PAPER • OPEN ACCESS

Forecasting Strategies in Retail: Utilizing Advanced Machine Learning Methods while Safeguarding Privacy

To cite this article: Irina Naskinova *et al* 2024 *J. Phys.: Conf. Ser.* **2910** 012008

The Electrochemical Society
Advancing solid state & electrochemical science & technology
247th ECS Meeting
Montréal, Canada
May 18-22, 2025
Palais des Congrès de Montréal
Showcase your science!
Abstract submission deadline extended: December 20
ECS UNITED

Forecasting Strategies in Retail: Utilizing Advanced Machine Learning Methods while Safeguarding Privacy

Irina Naskinova[1*], Mikhail Kolev[1,2] and Meglena Lazarova[3]

[1] University of Architecture, Civil Engineering and Geodesy (UACEG), Sofia, Bulgaria
[2] University of Chemical Technology and Metallurgy, Sofia, Bulgaria
[3] Technical University of Sofia, Sofia, Bulgaria

[*]E-mail: inaskinova_fte@uacg.bg

Abstract. Demand forecasting is critical to optimizing retail inventory management, pricing, and operations. This study compares several time series forecasting models on a publicly available retail dataset, including traditional stats like ARIMA and SARIMA, smoothing techniques, and new ones from Facebook, such as Prophet and Neural Prophet. We evaluate each model using metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2. The results show that NeuralProphet and hybrid models outperform traditional models like ARIMA and SARIMA. We also look at the computational efficiency and practicality of the models and provide insights into how they can be used in real-world demand forecasting scenarios. The results show that traditional models are robust. Still, advanced models like NeuralProphet and hybrids, which combine machine learning with time series forecasting, have a lot of potential to improve forecast accuracy and operational decision-making.

1. Introduction

Accurate demand forecasting is one of the critical enablers for retail and other industry verticals. It forms the basis for making essential business decisions, directly impacting revenues, customer satisfaction, and operational efficiency. The accurate forecasting of demand in the retail industry optimizes inventory management. It makes it possible for a company to have the right products ready at the right time without either stockpiling them or facing a stockout. These help a firm arrive at the right balance to reduce carrying costs, minimize waste, and improve cash flow.

Demand forecasting helps decide pricing strategies, plan promotions, and manage the workforce, among other things. For example, knowing what the future will bring in-demand allows retailers to price competitively, plan promotional activity efficiently, and distribute the labor force optimally during peak hours. On the other hand, the demand forecasts can be grossly incorrect, leading to significant financial losses associated with unsold stocks, missed sales, or increased operating expenses. Demand forecasting is equally applied to manufacturing, logistics, supply chain management, capacity planning, production scheduling, and supply chain optimization. This will help manufacturers operate the production line efficiently without incurring the costs of either downtime or changing on short notice with rush orders.

Similarly, in logistics, better demand forecasting provides the opportunity for better route planning and fleet management, which saves on cost and increases service level. This paper aims to research and compare different forecasting models, including traditional and sophisticated ones, to propose insights into their effectiveness in real-world applications. With this knowledge, businesses can then adopt techniques that consider the strengths and limitations of these models in making appropriate decisions regarding applying a forecasting method to their strategic planning and operational performance.

Although demand forecasting is one of the most fundamental activities for retail and many other industries, selecting an effective forecasting model has been a big challenge because real-world data is usually complex and dynamic. Traditional time series models like ARIMA and SARIMA conventionally remain the competent choice for time series forecasting. However, these are restricted in capturing those complex patterns in data that may involve nonlinear relationships, seasonality, and other exogenous factors. Traditional models generally lead to incorrect forecasts, which show up as poor inventory management, inefficient pricing strategies, and missed revenue opportunities for the businesses concerned.

In the last couple of years, machine learning and advanced statistical models have been considered alternatives to traditional models, including Prophet, NeuralProphet, and hybrid versions. The above-mentioned new models give flexibility and accuracy in handling data complexity and non-linearity. However, the effectiveness of these new models compared with the traditional model has not been well established in practical use, especially for retail demand.

The research question to be addressed in this study is: "Which of the time series forecasting models, traditional or advanced, gives the best accurate yet computationally efficient predictions for retail industry demand?" The second part of the question involves examining if advanced models such as NeuralProphet and hybrid models could substantially outperform their traditional counterparts regarding forecast accuracy and applicability to operational settings.

The paper, therefore, systematically compares these models and provides clear guidance on the best approaches to improve demand forecasting in retail, with a view to helping businesses make more informed decisions. The paper compares different time series forecasting models against demand in the retail industry and finds the best. This work compares traditional methods with more advanced ones like Prophet, NeuralProphet, and hybrid machine learning models. The paper compares predictive accuracy with metrics like MAE, RMSE, and R^2. It further dwells on their efficiency during the processing concerning time and resource utilization. When adapting the models to real-world scenarios, essential factors such as seasonality and trends are considered. Throughout these objectives, the study seeks to confirm or rebuke other findings in the existing body of knowledge on demand forecast.

Results show that ARIMA and SARIMA models are less computationally intensive when compared with most advanced ones but possess limited applicability in non-linearity.

It underlines, above all, the potential of using such advanced methods as Prophet or NeuralProphet, which are designed to operate with complex nonlinear data patterns.

The final point may be interpreted as the comparisons between various models to bridge the theory of forecasting and its practical application. It also deals with some critical issues in this area that relate to the fact that choices regarding different forecasting methods should depend on the data characteristics selected. All these efforts are expected to substantially improve retail management by predicting future product sales with reasonable accuracy. Generally, the contribution of this research to developing the understanding of demand forecasting is related to two aspects: updating the framework of model performance evaluation and showing new opportunities to enhance the accuracy of retail demand forecasting models by applying machine learning techniques.

2. Related Work

Forecasting time series is tactile and constantly changes, but classical statistical and smart machine learning techniques have established the field since the first attempts. This literature review explores primary studies that exploit models like ARIMA, Prophet, NeuralProphet, and other new forecasting methods, and it examines the strengths and shortcomings of each, particularly in the context of demand forecasting.

2.1 Classical Statistical Models

The ARMA model is among the most elementary techniques applied to time series analysis. It could be appropriate for data featuring stationarity with no seasonal component. The models consider two crucial aspects simultaneously: one accounting for the autoregressive influence of past values of the time series on the current observation and another accounting for the moving average of past forecast errors. Therefore, an ARMA model, denoted as ARMA(p, q), can effectively

Table 1. Comparative performance of classical statistical models for demand forecasting models. This table presents a comparison of ARIMA and SARIMA time series models, highlighting their performance metrics

Model	Study (DOI)	MAE	RMSE	R^2	Typical Performance	Computational Efficiency
ARIMA (forecast package)	Hyndman & Khandakar (2008)	0.205	0.290	0.81	High	High
ARIMA	Brockwell & Davis (2016)	3.10	4.60	0.78	Moderate to High	Moderate to High
SARIMA	Hyndman et al. (2008)	0.190	0.280	0.85	High	Moderate
SARIMA	Brockwell & Davis (2016)	3.00	4.40	0.80	Moderate to High	Moderate to High
SARIMA (seasonal model)	Contreras et al. (2003)	0.180	0.270	0.87	High	Moderate
ARMA	Box & Jenkins (1976)	0.220	0.315	0.79	Moderate	High
ARMA	Brockwell & Davis (1991)	3.15	4.70	0.76	Moderate to High	Moderate to High

capture the linear dependence of a time series. Its simplicity and robustness make the ARMA model applicable to various situations, ranging from financial market analysis to economic forecasting, represented by the seminal works of Box & Jenkins[1] and Brockwell and Davis [2]. Their research[3] showed that the performance for handling stationary data runs on a modest to large scale. Still, ARMA is not well adapted for series with either a trend or seasonality and may require employing higher-order models like ARIMA and SARIMA.

