Enhancing sell-in and sell-out forecasting using ensemble machine learning method

Vishal Das, Tianyi Mao, Zhicheng Geng, Carmen Flores, Diego Pelloso and Fang Wang

Abstract—Accurate sell-in and sell-out forecasting is a ubiquitous problem in the retail industry. It is an important element of any demand planning activity. As a global food and beverage company, Nestlé has hundreds of products in each geographical location that they operate in. Each product has its sell-in and sell-out time series data, which are forecasted on a weekly and monthly scale for demand and financial planning. To address this challenge, Nestlé Chilein collaboration with Amazon Machine Learning Solutions Labhas developed their in-house solution of using machine learning models for forecasting. Similar products are combined together, such that there is one model for each product category. In this way, the models learn from a larger set of data and there are fewer models to maintain. The solution is scalable to all product categories and is developed to be flexible enough to include any new product or eliminate any existing product in a product category based on requirements.

We show how we can use the machine learning development environment on Amazon Web Services (AWS) to explore a set of forecasting models and create business intelligence dashboards that can be used with the existing demand planning tools in Nestlé . We explored recent deep learning networks (DNN) which shows promising results for a variety of time series forecasting problem. Specifically, we used DeepAR autoregressive model that can group similar time series together and provides robust predictions. To further enhance the accuracy of the predictions and include domain specific knowledge, we designed an ensemble approach using DeepAR and XGBoost regression model. As part of the ensemble approach, we interlinked the sell-out and sell-in information to ensure that a future sell-out influences the current sell-in predictions. Our approach outperforms the benchmark statistical models by more than 50%. The machine learning (ML) pipeline implemented in the cloud is currently being extended for other product categories and is getting adopted by other geomarkets.

Keywords—Sell-in and sell-out forecasting, demand planning, DeepAR, retail, ensemble machine learning, time-series.

I. Introduction

FORECASTING is a critical element of any business. It helps companies to plan for the future and make informed decisions. In the retail industry, forecasting is particularly important, as it allows retailers to anticipate consumer demand and stock the products or increase production accordingly [1]. Retail forecasting depends on sell-in and sell-out forecasting. Sell-in forecasting refers to the process of predicting the demand for a product or service from retailers or distributors, while sell-out forecasting predicts the demand from end consumers. Both sell-in and sell-out forecasting are important for supply chain management and inventory planning. Sell-in forecasting can be more difficult and less accurate than sell-out forecasting, due to the complexity of consumer demand.

Prior to this work, the sell-in and sell-out forecasting at Nestlé Chilewere done using traditional time-series analysis - exponential smoothing and by incorporating business knowledge and expert opinions from the demand planners. However, this method is highly subjective and is very hard to scale over time. In this work, we have used ML techniques, particularly DNNs for sell-in and sell-out forecasting. We developed a ML pipeline that drives accuracy in forecasting and also helps scale the method to different products and geo-markets. We demonstrate the applicability of the method using two major product categories in one of the geo-markets of Nestlé . The paper is organized as follows: First, we will review the current state of the art in the field, and elaborate the motivation and business value behind this study. Second, we will discuss the methodology used in this work. We will state the benchmark model for this study and explain the different ML algorithms used. Third, we will describe the dataset and the experiment setup for these ML algorithms. Finally, we will discuss the results of the ML pipeline to real-world data from Nestlé . We also discuss the path to production of the methodology developed in this paper and the scalability of the ML pipeline for different product categories and the other Nestlé geo-markets.

Overall the goal of this paper is to define the state of the art in sell-in and sell-out forecasting using ML algorithms, and to demonstrate the potential of these algorithms in improving the forecasting accuracy in the retail industry.

II. Motivation and related work

Major retail and manufacturing companies, such as Nestlé have a variety of time-series data on sell-in and sell-out volumes and values for their products. These time-series data exist at various levels of granularity, for example: product level, client level and at different frequencies ranging from daily to monthly scales. The forecasting requirements from the business are diverse with forecast horizons ranging from weekly to monthly or yearly levels. These requirements primarily come from the demand planners for inventory management, and from the financial teams for fiscal management. The time-series data show strong seasonality patterns that are different for different products. Several new products are constantly launched by these companies and existing products are phased out or modified that impact the patterns in the data. This is in

Vshal Das, Tianyi Mao, Zhicheng Geng and Fang Wang are with Amazon Machine Learning Solutions Lab, Chicago, USA. (email: vishddas@amazon.com, tianyim@amazon.com, zcgeng@amazon.com, fwfang@amazon.com)

Carmen Flores is with Nestlé Chile, Santiago, Chile. (email: Carmen.Flores@cl.nestle.com)

Diego Pelloso is with Nestlé Brazil, São Paulo, Brazil. (email: Diego.Pelloso@BR.nestle.com

addition to the effect of external factors such as holidays and promotions.

