Product demand forecasting with time series models

Anastasiia Bakhmutova
Computer Science and Engineering
Digital Transformation Management
Bologna University
Email: bakhmutovaaa@gmail.com

***Abstract*—This study proposes a machine learning approach to forecast product demand in the food retail sector using time series data. Focusing on a highly active distribution center, the research investigates demand patterns for a specific high-volume product—Thai beverage—from historical order data. The methodology encompasses data cleaning, exploratory data analysis, seasonal-trend decomposition, and feature engineering. Both statistical models (ARIMA, SARIMA) and machine learning algorithms (Random Forest, XGBoost) are applied under two seasonal assumptions: zero seasonality and a 72-week seasonal cycle. Model performance is evaluated using MSE, RMSE, and MAPE metrics. The non-seasonal XGBoost model demonstrates the highest predictive accuracy and is used to generate an 8-week demand forecast. Despite its performance, certain limitations are noted, such as reliance on features computed from unavailable future values. The study concludes with recommendations for enhancing model performance using alternative hyperparameter tuning strategies and advanced models like LSTM, TCN, and Facebook Prophet.**

1. Introduction

The *identification of consumer demand* is a crucial aspect for retail businesses, as it enables them to align their product assortment with the actual needs of customers. This helps identify the most sought-after items, make efficient use of selling space, and adjust purchasing strategies in line with current customer preferences. A detailed analysis of demand also supports decision-making in pricing, the development of personalized promotions, and negotiations with suppliers.

Demand forecasting, in turn, plays a key role in effective inventory management, cost reduction, and improved customer service. Anticipating customer needs in advance allows retailers to avoid both overstocking and stockouts, optimize logistics, and ensure the availability of desired products during peak periods. Reliable forecasts also support long-term planning for procurement, staffing, and marketing campaigns.

This *Machine Learning project* is dedicated to building a robust time-series forecasting model capable of distinguishing between time-dependent demand peaks. The project focuses on analyzing the dependencies of several data points between themselves, as well as studying and applying various machine learning algorithms. The goal is to develop a deep model that can take into account trend and seasonal changes in demand for products, and make forecast for the next 8 weeks.

The implementation is organized into four main parts, with each addressing a different aspect of the functionality of the project:

1) **Data Import**. The initial stage involves importing an archive called Food Demand Dataset from *Kaggle*. The archive includes five files, three of which will be used: the first file contains data on the orders requested earlier, the second file provides information on the locations of the various retail centers, and the third file details information on the products sold by these centers.
   Since our goal is to predict the order level, the object of the study was chosen to be the store with the largest number of orders and also with the highest costs for them, which gives us more historical data for training models.
2) **Data Exploration**. This section delves into exploring and analyzing the dataset. This section presents an initial exploratory analysis of the study target - Distribution Center №13 - selected due to its high transaction frequency and volume. The product ordered most frequently, the Thai beverage, is identified as the primary target variable for forecasting. Moreover, as part of the exploratory phase, we examine the structure of imported tables with a focus on the data types present in each column.
3) **Data Preprocessing**. This stage includes a standard data cleaning step (removing unnecessary for forecasting columns and rows of the table, detecting and replacing outliers), conducting a correlation analysis to identify the level of dependence between various characteristics, as well as decomposing a series of demand data with the determination of trend and seasonality.
4) **Machine Learning Modeling**. In this stage of the project, various machine learning models are trained and tested to predict product demand. This section provides a detailed description of the imple-

mentation of the model evaluation function and the construction of forecasting algorithms, including *ARIMA*, *SARIMA*, *Random Forest*, and *XGBoost*. Based on the previously conducted seasonal analysis, the models are developed under two different assumptions regarding seasonality: the absence of a seasonal component (zero seasonality) and the presence of a 72-week seasonal pattern. This setup enables a comparative assessment of model effectiveness under different assumptions about the seasonal structure of the data.