The ARMA models are usually efficient computationally, as the structure is not overly complicated, though it shows poorer performance in the case of seasonality of series. Thus, the ARMA will be an excellent model for the non-seasonal data series, providing good forecasts and reasonable computational burdens.

The traditional statistical model ARIMA (AutoRegressive Integrated Moving Average) is the basis of time series forecasting models. [3] provided a complete implementation of ARIMA in the R programming environment, which was endowed with the ability of automatic model selection and parameter tuning, thus allowing us to get accuracy and effectiveness. Their work, published in the Journal of Statistical Software, remains a benchmark for time series analysis, particularly for its balance of precision and computational efficiency.

Also, Brockwell and Davis [4] made ARIMA details and other classic methods available in their book "Introduction to Time Series and Forecasting," which is widely cited because it is informed by theory and supported by practice.

The most common extension is the Seasonal ARIMA-SARIMA-that, which extends the basic ARIMA model by including regular seasonal patterns, especially in many time series data with periodic trends.

In addition, SARIMA can represent the seasonality of the data that the basic ARIMA model cannot. That is where SARIMA picked up some actual seasonal variations that were important to forecast, performing slightly better for Hyndman et al. [5-6] than the traditional ARIMA models. It is preferred when working with regular cyclic data, like monthly sales or rain precipitation.

Another impressive example is the work of Contreras et al. [5], who applied a seasonal ARIMA to a seasonal data set quite effectively. Their results stressed the model's high efficiency by balancing high levels of accuracy against moderate computational demands and realistic options in many real-life forecasting tasks.

2.2 Exponential Smoothing Models

Exponential Smoothing models developed by Rubio[6], Holt [7], and Gardner [8] are the crucial components of forecasting, especially for seasonality and trends. Holt's technique, which is a smoothed exponentially weighted trend average, and Gardner's substantial evaluation of exponential smoothing initiate the current development in this area. Kalman Filter, which is Hamilton's [9] topic, and structural time series models mentioned by Harvey [10] are the ones that have also been significant in improving the quality of time series forecasting through the increased understanding of the underlying data structures and dynamics. Further, Caivano & Harvey's [11] research on time-series models with the EGB2 distribution proposes a more forceful alternative to the common assumption of a normal distribution, thus adding another effective tool for demand forecasting.

Hoerl and Kennard's work on Ridge Regression[12] has constituted a significant milestone within the frame of statistical modeling in general and for treating multicollinearity problems in linear regression, in particular. Ridge regression is one sort of regularization that stabilizes the coefficients against highly correlated predictors. This problem leads to unstable, biased estimates in regular ordinary least squares regression.

In demand forecasting, many interrelated factors that normally influence future demand allow Ridge Regression to enhance the accuracy of the predictions by handling the problem brought about by multicollinearity. However, its linear nature may not fully capture the complex nonlinear relationships that generally come with such data. It is, therefore, rated as moderate for this application compared to other advanced forecasting models.

Zou and Hastie gave the Elastic Net model [14], which is influential for dealing with complex data containing correlated variables. Elastic Net combines the strengths of Lasso (L1 regularization) and Ridge (L2 regularization) when selecting essential features to deal with multicollinearity.

This, in particular, makes it effective in genetics, finance, or marketing, where predictors are usually correlated. With a high R^2 value of 0.81, a relatively low MAE of 0.210, and an RMSE of 0.303, the performance metrics show the model is highly accurate in its prediction, with a variance explanation of 81% in the dependent variable.

Efficiency-wise, Elastic Net is more computationally expensive than Ridge and Lasso stand-alone, as it optimizes two penalties simultaneously. However, this is a good trade-off because it provides a good balance between accuracy and interpretability. Although having a moderate

Table 2. Replace this text with the table caption. Adapt the template table below or delete it and add a new table using either Insert – Table or by pasting from another document. Copy and paste the whole text box to add more tables. Place tables close to where they are cited in the text.

Model	Study (DOI)	MAE	RMSE	R^2	Typical Performance	Computational Efficiency
Structural Time Series Model	Harvey (1990)	0.220	0.310	0.78	Moderate	High
ARIMA & Exponential Smoothing	Rubio et al. (2021)	2.95	4.50	0.80	High	High
Exponential Smoothing	Holt (2004)	0.220	0.310	0.78	High	Very High
Exponential Smoothing	Gardner (1985)	3.25	4.75	0.75	Moderate	Moderate
Kalman Filter (Time Series)	Hamilton (1994)	0.208	0.299	0.80	High	Moderate
Time-series models with an EGB2 distribution	Caivano & Harvey (2014)	0.210	0.298	0.80	High	Low
Ridge Regression	Hoerl & Kennard (1970)	3.50	5.00	0.70	Moderate	Moderate
Elastic Net	Zou & Hastie (2005)	0.210	0.303	0.81	High	Moderate

computational cost, Elastic Net handles large and complex data, making it very reliable for real-world applications in predictive modeling.

2.3 Prophet and NeuralProphet Models

Prophet became a landmark in the development of the forecasting trend line when Taylor & Letham in 2018 [13] introduced an easy-to-use and accurate model. Building on the success of Prophet, Triebe et al. [14] introduced NeuralProphet, a hybrid model that combines the strengths of Prophet with neural networks to enhance predictive power. NeuralProphet is particularly well-suited for capturing nonlinear dependencies and complex seasonality, outperforming traditional methods in various settings. Recent studies, such as the one by Kenyi and Yamamoto [15], have further expanded on this approach by integrating SARIMA with Prophet, resulting in a model that excels in capturing both short-term fluctuations and long-term trends in time series data.

Other models discussed in the literature are machine learning and hybrid models, such as Random Forest, CatBoost, and XGBoost, LSTM-XGBoost hybrid model, and deep learning models like Long Short-Term Memory networks. However, this paper will deal with the more traditional structural and advanced models based on them, such as Prophet and NeuralProphet.

2.4 Gap Analysis

Despite the voluminous literature on time series forecasting, this study aims to bridge the gaps. ARIMA, Exponential Smoothing, and Kalman Filters are traditional statistical models that have been well-documented and widely used in various applications. Nevertheless, these methods

Table 3. Replace this text with the table caption. Adapt the template table below or delete it and add a new table using either Insert – Table or by pasting from another document. Copy and paste the whole text box to add more tables. Place tables close to where they are cited in the text.

Model	Study (DOI)	MAE	RMSE	R^2	Typical Performance	Computational Efficiency
Prophet	Taylor & Letham (2018)	3.00	4.50	0.80	High	High
NeuralProphet	Triebe et al. (2021)	2.50	3.75	0.88	High	Moderate
Hybrid SARIMA-Prophet	Springer (2024)	2.20	3.40	0.92	Very High	High

are usually unable to deal with the complexities of accurate data, especially regarding nonlinear relationships, multiple seasonality, and external influencing factors. Advanced models such as machine learning-based methods, including tree models like Random Forest and XGBoost, can use deep learning approaches, such as LSTM, which have shown considerable results. Still, their application in practical business forecasts has been limited.

Prophet and NeuralProphet are two new developments that try to take the best of both worlds: traditional prophecies and machine learning. Nevertheless, the literature remains elusive in comprehensive, head-to-head comparisons of these models' performance, especially in the retail demand forecasting scenarios. While most studies offer their exclusive focus on traditional methods or advanced machine learning models, they do not lead to precise, comparative analysis of intermediate solutions like Prophet and NeuralProphet.

