Machine Learning with Applications 7 (2022) 100239

Contents lists available at ScienceDirect

Machine Learning with Applications

journal homepage: www.elsevier.com/locate/mlwa

Machine Learning Outperforms Classical Forecasting on Horticultural Sales Predictions

Florian Haselbeck [a,b,*], Jennifer Killinger [a,b], Klaus Menrad [c,d], Thomas Hannus [e], Dominik G. Grimm [a,b,f,*]

[a] Technical University of Munich, TUM Campus Straubing for Biotechnology and Sustainability, Bioinformatics, Schulgasse 22, D-94315 Straubing, Germany
[b] Weihenstephan-Triesdorf University of Applied Sciences, Bioinformatics, Petersgasse 18, D-94315 Straubing, Germany
[c] Technical University of Munich, TUM Campus Straubing for Biotechnology and Sustainability, Marketing and Management of Biogenic Resources, Schulgasse 22, D-94315 Straubing, Germany
[d] Weihenstephan-Triesdorf University of Applied Sciences, Marketing and Management of Biogenic Resources, Am Essigberg 3, D-94315 Straubing, Germany
[e] Weihenstephan-Triesdorf University of Applied Sciences, Retail Management, Am Staudengarten 10, D-85354 Freising, Germany
[f] Technical University of Munich, Department of Informatics, Boltzmannstr. 3, D-85748 Garching, Germany

ARTICLE INFO

Dataset link: https://github.com/grimmlab/HorticulturalSalesPredictions

Keywords:
Sales Forecasting
Time Series Forecasting
Comparative study
Horticulture
Machine Learning

ABSTRACT

Forecasting future demand is of high importance for many companies as it affects operational decisions. This is especially relevant for products with a short shelf life due to the potential disposal of unsold items. Horticultural products are highly influenced by this, however with limited attention in forecasting research so far. Beyond that, many forecasting competitions show a competitive performance of classical forecasting methods. For the first time, we empirically compared the performance of nine state-of-the-art machine learning and three classical forecasting algorithms for horticultural sales predictions. We show that machine learning methods were superior in all our experiments, with the gradient boosted ensemble learner XGBoost being the top performer in 14 out of 15 comparisons. This advantage over classical forecasting approaches increased for datasets with multiple seasons. Further, we show that including additional external factors, such as weather and holiday information, as well as meta-features led to a boost in predictive performance. In addition, we investigated whether the algorithms can capture the sudden increase in demand of horticultural products during the SARS-CoV-2 pandemic in 2020. For this special case, XGBoost was also superior. All code and data is publicly available on GitHub: https://github.com/grimmlab/HorticulturalSalesPredictions.

1. Introduction

Predicting future demand to support corporate analysis and decision-making is a potential competitive advantage in many domains. One solution for forecasting customer demand are time series prediction methods. With accurate estimations, company managers can quickly react to changing market signals and consequently adjust their procurement and production plans. These possibilities may lead to increased revenues when early adoption due to a rising demand is applied or decreasing costs as a response to a decline (Ivanov et al., 2019). This becomes even more relevant when dealing with goods that have a limited shelf life and can therefore only be kept in stock for a few days (Duan et al., 2012). Horticultural products are strongly influenced by these issues.

Currently, there are no scientific publications regarding time series prediction for sales of horticultural products, although total sales of ornamental plants are worth €9.4 billion in 2020 in Germany (Zentralverband Gartenbau e.V, 2020). Horticultural sales are usually characterized by a strong seasonality. In addition, sales cycles of certain products are shaped by abrupt changes in both directions of rising and decreasing numbers. Moreover, various external factors such as holidays or regional events affect demand. In addition, some of these factors, such as weather forecasts, are uncertain and only available for a short period (Behe et al., 2012). Furthermore, the short shelf life of horticultural products, particularly of cut flowers, is a challenge for operational decisions. In practical operations, this may cause out-of-stock situations with possibly missed sales as well as excess-stock cases, which often lead to the disposal of products. Besides financial loss, the latter induces environmental damage due to wasted resources during production and transport.

[*] Corresponding authors at: Technical University of Munich, TUM Campus Straubing for Biotechnology and Sustainability, Bioinformatics, Schulgasse 22, D-94315 Straubing, Germany.

E-mail addresses: florian.haselbeck@hswt.de (F. Haselbeck), jennifer.killinger@tum.de (J. Killinger), klaus.menrad@hswt.de (K. Menrad), thomas.hannus@hswt.de (T. Hannus), dominik.grimm@hswt.de (D.G. Grimm).

https://doi.org/10.1016/j.mlwa.2021.100239
Received 22 October 2021; Received in revised form 3 December 2021; Accepted 15 December 2021
Available online 21 December 2021

Classical forecasting methods such as Autoregressive Integrated Moving Average and Exponential Smoothing are still widely applied in research and industry. Despite their rather simple concept, they often show a competitive performance. As demonstrated in several publications and contests such as the M-Competitions, they are even able to outperform more complex machine learning (ML) approaches, e.g. Artificial Neural Networks (Kolassa, 2021; Makridakis et al., 2020, 2021). Thus, although ML algorithms are becoming more common in the forecasting literature, it is not clear that they are superior, but dependent on the application and data.

These horticultural product-specific aspects and unsolved questions in forecasting encourage conducting research in this domain. Hence, in this paper, we present a first-time comparison of ML-based and classical time series prediction methods to forecast sales for products such as potted plants, cut flowers and shrubs. We did not consider non-herbal products merchandized in this industry, e.g. plant pots, and edible products such as vegetables and fruits, which were already taken into account by other researchers (Arunraj & Ahrens, 2015; Priyadarshi et al., 2019; Sankaran, 2014; Shukla & Jharkharia, 2011). First, we want to answer the question whether ML is superior compared to classical forecasting approaches for horticultural sales predictions. Second, we investigate potential improvements by multivariate concepts making use of external factors, such as weather or holiday data, by comparing with classical univariate methods. Third, we examine the computational resource consumption of the applied methods, a critical issue with the necessity of model refittings due to potentially changing data distributions during the live operation of a forecasting system in mind. In addition, we evaluate whether the models are able to capture sudden change in data, such as strong increases in demand during the SARS-CoV-2 pandemic in 2020.

To this end, we present a comparative study of nine state-of-the-art ML and three classical forecasting methods. Regarding classical forecasting approaches, we used the univariate techniques Exponential Smoothing and Seasonal Autoregressive Integrated Moving Average as well as the multivariate extension of the latter. Furthermore, we show a comparison to Multiple Linear Regression models with different regularizations. We included nonlinear ML methods such as Artificial Neural Networks as well as Long Short-Term Memory Networks. Moreover, we applied the ensemble learner Extreme Gradient Boosting (XGBoost), and a nonparametric Bayesian approach by implementing Gaussian Process Regression. Furthermore, we implemented a setup with a regular refit of the forecasting model in order to simulate a potential scenario for a productive operation with a continuous update of company-specific sales data and a potentially changing data distribution in mind. Finally, we are able to provide initial insights in the new forecasting domain of horticultural sales.

The remainder of this paper is organized as follows. In Section 2, we provide an overview of related work. Afterwards, we describe the materials and methods, see Section 3. In Section 4, we outline and discuss our results, before we draw conclusions.

2. Related Work

In the following section, we summarize classical forecasting and ML methods that can be used for predicting demand. Then, we describe empirical comparisons and related work from similar domains, such as the food and tourism sector.

2.1. Machine Learning-based and Classical Time Series Forecasting Methods

In Table 1, we provide an overview of the approaches discussed in this paper. Classical forecasting methods use chronologically ordered time series data and try to predict future sequences, usually by projecting statistical information recovered from historical data. Methods can be divided into univariate and multivariate approaches, with the latter one using external information in addition to the time series itself. Two common univariate methods are Exponential Smoothing (ES) and Autoregressive Integrated Moving Average (ARIMA) (Box et al., 2016; Holt, 1957; Winters, 1960). In its simple form, ES is a weighted sum of past observations of a time series with weights decaying exponentially. In contrast, modeling autocorrelations – the correlation between a series and a lagged version of itself – is the key idea of ARIMA. A common extension, SARIMA, involves seasonal parts (Gardner, 2006; Hyndman & Athanasopoulos, 2018). Furthermore, a multivariate version involving external factors called SARIMAX can be formulated (Arunraj et al., 2016).

Sales forecasting can also be defined as a regression task for ML methods. In ML, we can distinguish between Frequentist and Bayesian formulations of various models with the latter one referred to as probabilistic in this work. Here, Frequentist ML methods minimize the empirical risk for a certain loss function including different regularization terms to determine point estimates of the most likely model parameters. Probabilistic approaches instead use Bayes' theorem to calculate a full posterior distribution over the parameters given the data and a prior (Bishop, 2009; James et al., 2017). For regression tasks, Multiple Linear Regression (MLR) is often used as a baseline. Common approaches are Ridge, Lasso and Elastic Net Regression, which involve different regularization terms in order to prevent overfitting (Hoerl & Kennard, 1970; James et al., 2017; Santosa & Symes, 1986; Tibshirani, 1996; Zou & Hastie, 2005). MLR can further be defined in a Bayesian perspective as follows: $p(y|\boldsymbol{x},\boldsymbol{w}) = \mathcal{N}(y|\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{w},\sigma^2)$, where the target variable y follows a Gaussian distribution with the mean $\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{w}$ (determined by the predictors $\boldsymbol{x}$ and the weights $\boldsymbol{w}$) and the variance σ^2. If the weights of the model are constrained to a zero-mean Gaussian prior, the solution is equivalent to Ridge Regression and therefore called Bayesian Ridge Regression (BayesRidge). Automatic Relevance Determination (ARD) is related to it, but introduces an individual variance for each weight (Bishop, 2009; James et al., 2017; Tipping, 2001). A widely applied approach in time series prediction are Artificial Neural Networks (ANN) (Rosenblatt, 1958; Zhang et al., 1998). Especially Recurrent Neural Networks (RNN), such as Long Short-Term Memory Networks (LSTM), are suitable for time series data, since they are able to capture temporal dependencies by definition (Hewamalage et al., 2021; Hochreiter & Schmidhuber, 1997). Several publications and forecasting competitions indicate a superiority of combined methods, e.g. achieved via ensemble learners (Bojer & Meldgaard, 2021; Petropoulos et al., 2018). XGBoost (XGB) is an ensemble technique that uses gradient boosted regression trees. With this approach, decision trees are sequentially added in a greedy manner based on the gradient of the loss function in order to correct the errors made by the current ensemble (Chen & Guestrin, 2016). With regard to the practical use of demand forecasts, the uncertainty of a prediction value seems profitable. Providing those is a main advantage of the nonparametric Bayesian method Gaussian Process Regression (GPR) (Williams & Rasmussen, 1996). The determining parameters of a Gaussian Process are the kernel $k(\boldsymbol{x},\boldsymbol{x}')$, which consists of the covariance value between any two sample points $\boldsymbol{x}$ and $\boldsymbol{x}'$ resulting in a $n \times n$ matrix for a training set length of n and the mean function $m(\boldsymbol{x})$. The assumption is that the similarity between samples reflects the strength of the correlation between their corresponding target values. Therefore, the function evaluation can be seen as a draw from a multivariate Gaussian distribution defined by $m(\boldsymbol{x})$ and $k(\boldsymbol{x},\boldsymbol{x}')$. Thus, Gaussian Processes are rather a distribution over functions (Rasmussen & Williams, 2008; Roberts et al., 2013). More detailed descriptions can be found in Appendix A.

