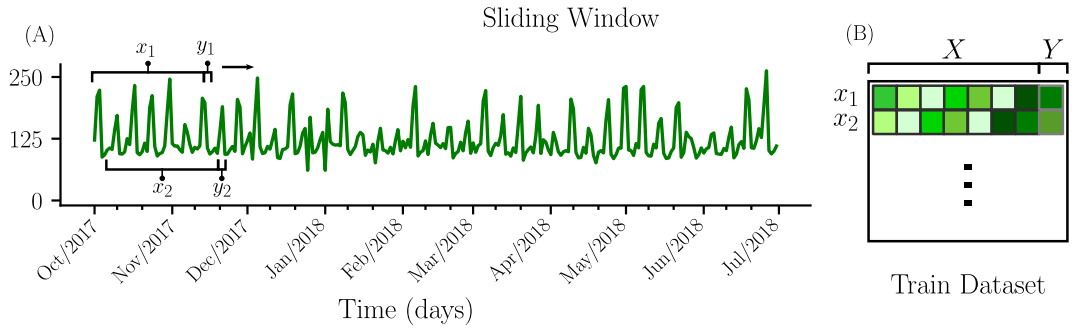

Figure 2. Example of applying the Sliding window technique to construct a labeled dataset from univariate time series. (A) The time series represents the daily sales of a retail product. Variables x_1 and x_2 represent input values at different time points, while y_1 and y_2 represent corresponding target values associated with these intervals. Sliding window constructs a labeled dataset from a time series by segmenting it into overlapping intervals. In (B) Representation of the training dataset generated from sliding window, where X contains the input values and Y the corresponding output values for the forecasting task.

Figure 3. The diagram shows the multiple steps ahead forecast using the Iterative method, where the model's predictions are used as inputs for subsequent steps, allowing the model to predict the next h future periods.

variability in forecasts, though this finding is specific to the datasets employed in the experiments.

Figures 6 and 7 in Appendix A illustrate two PACF plots for both datasets: Figure 6 shows the PACF of 20 products from the retail dataset, while Figure 7 presents the PACF of 20 bus lines from the mobility dataset. In PACF analysis, we look for the highest significant correlation lag, as smaller lags tend to show higher correlation. We chose this approach to provide an overall performance measure of the models across various scenarios, as individual time series can exhibit unique patterns, such as specific seasonalities (weekly, biweekly, and monthly).

For the retail data, the stabilization point of the window size, for most products, is 7, while for the mobility data, it is 24. These values corroborate the intuition about the data: many retail products exhibit weekly seasonality, whereas mobility patterns tend to repeat every 24 hours. However, less frequent patterns are also important for maintaining prediction quality and should be considered.

Choosing the window size (w) is particularly challenging. Very large windows can lead to models that become overwhelmed by the volume of data, resulting in poor-quality predictions. Therefore, it is essential to study the window size in detail to achieve the best possible results in time series forecasting.

4.2 Test Result

Although the analysis of the Partial Autocorrelation Function (PACF) is a valuable tool for estimating the appropriate window size in time series forecasting problems, it is not infallible. The PACF can indicate significant lags that suggest possible window sizes, but it does not account for all the nuances of the data and the variables involved in predictive modeling. Therefore, it is essential to conduct exhaustive experiments varying the window size to fully understand its impact on the performance of machine learning models. This experimental approach allows for an empirical evaluation of how different window sizes affect prediction accuracy, providing a more robust basis for choosing the ideal w value.

Figure 5 illustrates the results of the experiments following the methodology detailed in Section 3 of this article. Predictions were made for forty time series using four machine learning models. In (A), we can observe the *SMAPE* values for various window sizes, w, in experiments conducted with retail data. There was a decrease in *SMAPE* across all models for $w \leq 30$. For higher w values, the models maintained *SMAPE* values with less variation. In the case of *LSTM*, a greater variation in *SMAPE* was observed for $w > 30$.

In Figure 5B, the performance of the models for the mobility data can be seen. It is noticeable that the forecast improved as the window size increased, with the improvement becoming stable after a certain window size for some models. The *Random Forest* was the best-performing model for this dataset, but the *LSTM* showed the most performance improvement (reduction in *SMAPE*) with an increase in w. The *AdaBoost* model exhibited the least variation in improvement.