An effective sell-in and sell-out forecasting method should have the following characteristics:

- combine time-series that are similar in characteristics, for example: all time-series related to the same product category
- flexible and adaptable to include/ exclude products
- scalable to expand to new product categories and different geographical markets
- incorporate external factors such as holidays and promotions
- interlink the sell-out and sell-in predictions

Our goal as part of this work is to provide an end-to-end forecasting pipeline that ingests time-series data for a product category at the desired level of granularity and forecasts the sell-in or sell-out with the desired prediction window. The output of the forecasting pipeline is a self-serve business intelligence dashboard that can be used directly by the demand planners and financial planning team. The pipeline should be easily trainable or updated with new data.

Traditionally, retail forecasting has been done using statistical methods, such as time series analysis [2]. [3] provides a detailed overview of the methods used in retail forecasting. With the advent of machine learning, it is now possible to use more advanced techniques to make accurate forecasting [4]. Statistical methods perform well under stable economic conditions, Neural networks that incorporate non-linearity in their formulation outperform the statistical methods under volatile economic conditions [5]. Machine learning algorithms can learn from historical data and make predictions based on patterns and trends. The algorithms are more flexible and adaptable than conventional statistical methods. Additionally, the algorithms can take into account a variety of factors that may affect sales, such as seasonality, promotions, and consumer trends. [6] showed in their study that a combination of forecasting models outperformed the performance of any individual forecast model. The ML algorithms in these studies were mostly focused on feed-forward networks and tree-based regression. [7] used modern DNNs - Long Short-Term Memory (LSTM) in addition to feed-forward networks and tree-based regression methods. They found that the accuracy of forecasting improved significantly using LSTM based networks. [8] used a hybrid approach using convolutional neural network (CNN) with bi-directional LSTM to further improve the accuracy. Recently, DeepAR [9] model which is an autoregressive recurrent network has been found to outperform other DNNs for time-series forecasting. DeepAR is particularly useful in cases where there are related time-series that can be modeled with a global model.

With the background of the existing data driven approaches for retail forecasting and to support Nestlé Chile's sell-in and sell-out forecasting requirements, we developed a ML pipeline using an ensemble based approach. We combined DeepAR predictions with XGBoost tree-based regression model. The sell-out and sell-in information were combined in the ensemble model to improve the overall accuracy of the forecasts. Our contributions include:

- Development of a scalable ML pipeline for sell-in and sell-out forecasting,
- Application of DNN, DeepAR autoregressive model for sell-in and sell-out forecasting,
- Ensemble strategy (two-step approach) using DeepAR and XGBoost to improve accuracy and incorporate domain knowledge,
- Application of the method to a real data and systematic evaluation of the results with existing (benchmark) model.

III. Methodology

The two ML approaches used in this paper are DeepAR and XGBoost. The results in this paper have been benchmarked against the exponential smoothing method [10]. In this section, we first give a brief introduction to DeepAR and XGBoost for time series forecasting. Then we illustrate how we implement a two-step model ensemble using DeepAR and XGBoost.

A. DeepAR

DeepAR [9] is an autoregressive recurrent neural network (RNN) based deep learning algorithm for time series forecasting. In general, DeepAR is a global algorithm, in the sense that it trains a global model over a set of time series data along with target time series at the future time steps. Based on the inputs and the targets, DeepAR learns a probability distribution so that it could generate not only a set of possible future time series but also the corresponding probabilities. The architecture of DeepAR involves multiple long short-term memory (LSTM) cells, which are designed to handle long dependencies in time series data. The output of these LSTM cells is utilized to model the probability distribution over the target time series. DeepAR algorithm is superior at processing missing values, multiple related time series, and digesting meta data information than traditional methods and other DNNs for time series forecasting. In addition, its powerful ability to scale to large datasets and multiple machines makes it suitable for production environments.

B. XGBoost

XGBoost [11] is an efficient and scalable machine learning algorithm based on gradient boosting technique that combines multiple weak models to obtain a strong model. The core idea of XGBoost is to sequentially ensemble decision trees, where each new tree is added to correct the errors of the previous trees. XGBoost starts by fitting a simple decision tree to the data, and then fits another tree to the errors of the first tree. This process is repeated by a fixed number of iterations or until meeting a certain stopping criterion. At each iteration, the algorithm tries to minimize the error by tuning the parameters of the decision tree, such as the depth and the number of splits. To apply XGBoost to time series forecasting, the data needs to be prepared in a sliding window fashion, where the window of inputs and target outputs is shifted through time to create different samples for training [12].