2.2. Time Series Forecasting Competitions and Related Domains

There are several publications regarding empirical comparisons of forecasting approaches, e.g. the influential M-competitions that started in 1982 (Bojer & Meldgaard, 2021; Hong et al., 2019; Lloyd, 2014;

Table 1
Overview of classical forecasting and machine learning methods with their abbreviations and certain characteristics.

Category	Algorithm	Abbreviation	Univariate	Multivariate	Probabilistic
Classical forecasting	Exponential Smoothing	ES	×		
	Seasonal Autoregressive Integrated Moving Average	SARIMA	×		
	Seasonal Autoregressive Integrated Moving Average with external factors	SARIMAX		×	
Machine learning	Lasso Regression	LassoReg		×	
	Ridge Regression	RidgeReg		×	
	Elastic Net Regression	ElasticNet		×	
	Artificial Neural Network	ANN		×	
	Long Short-Term Memory Network	LSTM		×	
	Extreme Gradient Boosting (XGBoost)	XGB		×	
	Bayesian Ridge Regression	BayesRidge		×	×
	Automatic Relevance Determination	ARD		×	×
	Gaussian Process Regression	GPR		×	×

Makridakis et al., 1982, 1993; Makridakis & Hibon, 2000; Makridakis et al., 2018, 2020, 2021; Stepnicka & Burda, 2017). Nevertheless, these competitions do not show a general superiority of a specific approach. While the M4-competition did not yield advantages for single ML models, but for approaches combined with classical forecasting methods, gradient boosted trees were superior for the recent M5-study with strongly correlated time series and explanatory variables (Makridakis et al., 2018; Seaman & Bowman, 2021). This is in accordance with a recent review of several *Kaggle* forecasting competitions (Bojer & Meldgaard, 2021). So in summary, combined respectively ensemble methods were advantageous in many empirical comparisons without yielding a generally superior technique.

Besides these broad forecasting competitions, there are publications in similar domains such as food and tourism demand forecasting. Food demand forecasting is a domain with similarities to horticultural sales, especially when dealing with agricultural products such as vegetables or fruits, which are also perishable and seasonal. Arunraj et al. showed the superiority of SARIMAX supplemented with holiday and promotional features over its univariate version SARIMA when predicting sales of bananas in a retail store (Arunraj et al., 2014). A comparison of several ML methods forecasting food sales of a Japanese supermarket chain with an emphasis on the influence of meteorological data was done in (Liu & Ichise, 2017). It yielded a better performance of a LSTM compared to others, e.g. Random Forest. A recently published comparative study on bakery sales predictions, including LSTM and gradient boosted regression trees, indicated a superiority of ML algorithms (Huber & Stuckenschmidt, 2020). Besides that, the demand for ornamental plants is characterized by a strong seasonality with abrupt changes in requests and influenced by various external factors, such as the weather and holidays. Thus, tourism is another domain, which shares similar attributes. Athanasopoulos et al. compared the performance of several univariate time series algorithms with their multivariate extensions based on 366 monthly, 427 quarterly and 518 annual series. They concluded that the first ones deliver better results, but also mentioned that this is in contrast to other publications (Athanasopoulos et al., 2011). Jiao and Chen provided a review of methodological developments in tourism forecasting during the 2008 to 2017 time period. According to them, econometric and time series models are still widely applied and often lead to competitive predictions Besides that, they observed a trend in enhancing time series with additional features as well as an increasing use of ML methods. Furthermore, combining different approaches often seems to yield better results than single ones (Jiao & Chen, 2019). In summary, these findings in similar domains provide some guidance for horticultural demand forecasting, but also do not show a general superiority of classical or ML-based approaches.

3. Materials and Methods

In the next section, we provide insights into the data and its preparation. Afterwards, we describe the model selection and training. Finally, we outline the evaluation metrics and baselines. All code and data is publicly available on GitHub: https://github.com/grimmlab/HorticulturalSalesPredictions

3.1. Data Preparation

In the following, we provide an overview on the datasets. Afterwards, we summarize the feature engineering and data preprocessing steps in detail.

3.1.1. Datasets

For our analysis, we used typical horticultural retail sales data from Germany. We distinguished between five datasets based on two data sources, as shown in Fig. 1. *OwnDoc* was manually created by a daily documentation of sales numbers of tulips. These datasets have scientifically interesting characteristics, such as a strong seasonality in early spring with abrupt changes in demand. Regarding predictions, we focused on private customer sales of tulips, subsequently called *SoldTulips*. Records for the *OwnDoc* dataset begin on February 7, 2020 and last until May 11, 2020, as the tulip season usually ends in mid-May. Thus, data exists for one seasonal cycle lasting about three months. *OwnDoc* shows a detailed product aggregation level of the target variable with the plant species and delivers quantities. As indicated in Fig. 1, we generated two datasets based on *OwnDoc*. There is a 16 day long documentation gap starting on April 10. Thus, we derived a variant for which we only use values before that period (*OwnDoc_SoldTulips_short*), whereas we imputed them for the long one (*OwnDoc_SoldTulips_long*).

CashierData was created based on an electronic cashier system providing a summary of all sales. These sales figures, which are aggregated into product groups, were accumulated to daily turnovers. As we focused on herbal products, we selected cut flowers (*CutFlowers*) and potted plants (*PotTotal*) as target variables. Beyond that, *CashierData* differs from *OwnDoc* regarding several properties. It ranges from December 2016 to August 2020 and as a result contains several seasonal cycles. Additionally, due to the aggregation into product groups, we can only see patterns of these whole clusters instead of individual plants. Since *CashierData* is based on exports from an electronic cashier system, for which a gapless logging is required by tax law, there are no missing values. Furthermore, numbers in *CashierData* represent the turnover in euros. For *CashierData*, we derived three datasets, see Fig. 1. The first one reflects sales of *CutFlowers* over the whole period *(CashierData_CutFlowers)*. Besides this, we observe a sharp increase of revenues for *PotTotal* in the first half of 2020, probably caused by the SARS-CoV-2 pandemic. Therefore, we separated an alternative version ending 12/2019 (*CashierData_PotTotal_short*) in addition to the one ranging over the whole period (*CashierData_PotTotal_long*).

Table 2 shows characteristics for all datasets on a daily basis as well as on a weekly one for all *CashierData* versions. As already mentioned, *CashierData* does not contain missing values in contrast to *OwnDoc*. Furthermore, the standard deviations and maximum values are high compared to the mean, reflecting a dataset with strong variations. Due to this and the practical usefulness of predictions, we resampled all variants based on *CashierData* to a weekly cycle.

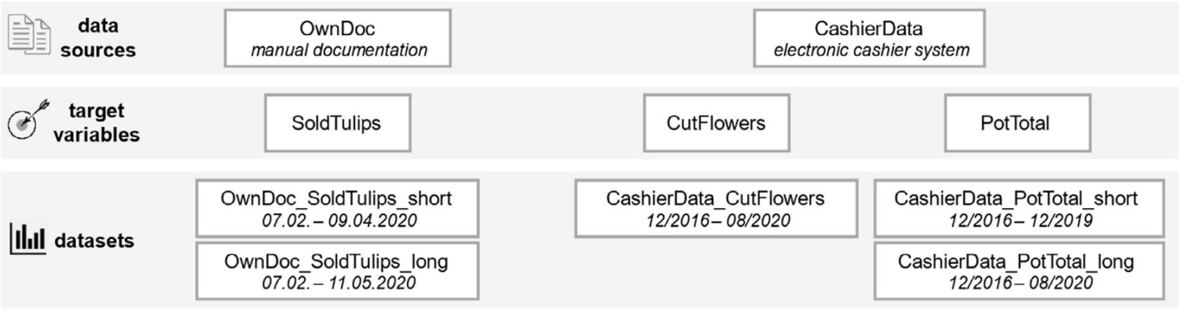

Fig. 1. Overview of all five datasets based on the two data sources OwnDoc and CashierData. OwnDoc_SoldTulips_short was derived due to a 16 day long period with missing values, CashierData_PotTotal_short because of a strong sales increase in 2020.

Table 2
Characteristics for all datasets. Statistics are on a daily basis as well as weekly resampled for CashierData, with the weekly values shown below the daily ones.

Dataset	Period	Samples	Target variable characteristics			
			Missing value ratio	Mean	Standard deviation	Maximum
OwnDoc_SoldTulips_short	07.02.–09.04.2020	63	1/63	158.90	107.47	428.00
OwnDoc_SoldTulips_long	07.02.–11.05.2020	95	17/95	145.36	103.36	428.00
CashierData_CutFlowers	12/2016–08/2020	1359	0/1359	244.00	244.74	2346.45
		195	0/195	1700.47	719.31	4828.85
CashierData_PotTotal_short	12/2016–12/2019	1115	0/1115	160.94	210.41	1605.30
		160	0/160	1121.58	1041.74	5358.70
CashierData_PotTotal_long	12/2016–08/2020	1359	0/1359	189.55	267.04	1926.40
		195	0/195	1321.04	1384.95	7529.25

3.1.2. Feature Engineering

All datasets have been enriched with certain features. To examine whether external factors support the forecasting of horticultural sales, we added daily weather and holiday information. Historical observations for the former were provided by the German Meteorological Service. Regarding holiday information, we considered public as well as school holidays. The included features and their explanations can be found in Table 3.

Further, two categories of additional features were derived: (i) calendric and (ii) statistical features, as summarized in Table 3. Calendric features are date-based properties such as the weekday and typically outstanding selling days, e.g. Valentine's and Mother's Day. Furthermore, we created counters for these special days and public holidays as sales might be higher close to such an event and lower afterwards. Statistical features are, among others, lagged variants of the target variables (both seasonal and non-seasonal). Moreover, we added lagged weather information (mean temperature, precipitation amount and sun duration), because the conditions prior to a date of sale might be influential. Beyond that, we derived rolling statistical values (both seasonal and non-seasonal), such as its mean over a fixed period prior to a sales date. Finally, for daily data, we calculated rolling statistical numbers for a specific weekday, e.g. the mean sales of the last four instances. Eventually, we divided these features into four different featuresets, see Table 3. We included raw features from external sources for all four. Beyond that, *sub2* was focused on calendric features and *sub3* on statistical ones. The *full* featureset contained all features.