Analyzing the error curves for the mobility data, all models showed a considerable reduction in error at w equal to 168, where it is possible that the models captured the weekly pattern in the behavior of each line.

In both datasets, a positive impact on predictions was observed with an increase in window size. After a certain w size, no clear reduction in error was observed. However, it is important to consider whether using much larger window sizes is worthwhile, as increasing the window size also increases the size of the input data and, consequently, the training time of the models. In the mobility dataset, the *Adaboost* model presented the highest error and the least reduction in error with an increase in window size. This may be related to the fact that the forecasting horizon is larger than in the retail dataset.

In addition to the results presented, it is important to emphasize that the concept of an optimal window size is not absolute and should not be considered as a fixed value for all datasets or models. Instead, the window size (w) is a hyperparameter whose effect is highly dependent on the characteristics of the dataset and the machine learning model being used. Our experiments demonstrated that, while model performance improves as w increases, this improvement reaches a point of stabilization, beyond which further increases in window size do not result in significant gains in prediction accuracy.

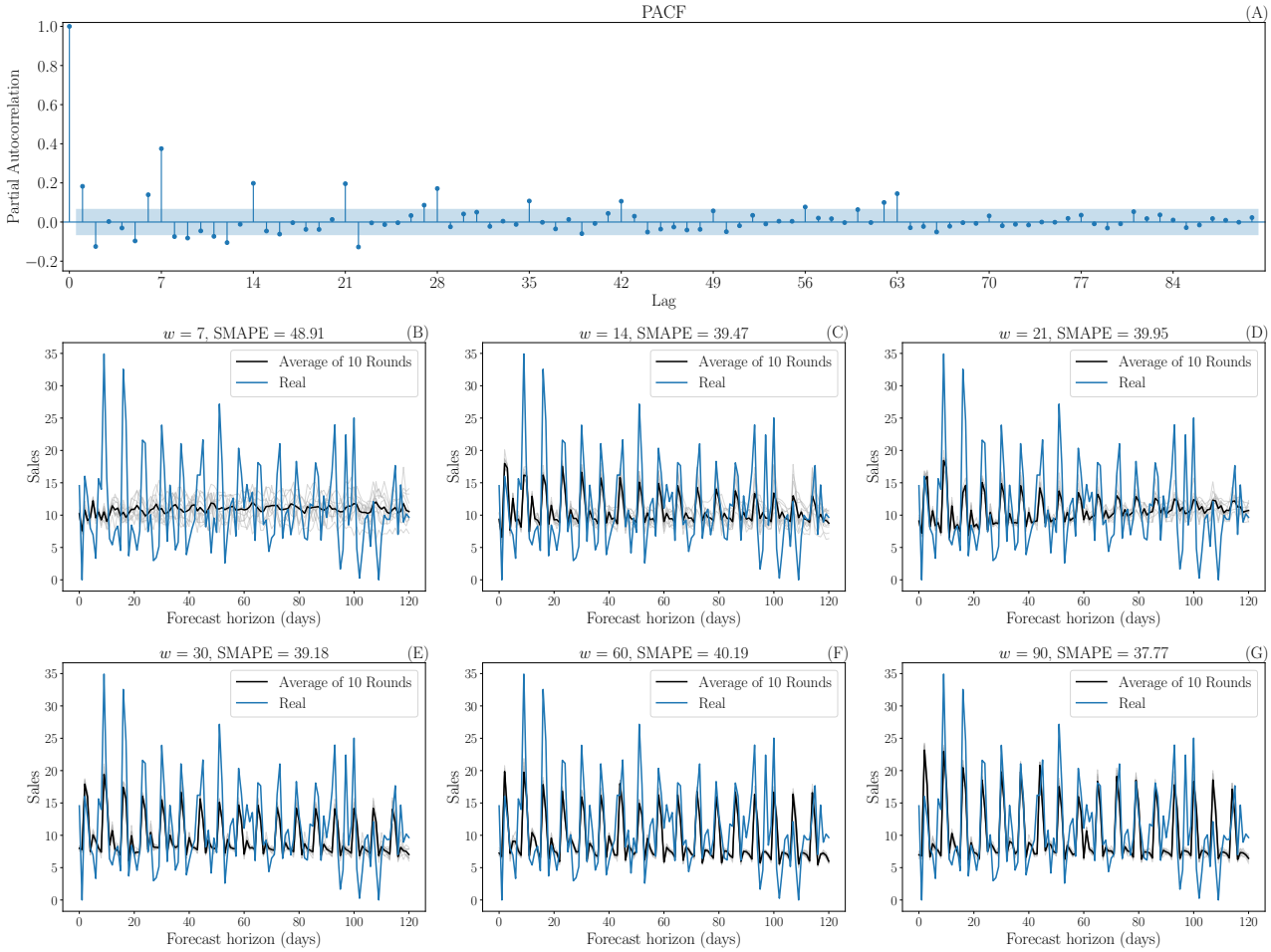