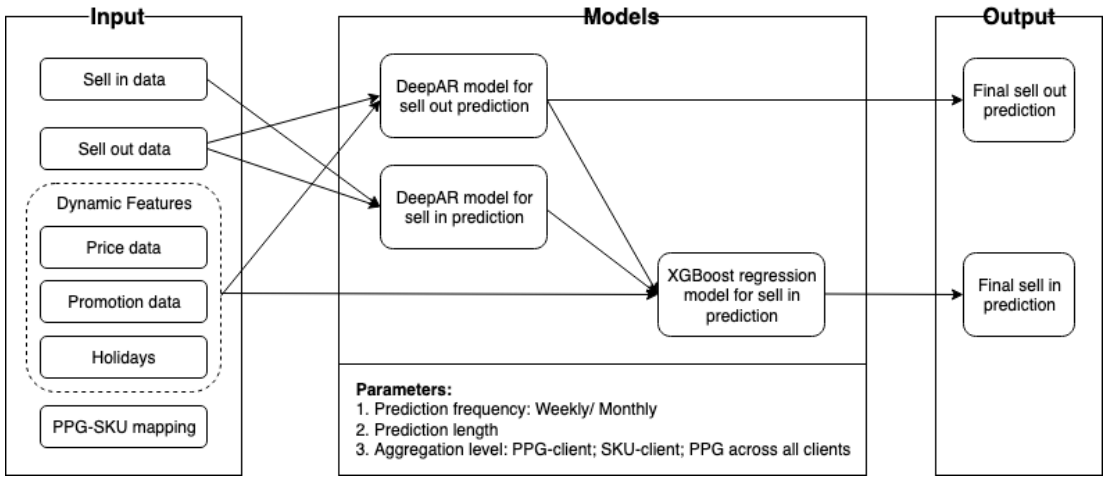

Fig. 1. Machine learning pipeline used for sell-in and sell-out forecasting. The figure illustrates the different input data used by the different models. The two-step ensemble method using DeepAR and XGBoost for final sell-in prediction is also presented in the figure.

C. Model ensemble using DeepAR and XGBoost

Ensemble learning is a powerful technique to improve the model performance and robustness by training and combining multiple models. In our method, we adopt model ensemble technique and combine the strengths of DeepAR and XGBoost. Figure 1 gives the flow diagram of our purposed ensemble method. We first train two DeepAR time-series forecasting model to jointly model all time series corresponding to the sell-in and sell-out data. The first DeepAR model is for sell-out prediction. This uses the sell-out historical data, along with dynamic features. This model also outputs the final sell-out predictions. The second DeepAR model is for sell-in prediction. This model combines the historical sell-in and sell-out data. The predicted results from the two DeepAR models are taken as input features along with the other dynamic features in the XGBoost regression model for final sell-in prediction. We performed feature engineering to introduce lag and lead to the outputs of the DeepAR models. These are then input to the XGBoost model as input features. This ensemble strategy has been formulated to incorporate the business knowledge that a future or past sell-out influences the current sell-in. The models are parameterized such that the user can define the frequency scale of prediction, the prediction length in units of the frequency scale and also the aggregation level at which the user wants the predictions. Our experiments demonstrate that the final predictions obtained from the ensemble approach are better than the predictions obtained from the model individually. A detailed discussion of the results are provided in Section 5 of the paper.

IV. Experiments

A. Dataset

In this paper, we focus on the retail forecasting dataset, particularly sell-in and sell-out dataset. To protect the privacy of the data from Nestlé while using a realistic dataset, we have removed the details of the product id and have normalized the values. The results reported in this paper remain unchanged with these data obfuscation. The dataset contains two product categories from four major retailers that comprise 60% of the market share for Nestlé Chile. The dataset contains 300 Stock Keeping Units (SKU) or unique products. Each SKU has a time-series corresponding to sell-in and sell-out volume and value recorded in a weekly frequency for sell-out and monthly frequency for sell-in, covering the time range from 2020-01-01 to 2022-09-30. On quality check of the dataset, some of these SKUs were found to be "in-and-out" or test products introduced by the company that do not have sufficient information. These products are out of commission without business value and are eliminated from further analysis. 158 SKUs are considered in the final analysis in this work. These SKUs were further aggregated into product group (PPG) level based on requirements from the business teams. Multiple SKUs form a PPG and each SKU is uniquely mapped to a PPG. There are 19 total PPGs in the dataset for both the product categories. The results presented in this paper are from the first product category. The results from the second product category are similar to the first product category and were primarily used for validation of the proposed methodology.

We also included the sell-in price, the sell-out price, promotion and holidays as additional dynamic time-series features in the models. The sell-in price information was available for each SKU. The sell-out price was unavailable for each SKU and was calculated by diving the sell-out value by the sell-out volume for each data point. The promotion calendar was available which indicated the time period during which a promotion was run. This information was converted to a binary indicator feature - 0 means no promotion and 1 means promotion. All these dynamic features were aggregated from SKU to PPG level using simple averaging. The holiday dynamic feature was calculated using squared exponential kernel based on the holiday calendar of Chile. The exponential kernel can be expressed as:

$$K = \exp^{-\alpha |D - D_H|^2}, \tag{1}$$

where D_H denotes the date of the holiday and D denotes the

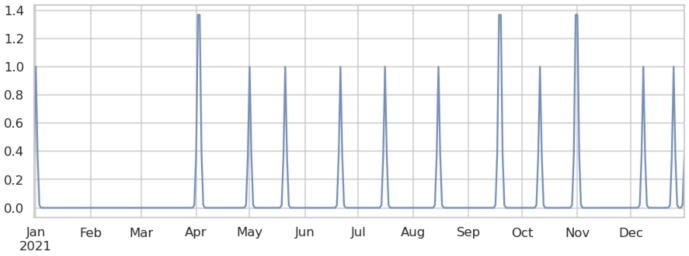

Fig. 2. Holiday feature for the year 2021 obtained using squared exponential kernel on the holiday calendar of Chile. The peaks denote the holidays.

date that the kernel value is computed on. In our experiment, α is set as 1.0. Figure 2 shows how the kernel varies over time for the year 2021. As we can see, the closer to holidays, the larger the kernel values. On dates that are not close to any holidays, the kernel values are zero. Some of the peaks are higher than others since there are two or more coinciding holidays during the period, for example the month of April in Figure 2.

