24th International Conference on Business Information Systems (BIS 2021)

Artificial Intelligence

https://doi.org/10.52825/bis.v1i.57

Published: 02 July 2021

Supporting an Expert-centric Process of New Product Introduction with Statistical Machine Learning

Shima Zahmatkesh[1][https://orcid.org/0000-0002-7832-0288], Alessio Bernardo[1][https://orcid.org/0000-0002-3492-0345], Emanuele Falzone[1][https://orcid.org/0000-0002-2699-2357], Edgardo Di Nicola Carena[2][https://orcid.org/0000-0001-6856-6957], and Emanuele Della Valle[1][https://orcid.org/0000-0002-5176-5885]

[1]DEIB, Politecnico di Milano, Italy

[2]Abstract s.r.l., Milano, Italy

Abstract. Industries that sell products with short-term or seasonal life cycles must regularly introduce new products. Forecasting the demand for New Product Introduction (NPI) can be challenging due to the fluctuations of many factors such as trend, seasonality, or other external and unpredictable phenomena (e.g., COVID-19 pandemic). Traditionally, NPI is an expert-centric process. This paper presents a study on automating the forecast of NPI demands using statistical Machine Learning (namely, Gradient Boosting and XGBoost). We show how to overcome shortcomings of the traditional data preparation that underpins the manual process. Moreover, we illustrate the role of cross-validation techniques for the hyper-parameter tuning and the validation of the models. Finally, we provide empirical evidence that statistical Machine Learning can forecast NPI demand better than experts.

Keywords: Demand Forecasting, New Product Introduction, Statistical Machine Learning, Gradient Boosting, XGBoost

Introduction

In several industries (e.g., Fashion), the period in which the products are saleable is likely to be short and seasonal. Since the products of these industries are replaced every new season by new ones, there are little relevant historical data available. Moreover, the industries may carry on many products with various futures corresponding to stock-keeping units (SKUs). Demand for these products is hardly stable or linear. It may be influenced by the fluctuations of many factors like weather conditions, holidays, marketing strategy, fashion trends, films, or even by celebrities and footballers. These factors make it challenging to forecast the demands for New Product Introductions (NPI). The major part of the companies uses manual efforts to predict the demands for NPI, the sales for existing manufactured goods, and the budget quantity or to benchmark the various products among them. Being manual processes, they can lead to inaccurate predictions, and so, failing in forecasting supply chain demands can cause under-staffing or over-staffing, incorrect operation budgeting, loss of credibility, failure in customer experience, economic loss in expenses, and waste of unsold products/services.

Several studies investigate NPI demand forecast in the literature, but most are not tailored for short life cycle environments. Most of the researches are based on diffusion theory, mainly including Bass [1] and Norton[2] models. Several methods have been developed based on these models, and different approaches are proposed in which analogical approaches are the

most important category. In this group of approaches, the assumption is that the diffusion patterns of the new products are similar to the analog products.

However, these approaches have some limitations. For example, experts define the similarity between products, and they often struggle to find a suitable benchmark. Some studies tried to overcome these limitations by using statistical Machine Learning [3], and Deep Learning [4] approaches focusing on optimizing the parameters of the Bass model.

In this paper, we aim to improve the judgment of human experts in forecasting NPI by utilizing statistical Machine Learning approaches (ML). In particular, we exploit statistical ML methods to automatically find the similarities between products and use them for forecasting NPI demands. In this way, the process no longer depends on expert judgments for the selection of similar products. As a result, we improve the forecast accuracy beyond the traditional methods, helping the short life cycle industries in making fast adaptations to their products to compete successfully in the market.

To this extent, we investigate the following research question:

RQ. *Utilizing the statistical Machine Learning, is it possible to improve the demand forecast for New Product Introduction done by the experts?*

In more details, the main contributions of this paper are:

- A characteristics analysis of a typical dataset collected in a short life-cycle industry;
- A data exploration task focusing on understanding and trying to enrich the data;
- The proposal of seven different approaches for predicting sales quantity using the Gradient Boosting and XGBoost statistical ML models;
- The exploration of the sensitivity to different cross-validation methods, hyper-parameters values, and feature encoding options; and
- A comparison, using the MAPE metric, of the results achieved by the proposed approaches with both a baseline and manual predictions, positively answering to **RQ**.

The remainder of the paper is structured as follows. Section Background presents the details of the statistical ML techniques used. Section NPI predictive analytics introduces the dataset used in this study and discusses the pre-processing analysis and the details of proposed approaches. Section Experimental Settings introduces the hypothesis tested and focuses on the design of the experiments that are carried out within this work, while Section Results and Discussion shows and discusses the results achieved. At the end, Section Related Work reviews the related work, and Section Conclusion concludes the paper.