Figure 4. This panel presents various forecast, using Random Forest, analyses over time with different window sizes. Each graph shows the actual sales time series compared with forecasts using different window sizes: 7 days (B), 14 days (C), 21 days (D), 30 days (E), 60 days (F), and 90 days (G). In the graphs from (B) to (G), the blue line represents the actual data, the light gray lines represent the predictions of 10 iterations, and the black line represents the average of these 10 iterations' predictions. Panel (A) presents the Partial Autocorrelation Function (PACF) of the same product, highlighting the correlation of a time series with its own lagged versions, showing the most significant lags in the sales time series.

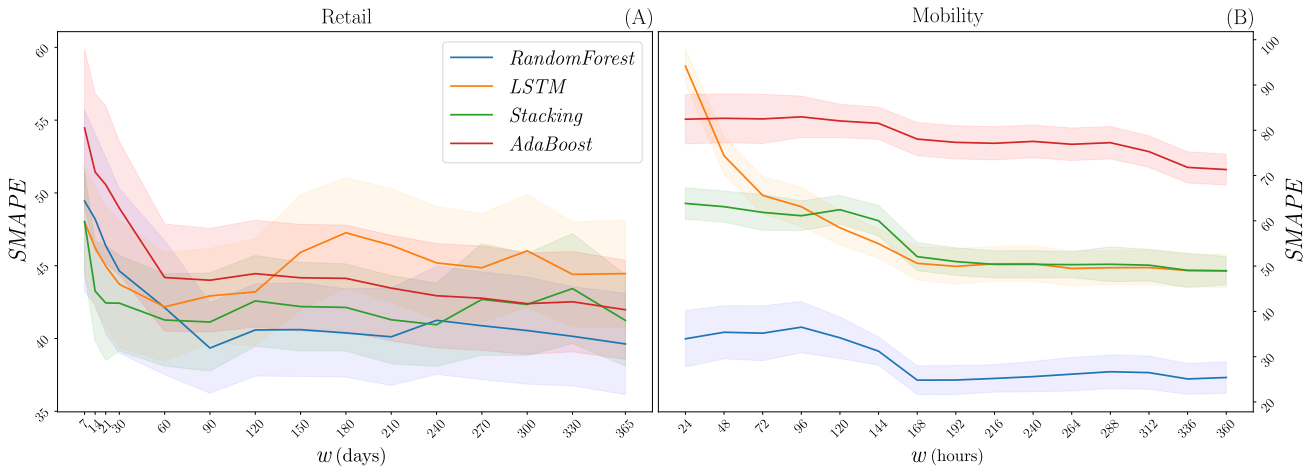

Figure 5. Model performance, with (A) showing the retail results and (B) showing the mobility results. The blue line represents the performance of *Random Forest*, the orange line represents the performance of *LSTM*, the green line represents the performance of *Stacking*, and the red line represents *AdaBoost*. The shaded areas represent the standard mean error for each model. On the x-axis is the window size, and on the y-axis is the *SMAPE* value.

.

5 Conclusion

This article focused on understanding the impact of window size on the performance of machine learning algorithms in univariate time series forecasting problems. Through experiments conducted on 40 time series from two different domains, using variations in window size across four machine learning algorithms, some important findings emerged.