B. Model setup and training

The data before April 2022 are used for training and hyperparameter optimization (HPO), and the rest of the months are used for blind test. As described in Section 3.1, DeepAR is a probabilistic method that requires a choice of the data distribution while modeling. We used the standard Zero-Inflated Negative Binomial (ZINB) distribution to model the sell-in data, and the standard gamma distribution to model the the sell-out data. These distributions are used to evaluate observations and sample predictions once the models are trained. The distributions are selected based on the nature of the data. It should be noted that the selection of the correct data distribution is a crucial step in using DeepAR model for accurate predictions. ZINB distribution extends the negative binomial distribution to account for excess zeros in count data. It assumes that the count variable has two components: (1) a zero-inflated component representing the likelihood of observing zero count, and (2) a negative binomial component representing the count distribution for positive values. The observed sell-in data matched with this distribution. Gamma distribution is the distribution of a sum of independent exponential random variables. It is used in cases where the variables are always positive and the results are skewed (unbalanced). This was the case with the observed sell-out data. We scale the target time series with their historical mean and rescale them back, with the help of MeanScaler implemented in GluonTS [13]. Note that in this business case, the sell-in prediction from the DeepAR model is not used as a direct output but as intermediate output that will serve as a feature of the ensemble model.

For training the DeepAR models, we used negative log likelihood loss function and ADAM optimizer. The metadata about each time series, such as the retail client and SKU/PPG group information, are fed into the model in the form of categorical variables. The companion time series, such as the price, discount and holiday information, are fed into the model in the form of dynamic real features. We used mean squared loss function for training the XGBoost regression models. The loss function is optimized with the approximate gradient boosting algorithm [11], which comes as default in the XGBoost package. We perform feature engineering by introducing leads and lags to the input features of the XGBoost model. Note that a lead feature can only be used for something that is stamped with a future time but is obtainable at the time point when the predictions are made. Examples of such features are holidays, discounts (because the business has a plan ahead), price (that is predefined by the business) and predictions made by auxiliary DeepAR models. In contrast, actual sell-in and sell-out volumes can only be engineered into lagged features.

All trainings were done using Amazon SageMaker ml.c5.4xlarge instances with 16 vCPUs and 16 GB of memory. It takes approximately 15 minutes for one round of DeepAR model training, and less than a minute for XGBoost training. Note that the HPO takes 30 rounds of training for DeepAR models and there are two DeepAR models involved for the sell-in prediction. The HPO takes 100 rounds of training for the XGBoost model. The end-to-end training time of the entire ML pipeline, including HPO is about 16 hours.

We use Weighted Mean Absolute Percentage error (WAPE) or Normalized Deviance (ND), defined by

$$WAPE = \frac{\sum |A_t - F_t|}{\sum |A_t|}, \tag{2}$$

for time series with actual value $\{A_t\}$ and forecasts $\{F_t\}$, as the evaluation metric of the model performance, which is the standard metric used at Nestlé . WAPE is more effective when it is important to prioritize popular items and reduce the error effect of non-popular items over the evaluation period. In this use case, the highest priority is to have an accurate prediction to make informed planning at the product category level, and WAPE better reflects the business impact of the forecasting accuracy. We also report the normalized root mean squared error (NRMSE) for completeness.

C. Hyperparameter optimization

The hyperparameters are optimized using the Optuna HPO library [14]. Optuna is a next generation HPO tool that helps to effectively search the hyperparameter space and come to the best model with optimized hyperparameters. Optuna uses state-of-the-art algorithms for sampling hyperparameters from the defined search space and efficiently pruning unpromising trials. This helps to reach optimum solution faster without an exhaustive grid search of the search space. We used Bayesian algorithm in Optuna for HPO in this study. Table I lists the parameters with their search ranges for the models used in this study. A total of 30 trials are made for each DeepAR model and 100 trials for each XGBoost model. The objective function used in the hyperparameter optimization is to minimize WAPE. The best parameters selected using HPO vary across dataset of products of the different categories. See Table II for an example of the optimized hyperparameter values for one category of products.

TABLE I
SEARCH RANGES OF HYPERPARAMETERS. FOR CATEGORICAL FEATURES, THE SEARCH WAS DONE TO DECIDE WHETHER TO INCLUDE EACH OF THEM.