3.1.3. Data Imputation and Dimensionality Reduction

Many algorithms cannot handle missing values, except for XGB. Therefore, we applied different strategies for data imputation: Mean, K-Nearest-Neighbors (KNN) and Iterative Imputation. The featuresets *sub3* and *full* introduce additional missing values at the beginning of the dataset because statistical values such as lagged variables cannot be calculated. We imputed these as well and did not drop samples as the amount of lacking seasonal features would lead to a large information loss. There are no missing values for *CutFlowers* and *PotTotal* on *CashierData* and only few in the raw weather data. Moreover, we did not insert new ones using featuresets *sub1* and *sub2* as both do not contain statistical features. Therefore, the effect of different imputation methods seemed negligible, and we only considered Mean Imputation for these setups. In summary, we can distinguish five different feature settings, including the univariate case, which were combined with the data imputation approaches. Fig. 2a provides an overview of all setups we considered.

To examine the effect of dimensionality reduction on the multivariate feature settings, we considered two variants: using the original featureset or running a Principal Component Analysis (PCA) selecting the components which explain at least 95% of the variance (Jolliffe & Cadima, 2016). As shown in Fig. 2a, we did not include PCA for the *sub1* featureset due to the low dimensionality.

3.2. Model Selection and Training

We included all algorithms summarized in Table 1 to show a comparative study of ML and classical forecasting methods. ES and SARIMA were selected as two common univariate techniques. SARIMAX was included as a multivariate classical alternative for a fair comparison with ML methods using additional features. MLR approaches were taken into account as a baseline for regression problems. Furthermore, ML methods able to capture nonlinear relationships were considered. ANNs have proven their suitability for time series predictions (Zhang et al., 1998). As temporal dependencies might be important, RNNs were included and designed with LSTM cells as preceding research suggests (Hewamalage et al., 2021). Beyond that, combined methods were superior in many comparisons (Bojer & Meldgaard, 2021; Petropoulos et al., 2018). Therefore, we included XGB. Eventually, with GPR, a nonparametric Bayesian method with its inherent ability to model uncertainty was selected. Furthermore, we applied normalization and standardization methods such as the Yeo–Johnson power transformation (Yeo & Johnson, 2000) or logarithmic scaling as certain methods are sensitive to input distributions. A summary of all hyperparameters and transformations for all algorithms is shown in Appendix B (Tables B.1 to B.8). Hence, we randomly sampled parameter combinations for model and preprocessing configuration.

We employed time series cross-validation with a regular model refit in order to simulate a potential scenario for the productive operation

Table 3
Overview of features and structure of the featuresets. Features are based on external sources or derived based on others. With regard to raw features from external sources, holiday and weather information was added. Furthermore, calendric and statistical features were derived. Based on that, four multivariate featuresets were designed. Their structure can be seen in the last four columns.

Source	Category	Feature	Explanation	Featureset			
				sub1	sub2	sub3	full
Features from external sources	Raw features	Public holiday	Name of public holiday	×	×	×	×
		School holiday	Name of school holiday	×	×	×	×
		Mean temperature	Daily mean air temperature	×	×	×	×
		Mean humidity	Daily mean relative humidity	×	×	×	×
		Precipitation amount	Daily mean and total precipitation amount	×	×	×	×
		Precipitation flag	Ratio of hours with precipitation and flag for full day	×	×	×	×
		Sun duration	Daily mean and total sun duration	×	×	×	×
Derived features	Calendric features	Date based features	Day of the month, weekday, month, quarter		×		×
		Working day	Flag showing if the day is a working day		×		×
		Special days	Valentine's and Mother's Day added to public holidays		×		×
		Public holiday counter	Counter for a public holiday starting seven days before and ending three days after with separate counters for Easter as well as Valentine's and Mother's Day		×		×
	Statistical features	Lagged variables	Lagged versions of target variables and weather features mean_temp, total_prec_height_mm and total_sun_dur_h			×	×
		Seasonal lagged variables	Seasonal lagged versions of same variables as above			×	×
		Rolling statistics	Rolling mean, median and maximum within a window for same variables as above			×	×
		Seasonal rolling statistics	Seasonal rolling mean, median and maximum within a window for target variables			×	×
		Rolling weekday statistics	Rolling mean, median and maximum within a window calculated for each weekday on target variables			×	×

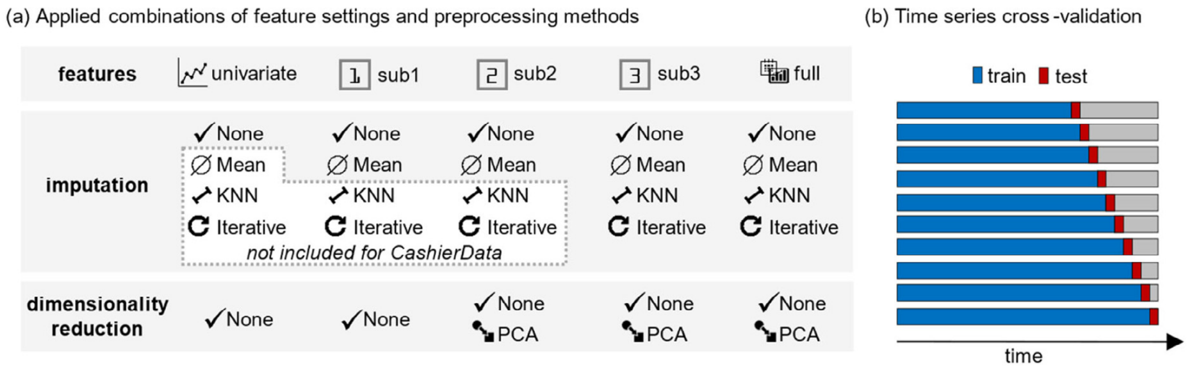
(a) Applied combinations of feature settings and preprocessing methods
features
univariate
sub1
sub2
sub3
full
imputation
✓None
∅ Mean
KNN
Iterative
not included for CashierData
dimensionality reduction
✓None
PCA
(b) Time series cross-validation
train
test
time

Fig. 2. (a) Overview of the applied combinations of the five feature settings and the preprocessing methods. The overview shows the preprocessing methods combined for each of the feature settings. For CashierData, certain imputation strategies are excluded for settings with no or few missing values. PCA was considered in multivariate cases and not performed on featureset sub1 due to its low dimensionality. (b) Time series cross-validation. Visualization of the evaluation on a rolling forecasting origin, which we employ to simulate a productive operation of a forecasting system with a continuous data update.

of a forecasting tool with a continuous update of company-specific sales data and a potentially changing data distribution in mind. Fig. 2b shows a visualization of this approach. First, we separated a training set covering 80% of the whole dataset to select the best working hyperparameter combination. Then, we simulated an online scenario using the remaining data. Whenever a new sample was available, we forecasted the next target value and refitted the model parameters keeping the already optimized hyperparameters fixed. A model configuration was evaluated by the average performance across the whole test set according to the evaluation metrics described in Section 3.3 (Hyndman & Athanasopoulos, 2018). For the best working solutions, we ran an in-depth optimization selecting more parameter combinations on a denser grid.

Beyond that, we compared the runtime needed by each algorithm. For this purpose, we selected *OwnDoc_SoldTulips_long* with *sub1* as well as *CashierData_CutFlowers* with *full* to include opposing examples regarding size. Furthermore, we randomly sampled 100 parameter combinations for every algorithm. Then, we ran the whole optimization for one training and prediction loop and calculated the average computation time. All runs for these experiments were executed on the same machine with four 4.0 GHz *Intel i7-6700K* CPUs, 62 GB memory and two *Nvidia GeForce GTX 1080 Ti* GPUs used for ANN and LSTM optimization.

The source code is written in Python 3.8 and published on GitHub: https://github.com/grimmlab/HorticulturalSalesPredictions. A list of all used packages and their versions can be found in this repository, including libraries such as GPflow (Matthews et al., 2017), Matplotlib (Hunter, 2007), NumPy (Harris et al., 2020), pandas (McKinney, 2010), PyTorch (Paszke et al., 2019), scikit-learn (Pedregosa et al., 2011), SciPy (Virtanen et al., 2020), seaborn (Waskom, 2021) and statsmodels (Seabold & Perktold, 2010). We used Docker to ensure a standardized working environment on different servers. A Dockerfile for its setup is included in the repository.

3.3. Evaluation Metrics and Baselines

For evaluation, we used the scale-dependent metric Root Mean Squared Error (RMSE) as well as the relative measures Mean Absolute Percentage Error (MAPE) and Symmetric Mean Absolute Percentage Error (sMAPE). With the forecast value $\hat{y}_i$, the true value y_i and the length of the evaluated set n, they are defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}$$

$$sMAPE = \frac{100\%}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{\left(|y_i| + |\hat{y}_i|\right)/2}$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

For all three evaluation measures, a lower value reflects a better performance. In order to prevent division by zero when using MAPE, we added a constant (0.1) to the denominator. MAPE values can get large, if sales numbers are close to zero (common in datasets with daily observations). This circumstance is less severe for sMAPE, because it is divided by the mean of y_i and $\hat{y}_i$, but is still present as the predicted value might also be close to zero in these cases. Furthermore, due to the quadratic term, RMSE is sensitive to outliers. On the basis of these weaknesses and the lack of a universal evaluation metric for forecasting, it is common to assess the performance compared to baseline methods (Hyndman & Koehler, 2006). To that end, we used the ones described in Table 4.

Table 4
Baseline methods with length of training set T and seasonal period m.

Historical Average (HA)	$\hat{y}_t = \frac{1}{t-1}\sum_{i=1}^{t-1} y_i$
Random Walk (RW)	$\hat{y}_t = y_{t-1}$
Seasonal Random Walk (seasRW)	$\hat{y}_t = y_{t-m}$

HA predicts the mean of all known values. RW predictions are equal to the last value, whereas seasRW uses the known observations of the last season (Hyndman & Athanasopoulos, 2018).

4. Results and Discussion

In the next section, we give an overview of our results followed by a more detailed analysis of the best performing algorithms. Afterwards, the influence of different features as well as the runtime are analyzed. Finally, a discussion of the results is provided.

4.1. Results Overview

For a full analysis of our experiments, we generated an overview of the best results for all five datasets with respect to the three evaluation metrics (15 comparisons in total) for all baselines, algorithms and featuresets, see Appendix C and Supplementary 1. Our results show that a baseline (RW, seasRW and HA) was never best overall. However, MLR approaches (LassoReg, RidgeReg, ElasticNet, BayesRidge and ARD) were inferior in several cases on both *OwnDoc* variants and *CashierData_PotTotal_long*. If we exclude MLR algorithms, only in three out of the 15 comparisons, one of the baselines performed better than at least one ML method, with two of these cases occurring on the smallest dataset (*OwnDoc_SoldTulips_short*).