It was observed that increasing the window size can lead to improvements in evaluation metric values up to a point of apparent stabilization. Beyond this limit, further increasing the window size does not result in better predictions. This suggests that there is a stabilization point in the window size that depends on each specific time series and the model being applied, beyond which additional historical data do not significantly contribute to model performance and can even lead to instability due to the increase in irrelevant attributes for modeling.

This finding highlights the challenge of generalizing a specific optimal window size across different domains or models, as the appropriate w is contingent on the interplay between the data and the learning algorithm. Therefore, our study reinforces the notion that window size optimization is a dynamic process, and practical applications should focus on identifying this stabilization point rather than seeking a universally optimal value.

The study revealed that Long short-term memory (LSTM) architectures did not outperform ensemble models in various univariate time series forecasting scenarios. While LSTMs are known for their ability to capture sequential dependencies and long-term patterns, ensemble models demonstrated similar or superior performance at different window sizes. Since univariate time series contain only a single variable over time, it is natural that there is less information available for the model to learn and capture complex patterns. LSTMs, due to their ability to handle sequential dependencies, can be particularly effective in forecasting time series with a large amount of multivariate data. In such cases, the model has the opportunity to explore correlations between different variables and capture more subtle nuances in temporal patterns. However, in univariate time series, this advantage may be minimized due to the lack of data variability. On the other hand, ensemble models leverage the diversity of different algorithms or data samples to improve predictive performance. This approach can help mitigate the challenges presented by univariate time series due to their limited nature. By combining multiple perspectives and learning strategies, ensemble models can compensate for the lack of multivariate data, resulting in equivalent or more accurate predictions.

Additionally, the analysis of the Partial Autocorrelation Function (PACF) proved to be a valuable tool for estimating the appropriate window size. Through PACF, it was possible to identify significant lags that suggest possible window sizes, aiding in the initial choice of w. However, PACF does not consider all the nuances of the data and the variables involved in predictive modeling. Therefore, the experimental approach, varying the window size, complements this analysis by providing an empirical evaluation of how different window sizes affect prediction accuracy. This robust approach is essential for choosing the ideal w value.

The results of this study indicate that the appropriate choice of window size is essential for accurate univariate time series forecasting and that a combination of theoretical analysis (such as PACF) and practical experiments provides a solid basis for this choice. These conclusions are especially relevant for practical applications in various domains, where accurate time series forecasting can lead to more informed and effective decision-making.

Funding

JF, CP, RB and CC have been funded by the Brazilian Agencies: Conselho Nacional de Desenvolvimento Científico e Tecnológico (www.cnpq.br) and Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (www.funcap.ce.gov.br).

Authors' Contributions

CC and CP contributed to the conception of this study. JF performed the experiments. All authors contributed to the writing of this manuscript. All authors read and approved the final manuscript.

Competing interests

The authors have declared that no competing interests exist.

Availability of data and materials

The data that support the findings of this study are available in Zenodo with the identifier https://zenodo.org/doi/10.5281/zenodo.12665354.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., *et al*. (2016). Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016)*, pages 265–283, Savannah, GA, USA. USENIX Association. DOI: http://dx.doi.org/10.48550/arXiv.1605.08695.

Abolghasemi, M., Beh, E., Tarr, G., and Gerlach, R. (2020). Demand forecasting in supply chain: The impact of demand volatility in the presence of promotion. *Computers & Industrial Engineering*, 142:106380. DOI: http://dx.doi.org/10.1016/j.cie.2020.106380.

Araújo, J. L., Oliveira, E. A., Lima Neto, A. S., Andrade Jr, J. S., and Furtado, V. (2023). Unveiling the paths of covid-19 in a large city based on public transportation data. *Scientific Reports*, 13(1):5761. DOI: http://dx.doi.org/10.1038/s41598-023-32786-z.

Azlan, A., Yusof, Y., and Mohsin, M. F. M. (2019). Determining the impact of window length on time series forecasting using deep learning. *International Journal of Advanced Computer Research*, 9(44):260–267. DOI: http://dx.doi.org/10.19101/IJACR.PID77.

Bergström, C. and Hjelm, O. (2019). Impact of time steps on stock market prediction with lstm.