	Description	Type	Range
Features	Holiday leads/lags	Integer	[-8,8]
	Promotions leads/lags	Integer	[-8,8]
	Price lags	Integer	[1,16]
	Actual sell-in lags	Integer	[1,16]
	Actual sell-out lags	Integer	[1,16]
	DeepAR sell-out pred. leads	Integer	[-2,-1]
	DeepAR sell-in pred. leads	Integer	[-2,-1]
	Categorical features	Boolean	Yes/No
DeepAR Param.	Number of layers	Integer	[3,8]
	Number of cells	Integer	[30,200]
	Context length multiplier	Integer	[1,8]
	Training epochs	Integer	[10,100]
	Learning rate	Float	[1e-4,1e-2]
	Batch size	Integer	[64,1024]
XGBoost Param	Number of estimators	Integer	[1,10]
	Learning rate	Float	[5e-4,0.5]
	Subsample rate	Float	[0.5,1]
	Max depth	Integer	[2,10]
	L1 regularization coeff.	Float	[0,0.1]
	L2 regularization coeff.	Float	[0.1,2]

TABLE II
OPTIMAL HYPERPARAMETERS FOR ONE PRODUCT CATEGORY, SELL-IN PREDICTION PROBLEM.

	Description	Optimal value
Features	Holiday leads/lags	[-4,1]
	Discount leads/lags	[-5,2]
	Price lags	[1,5]
	Actual sell-in lags	[1,3]
	Actual sell-out lags	[1,4]
	DeepAR sell-out pred. leads	[-2,-1]
	DeepAR sell-in pred. leads	[-2,-1]
	Categorical features	Client, PPG
DeepAR parameters, sell-out model	Number of layers	5
	Number of cells	126
	Context length multiplier	1
	Training epochs	47
	Learning rate	0.0032
	Batch size	64
DeepAR parameters, sell-in model	Number of layers	7
	Number of cells	177
	Context length multiplier	1
	Training epochs	42
	Learning rate	0.00018
	Batch size	1024
XGBoost parameters	Number of estimators	5
	Learning rate	0.33
	Subsample ratio	0.73
	Max depth	6
	L1 regularization coeff.	0.087
	L2 regularization coeff.	1.763

The lead/lags parameters are reported as an interval, meaning that the raw feature will be shifted by n periods for all integers n in the reported range. A negative number means we are using something with timestamps in the future.

It is worth mentioning that, for the sell-in prediction problem, we are doing the HPO in a two-stage process. Recall that the sell-in regression model requires outputs from two DeepAR models as inputs. To reduce the training overhead in both time and computation resource, we optimize and fix the parameters of the DeepAR models first, then use the fixed models to produce intermediate predictions that are fed to the regression model. The HPO for the regression model is then run only over the parameters related to feature engineering and XGBoost model.

V. RESULTS AND DISCUSSIONS

In this section, we first discuss the results obtained using the experiments listed in the previous section and then discuss some of the lessons learnt. The results of our experiments are listed in Table III. We first trained two DeepAR models - one for forecasting sell-in volumes and another for forecasting sell-out volumes. The sell-out models used weekly frequency scale and the sell-in models used monthly frequency scale based on availability of data and requirements stated by the business. The sell-out DeepAR model outperformed the benchmark exponential smoothing by 29%. The accuracy of the DeepAR model was further improved by 12% after HPO. The accuracy obtained using this sell-out model satisfied the requirements of the business. The results from the sell-in DeepAR model, although comparable to the exponential smoothing method, has a low accuracy of WAPE = 0.35. In order to use the model for forecasting, the business decision makers wanted a higher accuracy for sell-in forecasting. The HPO of the DeepAR sell-in model improved the result by 23%, but still did not meet the standards set by the business decision makers. We then experimented with XGBoost regression model for sell-in forecasting using only the manually engineered features, without introducing DeepAR prediction results as additional inputs. The results of the XGBoost model (Table III) showed better NRMSE as compared to DeepAR model, but has a higher WAPE (lower accuracy) as compared to DeepAR model. The observation remained the same even after HPO. Finally, we experimented using the two-step ensemble approach (Figure 1). The sell-in predictions using the ensemble approach outperfomed the rest of the methods including the benchmark exponential smoothing method by 40%. The accuracy was further improved by 29% using HPO.

The final sell-out forecasting model is a DeepAR model obtained using HPO and the final sell-in forecasting model is a ensemble model, combining DeepAR and XGBoost, obtained using HPO. Figure 3 shows an example of weekly forecast using the final sell-out model for a particular PPG-client pair. The WAPE for the prediction for 8 weeks using our sell-out model is 0.13 whereas the WAPE using the benchmark exponential smoothing model is 0.51. The prediction using the sell-out model matches the true value closely and picks up the trend particularly where the sell-out volume increases in mid-September. It should be noted that the trend of the sell-out volumes in the time period in the prediction interval differs largely from the previous two years (2020 and 2021) on which the model has been trained. Figure 4 shows an example of monthly forecast using the final sell-in model for a particular PPG-client pair. The WAPE for the prediction for 2 months using our sell-in model is 0.05 whereas the WAPE using the benchmark exponential smoothing model is 0.98. The benchmark model in this case failed completely to capture the trend or the true value. Similar observation as the sell-out example, in this case as well, the trends in the prediction interval varies largely when compared to the