Following this comparison with baseline methods, we selected the best performances of each algorithm, independent of featuresets and preprocessing methods. The results are summarized in Tables D.1 and D.2, see Appendix D.1. Fig. 3a–e visualize them for all datasets (rows) and evaluation metrics (columns). In all comparisons, a ML-based method performed best. The results of the ML and classical approaches were rather comparable for *OwnDoc*, whereas the advantage of ML techniques was larger for *CashierData*, probably due to the size of the datasets. In total, XGB was the leading method in 13 out of 15 comparisons, outperformed by LSTM two times for *CashierData_PotTotal_short*. Regarding univariate methods, SARIMA was competitive for both *OwnDoc* variants and the SARS-CoV-2-influenced *CashierData_PotTotal_long* dataset. Its multivariate extension, SARIMAX, achieved results close to the best ones for both *OwnDoc* versions. Furthermore, GPR and LSTM delivered competitive results in several cases, especially on all *CashierData* variants. All MLR methods as well as ES fell behind in most conditions, and ANN only kept up in a few instances. However, in summary, we can conclude a superiority of ML-based approaches.

4.2. Best Performing Algorithms

We subsequently focused on the best performing algorithms, so mainly ML-based methods. As described in Section 3.2, we conducted a further optimization of the best performers on a denser grid of hyperparameters. The results of these in-depth runs and a comparison with the results prior to them can be found in Tables 5 and 6. Overall, we again observed a superiority of ML-based approaches. Improvements based on the in-depth runs were rather small or mediocre with only one change regarding the overall ranking: XGB outperformed LSTM on RMSE for *CashierData_PotTotal_short.* In summary, XGB's predominance was extended, leading in 14 out of 15 comparisons after the in-depth optimization.

With regard to *OwnDoc* variants presented in Table 5, we observed that XGB was the clear winner. However, it was closely followed by SARIMAX for the shorter dataset. Besides that, GPR and LSTM were competitive for sMAPE. GPR was furthermore close for MAPE with two second best performances.

Table 5
Top results for OwnDoc after in-depth optimization. A lower value reflects a better performance. Results prior to in-depth optimization are given in brackets in case of an improvement. The best results overall are printed in bold. In-depth optimization was not done for SARIMA on OwnDoc_SoldTulips_short.

Algorithm	*OwnDoc_SoldTulips_short*			*OwnDoc_SoldTulips_long*		
	RMSE	SMAPE	MAPE	RMSE	SMAPE	MAPE
SARIMA	–	–	–	48.70 (52.02)	55.83	2366.01 (3675.31)
SARIMAX	58.06	52.58	85.75	48.45 (50.35)	52.37	203.34 (231.52)
LSTM	73.66	52.49	707.61	56.06	50.04	2884.77
XGB	**57.11**	**50.89**	**34.21** (35.98)	**43.48** (43.52)	**48.65**	**52.87** (54.22)
GPR	68.14	52.27	67.99	56.05	52.43	65.84

Table 6 shows a similar overview for all *CashierData* variants. For *CashierData_CutFlowers*, XGB was the top algorithm in all comparisons. Besides that, the ensemble learner improved its performance on RMSE for *CashierData_PotTotal_short* and consequently outperformed the previous best algorithm LSTM. However, the results of the LSTM were comparable for this dataset. Regarding *CashierData_PotTotal_long*, XGB performed best and GPR delivered the second-best results for RMSE and sMAPE.

Beyond that, we show example plots of the best performing algorithms in Fig. 4. Each of them displays the ground truth accompanied by the predictions of two of the top performing models. We observed that several predictions are close to the ground truth, but there are also problematic regions, e.g. the end of the test period on both *OwnDoc* datasets. Partially, this is due to its limitation to only one season. Because of this, we were not able to capture the high sales on May 10 at the end of *OwnDoc_SoldTulips_long*. All algorithms predicted low sales as it was a Sunday, but actually they were high because it was Mother's Day. Furthermore, the demand peaks of *CashierData_PotTotal_long* during the SARS-CoV-2 pandemic were hard

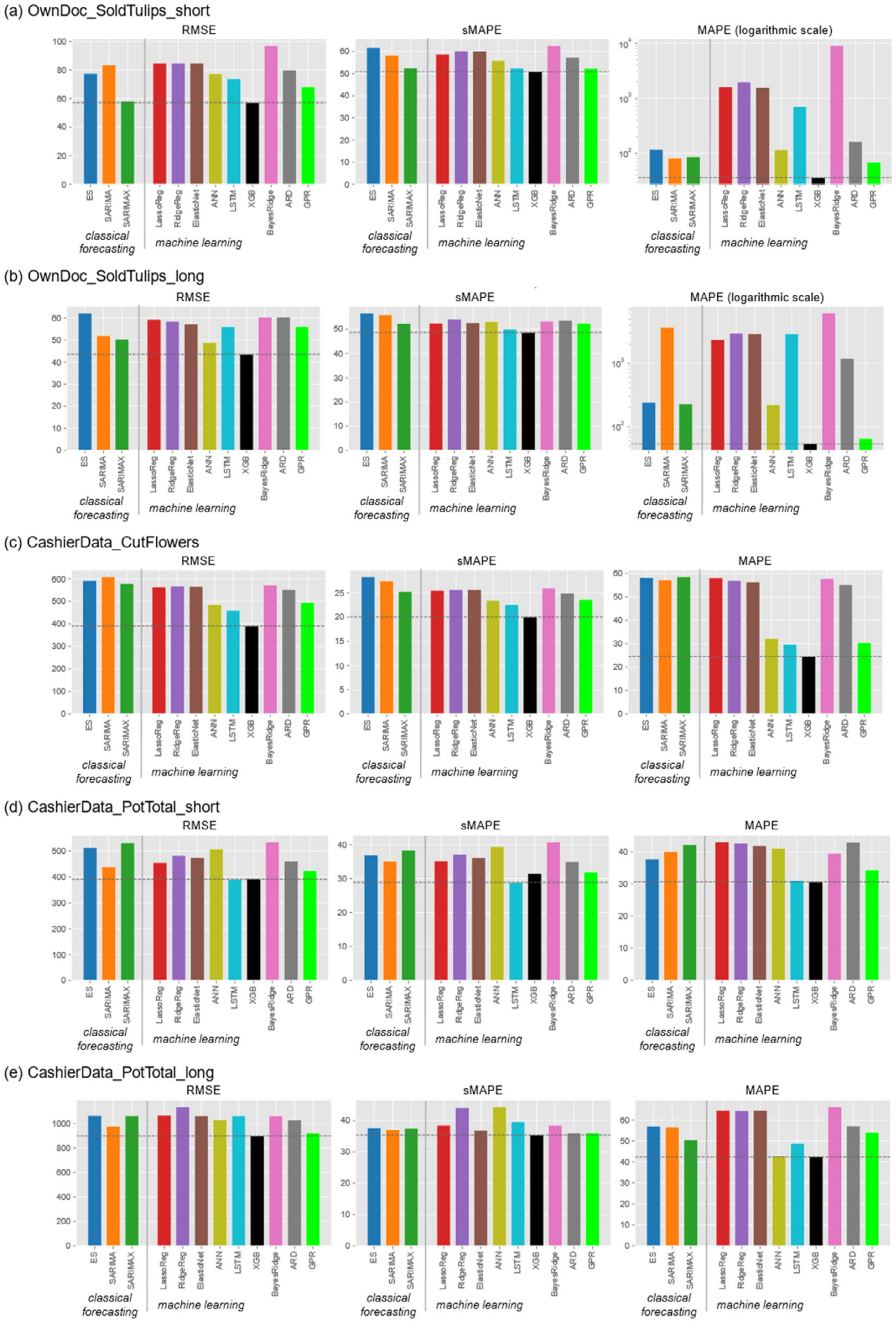

Fig. 3. Top results for each algorithm and dataset (rows) with respect to all evaluation metrics (columns). A lower value reflects a better performance. Classical forecasting and machine learning approaches are separated by a small horizontal offset and a vertical line. The best outcome in each plot is marked by a horizontal dashed line. For the exact values, see Tables D.1 and D.2 in Appendix D.1. MAPE values for both OwnDoc variants are plotted on a logarithmic scale as the results differ in a range from two- to four-digit numbers.

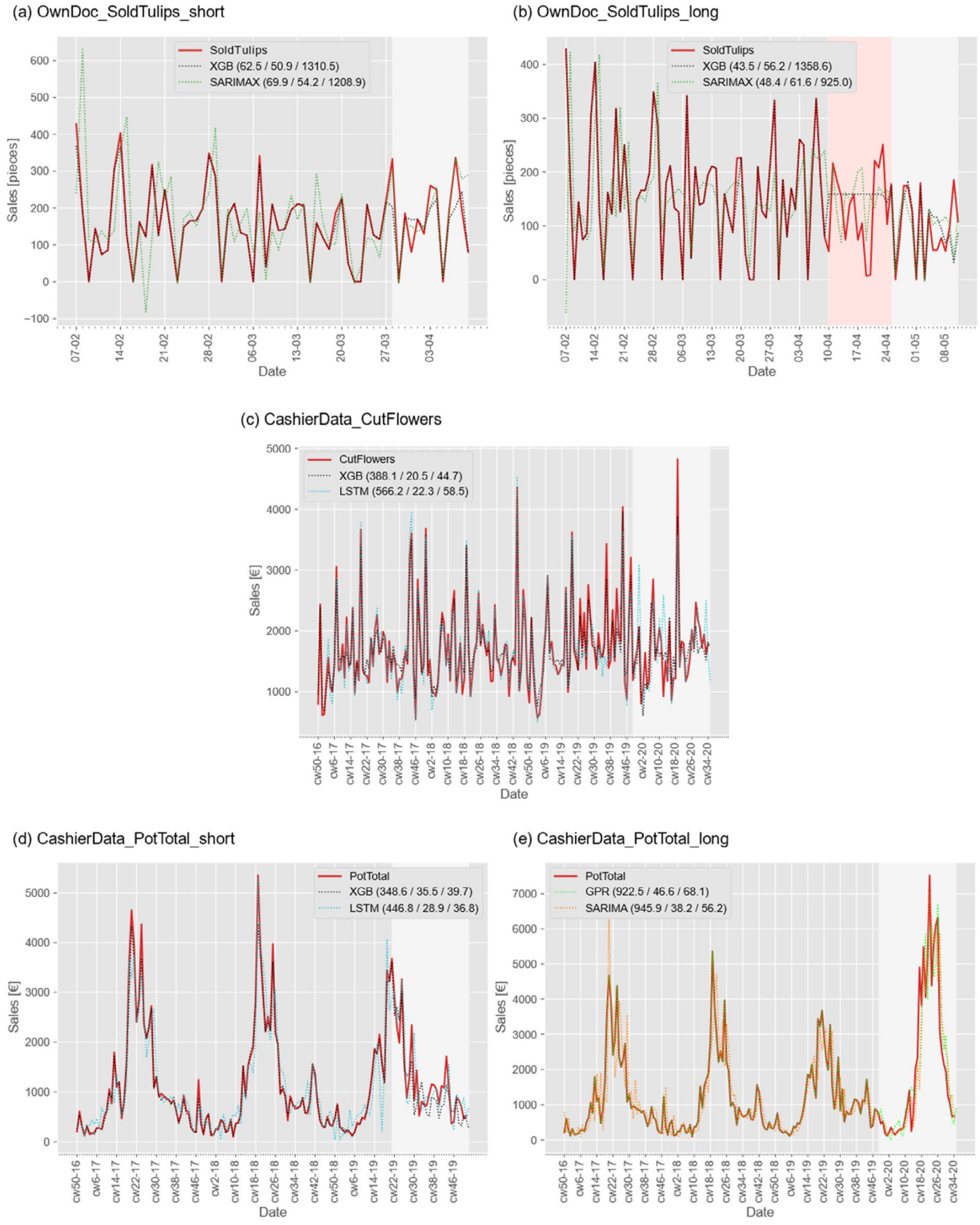

Fig. 4. Example plots of best performing algorithms. Predictions show selected examples of two of the top performers accompanied by the ground truth. The test periods are highlighted in light gray. Missing values for OwnDoc_SoldTulips_long were iteratively imputed and are marked with a red background. In the legends, the evaluation measures RMSE/sMAPE/MAPE are given in brackets. A lower value reflects a better performance. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

to forecast. Nevertheless, overall the plots confirm the promising results of the evaluation measures as most of the curves show a prediction close to the target.