Bomfim, R., Pei, S., Shaman, J., Yamana, T., Makse, H. A., Andrade Jr, J. S., Lima Neto, A. S., and Furtado, V. (2020). Predicting dengue outbreaks at neighbourhood level using human mobility in urban areas. *Journal of the Royal Society Interface*, 17(171):20200691. DOI: http://dx.doi.org/10.1098/rsif.2020.0691.

Box, G. E. P. and Jenkins, G. M. (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day, San Francisco.

Braga, D. J. F., da Silva, T. L. C., Rocha, A., Coutinho, G., Magalhães, R. P., Guerra, P. T., de Macêdo, J. A., and Barbosa, S. D. (2019). Time series forecasting to support irrigation management. *Journal of Information and Data Management*, 10(2):66–80. DOI: http://dx.doi.org/10.5753/jidm.2019.2037.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32. DOI: http://dx.doi.org/10.1023/A:1010933404324.

Caminha, C., Furtado, V., Pequeno, T. H., Ponte, C., Melo, H. P., Oliveira, E. A., and Andrade Jr, J. S. (2017). Human mobility in large cities as a proxy for crime. *PloS one*, 12(2):e0171609. DOI: http://dx.doi.org/10.1371/journal.pone.0171609.

Caminha, C., Furtado, V., Pinheiro, V., and Ponte, C. (2018). Graph mining for the detection of overcrowding and waste of resources in public transport. *Journal of Internet Services and Applications*, 9(1):1–11. DOI: http://dx.doi.org/10.1186/s13174-018-0094-3.

Caminha, C., Furtado, V., Pinheiro, V., and Silva, C. (2016). Micro-interventions in urban transportation from pattern discovery on the flow of passengers and on the bus network. In *2016 IEEE International Smart Cities Conference (ISC2)*, pages 1–6. IEEE. DOI: http://dx.doi.org/10.1109/ISC2.2016.7580776.

Chandra, R., Goyal, S., and Gupta, R. (2021). Evaluation of deep learning models for multi-step ahead time series prediction. *IEEE Access*, 9:83105–83123. DOI: http://dx.doi.org/10.1109/ACCESS.2021.3085085.

Cheng, H., Tan, P.-N., Gao, J., and Scripps, J. (2006). Multistep-ahead time series prediction. In *Advances in Knowledge Discovery and Data Mining: 10th Pacific-Asia Conference, PAKDD 2006, Singapore, April 9-12, 2006. Proceedings 10*, pages 765–774. Springer. DOI: http://dx.doi.org/10.48550/arXiv.2103.14250.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27. DOI: http://dx.doi.org/10.1109/TIT.1967.1053964.

De Gooijer, J. G. and Hyndman, R. J. (2006). 25 years of time series forecasting. *International journal of forecasting*, 22(3):443–473. DOI: http://dx.doi.org/10.1016/j.ijforecast.2006.01.001.

Drucker, H., Burges, C. J., Kaufman, L., Smola, A., and Vapnik, V. (1996). Support vector regression machines. *Advances in neural information processing systems*, 9:155–161.

Frank, R. J., Davey, N., and Hunt, S. P. (2000). Input window size and neural network predictors. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 2, pages 237–242. IEEE. DOI: http://dx.doi.org/10.1109/IJCNN.2000.857903.

Freitas, J. D., Ponte, C., Bomfim, R., and Caminha, C. (2023). The impact of window size on univariate time series forecasting using machine learning. In *Anais do XI Symposium on Knowledge Discovery, Mining and Learning*, pages 65–72. SBC. DOI: http://dx.doi.org/10.5753/kdmile.2023.232916.

Freund, Y., Schapire, R. E., *et al*. (1996). Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 29(5):1189–1232. DOI: http://dx.doi.org/10.1214/aos/1013203451.

Galton, F. (1889). *Natural inheritance*, volume 42. Macmillan, London.

Hamzaçebi, C., Akay, D., and Kutay, F. (2009). Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. *Expert systems with applications*, 36(2):3839–3844. DOI: http://dx.doi.org/10.1016/j.eswa.2008.02.042.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780. DOI: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67. DOI: http://dx.doi.org/10.1080/00401706.1970.10488634.

Huber, J. and Stuckenschmidt, H. (2020). Daily retail demand forecasting using machine learning with emphasis on calendric special days. *International Journal of Forecasting*, 36(4):1420–1438. DOI: http://dx.doi.org/10.1016/j.ijforecast.2020.02.005.