5) **Evaluation of Predictions**. This section evaluates the predictive performance of the developed models on the test set using standard error metrics: Mean Squared Error (*MSE*), Root Mean Squared Error (*RMSE*), and Mean Absolute Percentage Error (*MAPE*). These metrics provide a quantitative basis for comparing model accuracy. The model yielding the lowest error values across these indicators is considered the best-performing approach.
6) **Forecast**. In the final stage of the project, demand is forecasted for the upcoming *8 weeks*. A feature set is constructed to match the input requirements of the trained model, serving as the basis for generating predictions. The results are visualized through a forecast plot, providing insights into the model's projected behavior in the absence of ground-truth data.

The significance of this project lies in the practical application of time series analysis and machine learning techniques to address one of the key challenges in retail—demand forecasting. Accurate forecasts support more informed decision-making in inventory management by enabling timely product replenishment and reducing the risk of operational disruptions. Moreover, the project presents an approach for building adaptable models that can be easily transferred to other products, store locations, or forecasting horizons.

The subsequent sections of this paper will delve into the specific details of each project phase.

2. Related Work

Forecasting consumer demand for food products in retail settings has received increasing attention due to its direct impact on inventory management and waste reduction. Traditional time series models, such as ARIMA and Holt-Winters, remain popular choices for short-term demand prediction owing to their interpretability and computational efficiency.

Veiga, Catapan, and others [1] conducted a comparative analysis of Holt-Winters and ARIMA models in the context of food retail demand forecasting. Their results suggest that ARIMA performs particularly well when the data exhibits strong autocorrelation, offering stable and accurate forecasts for perishable products. Similarly, recent studies have highlighted the value of SARIMA in capturing seasonality in food sales data, making it a suitable choice for long-term forecasting in structured retail environments [2].

Nevertheless, classical models often struggle with complex, nonlinear trends that are common in real-world retail datasets. To overcome these limitations, researchers have increasingly adopted machine learning approaches such as Random Forest and XGBoost [3]. These ensemble models are capable of capturing nonlinear relationships and utilizing a wide range of engineered features. While their performance can surpass that of traditional models, recent evidence shows that machine learning techniques may not always yield significantly better results, especially when seasonal patterns are dominant and well-modeled by statistical methods [2].

In this context, hybrid approaches or careful model selection based on data characteristics can provide the best forecasting performance for food product demand in retail.

3. Proposed Methods

3.1. First acquaintance with data

The initial stage included importing and analyzing csv files, three of them were used for the project:

- The first file provided information about orders (order number, number of the week, center that made the order, products in orders, number of units ordered checkout price, base price, emailer for promotion, homepage featured);
- The second file provided information about the location of the retail center (region, city, type)
- The third file provided information about the purchased meals (name, category, cuisine)

As a first step, the orders table was filtered to include only records associated with *Distribution Center №13*. This center was selected as the focus of the analysis due to its consistently high ordering frequency and volume, making it the most active node in the dataset. The abundance and stability of its historical data provided a solid foundation for training and validating forecasting models. We got 7.046 rows.

Studied the file with the description of the centers, we found that center №13 is located in city 590 and has the type B. The total number of ordered products is 4.296.545 units, and the cost of expenses for all orders is 1.127.045.001 monetary units.

To perform demand forecasting, it was first necessary to select a specific product as the subject of analysis. Using the orders description file, we calculated the number of orders for each food item. The results revealed that all products were ordered exactly 145 times, indicating a consistent weekly ordering pattern throughout the study period. Therefore, the selection was based on the total number of units ordered, leading us to choose product with ID 1885, a Thai beverage, which had the highest overall volume. After filtering the rows containing data only for center 13 and for meal 1885 we have 145 rows left.