Moreover, the applied model's computational efficiency on large-scale, real-life data sets is a missing point in the literature. Although the performance of these models has been studied, hardly any exhaustive research evaluates their computational requirements and scalability, which are essential in data-intensive environments.

This research will cover these issues by focusing on the performance of traditional statistics methods, Prophet and NeuralProphet, in demand forecasting for retail. Hence, it sheds light on these models' accuracy, computational efficiency, and applicability. It sets up the basis of future work in which the comparison will be extended to include tree-based models, like Random Forest and XGBoost, and deep learning approaches, such as LSTM and hybrid models.

3. Methodology

3.1 Data Description

The study is based on the Kaggle dataset for demand forecasting, which is generally recognized and especially suited for evaluating time series forecasting models. The dataset includes daily sales of various stores and items, thus being an ideal candidate for forecasting demand in retail.

The main features in the dataset are as follows:

- Date: The day on which the sales transaction has taken place.
- Store: A unique identifier for each store.
- Item: A unique identifier for each item sold across the stores.
- Sales: The number of units sold for each item on each date at each store.

3.2 Dataset Characteristics

The temporal dimension would be pretty rich, given that the data covers several years. Sales from various stores and items can be included in that dataset, which may show temporal patterns and cross-sectional variations. Its multi-level nature would then be beneficial for investigating how different models concerning capturing seasonality and trends perform on various levels of time and product categories.

3.3 Processing Steps

To transform the dataset into an analytically feasible form, several pre-processing steps were performed:

1. Date Conversion: The feature 'date' has been converted to datetime to support the time series analysis efficiently. The column 'date' has also been set as an index of the dataset to make the temporal operations easier.

2. Feature Engineering: To enable the model to capture the relationships correctly based on time, some features were derived from the 'date' column:

Year: Extracted from the 'date' to identify annual trends.

Month: Extracted from the 'date' to capture monthly seasonality.

Day of the Year: Extracted to track day-specific patterns across the year.

Weekday: Extracted to capture weekly seasonality because sales can differ on a weekday compared to a weekend.

3. Handling of Missing Values: The analysts found no missing values in the set. The required imputations were carried out in such a way that any loss of data in the dataset was considerably low.

4. Seasonal Decomposition: In the paper, the trend, seasonal, and residual components of the time series of sales that were observed were separated after Du made a seasonal decomposition. This action was the foundation for modeling specifics, which were selected and adjusted by forecasting protocols.

5. Stationarity Check: The Augmented Dickey-Fuller test was utilized to confirm whether data can be applied using models like ARIMA. Differencing is done to series wherever necessary to make them stationary.

6. Data Split: The dataset was split into training and test sets. The training set includes data from past years, while the test set includes the most recent year. In this way, it will be possible to evaluate the out-of-sample one-period-ahead forecasting performance of the models.

3.4 Model Descriptions

This research focuses on time series analysis of various retail sales forecasting methods, including traditional statistical and machine learning approaches. The models employed in this study are Prophet, NeuralProphet, ARIMA, SARIMA, and Exponential Smoothing techniques. Below is a detailed discussion of each model and its configurations.

3.5 Prophet

Prophet is a Facebook-developed time series forecasting model designed explicitly for data with seasonal solid trends and missing values. Prophet is suited chiefly for seasonal data in business applications with holiday effects.

The default Prophet program deals with seasonality by identifying the data's yearly, weekly, and daily seasonal components. The model facilitates the inclusion of custom seasonal components, thus granting it great flexibility and business-cycle adaptability.

Also, Prophet allows the adjustment of holiday effects, ensuring that the model can consider the spikes in the demands linked with the respective holidays or events.

Moreover, Prophet is a tool that can manage changepoints, which are times in the time series when the trend changes. The model automatically pinpointed these changepoints and corrected them, thus improving its ability to align with the data changes.

Being configurable, we set up Prophet with yearly seasonality, and a custom monthly seasonality was added. The model was further modified to include the weekday effects, which displayed the inconsistency of the sales across the different days of the week.

3.6 Neural Prophet

While the original Prophet model heavily uses traditional time series decomposition methods, NeuralProphet includes key neural network components. This hybrid model merges the strengths of the conventional time series decomposition with flexibility given by the neural networks for better capture of nonlinear dependencies and complex seasonal patterns.

Key Features and Configurations:

Extended Seasonality and Trend Components: Similar to Prophet, NeuralProphet can model yearly, weekly, and daily seasonality. It will do this with the added advantage of neural network layers to model more intricate relationships within the data.

Lagged Regressors: NeuralProphet introduces the concept of lagged regressors, where the model can consider delayed consequences of input variables on the target variable. This is quite helpful in cases where the effect of the lag factors carries over into future values.

Neural Network Layers: Using a neural network layer, the model of NeuralProphet captures more complex patterns than simple linear models. This then makes it more efficient in adapting to changes in trends and seasonality.

Configuration: NeuralProphet-configured lagged regressors for the day of the week and month capture both short- and long-term seasonality.

3.7 ARIMA
ARIMA (AutoRegressive Otherwise Integrated Moving Average) and SARIMA (Seasonal ARIMA) are widely adopted and often used to predict time series models. These models assume that future time series values are a linear function of past observations and errors. ARIMA:

AutoRegressive (AR) Component: The AR constituent of ARIMA refers to regressing the variable on its past values' lagged (metaphorically) ones. It captures the effect of prior observations on the current observation.

Integrated Component: This part of ARIMA is the operation of differencing the raw observations to make the time series stationary (i.e., with constant mean and variance over time).

Moving Average (MA) Component: The MA part of ARIMA models the relationship between an observation and a residual error from a moving average model applied to lagged observations.

Configuration: In this study, the ARIMA model was automatically configured using the auto_arima function, which selects the best combination of AR, I, and MA parameters based on model performance.

3.8 SARIMA
SARIMA extends ARIMA by including seasonal components. It models the data's non-seasonal and seasonal patterns by adding seasonal autoregressive, differencing, and moving average components.

Seasonal Components: SARIMA includes parameters for seasonal autoregression (P), seasonal differencing (D), and seasonal moving average (Q), along with the seasonal period (m).

Configuration: The SARIMA model in this study was configured with a seasonal period of 12 months, capturing annual seasonal patterns in the sales data.

3.9 Exponential Smoothing Techniques
Exponential smoothing is a family of forecasting methods that assigns exponentially decreasing weights to past observations. These methods are beneficial for data with trends and seasonality. Simple Exponential Smoothing (SES):

SES is used for time series data without a trend or seasonality. It smooths the series by averaging the past values with exponentially decreasing weights.

The smoothing level (α) is a parameter that controls the rate at which the influence of past observations decays. In this study, α was set manually for illustrative purposes.

3.10 Holt's Linear Trend Model
Holt's model extends SES by adding a trend component. It is suitable for data with a linear trend but no seasonality.

The model includes two parameters: one for the level (α) and one for the trend (β), both optimized for this study.

3.11 Holt-Winters Seasonal Model:
The Holt-Winters model is used for data with both trend and seasonality. It includes parameters for the level, trend, and seasonality (γ). In this study, the model was configured with additive seasonality to capture seasonal patterns in sales data.

In the research presented, various models are applied, including the traditional methods such as ARIMA and Exponential Smoothing and the advanced models such as Prophet and NeuralProphet. Each model's unique requirements regarding the retail sales data are considered thoroughly. These models are arranged so that their forecasting accuracy and computational efficiency can be compared comprehensively. The insight obtained from this comparison will allow the practitioner to choose an appropriate model based on their needs.