Background

To solve the NPI problem and predict the future sale quantities, we utilized two different statistical Machine Learning models: Gradient Boosting [5] and XGBoost [6]. We introduce them in the following two sections. Moreover, to evaluate our models, we used two different cross-validation techniques. We present them in the last section.

Gradient Boosting

Gradient Boosting [5] refers to a class of ensemble ML algorithms that can be used for regression predictive modeling problems. Ensembles are constructed from decision tree models. Trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models. Models are fit using any arbitrary differentiable loss function and gradient descent optimization algorithm. This gives the technique its name, ”gradient boosting,” as the loss gradient is minimized as the model is fit, much like a Neural Network in Deep Learning. One way to produce a weighted combination of ensembles that optimizes the cost function is by gradient descent in function space. After calculating the loss, we must add a tree to the model

that reduces the loss (i.e., follow the gradient) to perform the gradient descent procedure. We do this by parameterizing the tree, then modifying the tree's parameters and moving in the right direction by reducing the residual loss. The output for the new tree is then added to the output of the existing sequence of trees to correct or improve the final output of the model.

Naïve gradient boosting is a greedy algorithm and can overfit the training dataset quickly. However, it can benefit from regularization methods that penalize various parts of the algorithm and generally improve the algorithm's performance by reducing overfitting. There are three types of enhancements to naïve gradient boosting that can improve performance:

- *Tree constraints*: the weak learners have skill but remain weak. There are several ways in which the trees can be constrained, such as the number of trees used in the ensemble, the depth of each tree, the minimum number of samples required to split an internal node, or the minimum number of samples required to be at a leaf node.
- *Weighted updates*: the predictions of each tree are added together sequentially. The contribution of each tree to this sum can be weighted to slow down the algorithm's learning. This weighting is called a learning rate.
- *Random sampling*: a considerable insight into bagging ensembles and random forests was allowing trees to be greedily created from sub-samples of the training dataset. This approach also reduces the correlation between the trees in the sequence.

There are some advantages in using the Gradient Boosting algorithm:

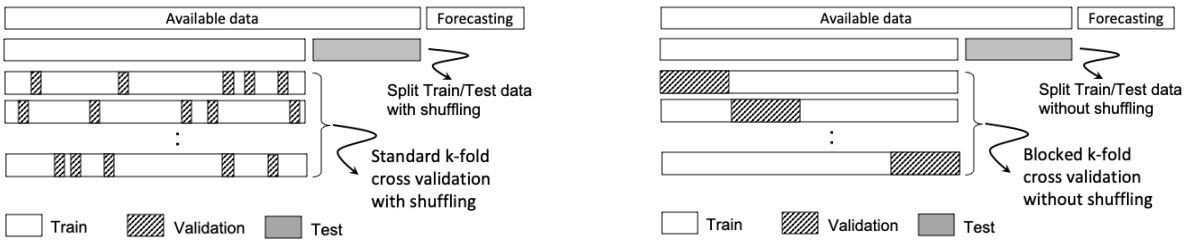
- *Better accuracy*: Gradient Boosting, compared with other regression techniques like Linear Regression, generally provides better accuracy. This is why it is used in most online hackathons and competitions.
- *Less pre-processing*: data pre-processing is one of the vital steps in ML workflow because it affects the model accuracy. However, Gradient Boosting requires minimal data pre-processing, which helps in implementing this model faster with lesser complexity.
- *Higher flexibility*: Gradient Boosting offers a wide range of hyper-parameters and loss functions. This makes the model flexible and usable for solving a wide variety of problems.
- *Missing data*: Gradient Boosting handles missing data[1] on its own. During the tree building phase, splitting decisions for a node are decided by minimizing the loss function and treating missing values as a separate category that can go either left or right.

XGBoost

eXtreme Gradient Boosting [6] is an optimized and distributed version of the Gradient Boosting algorithm. It improves upon the base Gradient Boosting framework through systems optimization and algorithmic enhancements. In particular, the system is optimized as follows.

- *Parallelization*: XGBoost approaches the process of sequential tree building using parallelized implementation.
- *Tree pruning*: the stopping criterion for tree splitting within the Gradient Boosting framework is greedy and depends on the negative loss criterion at the split point. XGBoost uses the *max-depth* parameter as specified instead of criterion first and starts pruning trees backward. This *depth-first* approach improves computational performance significantly.
- *Hardware optimization*: this algorithm has been designed to make efficient use of hardware resources. This is accomplished by cache awareness by allocating internal buffers in each thread to store gradient statistics. Further enhancements such as out-of-core computing optimize available disk space while handling big data-frames that do not fit into memory.

[1] In NPI forecasting, missing data is a severe problem that traditionally experts solve by selecting products similar to the one we want to predict demand for.



(a) Standard k-fold CV

(b) Blocked k-fold CV

Figure 1. Cross-validation approaches

While, algorithms are enhanced as follows.