4.3. Feature Analysis

We furthermore analyzed the performance influence of the different featuresets, see Table 3 regarding their structure. For that purpose, we focused on the top performing multivariate approaches SARIMAX,

Table 6
Top results for CashierData after in-depth optimization. A lower value reflects a better performance. Results prior to in-depth optimization are given in brackets in case of an improvement. The best results overall are printed in bold. In-depth optimization was not done for SARIMA on CashierData_CutFlowers and for ANN on the other two variants.

Algorithm	*CashierData_CutFlowers*			*CashierData_PotTotal_short*			*CashierData_PotTotal_long*		
	RMSE	SMAPE	MAPE	RMSE	SMAPE	MAPE	RMSE	SMAPE	MAPE
SARIMA	–	–	–	439.03	34.08 (35.05)	40.13	945.91 (979.10)	36.84 (37.03)	46.41 (56.59)
ANN	475.66 (485.60)	23.31 (23.43)	26.61 (32.19)	–	–	–	–	–	–
LSTM	460.61	21.78 (22.57)	29.65	390.98	**28.85**	29.71 (31.11)	1037.11 (1064.14)	39.65 (38.96)	47.70
XGB	**388.09** (390.96)	**19.71** (20.03)	**23.98** (24.52)	**348.60** (392.31)	29.86 (31.50)	**28.59** (30.70)	**892.04** (898.56)	**34.61** (35.35)	**36.69** (42.45)
GPR	493.58	23.61	30.45	421.46 (424.50)	31.00 (31.88)	32.91 (34.42)	922.46	36.03	52.35 (54.20)

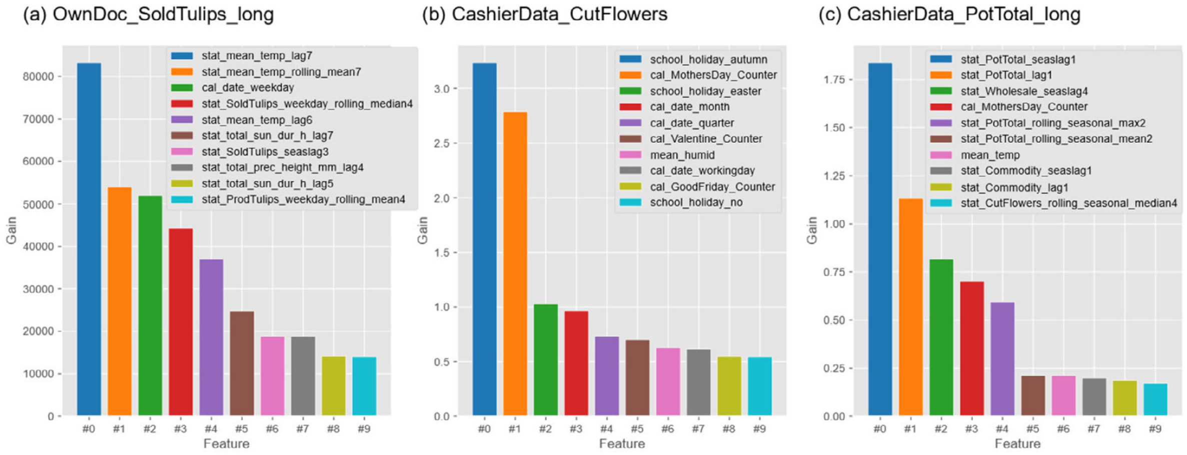

Fig. 5. Ten most important features of XGB according to the gain on selected examples of top performing configurations. The gain of a feature reflects the average improvement across all its usages. A higher value corresponds to a greater importance. (a) OwnDoc_SoldTulips_long with full featureset, (b) CashierData_CutFlowers with sub2 featureset, (c) CashierData_PotTotal_long with full featureset.

LSTM, XGB and GPR. To ensure a fair comparison, we used the predictions prior to the in-depth optimization, as this step was not done for all configurations. We compared the best results achieved with each featureset with respect to all evaluation metrics. An overview of the results (Tables D.3 and D.4) as well as their visualization (Fig. D.1) can be found in Appendix D.2. Based on this, we counted the number of times a featureset led to the best result for each algorithm and dataset, subsequently called a win. Table D.5 in Appendix D.2 summarizes this analysis. Counting all datasets, the win percentages were divided as follows: *sub1* (raw weather and holiday features) 11.7%, *sub2* (focus on calendric features) 41.7%, *sub3* (focus on statistical features) 21.7% and *full* (all features) 25.0%. However, the results varied depending on the dataset and algorithm. We discovered that the *full* featureset was superior for *OwnDoc* with the best result in 12 out of 24 comparisons. For GPR, *sub2*, *sub3* and *full* were on a par, whereas *full* was superior for the other three algorithms. Regarding *CashierData*, *sub2* led to the best result in 20 out of 36 cases (about 55%), followed by *sub3* with seven wins, *sub1* with six and *full* with three. If we exclude the *CashierData_PotTotal_long* dataset influenced by the SARS-CoV-2 pandemic, the advantage of *sub2* was even clearer with 15 wins in 24 comparisons (about 62%). XGB worked best with *sub3*, but *sub2* did so for the other three methods.

Beyond determining the top performing featuresets, we analyzed the feature importance of XGB as this was the best predictor. We present one example for each target variable in Fig. 5. They indicate both a benefit due to external factors such as weather and holiday data as well as derived features such as statistical values. In Fig. 5a, we can see that statistical quantities, both of the external factor weather and past sales numbers, were relevant. The best RMSE for *CashierData_CutFlowers* was achieved with the featureset *sub2* (focus on calendric features), for which we show the feature importance in Fig. 5b. We found out that calendric features were ranked high, e.g. the counters for the typical high sales days of Mother's and Valentine's Day. Furthermore, the school autumn holidays seemed important as they correlate with All Saint's Day on which religious traditions boost flower sales. Statistical features were the most frequent category for *CashierData_PotTotal_long*. As shown in Fig. 5c, the leading ones were the lagged sales numbers of the previous week and of the same one in the preceding season. Furthermore, delayed information of other product groups improved forecasting. With the mean temperature, a weather property was part of the leaderboard.

4.4. Runtime Comparison

Besides the prediction performance, the computational resource consumption of an algorithm is an important characteristic, especially with a regular model refit due to a changing data distribution in mind. Therefore, we compared the runtimes as described at the end of Section 3.2. The results are visualized in Fig. 6. In summary, the top performer XGB was in both cases in the middle range of the ranking and closer to the efficient methods, which is a further advantage. As we observe in Fig. 6a, ANN and LSTM required the longest runtime for *OwnDoc_SoldTulips_long*, followed by SARIMAX and GPR with about 80% less runtime than LSTM. The MLR approaches showed the lowest values, whereas XGB was comparable to ES and SARIMA. SARIMAX had by far the longest runtime for *CashierData_CutFlowers*, as shown in Fig. 6b, followed by its univariate version and LSTM. ES and the MLR approaches, except ARD, showed low runtimes. The results for GPR and ANN were comparable and XGB was closer to the efficient methods.

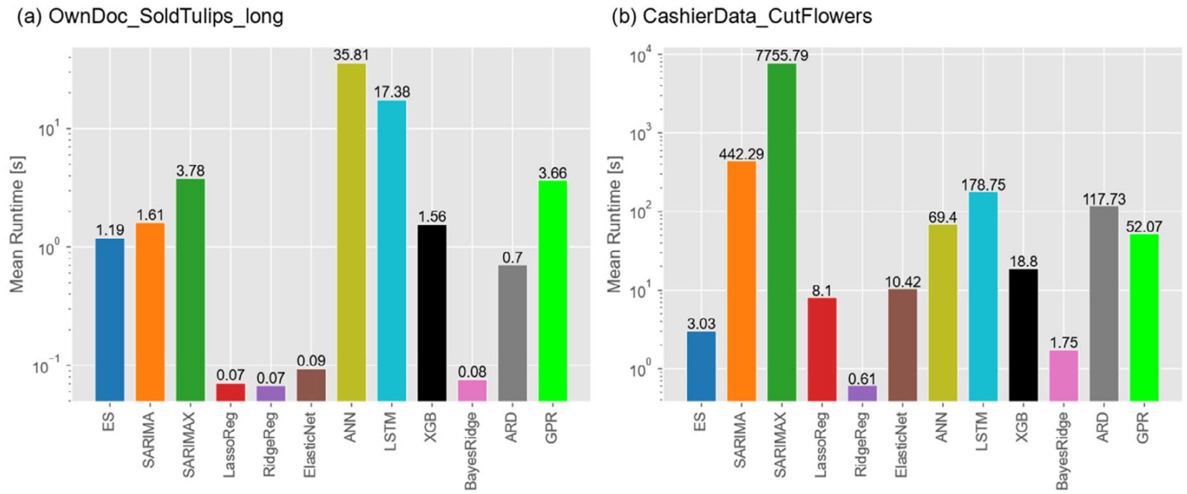

Fig. 6. Mean runtime of one optimization based on 100 random hyperparameter settings for every algorithm. Plots are shown on a logarithmic scale with the exact runtime in seconds above each bar. (a) OwnDoc_SoldTulips_long with sub1 featureset. (b) CashierData_CutFlowers with full featureset.

4.5. Discussion

In this paper, we show a first-time comparison of ML-based and classical forecasting approaches for horticultural sales predictions. Most of the methods outperformed simple baseline comparison partners. Further, our results show a superiority of ML-based methods compared to classical forecasting ones with the former delivering the lowest error in each of the 15 comparisons. The ensemble learner XGB performed best in 14 cases. We assume that its combination of multiple weak learners is beneficial for capturing different effects, which improves the quality and robustness of the final result. LSTM was first in one out of the 15 cases. In summary, LSTM together with GPR were most competitive compared to XGB. The performance of ML algorithms is usually highly influenced by the amount of available data. *CashierData* contains 1359 daily observations, which is about 14 times more than the *OwnDoc* dataset. Nevertheless, XGB also delivered the lowest errors for *OwnDoc*, but the advantage tended to be smaller. With only a few exceptions, the advantage of ML techniques increased on the larger *CashierData* datasets. We assume that, additionally to the dataset size, multivariate patterns and nonlinear relationships in the data, which ML models are rather able to capture, are more pronounced in these datasets that contain several seasons.