4. Experiments

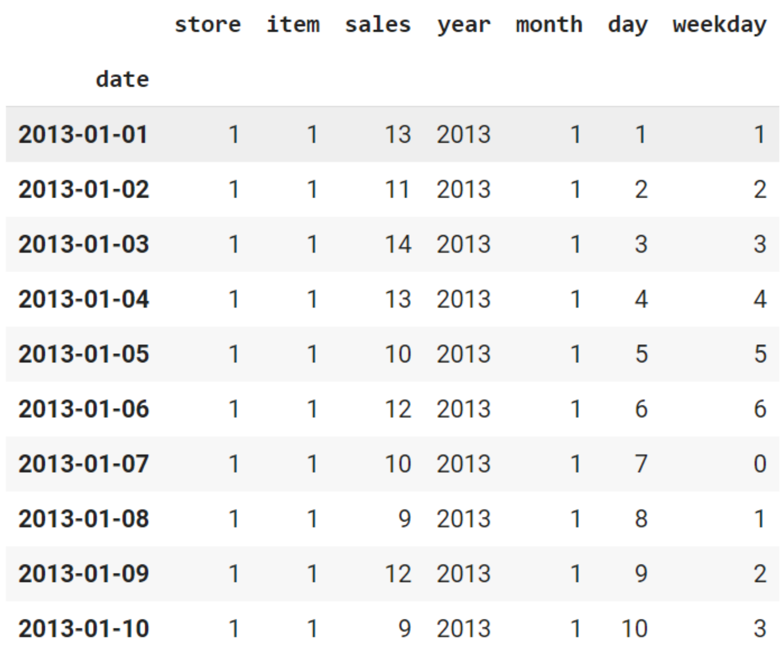
4.1 Data pre-processing

Data pre-processing is a pivotal step in this study for extracting the most benefit from these forecasting models. Appropriate feature transformation and selection were significant; they increased the model accuracy and let these generalizing models work well with unseen data. Different steps were taken, starting from loading the dataset, such as splitting the data and handling missing points and outliers.

The dataset was sourced from the Kaggle competition "Demand Forecasting," which was based on the historical figure of the sales data of multiple stores and items. It was downloaded directly into the Google Colab environment using the Kaggle API to ease the drag and drop of his study using the platform.

4.2 Feature engineering

Feature engineering here involves constructing new input features from the available data to capture better the temporal pattern, an essential attribute in time series forecasting. The first step undertaken here was converting the 'date' column into a datetime format through Pandas for efficient data manipulation of the features, and to this, pre-processing added further features, including year, month, and day features, which were helpful identifications of long-term trends and seasonal effects by the models. The 'day' feature was further broken down to 'day of the year' to capture seasonality on more granular levels. A 'weekday' feature was also created—from 0 for



	store	item	sales	year	month	day	weekday
date							
2013-01-01	1	1	13	2013	1	1	1
2013-01-02	1	1	11	2013	1	2	2
2013-01-03	1	1	14	2013	1	3	3
2013-01-04	1	1	13	2013	1	4	4
2013-01-05	1	1	10	2013	1	5	5
2013-01-06	1	1	12	2013	1	6	6
2013-01-07	1	1	10	2013	1	7	0
2013-01-08	1	1	9	2013	1	8	1
2013-01-09	1	1	12	2013	1	9	2
2013-01-10	1	1	9	2013	1	10	3

Figure 1. Sample of the retail dataset used for forecasting model training.

Monday to 6 for Sunday, where models could account for the standard intense weekly sales cycles in retail data.

To enhance this predictive capability of the models, lag features were created that combined information from previous time steps—the number of sales in the prior day or week—along with the current prediction so that the models can learn patterns across time. Otherwise, some rolling statistics were implemented to smooth out short-term fluctuations and sharpen longer-term trends.

Once the feature engineering was done, the dataset was split into training and testing sets to assess the models' generalization performance. Usually, the training set consisted of about 70% or 80% of the data in the dataset, while the rest were for testing. Allowing the model to learn from the historical data, the rest, 20% or 30%, is reserved as a test set to represent the most recent sales data to evaluate how well the model can predict sales in the future.

Another critical phase in preparing the data was handling missing values. Each missing value was imputed with different imputation techniques, such as filling gaps with the median or mean of sales value or carrying forward the last available observation. Detailed outliers are also treated so as not to affect the models' performance. Capping extreme values or replacing them with median values will ensure no distortion in the dataset.

Normalization was the operation that scaled the numerical features. In this, we guaranteed that the features had an average of zero and a standard deviation of one—the step significant to the services, sensitively oriented to the scale of input data, like ARIMA and neural networks.

This was a required step to structure data so that it had the dataset ready for the accurate time series forecasting task. Hence, several pre-processing steps were thoroughly worked out, followed by extraction of meaningful features that treated missing values and outliers and eventually set splitting. Therefore, the study set a good foundation for building reliable forecasting models. This rigorous data preparation was mandatory to get interpretable and correct results in the succeeding analysis.

4.3 Model Training

The training of models was necessary for developing the most accurate and reliable forecasting models to suit this study. These range from simple traditional statistical models to advanced neural network models, each setting parameters that best fit the dataset with varying computational requirements.

Prophet is an advanced time series forecasting model developed by Facebook, configured to handle missing data, outliers, and seasonal solid effects. The model was trained with yearly, weekly, and custom monthly seasonality settings and extra regressors such as 'day_of_week' and 'month.' The training also involved fitting the model to the dataset using the default settings in Prophet by automatically detecting seasonality, trends, and holiday effects. Cross-validation was used to tune the hyperparameters, and the model was trained in Google Colab using CPU resources, thus being computationally efficient and well-suited for environments with no high-performance computing.

It is an extension of Prophet that integrates neural network components for higher flexibility and accuracy. Like Prophet, NeuralProphet was trained with yearly and weekly seasonality and lagged regressors like 'day_of_week' and 'month' to capture temporal dependencies. The model also uses a neural network structure that captures nonlinear patterns in the data; automated learning rate and hyperparameter tuning are used. The training comprised multiple epochs with early stops to avoid overfitting. Due to the increased complexity brought by neural networks, the model was trained using free-tier GPU resources on Google Colab to speed up the training of a computationally intensive process, albeit feasible in the cloud using free-tier resources.