- *Regularization*: it penalizes more complex models through both LASSO (L1) and Ridge (L2) regularization to prevent overfitting.
- *Sparsity awareness*: XGBoost naturally admits sparse features for inputs by automatically learning the best missing values depending on training loss and handles different types of sparsity patterns in the data more efficiently.
- *Weighted quantile sketch*: XGBoost employs the distributed weighted quantile sketch algorithm to find the optimal split points among weighted datasets effectively.
- Cross-validation: the algorithm comes with a built-in cross-validation method at each iteration, taking away the need to explicitly program this search and to specify the exact number of boosting iterations required in a single run.

Cross-Validation

One way to tune ML models hyperparameters is to split data into train and validation sets (keeping the test set only for the final evaluation). The models can be trained on a smaller train set, and the evaluation set can be used for evaluating the models. The standard k-fold cross-validation [7] technique, shown in Fig. 1(a), firstly randomly splits the data into k distinct subsets, called folds, and then it trains and evaluates the model k times, each time selecting one of the folds as the validation set and the rest of them as the training set. Then, it saves the evaluation score, and it discards the model. The scores are averaged over the rounds to give an estimate of the model's predictive performance. The goal of cross-validation is to test the model's ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or selection bias [8] and to give an insight on how the model will generalize to an independent dataset (i.e., an unknown dataset).

However, in the literature, some authors have raised some theoretical problems about using the k-fold cross-validation approach in time series prediction[2]. This led to the introduction of new methods to overcome these problems. They can be divided into three categories: 1) cross-validation based on the last block such as forward validation [9], [10], 2) cross-validation with omission of dependent data [7], and 3) cross-validation with blocked subsets [11].

In this study, besides the standard k-fold cross-validation, we also test the blocked k-fold cross-validation approach [11], shown in Fig. 1(b), in which the k-fold cross-validation is done without shuffling the training set at the beginning, i.e., the training set is temporally sorted and the order of products on time is preserved. Unlike the standard k-fold CV, this new version, avoiding the initial shuffling and using blocks of data contiguous in time as validation sets, does not use obvious temporal dependencies in the short term.

[2]Indeed, the typical datasets used in NPI demand forecast are time-series.

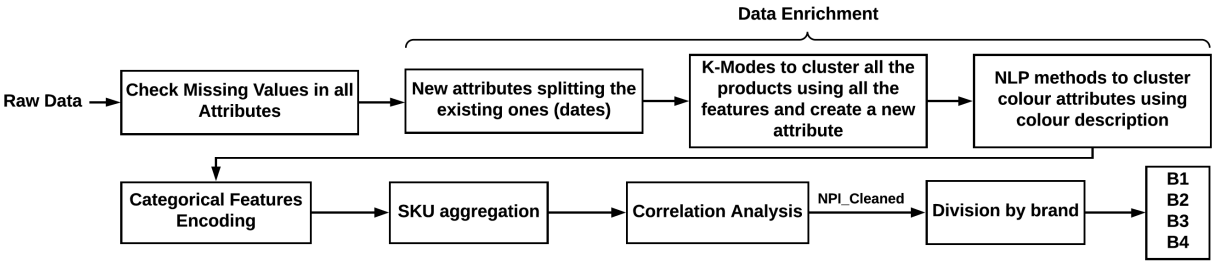


Figure 2. Data preparation pipeline

New Product Introduction predictive analytics

In this section, we present a dataset of past NPI sale quantities, and we describe the proposed solution for the NPI problem in detail. We divide it into two parts: 1) the data ingestion and preparation phase, and 2) the modeling phase.

NPI dataset

NPI dataset contains $30,750$ products introduced (or to be introduced) on the fashion market from 2017 to 2019. It aims at estimating new products' number of sales before introducing them on the market in 2019. From a business perspective, the company shall use the estimation to decide if it is worth or not introducing those products on the market. Each product has *67* features, and the label (ORDER-QTY) represents the number of sales in the first three months after the introduction. The features are divided into four groups:

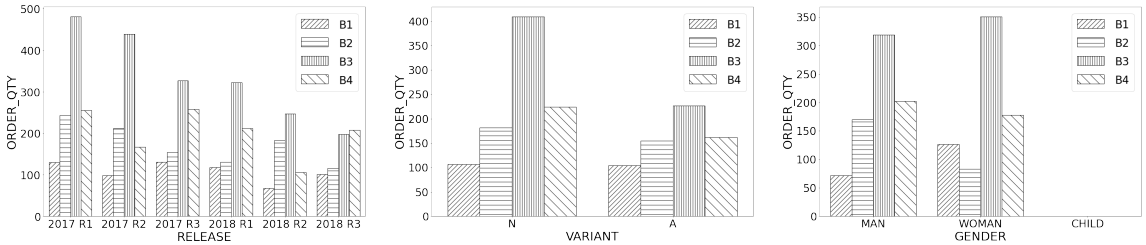
- features related to the product's characteristic, i.e., brand, model, color, material, price,
- time-related features, i.e., main release, and first availability date,
- benchmark features representing similar products from past collections *manually selected by experts*, i.e., benchmark-1, benchmark-2, and
- aggregated features, i.e., number of models by brand release, number of colors by model, and number of sizes by model.