One of the research questions of this paper is whether the applied methods cope with the sales increase during the SARS-CoV-2 crisis in *CashierData_PotTotal_long*. We observed that the relative measures sMAPE and MAPE were larger compared to the other two *CashierData* datasets. Beyond that, Fig. 4 suggests that the algorithms were limited in capturing this phenomenon. However, with XGB, an ML approach also performed best for this special case. We assume that the regular refit of the models is beneficial, as recent samples are taken into account. Nevertheless, the samples which comprise this sudden change are only a few compared to the whole train set. Therefore, the algorithms tended to follow the majority of the training instances, which reflect the sales prior to the SARS-CoV-2 pandemic. The comparably good performance of SARIMA might be caused by external factors that correspond to a different data distribution in the training set and consequently could even impede forecasting. But in general, our results indicate that external factors such as weather information as well as derived features, e.g. statistical measures, lead to an improved forecasting performance as all superior methods were multivariate. This additional info is especially useful for ML-based methods. However, regarding the influence of different features, see Section 4.3, a configuration being superior for all algorithms and datasets could not be determined.

Beyond that, the runtime comparison in Section 4.4 shows that SARIMAX – the most competitive classical approach – has poor scalability on larger data- and featuresets. By contrast, XGB was efficient in both cases. This is an additional advantage of the ML-based method XGB, especially when considering regular refits of the model.

There are several publications on competitions including classical forecasting and ML techniques, based on which a general superiority of one of them cannot be concluded (Bojer & Meldgaard, 2021; Makridakis et al., 2020, 2021) . Nevertheless, our conclusion that ML performs better agrees with recent publications on food and tourism demand forecasting (Huber & Stuckenschmidt, 2020; Jiao & Chen, 2019). Several papers have shown a superiority of combined approaches (Bojer & Meldgaard, 2021; Makridakis et al., 2020; Petropoulos et al., 2018). Hence, our results with XGB performing best confirms previous work for the new forecasting domain of horticultural sales. Beyond that, GPR produced competitive results. For this reason and its advantage of providing prediction uncertainties, it might be interesting for horticultural sales prediction and similar domains.

Besides this, we made certain simplifications for our analysis. First, we used historical weather data as external information, but when predicting the demand in production-ready systems, only short-time weather forecasts are available. However, since historical weather forecasts were not retrievable for the whole period, this is a reasonable approximation. Second, the datasets provide sales numbers, which do not fully reflect the potential demand as out-of-stock situations (as an indicator for a higher demand of customers) were not documented. Third, we used holiday and weather data as external information, but there could be additional features which might improve prediction performances (e.g. promotional and communication activities of the company).

This study is the first analysis of horticultural demand forecasting based on typical retail data. So far, it is not obvious if our results generalize, e.g. for companies of the whole value chain. Nevertheless, we provide first insights, which need to be further evaluated before drawing more general conclusions. In accordance with literature, an ensemble method led to the best result. This suggests that combined approaches might also be superior for horticultural demand forecasting. Therefore, the integration of further combined methods, e.g. those including ML-based as well as classical forecasting techniques, is interesting for further research. Beyond that, we employed a computationally expensive approach with a model refit when new samples arrive. This seems reasonable for a first-time adoption with potential changes in data and a productive operation of a forecasting tool with a continuous update of company-specific sales data in mind. Nevertheless, the necessity of model refits should be further examined with multi-step-ahead forecasts and a periodical retraining. The focus of this paper is on horticultural retail sales. It is interesting whether including data of businesses along the whole value chain, e.g. from wholesalers or producers, might improve forecasts. Furthermore, as horticultural companies are typically small and medium-sized, datasets are heterogeneous and rather small. Thus, novel approaches that combine data of diverse sources and consequently benefit from the information derived from several smaller datasets are an interesting research direction. This might also be beneficial for other domains such as e.g. agricultural producers or small and medium-sized food processing companies. The influence of the SARS-CoV-2 pandemic resulted in a sudden rise in demand for potted plants. Most of the algorithms were not able to deal with this change of the data distribution. Therefore, research based on methods enabling them to cope with such phenomena such as EVARS-GPR could be useful for various applications (Grande et al., 2017; Haselbeck & Grimm, 2021; Liu et al., 2020; Ni et al., 2012).

5. Conclusion

In this paper, we presented a first-time comparative study for horticultural sales predictions with nine state-of-the-art ML and three classical methods. Our study was based on typical horticultural retail data with distinct characteristics, e.g. regarding size and seasonality. We employed a forecasting setup with a regular refit of the model parameters in order to simulate a productive operation of a forecasting system with a continuous data update and a potentially changing data distribution. Our findings show a superiority of ML approaches, especially of the ensemble learner XGB. This advantage increased for larger datasets containing multiple seasons. Beyond that, we experienced a performance increase by including additional features, such as weather or holiday data. Finally, we showed that the top performer XGB is computationally efficient. Based on these first results, a plethora of new research questions arise with a lot of potential for future research. Especially the transfer and verification of these results within the same sector are of general interest and might allow the generalization of our conclusions.

List of abbreviations

ANN — Artificial Neural Network
AR — Autoregressive part
ARD — Automatic Relevance Determination
ARIMA — Autoregressive Integrated Moving Average
ES — Exponential Smoothing
GPR — Gaussian Process Regression
HA — Historical Average
KNN — K-Nearest-Neighbors
LSTM — Long Short-Term Memory
MA — Moving average part
MAPE — Mean Absolute Percentage Error
ML — Machine learning
MLR — Multiple Linear Regression
PCA — Principal Component Analysis
ReLU — Rectified Linear
RMSE — Root Mean Squared Error
RNN — Recurrent Neural Networks
RW — Random Walk
SARIMA — Seasonal Autoregressive Integrated Moving Average
SARIMAX — Seasonal Autoregressive Integrated Moving Average with external factors
seasRW — Seasonal Random Walk
sMAPE — Symmetric Mean Absolute Percentage Error
XGB — Extreme Gradient Boosting (XGBoost)

CRediT authorship contribution statement

Florian Haselbeck: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Visualization. **Jennifer Killinger:** Validation, Investigation. **Klaus Menrad:** Writing – review & editing, Project administration, Funding acquisition. **Thomas Hannus:** Writing – review & editing, Project administration, Funding acquisition. **Dominik G. Grimm:** Conceptualization, Methodology, Investigation, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All code and data is publicly available on GitHub: https://github.com/grimmlab/HorticulturalSalesPredictions.

Acknowledgments

We would like to thank the project funder for supporting our research as well as all project partners. In addition, the authors gratefully acknowledge the compute resources provided by the Leibniz Supercomputing Centre (www.lrz.de). Furthermore, we thank the German Meteorological Service for providing historical weather data.

Funding

The project is supported by funds of the Federal Ministry of Food and Agriculture (BMEL), Germany based on a decision of the Parliament of the Federal Republic of Germany via the Federal Office for Agriculture and Food (BLE) under the innovation support program [grant number 2818504A18].

Appendix A. Time Series Forecasting Methods

A.1. Classical Forecasting Methods

Classical forecasting methods use chronologically ordered time series data and try to predict future sequences, usually by projecting statistical information recovered from historical data (Hyndman & Athanasopoulos, 2018). In Table 1 of the main document, we provide an overview of the approaches discussed in this paper. Methods can be divided into univariate and multivariate approaches, with the latter one using external information in addition to the time series itself. Two common univariate methods are Exponential Smoothing (ES) and Autoregressive Integrated Moving Average (ARIMA) (Box et al., 2016; Holt, 1957; Winters, 1960). Gardner (2006) gave a detailed overview about the theoretical background of ES and its application (Gardner, 2006). In its simple form, ES is a weighted sum of past observations of a time series with weights decaying exponentially. Therefore, recent values have more influence on the output. More elaborate versions include linear or damped trend as well as seasonal components, which are characteristics that many real-world time series possess. Both components can be formulated in an additive or multiplicative way, depending on the characteristics of the time series.

In contrast, modeling autocorrelations – the correlation between a series and a lagged version of itself – is the key idea of ARIMA. Thereby, autoregressive (AR) and moving average (MA) parts are calculated. The first is a linear combination based on the time series itself, whereas the latter is a weighted sum of past model errors. One of the algorithm's assumptions is stationarity of the input, so trend and seasonality should be removed. One way to achieve this property is differencing prior to optimizing the AR and MA parameters. This step is called the "integrated" part of ARIMA. The determining parameters are thus the degree of differencing and the lags considered when constructing the

AR and MA elements. A common extension, SARIMA, involves seasonal parts. Their formulation is similar to the non-seasonal ones, but uses the observations of previous seasons. Therefore, besides the arguments of the seasonal differencing, AR and MA components, the length of such a period needs to be defined (Hyndman & Athanasopoulos, 2018; Shumway & Stoffer, 2000).

SARIMA is univariate by definition. Consequently, external factors potentially providing useful information are not included. Hence, a multivariate extension called SARIMAX exists. One way to implement it is combining multivariate regression depending on external factors and a SARIMA model, which is driven by the errors of the first one. Usually, both elements are optimized jointly delivering a final output after summation (Arunraj et al., 2016).

A.2. Machine Learning Methods

Sales forecasting can also be formulated as a regression task for ML methods. These data-driven approaches are able to discover complex relationships, which are more likely to be present in multivariate datasets. Therefore, as stated in Table 1 of the main document, we only consider ML approaches in such setups.

For regression tasks, Multiple Linear Regression (MLR) is often used as a baseline. MLR models try to predict a target variable by building a weighted sum of several features and an intercept. Typically, the sum of squared errors between the observed and predicted value is optimized with Ordinary Least Squares during training. With an increasing number of features, this might lead to overfitting. To prevent this, we can use regularized regression approaches by adding a penalty term to the loss function. The goal of this is to learn models which generalize better to unknown data. Common approaches are Ridge, Lasso and Elastic Net Regression (Hoerl & Kennard, 1970; Santosa & Symes, 1986; Tibshirani, 1996; Zou & Hastie, 2005). The first one – also called L2-regularization – introduces a quadratic term penalizing the weights' size. This usually leads to a lower influence of correlated features by distributing the weight among them. However, Ridge Regression does not push the weights of irrelevant features exactly to zero, but reduces their influence. Lasso Regression uses L1-norm (the absolute value of the weights instead of the quadratic ones) as penalty term. This leads to the effect of forcing the weights of unimportant features to zero, which can be seen as an automatic feature selection process (James et al., 2017). Elastic Net Regression combines both L1- and L2-regularization with an additional hyperparameter controlling the influence of each one. Consequently, the variable selection effect of Lasso as well as the grouping mechanism for correlated features of Ridge are included (Zou & Hastie, 2005).