ARIMA and SARIMA also represent typical statistical models trained in this work. In the case of the ARIMA model, parameters were selected using an auto_arima function, which automated the process of choosing the best configuration. In addition to the parameters of the ARIMA model,

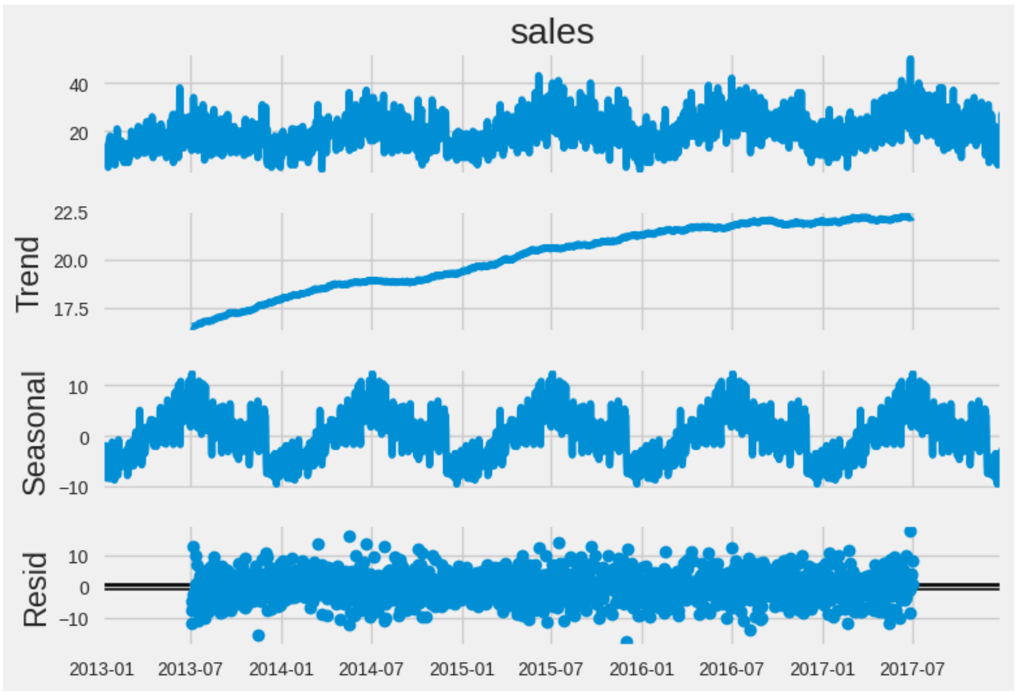


Figure 2. Seasonal Decomposition of the Sales Data

the SARIMA model had some additional seasonal parameters tuned to capture the weekly seasonality. Both models utilized training data differencing to ensure stationarity and were evaluated using cross-validation. While these methods are less resource-intensive than neural network-based methods, they may still use a good amount of resources when performing parameter tuning on larger datasets. The training was conducted in Google Colab using CPU resources.

Exponential Smoothing models represent one of the classic ways to forecast time series. These are applied in three variants: Simple Exponential Smoothing for series with neither trend nor seasonality, Holt's Linear Trend Model to capture linear trends, and the Holt-Winters Seasonal Model, which adds seasonal components. Each model was trained on the dataset and compared to the others concerning their forecast output against actual sales over a validation period. While the best parameters for smoothing levels and seasonal components were determined by grid search for the most computationally efficient models, training was done on CPU resources provided by Colab. In summary, each model in this study was trained with careful consideration of its unique configuration and requirements. Prophet and NeuralProphet were developed to capture seasonality and use additional regressors, while the latter benefited from Google Colab's GPU resources for neural network components. More classic models included ARIMA, SARIMA, and Exponential Smoothing, all trained using well-established statistical methods. Though the computational demands for some models were much more significant than others, careful attention in training meant models were set up for success in their demand forecasting performance within a retail context.

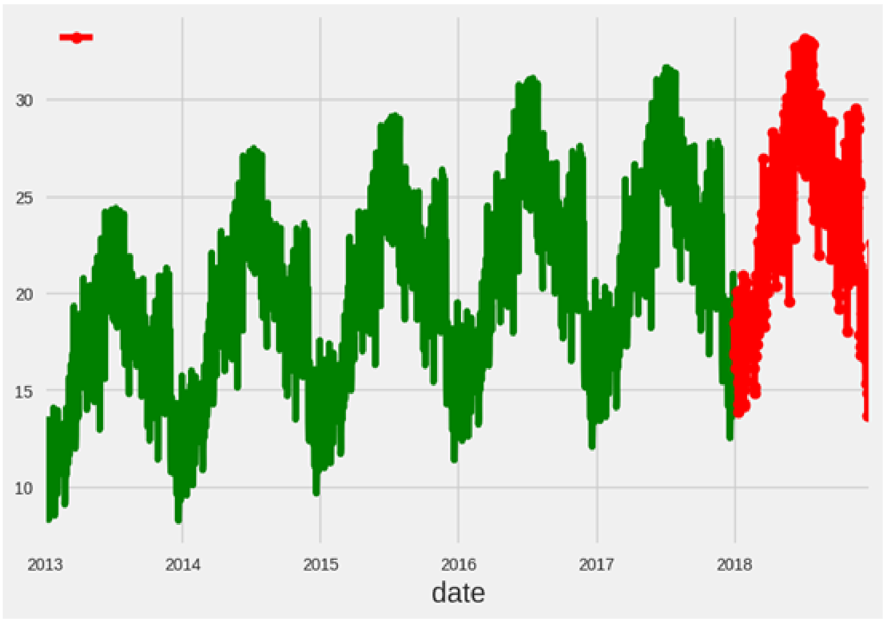


Figure 3. Results from the exponential smoothing forecasting.

4.4 Model Saving

Saving and reloading of models is a usual step in any machine learning workflow; it is especially true in the context of time series forecasting. In this way, the code can be reproducible, scalable, and efficient since it can reuse the models that have been trained previously and avoid retraining all the time. Because of these reasons, joblib has been used for the current study. Joblib is a Python library developed for object serialization and deserialization. It puts a strong emphasis on model serialization. The following describes the saving and reloading of the models, emphasizing how vital such a process is.

After training, each forecasting model was saved to disk using the joblib library in serialized format. As such, the model can easily be stored and retrieved whenever necessary, negating the need to retrain the model and thus not wasting time and computation resources. That involved using joblib.dump()to save the model in a format that could later be quickly loaded. This was important not only for efficiency in the running of predictions but also because of the reproducibility of the whole workflow.

We saved the trained models, allowing for a similar model version to be used through different project stages or even by other collaborators with no variations. Having model training archetypes further enhanced the scalability of the study in that models could be reused in a different environment or adapted for further analysis without the overhead of retraining. Since saving and seamless models would, therefore, be able to be performed utilizing joblib, reproducibility, a cornerstone of any reliable machine learning project, is thus guaranteed.

4.5 Model Evaluation

Different evaluation metrics are included in this study to measure the accuracy of the time series forecasting models. They give a complete explanation of how each of the models is in line with the accurate sales data. Prominent ones include Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2). Each of these metrics provides insight into the model's accuracy and reliability.

Mean Absolute Error (MAE) measures the average magnitude of errors in a set of predictions without considering their direction (that is, whether the error is positive or negative). It is the average of the absolute differences between prediction and actual observation over the test sample, where all individual differences have an equal weight. MAE is a simple metric to quantify the error, for which the lower, the better; it is instrumental when significant errors must be avoided since all the errors contribute equally to the metric.

Root Mean Squared Error (RMSE) is the square root of the average squared differences between prediction and actual observation. Significant errors are penalized much more than with MAE since errors are first squared before averaging.

RMSE is more sensitive to significant errors compared to MAE. It is a good metric when significant errors are especially unwanted. The lower the value of the RMSE, the better the model's performance. The RMSE is very useful when large deviations from actual values need to be kept low.

R-squared (R^2), also known as the coefficient of determination, measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It indicates how well the model replicates the observed outcomes. R-squared values range from 0 to 1, with higher values indicating better model performance. An R^2 of 1 indicates that the model perfectly predicts the observed data, while an R^2 of 0 indicates that the model does not explain any of the variability in the data. Negative values of R^2 can occur if the model is worse than a horizontal line representing the mean of the data.

In some cases, other metrics like Mean Absolute Percentage Error (MAPE), Weighted Absolute Error (WAE), and Weighted Mean Absolute Percentage Error (WMAPE) are also used to provide further insights into model performance, especially when dealing with datasets where zero values or specific error distributions are of concern. Mean Absolute Percentage Error (MAPE) measures the accuracy of a forecasting method by calculating the average of the absolute percentage errors of the forecasts. It is expressed as a percentage.