Moreover, there is also an attribute named BGT-QTY representing the manual estimations of the experts.

Data ingestion and preparation

Fig. 2 shows a series of operations done during the data ingestion and preparation phase. Starting from the previous section's dataset, we first performed a missing values analysis all over the attributes finding seven attributes (five of them are aggregated features) having a lot of missing values. So we replaced them with ND, standing for "not defined," in case of nominal features, with the mean of the not null values in case of numerical features, and with a meaningful date format, to avoid formatting errors, in case of date features.

Moreover, we studied the ORDER-QTY demands distribution w.r.t. the different features. Fig. 3 shows the ORDER-QTY demands divided for brand respect to the RELEASE, VARIANT and GENDER attributes. In particular, from Fig. 3(a), we can notice different sales trend for brands. In brand B3, there is a decreasing trend in sales, while in brand B1 the sales are stable across the years. In brand B4, the sales increase in the last quarters of the year and then they decrease, while in brand B2, the sales slightly decrease during the years quarters. Fig. 3(b) shows that, for all brands, there were introduced more new models than variants of already existing ones. In particular, the most introduced models are from brand B3, followed by brand B4. Finally, Fig. 3(c) shows that brands B3 and B4 sold similarly among men and women. Brand B2 sold more among men, while brand B1 sold more among women. The second task of the pipeline refers to a data enrichment operation. For example, we split the *main release*, and the *alternative release* date attributes into, respectively, main release year, main release



(a) ORDER-QTY for Release (b) ORDER-QTY for Variant (c) ORDER-QTY for Gender

Figure 3. The ORDER-QTY divided for BRAND respect to the RELEASE, VARIANT and GENDER attributes

quarter, alternative release year, and alternative release quarter. We did the same with the *first availability date week*, *benchmark launch 1* and *benchmark launch 2* attributes. After that, we also decided to add another attribute to group similar products into the same cluster. Since the dataset has mixed numerical and nominal features, we used the K-Modes [12] algorithm to create some product clusters, and, through the Elbow curve [13], we selected the exact number of clusters to use (17 in our case). The last step consisted of adding 3 other attributes that cluster the products based on some features' descriptions.

In the next step, we performed a nominal feature encoding. We used the label encoder as a classical approach to encoding the nominal features into numerical values. In many ML approaches, numerical encoding would not perform very well as the nominal features do not necessarily have the ordinal relationship introduced when assigning them a number. However, for the ML methods that we use, this approach does not affect the models' performances since they inherently perform very well on nominal features. However, in some of the experiments, we applied the one-hot encoding procedure (detailed in the next section) to avoid the ML algorithm to learn in-existing patterns out of the ordinal relationship. The next step aggregated the products by the *SKU* attribute that combines in one attribute the *model*, *size* and *colour*. Then, we performed a correlation analysis to know the most correlated features to the label. Unfortunately, we discovered that there were no highly correlated features to the label. For this reason, we used all the features to train and test the models. The result of all these tasks was the so-called *NPI-Cleaned* dataset. The last task split it by brand (B1, B2, B3, B4), creating four other datasets that will be all used to train and test the models.

Data Modeling

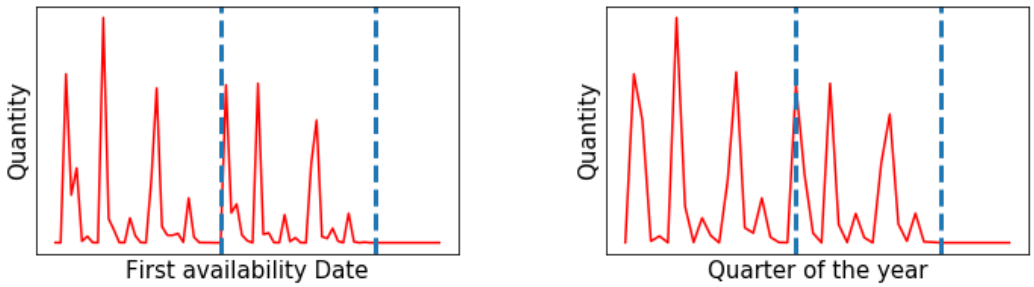
Starting from the five datasets created, we introduce seven approaches for each couple of learners (Gradient Boosting, XGBoost). They combine different cross-validation methods, hyper-parameters tuning values, and encoding. In the following, we introduce them in detail.