3.2. Data Preprocessing

3.2.1. Data Cleaning. Upon initial inspection of the dataset, we observed that the main table of orders contained no missing values, duplicate rows or columns, and that the order ID field consisted of unique entries. However, further data cleaning was required to prepare the dataset for modeling. This included removing irrelevant rows and columns, renaming selected features to ensure semantic clarity, and performing an outlier analysis to detect anomalous values.

Following the initial cleaning, **outlier detection** was performed using the *Interquartile Range* (IQR) method, with $Q_1 = 0.25$ and $Q_3 = 0.75$. This analysis revealed seven outliers: one corresponding to a significant decrease in the number of units ordered, and the remaining six indicating unexpected spikes in demand. These anomalies were distributed across the dataset with one occurring in the first year, one in the second, and five in the third year. Notably, the outliers did not align with the same calendar months, suggesting a lack of seasonal synchrony. Furthermore, the anomalies did not coincide with holiday periods, ruling out the hypothesis of holiday-driven demand surges. This motivated a subsequent correlation analysis to examine the potential relationship between demand fluctuations and product pricing.

A **correlation analysis** was conducted to assess the relationship between two variables: meal price and the number of beverages ordered. The objective was to determine whether there exists any dependency, such as lower prices being associated with larger order volumes. The first step involved a *visual examination* via a scatter plot (figure 1), which revealed no clear relationship between two variables (prices mostly stay within a narrow range (140–150) regardless of the volume ordered).

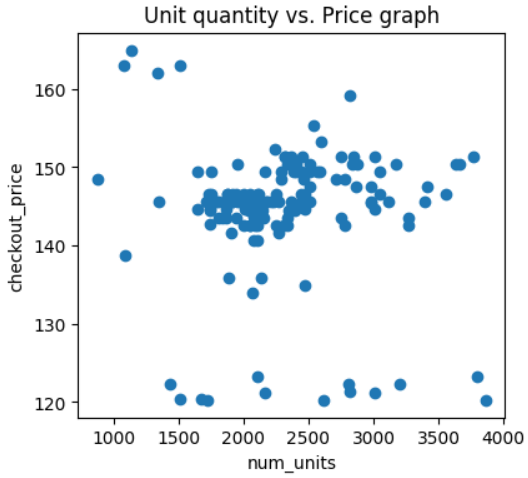

Figure 1: Relationship between Order Quantity and Checkout Price

Subsequently, the *Pearson correlation coefficient* was calculated as $r = -0.08$, indicating a very weak linear relationship. The associated $p - value = 0.335$ suggested that this correlation is not statistically significant.

To further investigate potential causal influence, the *Granger Causality Test* was applied to determine whether past values of one variable help predict the other. The SSR-based F-test, along with other test statistics, yielded $p > 0.05$, confirming the absence of any statistically significant predictive relationship.

In summary, the variables, product price and order volume, appear to evolve independently. As a result, price was excluded from the feature set used in subsequent demand forecasting models.

Given these findings, we returned to the **handling outliers** without keeping in their original form. It is important to note that no external information was available regarding potential business events such as sales or promotional campaigns that could justify the observed demand anomalies. For the two consecutive outliers, the values were replaced with the average of their immediate neighboring points. The remaining five anomalies were adjusted using a three-point median filter, computed over the previous, current, and subsequent observations. After all modifications we had 145 rows and 2 columns (number of week and number of ordered units).

3.2.2. Seasonal–Trend Decomposition. The first step in time series analysis is to identify the underlying **trend**, as it plays a critical role in model selection and interpretation. To extract the global trend component, we applied the *Hodrick–Prescott (HP) filter*. Given that our data is weekly, the smoothing parameter λ was set to 129.600, following standard practice in time series research. The resulting trend captures the overall direction of demand change and appears relatively smooth.