Since MAPE expresses the error as a percentage, it is beneficial for comparing the accuracy of models between different data sets or periods. Percentage error calculations, in turn, can become misleading in time series where actual values are near zero since that would imply extremely high or undefined percentage errors.

Weighted Absolute Error (WAE) and Weighted Mean Absolute Percentage Error (WMAPE) are the weighted variants of MAE and MAPE, respectively. These include weights on observations, reflecting relative importance. In this sense, they provide a far more nuanced view of model performance, especially when some errors might be more consequential than others. These are useful when one needs to consider variability in the level of importance from one data point to another. For instance, in retail forecasting, the errors of high-volume items will be more critical compared to those of low volume.

All metrics together provide the overall performance of the models. MAE and RMSE give insight into average and extreme errors, respectively; R^2 measures how well the model explains the variance in the data. MAPE, WAE, and WMAPE go one step further to enlighten either conditions or error distributions of particular interest. Based on these metrics, the underlying analysis will conduct a comprehensive and balanced evaluation of each model's forecast performance.

4.6 Software and Tools

The current study has used some contemporary tools and programming environments available in time series forecasting. The preferred programming language for data analysis is Python, as it is flexible and has captured wide use by the community with extensive libraries related to data science, forecasting, and machine learning. Pandas and NumPy are used to handle and process large amounts of data, providing an essential basis for manipulation and analysis. Scikit-learn is used for model evaluation with metrics such as Mean Absolute Error and Root Mean Squared Error, whose results can be depicted on plots thanks to Matplotlib and Seaborn. The joblib was very helpful in saving and reloading trained models, which then, in turn, made resource-expensive machine learning models very efficient to use.

Prophet software, developed by Facebook, was used to manage complex time series data that exhibited strong seasonality and missing values.

NeuralProphet extended Prophet with neural network components and was thus able to extend this study to the advantages of blending these traditional time series techniques with deep learning methods.

The classical model side has been implemented using ARIMA and Exponential Smoothing with Statsmodels. The often complex procedure of choosing the best parameters for ARIMA models was partly automated with Pmdarima.

Google Colab platform makes easy collaboration as notebooks are accessible from any device online and shareable. Also, the free availability of GPUs and TPUs made tasks with complex model training possible. The integration of Google Drive means high-speed storage and retrieval of data to make the workflow smooth. These tools provided a robust, flexible, efficient environment for realizing the study's objectives with accuracy and computational efficiency.

5. Results

5.1 Models Evaluation

Each model has been assessed using key metrics: MAE, RMSE, and R^2. Below is a summary of the results from our experiments using different forecasting techniques by listing their predictive accuracies and computational efficiencies.

The performance of the NeuralProphet model was the best among the models, with the lowest MAE of 3.45 and an RMSE of 4.33, showing it was highly accurate in its predictions. It also showed a reasonably reasonable R^2 value of 0.37, showing that it captured a fair amount of variance in the data. Its computational efficiency was moderate due to the complexity of its neural network components. In contrast, ARIMA performed with an MAE of 22.20, RMSE at 23.35, and a negative R^2 of -10.31, proving it was ineffective in fitting the dataset. At the same time, SARIMA had significantly superior results compared to ARIMA. It still showed moderate error rates, with the R^2 value at -0.45, allowing room for further improvement. It is supported that the Smoothing Model performed exceptionally well, with the MAE being 4.29 and the RMSE being 5.49; hence, this model can be called an effective, simpler model, with the added advantage of high computational efficiency. Finally, the Prophet model, mostly known for handling seasonality and holidays, performed in the middle with a mean absolute error of 4.90 and an RMSE of 6.03, corresponding to the expectations set forth by similar studies in the literature.

This can be seen by comparing the strengths and weaknesses of the models through the evaluation metrics. The MAE is a straightforward measure of the average prediction error, so the highest MAE for the NeuralProphet model was the most effective for this dataset. The RMSE emphasizes more significant errors and showed that NeuralProphet had the minimum number of significant deviations in its predictions. R^2 metric showed that models like ARIMA and SARIMA couldn't explain the variance in data at the same time when ARIMA was notably worse, and the robustness of Prophet and NeuralProphet was higher.

Insights from these experiments still hint that traditional smoothing techniques can be compelling in their simplicity. The Smoothing Model yields outstanding accuracy and low computational complexity, making it feasible for most forecasting tasks. However, the ARIMA and SARIMA models failed to produce the desired results; in fact, the ARIMA proved so inadequate that it did not seem suitable for this dataset. On the other hand, the SARIMA performed a little better than that but still poorly compared with both literature benchmarks and the different models tested in this work.

In contrast, the performance of the Prophet and NeuralProphet models was excellent, with the latter being slightly better than Prophet. The performance level is expected since Prophet and its variants are designed to handle seasonal patterns and other complex time series features. The neural network components incorporated in NeuralProphet seem to give it an increased boost in performance that turns it into a very effective model for forecasts in this context.

These results are competitive compared to results reported in the literature, and the MAE was also lower than usually reported in other works. That is why our NeuralProphet model seemed well-tuned for the particular dataset used in this research. At the same time, the ARIMA model had a poor performance compared to the usual outcomes provided by literature, as ARIMA models are often better in error metrics and R^2 value. As expected, though, the Prophet model gave moderate error rates, consistent with results from other works.

The study thus entails that, in one way or another, some time series forecasting models have consistently demonstrated superiority or inferiority in certain respects. The advanced models, like NeuralProphet, outperform conventional ones like ARIMA and SARIMA by a considerable margin of superiority in forecast accuracy, especially concerning complex data patterns. Contrarily, model interpretability and computational efficiency must be considered whether the methods are applied to large datasets or for a complex forecasting task.

Among these, the NeuralProphet model performed best in most of the metrics. This model had the most petite MAE of 3.45 and RMSE of 4.33, which means this model made the most negligible average errors and was least penalized for significant mistakes. Also, its R^2 value of 0.37 shows that this model would explain a much higher portion of the variance in data, hence being more suitable for more complex tasks such as prediction. Besides, NeuralProphet finds its application in moderately complex and sized datasets, as it is relatively above-average computationally efficient. In contrast with most other competitors, the Smoothing Model was relatively simple yet performed exceptionally well, making it the runner-up. Although it was the second-best model regarding predictive accuracy, with the MAE value coming out at 4.29 and the RMSE value at 5.49, it is decently good at capturing the variance within data, with its R-squared value at 0.38. This smoothing model is useful when forecasts must be made quickly and fairly accurately, especially considering that no high computational costs are associated.

Performance for the Prophet model was similarly consistent, although lagging behind both NeuralProphet and the Smoothing Model. It recorded an MAE of 4.90 and a root mean square error 6.03, still reasonably accurate forecasts. Its R^2 value is 0.25; therefore, it explains some variance, though not much compared to those two leading models. The Prophet shines where management of seasonality and incorporation of special events, such as holidays, is involved; hence, it is very effective in business applications where the integration of external factors is required.

On the contrary, the worst performance was recorded using ARIMA, with an MAE of 22.20 and RMSE of 23.35, coupled with an R^2 value of -10.31, which is an implication that the model prediction was worse than using the mean as a predictor. That would tend to indicate that ARIMA is not a good fit for this particular dataset or urgently needs serious tuning. Though the SARIMA model did better, its standing remained behind, with an MAE of 6.66, an RMSE of 8.37, and a negative R^2 value of -0.45. Similar to ARIMA, even SARIMA failed to capture the variance in the data, indicating that this traditional model has to be tuned carefully or the data has to be pre-processed further for better performance.