same time period in the previous two years (2020 and 2021) on which the model has been trained. The patterns in the year 2020 are especially different due to the change in retail patterns due to the outbreak of global COVID-19 pandemic. Figure 5 highlights the overall performance of the sell-in model for the different client-product group combinations. The majority of the predictions have high accuracy. There are only couple of client-group pairs that have WAPE over 0.3. One such high client-group pair with high WAPE is Group5. This product is only sold at client3. A closer examination of these results were done by the business and proper explanations were drafted for the two anomalous client-group values with high WAPE. It was found that these two client-group data deviated from the overall pattern of the sell-in data for this product category and had insufficient history which caused the model to make predictions with lesser accuracy as compared to the other client-group values.

The results from the application of the state-of-the-art DeepAR model proved the business domain understanding that sell-in is a harder problem as compared to sell-out forecasting. Sell-in forecasting requires predicting the demand of a product at the wholesale level, rather than at the retail level. There are more factors that affect the demand of a product at a wholesale level such as the distributor's inventory levels, the retailer's merchandising plans, and the promotional activities. All of these factors add to the complexity of the problem. The sell-in ML pipeline hence needed special efforts to reach the desired level of accuracy. The DeepAR models provide new features to the XGBoost regression model that enhances its performance. We also found that HPO is effective in each step of the training.

We generated and executed the entire ML pipeline on AWS using services such as Amazon SageMaker, Amazon Simple Storage Service (S3) cloud storage, Amazon QuickSight. The solution architecture is shown in Figure 6. The input data is ingested manually into Amazon S3 buckets for this work. This step is currently being modified in production for automatic data ingestion. Once data is available, Amazon SageMaker Notebooks consume the data for data preprocessing and preparing the data for the ML training pipeline. The Amazon SageMaker training pipeline has the steps for the various models that are executed for the sell-out and sell-in forecasting. The models are trained and evaluated in this step along with HPO. The final model artifacts including the model weights are stored in Amazon S3 buckets. The trained model can be accessed using Amazon SageMaker inference endpoints for inference on new data. The predictions are stored in Amazon S3 bucket. The final output of this solution are Amazon QuickSight dashboards. The dashboards are prepared such that they have both the historical data and the predictions including the accuracy (WAPE) metrics. The dashboards are being used by the business users for their business decisions. The codes for the ML pipeline are developed as a software package. The package is designed to be extensible and easy to use. For this specific business case, we are only making weekly and monthly predictions, while the package has the capability to support predictions for any given fixed period, given data at the corresponding granularity. It also supports grouping the products at levels other than retail clients, SKUs and PPGs, or any hierarchical combination of those. As an extension of the business case, it can be used to forecast another time series of other categories of products, or those in a different geo-market.

TABLE III
PERFORMANCE METRICS OF THE PREDICTION MODEL FOR ONE PRODUCT CATEGORY.

	sell-out		sell-in	
Model	NRMSE	WAPE	NRMSE	WAPE
Exp. Smoothing	0.61	0.35	0.60	0.40
DeepAR	0.55	0.25	0.62	0.35
DeepAR+HPO	0.40	0.22	0.52	0.27
XGBoost	n/a	n/a	0.56	0.38
XGBoost+HPO	n/a	n/a	0.38	0.29
Ensembled	n/a	n/a	0.29	0.24
Ensembled+HPO	n/a	n/a	0.15	0.17

DeepAR models without hyperparameter tuning uses default settings with 2 RNN layers, 40 RNN cells, trained with learning rate 0.001, 50 epochs, batch size 32 and context length multiplier 6. XGBoost models without hyperparameter tuning uses default settings with 5 estimators, learning rate 0.3, subsample ratio 1, max depth 6, L1 regularization weight 0 and L1 regularization weight 1. Exponential smoothing models used optimized alphas of 0.42 for sell-out forecasting and 0.51 for sell-in forecasting.

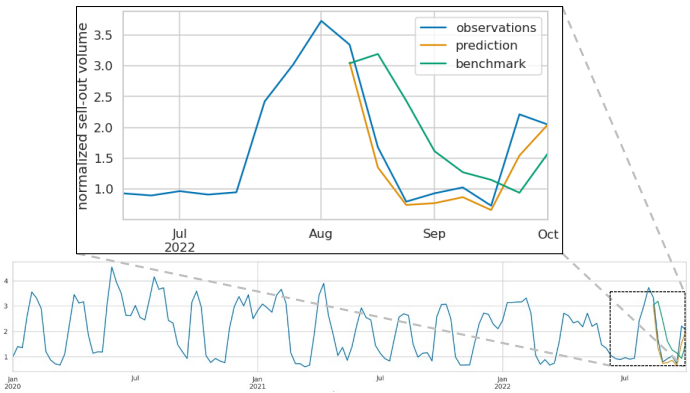

Fig. 3. An illustrative example for weekly sell-out prediction, 8 weeks in advance. Benchmark forecasts based on exponential smoothing. The numbers are normalized so that the sampled time series starts at 1.0