Baseline with cross-validation

Here, the standard 5-fold cross-validation approach over the train set was applied. So, we had 5 Gradient Boosting and 5 XGBoost models. Then, they were validated and tested on the respective validation and test sets. In the case of the two models trained on the *NPI-Cleaned* dataset, they were validated and tested, respectively, on the validation and test sets generated from the main dataset. Then the prediction results were divided by brand. Since this approach represents a first try that a typical data scientist would test, we used it as a baseline.

Partial one-hot encoding with K-fold CV (POH) and with blocked CV (POH-BCV)

In this approach, we applied some other feature engineering tasks w.r.t. the baseline. Firstly, we looked at the presence of the seasonality phenomena on the number of sales. Fig. 4 shows the number of products sold grouped by the *first availability date week*. Separating them by year (Fig. 4(a)), we can notice a sort of seasonality, but the peaks are not aligned over the



(a) ORDER-QTY grouped by the first availability date week

(b) ORDER-QTY grouped by a bi-weekly period

Figure 4. Seasonality phenomena in ORDER-QTY attribute

years. Instead, grouping them by every two weeks (Fig. 4(b)), we can notice that the peaks are more aligned over the years than before. So, we added a *bi-weekly* feature, and we removed all the other time-related features. Then, we also evaluated the benefits of using the *benchmarks* features selected by the experts. We inspected how often the NPI SKUs referred to the same benchmark SKUs, finding out that *Benchmark-1* is more present than *Benchmark-2* and that the experts often used the same few benchmarks, while a large number of benchmarks are used less often. We also inspected if there were any NPI SKUs that referred to older NPI SKUs as benchmark SKUs, finding out that only the $15 - 20\%$ of the products, later on, became benchmarks. It happened on average 15 times and at most 100 times. Moreover, the experts selected only a few products frequently or for a long period of time. Half of the products were seldom used, and still, others were never used. Our conclusion was that the *benchmarks* might be useless in the predictions, and they may even disturb the learning. So, we removed all the *benchmarks* features, and all the *aggregated* features, too. The last task was one-hot encoding the *categorical* features before training the two models. Since the major part of the categorical features had less than *10* distinct values, for this approach, we encoded only the features having more than *10* distinct values. At this point, we distinguished two approaches: POH and POH-BCV. In the former, we divided the datasets, and we applied the 5-fold cross-validation as in the baseline. Instead, in the latter, we used the 5-fold blocked cross-validation. Moreover, in both approaches, we explored a maximum tree depth between 6 and 12.

POH-BCV with one test set

This approach is based on the POH-BCV one, but instead of using the test and the validation sets, it uses only one test set. The whole training set was used to apply the 5-fold blocked cross-validation, so having more data points available during the training phase. The validation error is the mean of the errors during the 5-fold blocked cross-validation process.

Other approaches (POH-D4-6, OH, OH-D4-6)

We also tested three other approaches similar to the previously described ones: POH-D4-6 is the same as the POH approach, but it explores a maximum tree depth between 4 and 6; OH is based on the POH one, but it encodes all the categorical features, and OH-D4-6 is the same as the previous one, but it explores a maximum tree depth between 4 and 6.

Experimental Settings

This section firstly introduces the hypotheses to test, then, i) discusses the datasets splitting criteria applied, ii) proposes all the parameters used to train the models, iii) introduces the evaluation metric used for comparing the performances between the proposed approaches and the experts' prediction, and iv) describes the experimental environment.

Research Hypotheses

We formulated our hypotheses as follows:

- *Hp. 1*: Since the introduced Baseline with cross-validation approach is considered just as a baseline, the other approaches that use one-hot encoding, different types of cross-validations and perform more analysis on the data outperform baseline approach and generate more accurate predictions in terms of MAPE metric.
- *Hp. 2*: Applying statistical Machine Learning models, we can improve the forecasting accuracy with respect to the experts' prediction in terms of MAPE metric.

Splitting Criteria

In total, we have five datasets: one is the main dataset containing all the product sales, while the others are related to the specific brands (named B1- B4). We first discarded the 2019 data since they represent the products not already introduced on the market, so the products we need to predict the number of sales (forecasting). Then, we considered as the training set the data sold from 2017 R1 to 2018 R2 quarters and as the test set the data sold in the 2018 R3 quarter. Finally, we split the training set into the train and validation (70%-30%) sets before each method applies the two k-fold cross-validation approaches proposed.

Hyper-parameter Tuning

To improve the performance accuracy of our models, we used the two cross-validation approaches in a grid search to find the optimal values for each model's parameters. Grid search[3] fits different models using the range of defined values for the selected parameters and chooses the model parameter values that minimize the loss. Applying cross-validation to the grid search will help to avoid over-fitting.