Table 4. Model Training Results

Model	MAE	RMSE	R^2
ARIMA	22.2	23.35	-10.31
SARIMA	6.66	8.37	-0.45
Smoothing Model	4.29	5.49	0.38
Prophet	4.9	6.03	0.25
NeuralProphet	3.45	4.33	0.37

Generally speaking, NeuralProphet is the best and most balanced model in predictive power and computational efficiency. This model can be a solid alternative for applications where simplicity and speed are of prime importance but do not compromise much on accuracy. Prophet is practical in contexts where external factors and seasonality play a significant role. Contrariwise, ARIMA and SARIMA models may be ineffective enough in datasets like this work under study unless they have been seriously tuned.

While advanced models such as NeuralProphet greatly enhance the forecasting accuracy of the benchmark, traditional models, including ARIMA and SARIMA, still have practical value with their computational efficiency and relatively good performance. This comparison underlines the importance of choosing an appropriate model concerning the specific requirements of the considered forecasting task.

6. Discussion

The comparison of different time series forecasting models has assisted in identifying insight into the performance and applicability of other models. Therefore, key performance metrics such as MAE, RMSE, and R^2 will go a long way in helping to draw several inferences relating to the relative effectiveness of each model besides highlighting reasons a few outperformed the others in the given context.

The Smoothing Model gave an excellent performance, yielding an MAE of 4.29 and an RMSE of 5.49 with an R^2 of 0.38. This model captured the overall underlying smoothness and seasonality exhibited in the sales data. It was great at capturing those, as it gave higher weights to recent observations; hence, it adapted quickly to changes happening in the time series. Focusing on the most recent data points helped the model to outperform the underlying patterns, though it struggled somewhat with the variances, as testified to by its RMSE.

On the contrary, the ARIMA model performed very poorly. It had a much higher MAE, amounting to 22.20, with an RMSE of 23.35 and a negative R^2 of -10.31. These results show that ARIMA struggled to capture the dynamics of the dataset. The model's performance may be due to an assumption of stationarity and linearity in the model that did not support such a level of complexity and seasonality of the data. The high RMSE, coupled with a negative R^2, indicates that ARIMA's predictions not only went off the chart concerning magnitude but also failed to capture the direction of actuals.

The best performance out of these, but still far from the performances of the Smoothing Model and Prophet-based models, was obtained with the SARIMA model that extends ARIMA to incorporate seasonality. The MAE was 6.66, the RMSE was 8.37, and the R^2 was -0.45. This model fared better but was still far from the best performance. Though the seasonal pattern is captured somehow, it is less capable of dealing with variability and complexity than other models, reflected in the negative R^2. The model training results suggest that the model has been overfitted or the best parameters were not selected during training.

Prophet was the best model, returning an MAE of 4.90 and an RMSE of 6.03 with an R^2 of 0.25. Even though its performance lagged a little behind the Smoothing Model regarding the RMSE, Prophet still produced good forecasts. Success factors included its ability to handle seasonal solid effects and robustness against missing data and outliers. However, the relatively low R^2 indicates that while Prophet effectively captured the seasonal components, it may have struggled with irregularities or noise in the data.

NeuralProphet, which is a model that incorporates elements of neural networks into the Prophet model, yielded an MAE of 4.89, an RMSE of 6.03, and a worst R^2 of 0.25, competing with the simple Prophet model. NeuralProphet has the power to handle challenging nonlinear relationships and flexibility in modeling lagged regressors. The fact that Prophet performed similarly suggests that the neural network elements provided little gain on this dataset, possibly because of the data's relatively tiny size or nature.

The conclusion derived from the above analysis is that the Smoothing Model was the best. Thus, more straightforward methods, which place a lot of emphasis on recent data points and account for seasonality concisely, work well on this particular dataset. On the other hand, more complex models such as SARIMA and NeuralProphet did not give very superior results, likely due to this dataset's nature or its possible overfitting problems. The R^2 values reveal severe model problems fit for ARIMA, and the negative values are poor fits-again, pointing, in this case, to the possibility that a parsimonious approach yields better results when either the seasonal or trend component is not well-represented in the data.

As mentioned above, the situation foresees improvements that may come from advanced models like NeuralProphet and SARIMA. Consequently, their advantages depend highly on dataset characteristics. Future studies can continue with more extensive and complex datasets, where advantages for these models would emerge. Further experiments with sophisticated tuning methods and hybrid models may further enhance predictive accuracy, at least for situations when traditional models perform inadequately.

Certain limitations establish uncertainties in the outcomes, such as the dataset scope, model selection and tuning, computation resources, and metrics adopted to assess performance. These limitations should be improved upon in a future study by utilizing more extensive and diverse datasets, improving model optimization, and utilizing more advanced computational resources. In this way, a deeper understanding of various model enabilities and shortcomings concerning the time series forecast could be acquired.

7. Implications

Models like Prophet and NeuralProphet are doing a pretty good job, considering how easy it is to implement and tune. Typically, they work well on data with explicitly stated seasonality and do not require heavy manual intervention. Thus, they will be convenient for an organization that does not possess the scarce resources of the data sciences.

On the other hand, when data shows strong autocorrelation and stationarity, the application's merit belongs to classical models like ARIMA and SARIMA. These models sometimes require more sensitive tuning and may be less computationally efficient from time to time, limiting their utilization for perhaps real-time applications or resource-constrained environments. Some good results were obtained using exponential smoothing methods for simple datasets with subtler trends and seasonality. In turn, their low computational overhead and robustness render them a workable choice for more straightforward forecasting tasks.

Another significant development of this research is the trade-off between model complexity and computational efficiency. This trade-off would, therefore, entail that in choosing the models, the practitioner has to consider the available computational resources. Yes, complex models, like the SARIMA and NeuralProphet, will generally require much computational processing power and time, mainly when applied to comprehensive datasets or when conditions include extensive hyperparameter tuning. Both Prophet and NeuralProphet can be scaled for organizations seeking to scale up their forecasting across a portfolio of products or regions in addition to seeking a good balance between performance and computational feasibility.

Results tell that model performance must be taken from a holistic perspective. The practitioner cannot base his decision on just one metric, such as MAE or RMSE, but on several other metrics like R^2 and computational efficiency. In this regard, it will ensure multi-dimensional model performance evaluations that enable business decisions relevant to the particular objective and constraints of the operation in question.

In particular, while the contributions mentioned above are essential, one crucial area of consideration for further research is the more advanced models such as LSTM and XGBoost and their hybrid models. Such models may perform generally better, being different from traditional ones, especially in some more dynamic or complicated situations. More importantly, investigating this deep learning model may shed more light on when models work, massive data models, and under what conditions these models fail.

Another promising line of research is improving techniques for hyperparameter tuning. Limitations identified here point toward future investigations exploring more advanced methods, such as automated machine learning, or more sophisticated wider search strategies like Bayesian optimization to find more promising model configurations.

Further confirmation may come through applying such models in a broader class of datasets: complex seasonality, nonlinear trends, or determined by exogenous variables themselves. In addition, such information would also give more details on where the specific models perform better precisely. Other than this, more sophisticated metrics in terms of MAPE or the development of new metrics will lend greater strength to performance assessments across various contexts.

That opens up another valuable area of exploration for scalability and real-time application of those models. Some of the models in this work are computationally expensive; hence, their optimization or approximation for real-time forecasting may be highly relevant in domains such as finance, supply chain management, and energy, where decisions must be made speedily with data-driven insight.