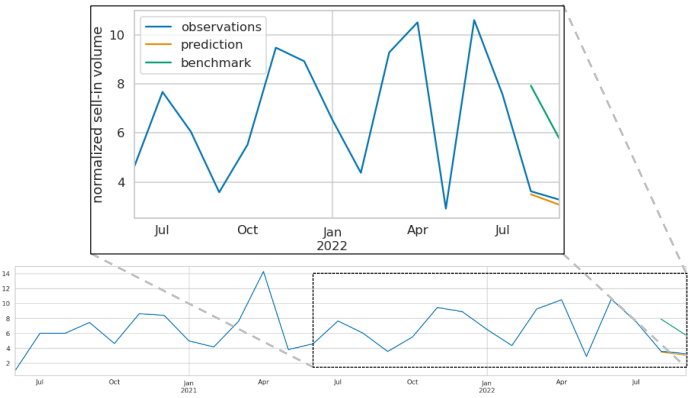

Fig. 4. An illustrative example for monthly sell-in prediction, 2 months in advance. Benchmark forecasts based on exponential smoothing. The numbers are normalized so that the sampled time series starts at 1.0

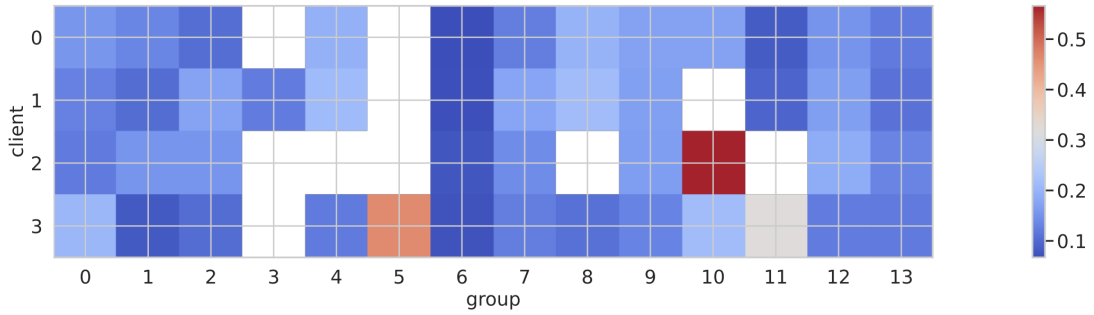

Fig. 5. Heat map showing WAPE for different clients and product groups combinations for one product category. White squares means n/a (that product is not sold at such client, or data is insufficient to make a prediction).

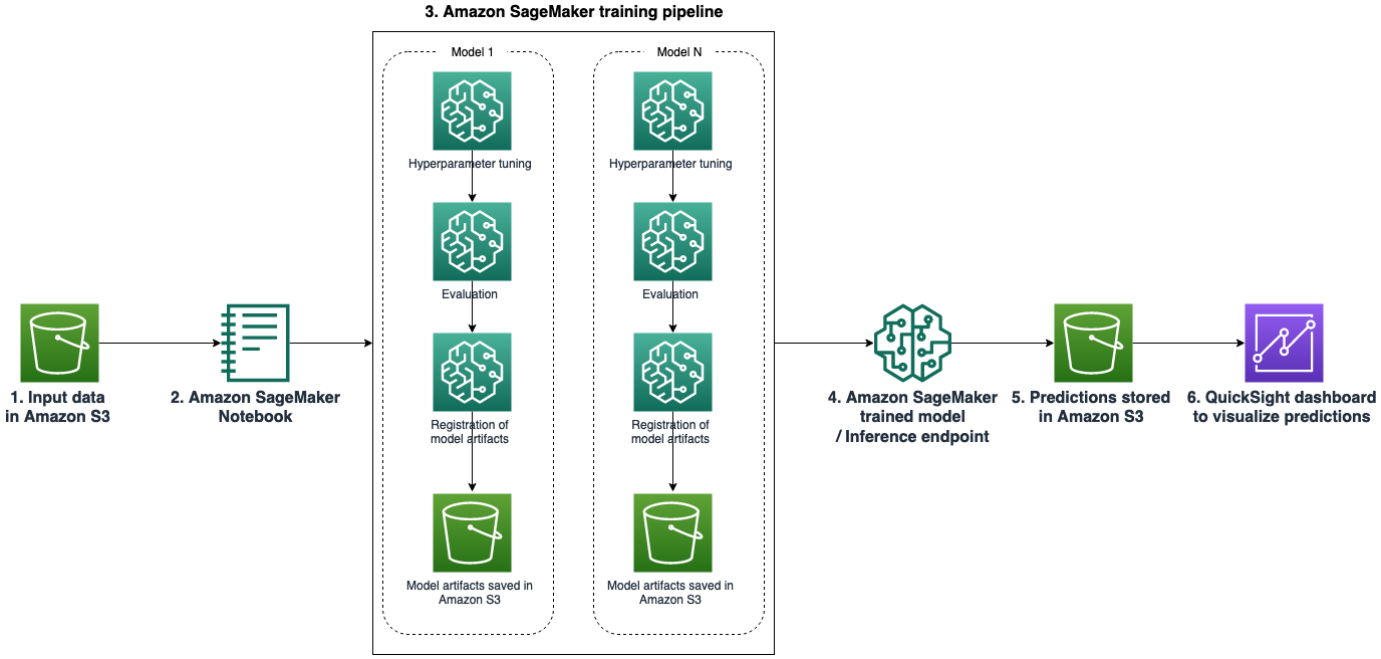

Fig. 6. The solution architecture of the end-to-end sell-in and sell-out forecasting workflow developed on AWS.