We tested the following for the Gradient Boosting and XGBoost parameters:

- number of estimators: 1000, 2000;
- minimum number of samples to split: 5, 15;
- minimum number of samples to be at a leaf node: 5, 15;
- learning rate: 0.05, 0.01; and
- sub-sample rate: 0.5, 0.8;

For the XGBoost parameters in grid search, we also consider the following parameters:

- sub-sample ratio of columns: 0.5, 0.8;
- L1 regularization term on weights: 0.1, 0.9;
- L2 regularization term on weights: 0.1, 0.9; and
- minimum loss reduction gamma: 0.5, 0.8.

Evaluation Metric

All the approaches' predictive performances were evaluated with the Mean Average Percentage Error (MAPE)[4] metric. It measures the accuracy as a percentage and can be calculated as the average absolute percent error for each actual value minus the forecasted one divided by the actual value. Where A_t is the actual value, and F_t is the forecast value, this is given by:

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|A_t - F_t|}{A_t} * 100 \tag{1}$$

In our case, A_t were the ORDER-QTY values, while F_t were the predicted values.

Moreover, we also calculated the MAPE between the ORDER-QTY values and the experts' predictions BGT-QTY done by the company, and we compared it with our MAPE.

[3] https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e

[4] https://en.wikipedia.org/wiki/Mean_absolute_percentage_error

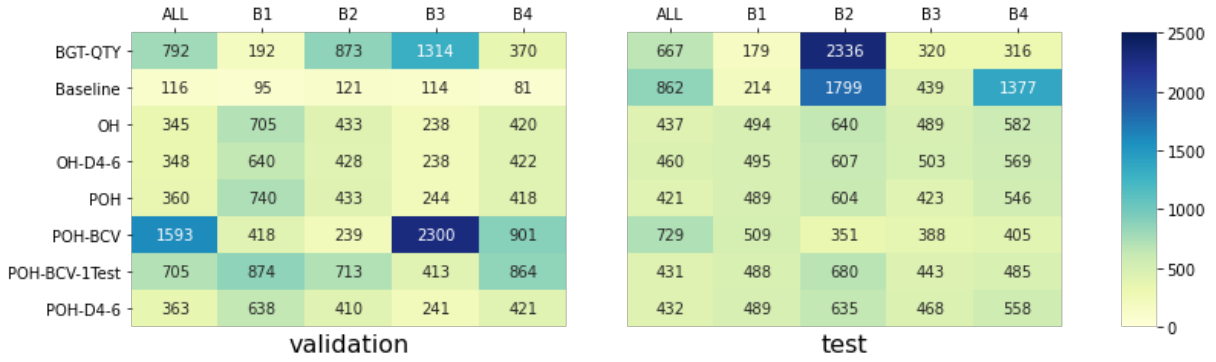


Figure 5. Result of Gradient Boosting Experiments based on MAPE metric

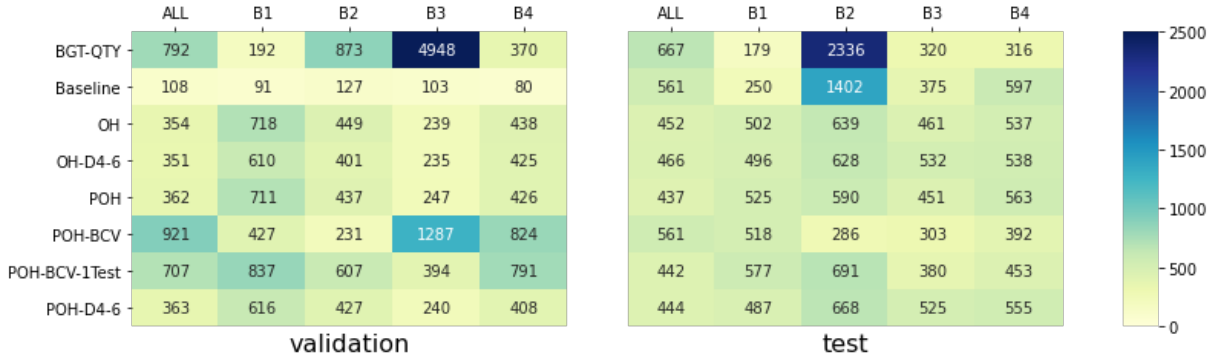


Figure 6. Result of XGBoost Experiments based on MAPE metric

Experimental Environment

All the tests were run on a machine having a CPU with 12 Core/24 Thread, 128GB RAM, and a GPU NVIDIA QUADRO P6000 with 24GB of RAM. In particular, we used Python 3 and the Scikit-learn[5] library for all the methods and approaches presented.

Results and Discussion

This section shows the results achieved by each proposed approach divided by the statistical ML model adopted.