Moreover, we used ML methods because they can cover nonlinearities. A common approach in time series prediction, as shown by Zhang et al. (1998), are Artificial Neural Networks (ANN) (Rosenblatt, 1958; Zhang et al., 1998). Their advantages, such as the ability to model nonlinear and complex relationships in a data-driven way, make them an appropriate alternative. As we work with time series data, it seems obvious to use Recurrent Neural Networks (RNN), which are able to capture temporal dependencies as they – intuitively speaking – possess a memory of previous states. Hewamalage et al. presented an overview of RNNs in time series forecasting (Hewamalage et al., 2021). They concluded that RNNs are competitive and that Long Short-Term Memory Networks (LSTM) show the best performance. Due to their memory and gating mechanisms, LSTMs can capture long-term dependencies and prevent the vanishing gradient problem (Hochreiter & Schmidhuber, 1997). Based on these scientific findings, it is common to design recurrent networks for time series forecasting with these cells. However, the flexibility of ANNs and RNNs might lead to overfitting. To prevent this, regularization techniques can be employed. A common regularization method is dropout. With this approach, a specified share of neurons are randomly ignored during each training iteration, leaving a reduced network (Srivastava et al., 2014). Another regularization procedure is early stopping. Thereby, the loss on a validation set (independent data) is monitored during training and if there is no improvement for a certain period, the optimization gets terminated (Bishop, 2009).

Many publications and forecasting competitions indicate a superiority of combined methods, e.g. achieved via ensemble learners and mechanisms such as Bagging or Boosting (Bojer & Meldgaard, 2021; Petropoulos et al., 2018). XGB is an ensemble technique that uses gradient boosted regression trees as weak learners. With this approach, decision trees are sequentially added in a greedy manner based on the gradient of the loss function in order to correct the errors made by the current ensemble. This is in contrast to algorithms using Bagging, e.g. Random Forests, which train several weak learners in parallel on bootstrapped samples. Furthermore, Boosting algorithms showed a good performance in forecasting competitions (Bojer & Meldgaard, 2021). XGB is a sparse-aware and computationally efficient implementation of this technique (Chen & Guestrin, 2016).

Besides the frequentist approach, for which we view probability as the relative frequency of an event in an experiment, we can interpret probability in a Bayesian perspective. Thereby, we define a prior belief before considering evidence as we assume that certain hypotheses are more plausible than others. Then, we take evidence into account, e.g. observations in an experiment, leading to a posterior belief. This is regularized by the prior, as we usually do not completely reject our previous belief. For ML methods, we can define the prior as a probability distribution over the model parameters and calculate their posterior based on observations in a dataset. Therefore, in contrast to a frequentist viewpoint with fixed parameter values, probability distributions over the model parameters reflecting their uncertainty are estimated (Bishop, 2009).

MLR can be defined in a Bayesian perspective as follows: $p(y|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}(y|\boldsymbol{x}^T\boldsymbol{w}, \sigma^2)$, where the target variable y follows a Gaussian distribution with the mean $\boldsymbol{x}^T\boldsymbol{w}$ (determined by the predictors $\boldsymbol{x}$ and the weights $\boldsymbol{w}$) and the variance σ^2. If the weights of the model are constrained to a zero-mean Gaussian prior, the solution is equivalent to Ridge Regression and therefore called Bayesian Ridge Regression (BayesRidge). Automatic Relevance Determination (ARD), also known as Sparse Bayesian Learning or Relevance Vector Machine, is related to it, but introduces an individual variance for each weight. If the variance of a feature is low, the corresponding weight is likely to be close to zero and can be pruned leading to sparser solutions (Bishop, 2009; James et al., 2017; Tipping, 2001).

With regard to the practical use of demand forecasts, the uncertainty of a prediction value seems profitable. Providing those by its definition is a main advantage of the nonparametric Bayesian method Gaussian Process Regression (GPR) (Williams & Rasmussen, 1996). To explain it, it makes sense to go back to the linear model. This is defined as

$$f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{w}, \qquad y = f(\boldsymbol{x}) + \epsilon,$$

with $\boldsymbol{x}$ being the input vector, $\boldsymbol{w}$ the vector of weights, the function value $f(\boldsymbol{x})$ and observed target value y with additive noise ϵ assumed to follow a zero-mean Gaussian. Combined with the independence assumption of the observation values, we get the likelihood, which reflects how probable the observed target values $\boldsymbol{y}$ are for the different inputs $\boldsymbol{X}$ and weights $\boldsymbol{w}$:

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) = \prod_{i=1}^{j} p(y_i|\boldsymbol{x}_i, \boldsymbol{w})$$

As usual for a Bayesian formulation, we define a prior over the weights, for which we again choose a zero-mean Gaussian. With the defined prior and the likelihood based on the observed data, we can use Bayes' rule to get the posterior of the weights given the data:

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X})}$$

This is also called the maximum a posteriori estimate which – provided the data – delivers the most likely set of weights $\boldsymbol{w}$. As $p(\boldsymbol{y}|\boldsymbol{X})$ is independent of $\boldsymbol{w}$, we can reformulate this equation expressing the posterior distribution with a Gaussian defined by a mean and covariance matrix:

$$p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y}) \sim \mathcal{N}(\overline{\boldsymbol{w}}, \boldsymbol{A}^{-1})$$

During inference, we marginalize out $\boldsymbol{w}$ and as a result take the average based on all possible $\boldsymbol{w}$ weighted by their posterior probability:

$$\begin{aligned} p\left(y_{Test}|\boldsymbol{x}_{Test},\boldsymbol{X},\boldsymbol{y}\right) &= \int p\left(y_{Test}|\boldsymbol{x}_{Test},\boldsymbol{w}\right) p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y})\, d\boldsymbol{w} \\ &= \mathcal{N}(\frac{1}{\sigma^2}\boldsymbol{x}_{Test}^T\boldsymbol{A}^{-1}\boldsymbol{X}\boldsymbol{y}, \boldsymbol{x}_{Test}^T\boldsymbol{A}^{-1}\boldsymbol{x}_{Test}) \end{aligned}$$

Therefore, we do not only get an output value, but also an uncertainty. So far, we reached the Bayesian formulation of linear regression with its limited expressiveness. To overcome this constraint to linearity, we can project the inputs into a high-dimensional space and apply the linear concept there. This transformation can be accomplished using basis functions $\phi(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^i$ leading to the following model with i weights $\boldsymbol{w}$:

$$f(\boldsymbol{x}) = \phi(\boldsymbol{x})^T\boldsymbol{w}$$

Conducting the same derivation as shown above results in a similar outcome:

$$\begin{aligned} p\left(y_{Test}|\boldsymbol{x}_{Test},\boldsymbol{X},\boldsymbol{y}\right) = \mathcal{N}(&\frac{1}{\sigma^2}\phi\left(\boldsymbol{x}_{Test}\right)^T\boldsymbol{A}^{-1}\boldsymbol{\Phi}(\boldsymbol{X})\boldsymbol{y}, \\ &\phi\left(\boldsymbol{x}_{Test}\right)^T\boldsymbol{A}^{-1}\phi(\boldsymbol{x}_{Test})) \end{aligned}$$

The need of inverting the ixi matrix $\boldsymbol{A}$ possibly causes computational problems if the dimension of the feature space i becomes large. To solve this, we can reformulate the above using the so-called "kernel trick". This leads to the formulation of a Gaussian Process, which is completely specified by its mean and covariance function:

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x},\boldsymbol{x}'))$$

$$m(\boldsymbol{x}) = \mathbb{E}\left[\boldsymbol{f}(\boldsymbol{x})\right], \quad \boldsymbol{k}(\boldsymbol{x},\boldsymbol{x}') = \mathbb{E}[(\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{m}(\boldsymbol{x}))(\boldsymbol{f}(\boldsymbol{x}') - \boldsymbol{m}(\boldsymbol{x}'))]$$

$k(\boldsymbol{x},\boldsymbol{x}')$ consists of the covariance value between any two sample points $\boldsymbol{x}$ and $\boldsymbol{x}'$ resulting in a nxn matrix for a training set length of n. The assumption is that the similarity between samples reflects the strength of the correlation between their corresponding target values. Therefore, the function evaluation can be seen as a draw from a multivariate Gaussian distribution defined by $m(\boldsymbol{x})$ and $k(\boldsymbol{x},\boldsymbol{x}')$. Thus, Gaussian Processes are a distribution over functions rather than parameters, in contrast to Bayesian linear regression. For simplicity, the mean function is often set to zero or a constant value. There are many forms of kernel functions, which need to fulfill certain properties, e.g. being positive semidefinite and symmetric. Furthermore, they can be combined, e.g. by summation or multiplication. The choice of the covariance kernel function is a determining configuration of GPR and its parameters need to be optimized during training (Rasmussen & Williams, 2008; Roberts et al., 2013).

Appendix B. Hyperparameters

Subsequently, the hyperparameters and transformations we used for all algorithms can be found. For every one, we randomly sampled hyperparameter combinations for optimization. All tables show the values used for the comparison before the best working solutions were analyzed in an in-depth optimization. Beyond that, we used different imputation strategies, from which KNN and Iterative Imputation needed to be parametrized. We empirically determined k = 10 for KNN and a maximum number of iterations of 100 for Iterative Imputation.

B.1. Exponential Smoothing (ES)

See Table B.1.

Table B.1
Hyperparameters Exponential Smoothing.

Parameter	Values	Notes
seasonal_periods	[7, 14, 21]	Only for *SoldTulips*: period for seasonality
trend	['add', None]	Trend component
damp	[False, True]	Damp trend
seasonality	['add', 'mul', None]	Seasonal component
remove_bias	[False, True]	Force average residual to zero
use_brute	[False, True]	Search starting values using brute force or use a naive set
transf	[False, 'log', 'pw']	Transformation method

B.2. Seasonal Autoregressive Integrated Moving Average (with external factors) (SARIMA(X))

See Table B.2.

Table B.2
Hyperparameters Seasonal Autoregressive Integrated Moving Average (with external factors).

Parameter	Values	Notes
$p/q/P/Q$	[0, 1, 2, 3]	Non-seasonal/seasonal lag for AR and MA component
d/D	[0, 1]	Non-seasonal/seasonal differencing
exog	[False, True]	Use exogeneous variables (SARIMA or SARIMAX)
transf	[False, 'log', 'pw']	Transformation method

B.3. Regularized Regression (LassoReg, RidgeReg, ElasticNet)

See Table B.3.

Table B.3
Hyperparameters Regularized Regression.

Parameter	Values	Notes
normalize	[False, True]	Normalization prior to regression
alpha	[10** x for x in range(−5, 5)]	Constant multiplying the regularization penalty term
l1_ratio	Np.arange(0.1, 1, 0.1)	Only for Elastic Net: ratio of L1-regularization

B.4. Artificial Neural Network (ANN)

See Table B.4.

Table B.4
Hyperparameters Artificial Neural Network.