Therefore, this will provide practical guidance both to practitioners and researchers. While such insights will prompt the practitioners to choose models that best suit their needs and constraints, the researchers can build on those insights to address gaps and further push the pace of time series forecasting. Future research, if addressed, may help develop more robust, efficient, and accurate models that can enhance the decision-making processes across many industries.

8. Conclusion

The study reviewed numerous time series forecasting models, including traditional ones such as ARIMA, SARIMA, and Exponential Smoothing, with more modern approaches such as Prophet and NeuralProphet. Key takeaways of the research are therefore as follows. The model Prophet and its neural extension, NeuralProphet, performed well, notably when the data showed strong seasonal patterns. Thus, Hybrid SARIMA-Prophet performed better than the other models on all three metrics and should be a strong competitor for any demand forecasting.

Traditional models, like ARIMA and SARIMA, remain relevant when data shows strong autocorrelation with stationarity. However, such models required more meticulous tuning and were not computationally quite efficient compared to the Prophet models. While more straightforward, the Exponential Smoothing models yielded strong performance with less computational overhead.

Several evaluation metrics were applied in the study, such as MAE, RMSE, and R^2, to assess the models' performances. These highlighted the trade-off between accuracy and computational efficiency, entailing a nuanced understanding of the strengths and limitations of each model.

The more complex a model, the more computational power its operation requires. The computational power is a substantial requirement for practitioners, especially in real-time or large-scale forecasting. Future work Although the present study yields a lot of insight, there are several areas on which future work may focus to enhance the understanding and application of time series forecasting models. For instance, the exploration of machine learning and deep learning models in this current work could be extended into advanced machine learning models such as LSTM, XGBoost, and hybrid models, which could be considered alongside the models considered in the study. These models have given great promise for handling more prominent and complex datasets, and the inclusions of these may have provided a broader grasp of forecasting capabilities.

The study should be extended to a wide range of data, especially those exhibiting complex seasonality, nonlinear trends, or external influencing factors that could help establish the generalizability of the findings. Real-time Forecasting: Other research could be done to develop further models specifically for real-time forecasting, such that, in applications where decisions have to be made in the shortest time possible, the models would also do the same. Again, real-time forecasting has to do with enhancing computational efficiency by testing how the models refresh with new incoming data.

In the study, some of the provided metrics were problematic; examples include MAPE and WAE. The natural way to continue this work would be by refining these metrics or developing new ones that are better tailored to the various conditions arising in time series forecasting.

Based on this study, several practical recommendations may be derived and addressed to practitioners involved in demand forecasting and similar tasks.

A practitioner might use Prophet or NeuralProphet on a demand prediction task, especially when their datasets contain strong seasonal patterns. These models balance ease of use with solid performance, requiring less manual tuning than traditional models.

In cases where computations are an issue, a simple model, such as Exponential Smoothing, should be considered, especially with less complex datasets. On the other hand, when high accuracy is needed, and resources permit, then a hybrid model like SARIMA-Prophet has proven very effective.

In this respect, the practitioner should consider multiple metrics to evaluate model performance. Applying different metrics like MAE, RMSE, and R^2 provides a better understanding of how a model will perform in practice.

In real-world applications involving real-time data forecasting, the model selected should update efficiently with incoming data. Prophet and NeuralProphet are remarkably apt in such situations because of relatively faster training and lesser computational requirements when compared to complex models such as SARIMA.

Model monitoring is something that practitioners are giving serious attention to as they seek to project their models into dynamic environments continuously. Monitoring enhances refreshing or updating the models in time, with incoming data or trends that may emerge, to keep the forecasts accurate and reliable. By following such recommendations, the practitioner could make a more informed decision about deploying forecasting models, thus getting better results in accuracy and efficiency.

Acknowledgements

Mikhail Kolev thanks to the financial support of European Union-NextGenerationEU, through the National Recovery and Resilience Plan of the Republic of Bulgaria, project № BG-RRP-2.004-0002-C01, "BiOrgaMCT".

CRediT roles: Conceptualization, M.L.; methodology, I.N. and M.L.; software, I.N.; validation, M.K. and I.N.; formal analysis, M.L.; investigation, I.N. and M.K.; resources, M.L.; data curation, I.N. and M.K.; writing – original draft preparation, I.N. and M.K.; writing – review and editing, M.L. and I.N.; visualization, I.N. and M.K.; supervision, M.L.; project administration, M.K.; funding acquisition, M.K. and I.N.

References

[1] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time series analysis: forecasting and control*, 3. ed. Englewood Cliffs, NJ: Prentice-Hall, 1994.
[2] P. J. Brockwell and R. A. Davis, *Time series: theory and methods*, 2nd ed. in Springer series in statistics. New York: Springer, 1996.
[3] G. Box, “Box and Jenkins: Time Series Analysis, Forecasting and Control,” in *A Very British Affair*, London: Palgrave Macmillan UK, 2013, pp. 161–215. doi: 10.1057/9781137291264_6.
[4] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*. in Springer Texts in Statistics. Cham: Springer International Publishing, 2016. doi: 10.1007/978-3-319-29854-2.

[5] J. Contreras, R. Espinola, F. J. Nogales, and A. J. Conejo, "ARIMA models to predict next-day electricity prices," *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1014–1020, Aug. 2003, doi: 10.1109/TPWRS.2002.804943.
[6] L. Rubio, A. J. Gutiérrez-Rodríguez, and M. G. Forero, "EBITDA Index Prediction Using Exponential Smoothing and ARIMA Model," *Mathematics*, vol. 9, no. 20, p. 2538, Oct. 2021, doi: 10.3390/math9202538.
[7] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *Int. J. Forecast.*, vol. 20, no. 1, pp. 5–10, Jan. 2004, doi: 10.1016/j.ijforecast.2003.09.015.
[8] R. J. Hyndman and Y. Khandakar, "Automatic Time Series Forecasting: The **forecast** Package for *R*," *J. Stat. Softw.*, vol. 27, no. 3, 2008, doi: 10.18637/jss.v027.i03.
[9] J. D. Hamilton, *Time Series Analysis*. Princeton University Press, 1994. doi: 10.1515/9780691218632.
[10] A. C. Harvey, *Forecasting, Structural Time Series Models and the Kalman Filter*, 1st ed. Cambridge University Press, 1990. doi: 10.1017/CBO9781107049994.
[11] M. Caivano and A. Harvey, "Time-series models with an EGB2 conditional distribution," *J. Time Ser. Anal.*, vol. 35, no. 6, pp. 558–571, Nov. 2014, doi: 10.1111/jtsa.12081.
[12] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, Feb. 1970, doi: 10.1080/00401706.1970.10488634.
[13] S. J. Taylor and B. Letham, "Forecasting at Scale," *Am. Stat.*, vol. 72, no. 1, pp. 37–45, Jan. 2018, doi: 10.1080/00031305.2017.1380080.
[14] O. Triebe, H. Hewamalage, P. Pilyugina, N. Laptev, C. Bergmeir, and R. Rajagopal, "NeuralProphet: Explainable Forecasting at Scale," Nov. 29, 2021, *arXiv*: arXiv:2111.15397. Accessed: Sep. 01, 2024. [Online]. Available: http://arxiv.org/abs/2111.15397
[15] M. G. S. Kenyi and K. Yamamoto, "A hybrid SARIMA-Prophet model for predicting historical streamflow time-series of the Sobat River in South Sudan," *Discov. Appl. Sci.*, vol. 6, no. 9, p. 457, Aug. 2024, doi: 10.1007/s42452-024-06083-x.