Fig. 5 shows the Gradient Boosting MAPE results for the validation and test sets, using every single brand and for all brands. The validation results show that the Baseline approach achieved the best MAPE among different approaches. However, this is not the case for test results, which show that the baseline approach overfitted the data. In the test results, the POH approach was the best for all the brands. The POH-BCV approach was the best approach among proposed approaches for brands B2, B3, and B4, but the experts' predictions BGT-QTY were better in the case of brands B3, and B4. Also, for brand B1, the best approach was the experts' one. It is worth noting that the validation MAPE in the case of experts' approach corresponding to each proposed approach had different values due to the way we generated the validation set for each approach. In Fig. 5 we put the minimum value between all the achieved results. Fig. 6 shows the XGBoost MAPE results for the validation and test sets, using every single brand and for all brands. Also here, the validation results show that the Baseline approach outperformed the other approaches, but, in the test results, it had the best performance between proposed approaches only for the brand B1, proving that the XGBoost model overfitted the data. The POH approach was the best approach for all brands, while the POH-BCV approach

[5] https://scikit-learn.org/stable/

outperformed other proposed approaches in brands B2, B3, and B4. Comparing the BGT-QTY experts' predictions to the proposed approaches, the former outperformed the latter in the case of B1 and B4 brands.

So, using both statistical ML models, we can say that the *Hp. 1* hypothesis is verified, while, about the *Hp. 2* one, we can say that in 3 cases out of 5 our approaches are better than the experts' one and so verify *Hp. 2*, too.

Related Work

Different studies on demand forecasting for the new product introduction have been investigated in the literature, but most of them are not tailored for short life cycle environments. The most commonly used market demand forecasting methods are based on diffusion theory, mainly including Bass [1] and Norton [2] models. Several methods have been developed based on these models, in which analogical approaches are the most critical category. This approach assumes that a new product will behave as similar products do. Mik et al. [14] introduced a survey on existing approaches for demand forecasting proposing various types of NPI and showing which approach performs better in which situation.

Beheshti-Kashi et al. [15] introduced a survey of sale forecasting and NPI mainly in fashion markets. In particular, for NPI approaches, they reviewed seven studies in fashion markets focused on pre-processing of time series and sales data, color forecasting, E-commerce, and fast fashion sales forecasting, but none of the works in this survey focused on ML approaches. Therefore, in the following, we review some of the studies with a focus on ML approaches.

Lee et al. [3] utilized statistical ML approaches to overcome the challenges existing in analogical approaches and predict the parameters of the Bass model. They created a reliable relationship between the attributes and diffusion characteristics of the existing products, so similar products are automatically selected without any human manipulation and used to forecast new products. Their experimental validation showed that most single prediction models and the ensemble model outperform the conventional analogical method. However, in our study, statistical ML approaches are directly used to finding the similarities between products.

Steenbergen et al. [16] proposed a novel new product forecasting method called DemandForest, which combines K-means, Random Forest, and Quantile Regression Forest. Their approach utilizes the historical demand of existing products and the product characteristics of both new and existing products to make pre-launch forecasts. Furthermore, the Quantile Regression Forest (QRF) algorithm quantifies the uncertainty of the demand and can be used to construct prediction intervals. Their proposed methods are evaluated on a synthetic dataset and five real-life datasets. They showed that DemandForest is a generalizable computational approach that also provides the uncertainty of demand for new products. Their focus on the uncertainty of the demand is out of the scope of this study.

Loureiro et al. [17] focused on the fashion retail industry and explored the use of a deep learning approach to forecasting sales, predicting the sales of new individual products in future seasons. They also compared the sales predictions obtained with the deep learning approach with other ML techniques such as Decision Trees and Random Forests. Their results demonstrated that the use of Deep Neural Network and other data mining techniques for performing sales forecasting in the fashion retail industry is auspicious.

Yin et al. [4] proposed a hybrid model for sale forecasting based on product similarity, which is measured through applying a quantitative similarity measurement method. Also, they employed an ensemble deep learning method to improve the low prediction accuracy caused by insufficient consideration of factors affecting the demand for the new product. Their empirical results proved that the forecasting accuracy could be improved using the deep learning method.

To conclude, several works in market demand forecasting have considered methods based on diffusion theory, such as the Bass model and similarities between products. Instead, in our work, we utilized the statistical ML approaches to automatically finding the similarities between products. Furthermore, the studies done in fashion markets have focused on different methodologies such as time series prediction or fast fashion sales forecasting. In contrast, in this work, we have exploited statistical ML approaches for the NPI problem to improve the forecast accuracy beyond the traditional methods. Lee et al. [3] also used statistical ML approaches, but unlike our approach, they applied it to predict the parameters of the Bass model. However, few works have applied other ML approaches, but none of them have our focus. For instance, Yin et al. [4] applied deep learning methods for sale forecasting.