Kil, R. M., Park, S. H., and Kim, S. (1997). Optimum window size for time series prediction. In *Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society.'Magnificent Milestones and Emerging Opportunities in Medical Engineering'(Cat. No. 97CH36136)*, volume 4, pages 1421–1424. IEEE. DOI: http://dx.doi.org/10.1109/IEMBS.1997.756971.

Kreinovich, V., Nguyen, H. T., and Ouncharoen, R. (2014). How to estimate forecasting quality: A system-motivated derivation of symmetric mean absolute percentage error (smape) and other similar characteristics. *Departmental Technical Reports (CS)*.

Leites, J., Cerqueira, V., and Soares, C. (2024). Lag selection for univariate time series forecasting using deep learning: An empirical study. *arXiv preprint arXiv:2405.11237*. DOI: https://doi.org/10.48550/arXiv.2405.11237.

Liu, Y., Wang, Z., Yu, X., Chen, X., and Sun, M. (2022). Memory-based transformer with shorter window and longer horizon for multivariate time series forecasting. *Pattern Recognition Letters*, 160:26–33. DOI: http://dx.doi.org/10.1016/j.patrec.2022.05.010.

Makridakis, S. (1993). Accuracy measures: theoretical and practical concerns. *International journal of forecasting*, 9(4):527–529. DOI: http://dx.doi.org/10.1016/0169-2070(93)90079-3.

Nikolopoulos, K. and Fildes, R. (2013). Adjusting supply chain forecasts for short-term temperature estimates: a case study in a brewing company. *IMA Journal of Management Mathematics*, 24(1):79–88. DOI: http://dx.doi.org/10.1093/imaman/dps006.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V.,

Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., *et al*. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Ponte, C., Carmona, H. A., Oliveira, E. A., Caminha, C., Lima, A. S., Andrade Jr, J. S., and Furtado, V. (2021). Tracing contacts to evaluate the transmission of covid-19 from highly exposed individuals in public transportation. *Scientific Reports*, 11(1):24443. DOI: http://dx.doi.org/10.1038/s41598-021-03998-y.

Ponte, C., Melo, H. P. M., Caminha, C., Andrade Jr, J. S., and Furtado, V. (2018). Traveling heterogeneity in public transportation. *EPJ Data Science*, 7(1):1–10. DOI: http://dx.doi.org/10.1140/epjds/s13688-018-0172-6.

Salles, R., Belloze, K., Porto, F., Gonzalez, P. H., and Ogasawara, E. (2019). Nonstationary time series transformation methods: An experimental review. *Knowledge-Based Systems*, 164:274–291. DOI: http://dx.doi.org/10.1016/j.knosys.2018.10.041.

Shynkevich, Y., McGinnity, T. M., Coleman, S. A., Belatreche, A., and Li, Y. (2017). Forecasting price movements using technical indicators: Investigating the impact of varying input window length. *Neurocomputing*, 264:71–88. DOI: http://dx.doi.org/10.1016/j.neucom.2016.11.095.

Taieb, S. B., Bontempi, G., Atiya, A. F., and 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: http://dx.doi.org/10.1016/j.eswa.2012.01.039.

Teixeira, J. P. and Fernandes, P. O. (2011). A insolação como parâmetro de entrada em modelo baseado em redes neuronais para previsão de série temporal do turismo. In *VI Congresso Luso-Moçambicano de Engenharia*.

Ughi, R., Lomurno, E., and Matteucci, M. (2023). Two steps forward and one behind: Rethinking time series forecasting with deep learning. *arXiv preprint arXiv:2304.04553*.

Wolpert, D. H. (1992). Stacked generalization. *Neural networks*, 5(2):241–259. DOI: http://dx.doi.org/10.1016/S0893-6080(05)80023-1.

A Partial Autocorrelation Function (PACF) Analysis for Products and Bus Lines

This appendix presents the Partial Autocorrelation Function (PACF) plots for retail products (Figure 6) and bus lines (Figure 7). These plots provide insight into the temporal relationships and dependencies within the data for both datasets, highlighting how the current values of the series relate to past values at different time lags. The PACF is useful in identifying the direct correlation of each lag, which helps in determining the optimal number of past observations to include in the forecasting models.