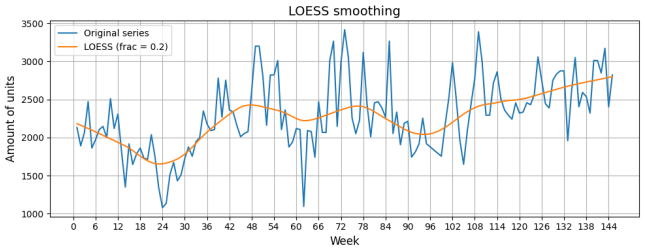

Figure 2: LOESS smoothing trend

However, since time-series data may also exhibit local fluctuations, we further employed *LOESS smoothing* to gain insights into more granular, short-term movements. The LOESS plot (figure 2) reveals that the trend is non-linear: the rate of growth and decline varies over time. Specifically:

- The trend slopes downward until approximately the 24th week, followed by an increase, a slight decline, and a final upward movement toward the end of the series;
- The amplitude of short-term fluctuations (LOESS residuals) remains approximately constant throughout, indicating that volatility does not scale with trend magnitude.

To assess potential **seasonality**, we examined the *autocorrelation* (ACF) and *partial autocorrelation* (PACF) plots, shown in Figures 3 and 4.

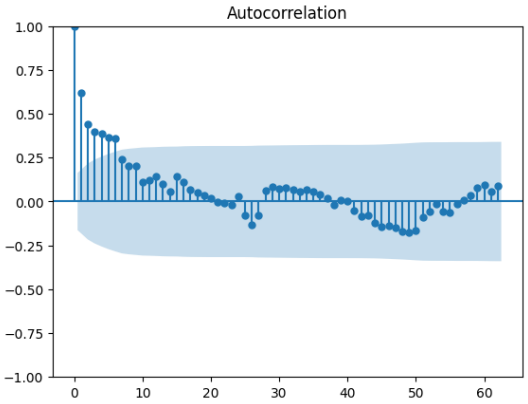

Figure 3: Autocorrelation plot

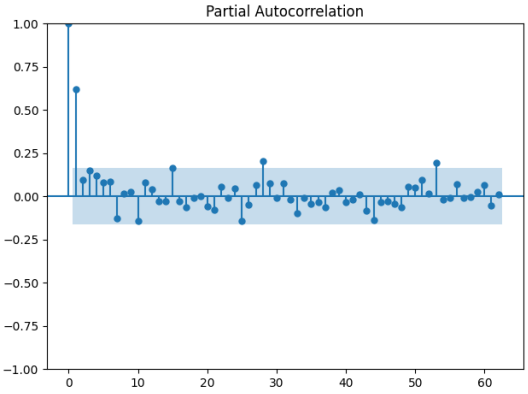

Figure 4: Partial autocorrelation plot

The analysis revealed the following:

- A rapid decay of autocorrelation after lag 6 suggests strong short-term dependencies (within 6 weeks), followed by a near-zero correlation at longer lags;
- No prominent spikes were observed at typical seasonal lags (e.g., 12 or 52 weeks), and all minor deviations remained within the confidence bounds, indicating no statistically significant seasonal effects.

To further validate the absence of standard seasonality, we performed spectral analysis using the *Fast Fourier Transform* (FFT). The analysis revealed a dominant spectral peak corresponding to a 72-week period, representing the strongest repeating cycle in the series.

3.3. Machine Learning modeling

Once the dataset was prepared and time series components analyzed, it was split into training and test sets. Machine learning models were trained on the training portion, and subsequently used to generate forecasts on the test set. This approach enabled the evaluation of predictive accuracy and the identification of the best-performing model.

We experimented with the following machine learning algorithms, each offering distinct advantages for time-series forecasting:

- *ARIMA (AutoRegressive Integrated Moving Average)* - a classical statistical model that captures both autoregressive and moving average components while accounting for non-stationarity through differencing;
- *SARIMA (Seasonal ARIMA)* - it extends arima by incorporating seasonal autoregressive and moving average components, making it well-suited for datasets with regular, repeating seasonal patterns;
- *XGBoost (Extreme Gradient Boosting)* - a powerful ensemble machine learning model that excels at capturing complex nonlinear relationships and interactions between features;
- *Random Forest* - a robust, nonparametric ensemble method that leverages multiple decision trees to capture nonlinearities and interactions within the data.