VI. Conclusion

In this paper, we show the effectiveness of ensemble machine learning methods in enhancing the accuracy of sell-in and sell-out forecasting. By combining multiple models, DeepAR and XGBoost, the ensemble approach was able to achieve better performance than the individual models, especially for the sell-in forecasting. The results show that the ensemble model improved the accuracy of sell-in forecasting by an average of 58%. Additionally, this study has highlighted the importance of feature selection and incorporating domain knowledge in improving forecasting performance. By carefully orchestrating the design of the ML pipeline using relevant features, the ensemble method was able to achieve even better results. The ML pipeline we built as part of this work is highly configurable and scalable, and can be extended globally across different geo-markets and for different product categories.

The results of this paper provide a solid foundation for future research in the area of sell-in and sell-out forecasting or retail forecasting using ensemble machine learning methods, and the potential benefits of this approach for businesses in various industries. However, it should be noted that the findings in this study are based on a small subset of data. Future research with a larger dataset will further validate the findings in this study.

Future work includes:

- Investigate the performance of the model when trained on a larger time period
- Comparison of the results with other DNNs that provide similar or improved accuracy for time-series forecasting problems
- Incorporate consumption data forecasting in the ML pipeline
- Evaluate the models using additional factors that might impact the forecasting. For example: macroeconomic features such as GDP of a country
- Comparison of different aggregation strategies - SKU/ brand/ category; weekly/ monthly granularity in predictions and its value to the business

ACKNOWLEDGMENT

The authors of this paper would like to thank the many colleagues at Nestlé and Amazon Machine Learning Solutions Labfor their contributions and feedback. Xin Chen, Elisabeth Cohen, Aldo Arizmendi thank you for your support and feedback

REFERENCES

[1] T. Boone, R. Ganeshan, A. Jain, and N. R. Sanders, "Forecasting sales in the supply chain: Consumer analytics in the big data era," *International Journal of Forecasting*, vol. 35, no. 1, pp. 170–180, 2019.

[2] D. M. Bechter, J. L. Rutner *et al.*, "Forecasting with statistical models and a case study of retail sales," *Economic Review*, vol. 63, no. Mar, pp. 3–11, 1978.

[3] R. Fildes, S. Ma, and S. Kolassa, "Retail forecasting: Research and practice," *International Journal of Forecasting*, vol. 38, no. 4, pp. 1283–1318, 2022.

[4] R. Fildes, S. Kolassa, and S. Ma, "Post-script—retail forecasting: Research and practice," *International Journal of Forecasting*, vol. 38, no. 4, pp. 1319–1324, 2022.

[5] I. Alon, M. Qi, and R. J. Sadowski, "Forecasting aggregate retail sales:: a comparison of artificial neural networks and traditional methods," *Journal of retailing and consumer services*, vol. 8, no. 3, pp. 147–156, 2001.

[6] G. C. Aye, M. Balcilar, R. Gupta, and A. Majumdar, "Forecasting aggregate retail sales: The case of south africa," *International Journal of Production Economics*, vol. 160, pp. 66–79, 2015.

[7] J. Huber and H. Stuckenschmidt, "Daily retail demand forecasting using machine learning with emphasis on calendric special days," *International Journal of Forecasting*, vol. 36, no. 4, pp. 1420–1438, 2020.

[8] R. V. Joseph, A. Mohanty, S. Tyagi, S. Mishra, S. K. Satapathy, and S. N. Mohanty, "A hybrid deep learning framework with cnn and bi-directional lstm for store item demand forecasting," *Computers and Electrical Engineering*, vol. 103, p. 108358, 2022.

[9] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "Deepar: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[10] E. S. Gardner Jr, "Exponential smoothing: The state of the art," *Journal of forecasting*, vol. 4, no. 1, pp. 1–28, 1985.

[11] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[12] S. Elsayed, D. Thyssens, A. Rashed, H. S. Jomaa, and L. Schmidt-Thieme, "Do we really need deep learning models for time series forecasting?" *arXiv preprint arXiv:2101.02118*, 2021.

[13] A. Alexandrov, K. Benidis, M. Bohlke-Schneider, V. Flunkert, J. Gasthaus, T. Januschowski, D. C. Maddix, S. Rangapuram, D. Salinas, J. Schulz *et al.*, "Gluonts: Probabilistic time series models in python," *arXiv preprint arXiv:1906.05264*, 2019.

[14] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2623–2631.