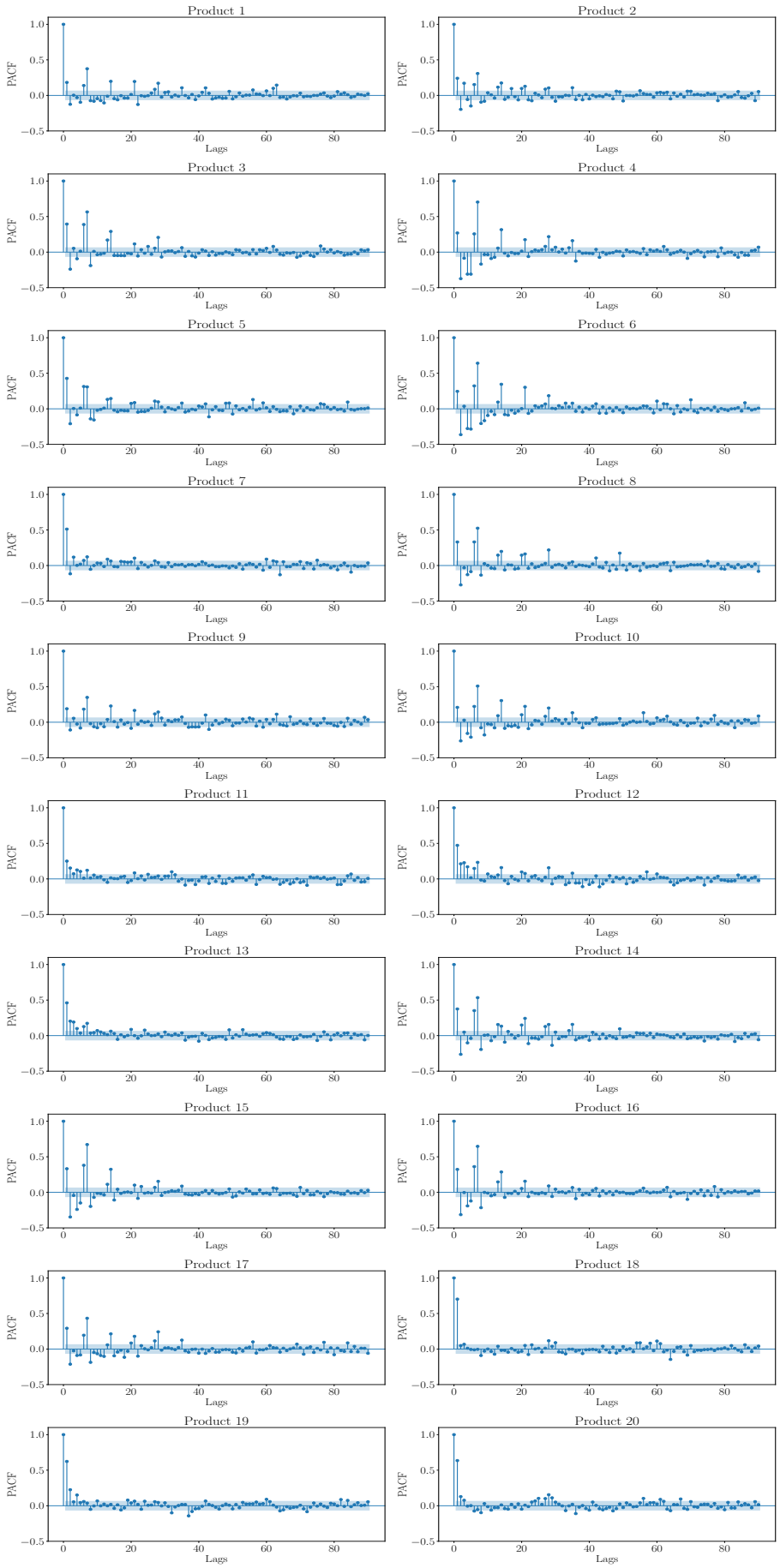

Figure 6. Partial Autocorrelation Function (PACF) for products. Each subplot represents the PACF for a specific product, indicating the temporal dependencies up to 90 lags.