Based on the earlier seasonality analysis, models were developed under two hypotheses: one assuming no seasonality and the other incorporating a 72-week seasonal pattern. The ARIMA model was constructed exclusively under the no-seasonality assumption, while the SARIMA model was designed to account for a 72-week seasonal structure. For XGBoost and Random Forest, both seasonal and non-seasonal versions were implemented to enable a comparative evaluation of their forecasting performance.

For both ARIMA and SARIMA models, an evaluation function (`evaluate_model`) was implemented using the *AIC* and *BIC* criteria to guide model selection. These information-theoretic measures assess the trade-off between model fit and complexity, enabling the identification of the most suitable parameter configuration for final forecasting.

3.3.1. ARIMA. To apply the non-seasonal ARIMA model, we first assessed the stationarity of the time series. The *Augmented Dickey-Fuller* (ADF) test indicated stationarity, whereas the *KPSS test* suggested the opposite. Due to this discrepancy, the series was considered conditionally stationary. Moreover, ARIMA models are capable of handling trend components via the integrated term I, particularly when drift is included. The key model parameters:

- p - the number of autoregressive terms;
- d - the number of nonseasonal differences needed for stationarity;
- q - the number of lagg for moving average order.

Parameter selection was performed both manually, using ACF and PACF plots, and automatically through the `AutoARIMA` function from the `StatsForecast library`.

Model evaluation was carried out using AIC and BIC criteria, and the optimal configuration was found to be ARIMA(1,1,1). As shown in Figure 5, the model produced a flat, "blank" prediction, which is a well-known characteristic of a pure h-step ARIMA(1,1,1) model without drift - essentially a random walk lacking directional movement.

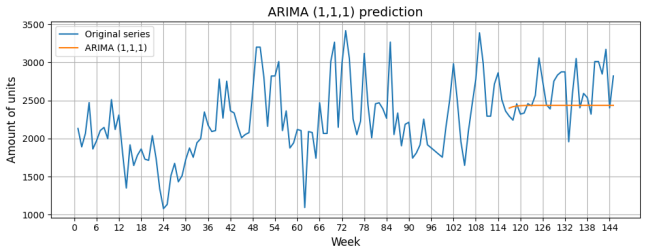

Figure 5: Prediction based on ARIMA (1,1,1)

3.3.2. SARIMA. An automatic parameter selection procedure was also applied to the SARIMA model. Unlike ARIMA, SARIMA incorporates a seasonal component, which in this case was set to 72 weeks based on prior spectral analysis. The automated optimization identified SARIMA(1,1,1) as the best-fitting configuration. Interestingly, the resulting AIC and BIC values matched those obtained for the ARIMA(1,1,1) model, indicating comparable predictive performance.

The forecast on the test set displayed behavior nearly identical to that of ARIMA. As shown in the corresponding figure 6, the predicted values remained flat over time - a typical outcome for models that do not include a trend (drift) component.

3.3.3. XGBoost. It is a gradient boosting algorithm based on decision trees, well suited for regression tasks such as time series forecasting - provided the features are engineered appropriately.

For the **non-seasonal model**, the seasonal length was set to 1, meaning that no seasonal lags were introduced. Feature engineering was a crucial step and included: a trend variable,

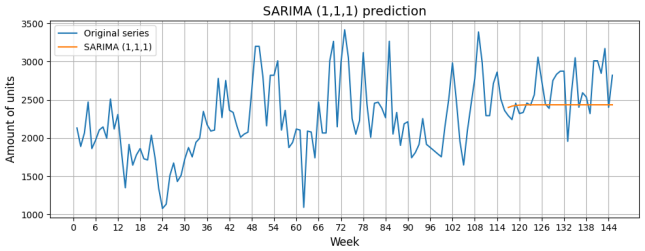

Figure 6: Prediction based on SARIMA (1,1,1)

lags from 1 to 10 weeks, exponential smoothing, and rolling means with 4, 8, and 12-week windows.