Conclusion

This study proposes a statistical ML solution for demand forecasting of the New Product Introduction. Using different data pre-processing methods such as data clustering, encoding, grid search, cross-validation approaches, and ML models like Gradient Boosting and XGBoost, we proposed seven approaches for predicting sales quantity in short life-cycle industries (e.g., Fashion). We compared our approaches with both a baseline and the experts' predictions from an industrial partner of ours.

As a result, we found that the baseline approach, which is the first try that a typical data scientist will test, suffers from overfitting. However, for all brands, our proposed approach with partial one-hot encoding and standard 5-fold cross-validation achieved the lower MAPE values in the prediction of validation and test sets (**RQ**). Furthermore, considering single brand evaluation, for brands B2 and B3, the PHO-BCV approach outperforms the experts' prediction. Moreover, although splitting data to different brands can help experts have a more accurate prediction, statistical ML approaches are not the case. Table 1 reports the winner approach comparing the MAPE values of all the experiments and the experts' prediction. In conclusion, we showed that combining statistical ML methods with different data pre-processing tasks can improve the experts' manual predictions and, more in general, can help industries to predict the demands for New Product Introductions better.

As future work, we intend to investigate the ensemble of different ML models and check if we can achieve more accurate predictions. Using ensemble forecasting, we can apply multiple forecast methods independently and finally come to the final forecast. Moreover, in this study, we only focused on the features of the products. However, it is worth paying attention to the customer-generated content. The intentions of potential customers may have some predictive value which can help to improve the prediction. We took into account neither any temporal dependency nor fluctuations nor not-stationarity among the data, too. Future versions of this work should take into account them.

Table 1. Summary of the winner approach

Model	Data	Brand				
		ALL	B1	B2	B3	B4
Gradient Boosting	Validation	Baseline	Baseline	Baseline	Baseline	Baseline
	Test	POH	Experts'	POH-BCV	POH-BCV	Experts'
XGBoost	Validation	Baseline	Baseline	Experts'	Baseline	Baseline
	Test	POH	Experts'	POH-BCV	POH-BCV	Experts'

References

[1] F. M. Bass, “A new product growth for model consumer durables,” *Management science*, vol. 15, no. 5, pp. 215–227, 1969.

[2] J. A. Norton and F. M. Bass, “A diffusion theory model of adoption and substitution for successive generations of high-technology products,” *Management science*, vol. 33, no. 9, pp. 1069–1086, 1987.

[3] H. Lee, S. G. Kim, H.-w. Park, and P. Kang, “Pre-launch new product demand forecasting using the bass model: A statistical and machine learning-based approach,” *Technological Forecasting and Social Change*, vol. 86, pp. 49–64, 2014.

[4] P. Yin, G. Dou, X. Lin, and L. Liu, “A hybrid method for forecasting new product sales based on fuzzy clustering and deep learning,” *Kybernetes*, 2020.

[5] J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” *Annals of statistics*, pp. 1189–1232, 2001.

[6] 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.

[7] S. Arlot, A. Celisse, *et al.*, “A survey of cross-validation procedures for model selection,” *Statistics surveys*, vol. 4, pp. 40–79, 2010.

[8] G. C. Cawley and N. L. Talbot, “On over-fitting in model selection and subsequent selection bias in performance evaluation,” *The Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010.

[9] U. Hjorth and U. Hjort, “Model selection and forward validation,” *Scandinavian Journal of Statistics*, pp. 95–105, 1982.

[10] J. U. Hjorth, *Computer intensive statistical methods: Validation, model selection, and bootstrap*. CRC Press, 1993.

[11] J. Racine, “Consistent cross-validatory model-selection for dependent data: Hv-block cross-validation,” *Journal of econometrics*, vol. 99, no. 1, pp. 39–61, 2000.

[12] Z. Huang, “Clustering large data sets with mixed numeric and categorical values,” in *Proceedings of the 1st pacific-asia conference on knowledge discovery and data mining,(PAKDD)*, Citeseer, 1997, pp. 21–34.

[13] T. M. Kodinariya and P. R. Makwana, “Review on determining number of cluster in k-means clustering,” *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.

[14] E. C. Mik and G. Koole, “New product demand forecasting,” *Vrije Universiteit Amsterdam, Amsterdam*, 2019.

[15] S. Beheshti-Kashi, H. R. Karimi, K.-D. Thoben, M. Lütjen, and M. Teucke, “A survey on retail sales forecasting and prediction in fashion markets,” *Systems Science & Control Engineering*, vol. 3, no. 1, pp. 154–161, 2015.

[16] R. van Steenbergen and M. Mes, “Forecasting demand profiles of new products,” *Decision support systems*, vol. 139, p. 113 401, 2020.

[17] A. L. Loureiro, V. L. Miguéis, and L. F. da Silva, “Exploring the use of deep neural networks for sales forecasting in fashion retail,” *Decision Support Systems*, vol. 114, pp. 81–93, 2018.