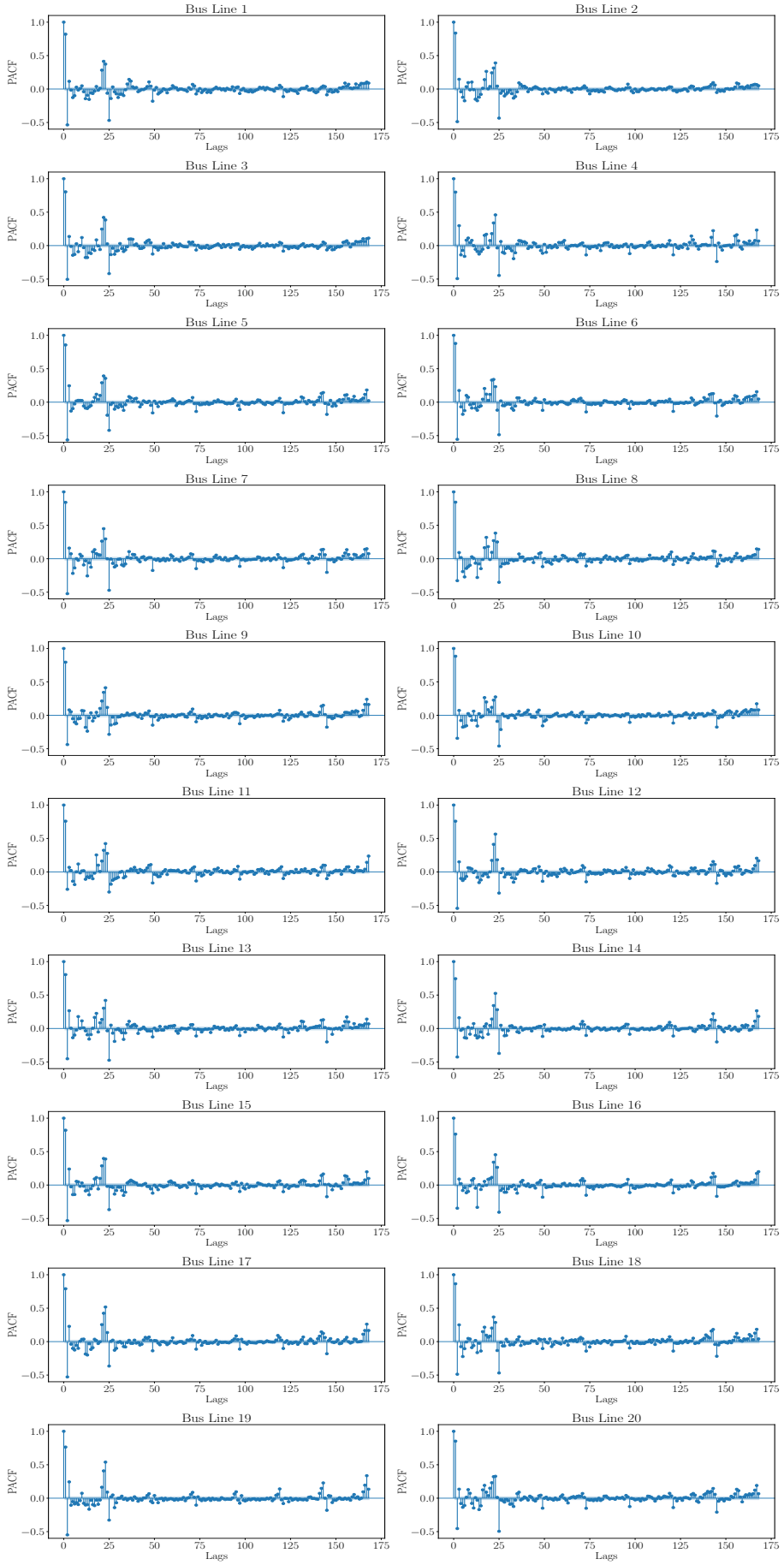

Figure 7. Partial Autocorrelation Function (PACF) for bus lines. Each subplot represents the PACF for a specific bus line, indicating the temporal dependencies up to 168 lags.