Hyperparameter tuning was performed using randomized search (`RandomizedSearchCV`) with `XGBRegressor`. The optimal configuration was determined as:

- `n_estimators` - number of boosting trees. Set 500;
- `learning_rate` - step size in gradient descent. Set 0.1;
- `max_depth` - maximum depth of individual trees. Set 3;
- `subsample` - fraction of samples used for each tree. Set 0.6;
- `colsample_bytree` - fraction of features used when constructing each tree. Set 1.0

Using these settings, the model was trained and used for predictopn. As shown in the plot 7, the model successfully captured the peaks and troughs of demand, indicating its ability to adapt to the temporal structure of the series.

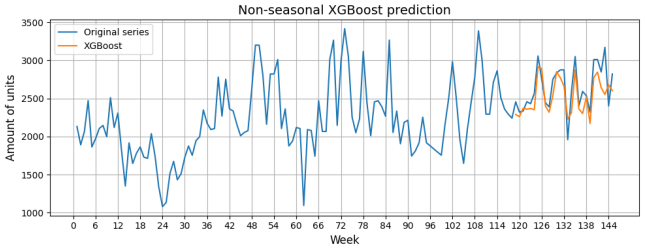

Figure 7: Prediction based on non-seasonal XGBoost

For the **seasonal XGBoost model**, several engineered features were introduced during dataset preparation. These included: a trend variable, standard lags from 1 to 6 weeks, and extended seasonal lags at 54 and 72 weeks to reflect potential annual and 72-week periodicities. Additionally, a 72-week rolling mean was computed, and sinusoidal and cosinusoidal terms were added to explicitly model cyclical patterns with a 72-week period.

Hyperparameter tuning was again performed using randomized search. The optimal configuration was:

- `n_estimators` - set 300;
- `learning_rate` - set 0.01;
- `subsample` - set 0.8;

- `colsample_bytree` - set 1.0;
- `reg_alpha` - L_1 regularization - adds to the loss function the sum of the absolute values of the weights of the tree leaves, multiplied by the coefficient reg_alpha. Set 0.5;
- `reg_lambda` - L_2 regularization - adds to the loss function the sum of the squares of the leaf weights, multiplied by the reg_lambda coefficient. Set 0.5

Despite incorporating seasonal features, the trained model did not demonstrate a notable improvement in forecast accuracy over the non-seasonal version. This suggests that neither the 72-week nor the 52-week seasonal structure contributed meaningfully to predictive performance (figure 8.

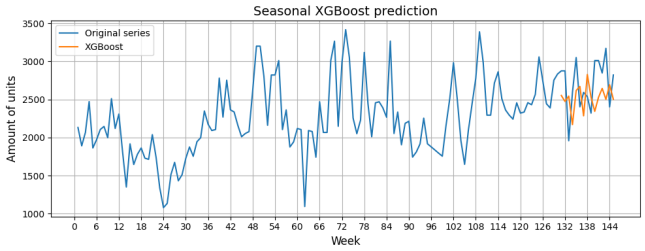

Figure 8: Prediction based on seasonal XGBoost

3.3.4. Random Forest. The model was employed to test both the no-seasonality hypothesis and the alternative assumption of 72-week seasonality. For the **non-seasonal configuration**, the dataset was enhanced with lag features from weeks 1 to 6, as well as additional lags at weeks 10 and 24. Rolling mean (window = 4) and rolling standard deviation (window = 8) were also included.

Hyperparameter tuning was conducted using randomized search with the `RandomForestRegressor`. Once the optimal parameters were selected, the model was trained and used to generate predictions on the test set:

- `n_estimators` - the number of decision trees to be included. Set 200;
- `max_features` - the maximum number of features considered for splitting at each node. Set '$log2$';
- `min_samples_split` - the minimum number of samples required to split an internal node. Set 4;
- `random_state` - sets the seed for the random number generator to ensure reproducibility of results across different runs. Set 42.

As illustrated in Figure 9, the forecast failed to accurately replicate the series' behavior. The predicted values lack the necessary variability and do not track the underlying trend, indicating that the model struggles to generalize well under the non-seasonal hypothesis.

For the **seasonal Random Forest model**, an extended feature set was engineered. This included lags from weeks 1 to 15, a 3-week rolling mean, and a 7-week rolling

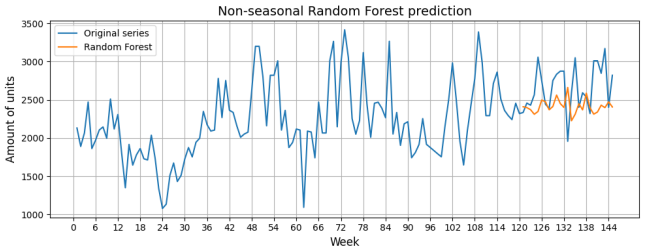

Figure 9: Prediction based on non-seasonal Random Forest

standard deviation. To model seasonality explicitly, a 72-week lag (shift) and sine/cosine transformations were added to capture periodic behavior. Hyperparameters were tuned using randomized search:

- `n_estimators` - set 200;
- `max_features` - set '$log2$';
- `min_samples_split` - set 4;
- `random_state` - set 42.

Despite incorporating seasonal features, the model failed to deliver improved prediction accuracy. As shown in Figure 10, the predictions do not adequately follow the actual demand fluctuations, and the model does not effectively capture either short-term variability or longer seasonal patterns.

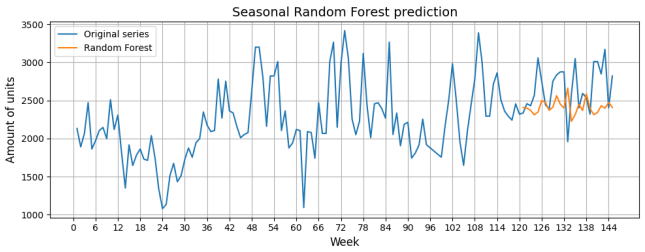

Figure 10: Prediction based on seasonal Random Forest

4. Results

After generating predictions from all Machine Learning models, we implemented an evaluation function using three metrics: *Mean Squared Error* (MSE), *Root Mean Squared Error* (RMSE), and *Mean Absolute Percentage Error* (MAPE) called `accurancy_metrics`.

Model	MSE	RMSE	MAPE
Arima (1,1,1)	118223.96	343.84	0.096
Sarima (1,1,1)	118223.96	343.84	0.096
XGBoost nonseasonal	43286.78	208.05	0.063
XGBoost seasonal	175164.23	418.53	0.146
Random Forest nonseasonal	174710.91	417.98	0.119
Random Forest seasonal	174710.91	417.98	0.119

TABLE 1: Forecast Accuracy Metrics for ML Models

Based on these metrics, the **non-seasonal XGBoost** model consistently achieved the lowest error values, indicating the highest predictive accuracy. As a result, this model was selected for the final forecast of future product demand.

To generate the forecast, a feature set was constructed to describe temporal patterns: a trend variable, lags from weeks 1 to 11, exponential smoothing, and rolling means with 4-, 8-, and 12-week windows. The previously trained model was then used to forecast demand for the next 8 weeks (two months). The results, along with a 95% confidence interval, are visualized lower, highlighting the expected value range under model uncertainty.

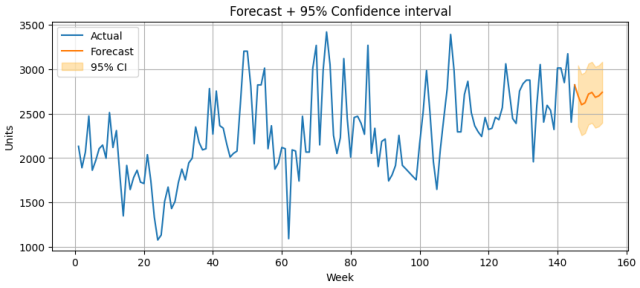

Figure 11: 8 week forecast with 95% confidence interval

5. Conclusion

In conclusion, a forecast for the next 8 weeks was generated using the **non-seasonal XGBoost** model, which previously demonstrated the highest prediction accuracy. Table 2 presents the estimated number of product units to be procured each week.

Week	Units
146	2698
147	2598
148	2621
149	2718
150	2737
151	2682
152	2698
153	2738

TABLE 2: Predicted demand volume for 8 weeks

However, the resulting forecast does not appear realistic from a business perspective. One likely explanation is that *exponential smoothing*, which emerged as a key feature during training, is based on future values that were not available at forecast time. Additionally, the underlying structure of the dataset may exhibit *random walk–like behavior*, which poses a significant challenge for model generalization.

As for the other models, **ARIMA** and **SARIMA** may have underperformed on the test set due to their inherent limitations — they are best suited for capturing linear trends, while the demand data likely contains more complex, nonlinear patterns. The **Random Forest model** also failed to deliver robust predictions, possibly due to insufficient tuning or its limited capacity to handle temporal dependencies effectively.

To improve forecasting performance in future work, several enhancements can be considered:

- applying *Grid Search* or *Bayesian Optimization* for hyperparameter tuning instead of randomized search;
- experimenting with other time-series forecasting models such as *SVR*, Facebook *Prophet*, or deep learning architectures like *LSTM* and *Temporal Convolutional Networks* (TCNs), which are designed to capture nonlinear and long-term dependencies in time series data.

References

[1] C. P. Da Veiga, C. R. Da Veiga, A. Catapan, U. Tortato, and W. V. Da Silva, "Demand forecasting in food retail: A comparison between the Holt-Winters and ARIMA models," *WSEAS Transactions on Business and Economics*, vol. 11, no. 1, pp. 608–614, 2014.

[2] O. Khalid, "Short-term and long-term product demand forecasting with time series models," *Journal of Trends in Financial and Economics*, vol. 1, no. 3, pp. 11–17, 2024.

[3] J. Korstanje, *Advanced Forecasting with Python: With State-of-the-Art Models Including LSTMs, Facebook's Prophet, and Amazon's DeepAR*. Maisons Alfort, France: Apress, 2021. ISBN: 978-1-4842-7149-0.

[4] B. V. Vishwas and A. Patel, *Hands-on Time Series Analysis with Python*. Berkeley, CA: Apress, 2020. ISBN: 978-1-4842-5991-7.

[5] A. C. Schuermann and N. P. Kannan, "A production forecasting and planning system for dairy processing," *Computers & Industrial Engineering*, vol. 2, no. 3, pp. 153–158, 1978.

[6] J. G. A. J. van der Vorst, A. J. M. Beulens, W. de Wit, and P. van Beek, "Supply chain management in food chains: improving performance by reducing uncertainty," *International Transactions in Operational Research*, vol. 5, no. 6, pp. 487–499, Nov. 1998.

[7] P. Doganis, A. Alexandridis, P. Patrinos, and H. Sarimveis, "Time series sales forecasting for short shelf-life food products based on artificial neural networks and evolutionary computing," *Journal of Food Engineering*, vol. 75, no. 2, pp. 196–204, Jul. 2006.

[8] M. Matsumoto and S. Komatsu, "Demand forecasting for production planning in remanufacturing," *Journal of Remanufacturing*, vol. 79, pp. 161–175, 2015.

Anastasiia Bakhmutova

May 30, 2025