Parameter	Values	Notes
activation_function	ReLu	Activation function after each neuron
optimizer	Adam	Optimizer for hyperparameter optimization
dropout_rate	[0.0, 0.5]	Rate for dropout layer
batch_size	[4, 8, 16, 32]	Batch size for training
learning_rate	[1e−4, 1e−3, 1e−2, 1e−1]	Learning rate for Adam optimizer
min_val_loss_improvement	[100, 1000]	Minimum validation loss improvement for early stopping
max_epochs_wo_improvement	[20, 50, 100]	Maximum epochs without improvement for early stopping
n_hidden	[10, 20, 50, 100]	Number of hidden neurons of input layer
num_hidden_layer	[1, 2, 3]	Number of hidden layer (including input layer)

B.5. Long Short-Term Memory Network (LSTM)

See Table B.5.

Table B.5
Hyperparameters Long Short-Term Memory Network.

Parameter	Values	Notes
activation_function	ReLu	Activation function after each neuron
optimizer	Adam	Optimizer for hyperparameter optimization
dropout_rate	[0.0, 0.5]	Rate for dropout layer
batch_size	[4, 8, 16, 32]	Batch size for training
learning_rate	[1e−3, 1e−2, 1e−1]	Learning rate for Adam optimizer
min_val_loss_improvement	[0.01, 0.1]	Minimum validation loss improvement for early stopping
max_epochs_wo_improvement	[20, 50, 100]	Maximum epochs without improvement for early stopping
lstm_hidden_dim	[5, 10, 50]	Dimensionality of the hidden state
lstm_num_layers	[1, 2]	Number of recurrent layers
seq_length	[1, 4, seasonal_periods]	Sequence length for input

B.6. Extreme Gradient Boosting (XGB)

See Table B.6.

Table B.6
Hyperparameters Extreme Gradient Boosting.

Parameter	Values	Notes
learning_rate	[0.05, 0.1, 0.3]	Learning rate / eta
max_depth	[3, 5, 10]	Maximum tree depth of a base learner
subsample	[0.3, 0.7, 1]	Subsample ratio of a training instance
n_estimators	[10, 100, 1000]	Number of gradient boosted trees
gamma	[0, 1, 10]	Minimum loss reduction for a partition on a leaf node
alpha	[0, 0.1, 1, 10]	Weight for L1-regularization
reg_lambda	[0, 0.1, 1, 10]	Weight for L2-regularization

B.7. Bayesian Regression (BayesRidge, ARD)

See Table B.7.

Table B.7
Hyperparameters Bayesian Regression.

Parameter	Values	Notes
normalize	[False, True]	Normalization prior to regression
alpha_1	[10** x for x in range(−6, 1)]	Shape parameter for Gamma distribution over alpha
alpha_2	[10** x for x in range(−6, −4)]	Rate parameter for Gamma distribution over alpha
lambda_1	[10** x for x in range(−6, 1)]	Shape parameter for Gamma distribution over lambda
lambda_2	[10** x for x in range(−6, 1)]	Rate parameter for Gamma distribution over lambda
threshold_lambda	[10** × for × in range(2, 6)]	Only for ARD: threshold for pruning weights

B.8. Gaussian Process Regression (GPR)

The determining parameter for GPR is the kernel function. We used different kernel functions as well as combinations of up to three kernels (sums and products). The base kernels we used can be found in the following list, which are formulated according to the used library GPflow (Matthews et al., 2017):

- *SquaredExponential()*
- *Matern52()*
- *White()*
- *RationalQuadratic()*
- *Polynomial(),*
- *Periodic(kernels=SquaredExponential(), period=seasonal_periods),*
- *Periodic(kernels=Matern52(), period=seasonal_periods),*
- *Periodic(kernels=RationalQuadratic(), period=seasonal_periods)]*

Furthermore, we used the subsequent hyperparameters.

Table B.8
Hyperparameters Gaussian Process Regression.

Parameter	Values	Notes
mean_function	[None, Constant()]	Mean function used for Gaussian distribution
noise_variance	[0.01, 1, 10, 100]	Noise variance added to diagonal of kernel matrix
standardize_x	[False, True]	Standardize features
standardize_y	[False, True]	Standardize target variable

Appendix C. Results Overview Tables

The tables, which reflect an overview of the results of the comparison performed in this paper, are published in supplementary fashion as *"Supplementary 1 - Results Overview Tables.xlsx"*. This file contains several tabs, of which each one presents the results of one dataset. These show an overview of the best optimization results with respect to the three evaluation metrics of every algorithm for each experimental setting. Every table is grouped in *"Univariate Methods and Baselines"* as well as *"Multivariate Methods"*. Besides that, every chart is structured by the used featureset, imputation strategy and dimensionality reduction. If a specific combination was not included in the comparison, the related cell is filled in gray. The best results are printed in bold, and the cells are highlighted with a green background.

Appendix D. Top Results Tables

The following tables show top results for several setups such as the best performance of every algorithm on each dataset over all featuresets and preprocessing methods, see Tables D.1 and D.2. As in all subsequent tables, the best results are printed in bold and highlighted in green. The results based on each feature- and dataset for the multivariate top performers are summarized in Appendix D.2. The evaluation is based on the results before the in-depth optimization, as this second optimization was not conducted for all configurations.

D.1. Top Results Overall

See Tables D.1 and D.2.

Table D.1
Top results overall for the OwnDoc datasets.

	OwnDoc					
	OwnDoc_SoldTulips_short			**OwnDoc_SoldTulips_long**		
algorithm	**RMSE**	**sMAPE**	**MAPE**	**RMSE**	**sMAPE**	**MAPE**
ES	77.63	61.84	116.77	62.17	56.61	241.40
SARIMA	83.31	58.19	81.52	52.02	55.83	3675.31
SARIMAX	58.06	52.58	85.75	50.35	52.37	231.52
LassoReg	84.65	58.86	1615.95	59.32	52.50	2354.94
RidgeReg	84.73	60.29	1978.83	58.47	54.12	2970.38
ElasticNet	84.73	60.16	1569.83	57.27	52.64	2888.18
ANN	77.48	56.06	115.73	48.93	53.09	223.10
LSTM	73.66	52.49	707.61	56.06	50.04	2884.77
XGB	**57.11**	**50.89**	**35.98**	**43.52**	**48.65**	**54.22**
BayesRidge	96.88	62.62	9153.80	60.25	53.28	6221.16
ARD	79.89	57.41	162.77	60.35	53.68	1194.57
GPR	68.14	52.27	67.99	56.05	52.43	65.84

Table D.2
Top results overall for the CashierData datasets.

	CashierData								
	CashierData_CutFlowers			CashierData_PotTotal_short			CashierData_PotTotal_long		
algorithm	**RMSE**	**sMAPE**	**MAPE**	**RMSE**	**sMAPE**	**MAPE**	**RMSE**	**sMAPE**	**MAPE**
ES	591.89	28.29	58.18	513.26	36.87	37.74	1066.36	37.63	57.12
SARIMA	608.13	27.50	57.23	439.03	35.05	40.13	979.10	37.03	56.59
SARIMAX	578.15	25.31	58.53	531.76	38.33	42.29	1063.66	37.52	50.57
LassoReg	563.27	25.53	58.12	455.28	35.19	43.05	1067.88	38.53	64.61
RidgeReg	566.51	25.66	56.98	482.28	37.14	42.68	1136.46	44.09	64.42
ElasticNet	565.27	25.63	56.43	475.17	36.16	41.85	1063.28	36.85	64.65
ANN	485.60	23.43	32.19	507.19	39.40	41.04	1031.24	44.36	42.82
LSTM	460.61	22.57	29.65	**390.98**	**28.85**	31.11	1064.14	39.65	48.90
XGB	**390.96**	**20.03**	**24.52**	392.31	31.50	**30.70**	**898.56**	**35.35**	**42.45**
BayesRidge	572.96	26.06	57.88	534.58	40.73	39.59	1062.94	38.53	66.32
ARD	550.98	24.91	55.24	460.64	34.94	42.93	1026.61	36.01	57.27
GPR	493.58	23.61	30.45	424.50	31.88	34.42	922.46	36.03	54.20

Table D.3
Top results per featureset of the OwnDoc datasets.

		OwnDoc					
		OwnDoc_SoldTulips_short			OwnDoc_SoldTulips_long		
algorithm	**featureset**	**RMSE**	**sMAPE**	**MAPE**	**RMSE**	**sMAPE**	**MAPE**
SARIMAX	sub1	74.39	57.46	222.57	53.40	52.37	585.06
	sub2	64.27	52.74	138.81	50.35	55.23	231.52
	sub3	60.37	54.98	110.87	71.98	61.57	237.38
	full	58.06	52.58	85.75	55.73	65.63	254.73
LSTM	sub1	86.28	56.27	4906.27	58.24	54.57	8237.34
	sub2	88.90	57.63	4450.57	58.74	54.88	6398.16
	sub3	73.66	52.49	914.54	56.21	50.87	5481.53
	full	83.90	55.25	707.61	56.06	50.04	2884.77
XGB	sub1	120.97	65.83	5936.07	47.90	54.22	4745.91
	sub2	67.67	57.19	43.90	48.15	54.03	**54.22**
	sub3	**57.11**	53.24	37.95	48.82	**48.65**	1221.83
	full	62.52	**50.89**	**35.98**	**43.52**	52.36	1346.26
GPR	sub1	96.01	59.95	79.00	58.14	52.82	74.03
	sub2	68.14	57.91	79.02	56.05	54.88	73.91
	sub3	76.63	55.79	67.99	63.10	52.61	65.84
	full	75.02	52.27	70.54	56.56	52.43	69.28

D.2. Featureset-specific Top Results for SARIMAX, LSTM, XGB and GPR

To visualize the top results per featureset, we derived the stacked bar plots shown in Fig. D.1 based on the information presented in Tables D.3 and D.4. Furthermore, Table D.5 summarizes the featuresets leading to the best result for each dataset.

Appendix E. Supplementary Data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.mlwa.2021.100239.

Table D.4
Top results per featureset of the CashierData datasets.

		CashierData								
		CashierData_CutFlowers			CashierData_PotTotal_short			CashierData_PotTotal_long		
algorithm	featureset	RMSE	sMAPE	MAPE	RMSE	sMAPE	MAPE	RMSE	sMAPE	MAPE
SARIMAX	sub1	706.69	30.93	60.95	587.51	38.33	50.16	1084.95	37.52	54.97
	sub2	578.15	25.31	58.53	531.76	39.05	42.29	1063.66	41.03	50.57
	sub3	679.88	30.93	60.95	587.51	39.29	50.32	1086.67	37.64	54.47
	full	585.82	26.13	63.34	545.43	39.74	44.81	1095.59	41.45	51.04
LSTM	sub1	460.61	23.74	34.15	401.52	**28.85**	34.59	1100.44	41.99	52.26
	sub2	487.53	22.57	37.74	391.28	30.69	34.81	1064.14	39.65	48.90
	sub3	577.10	25.16	34.22	407.37	29.95	31.11	1143.76	42.15	53.06
	full	511.28	24.61	29.65	**390.98**	30.83	31.77	1116.59	41.71	50.47
XGB	sub1	582.96	21.95	**24.52**	512.48	37.92	37.79	1494.98	47.08	51.60
	sub2	**390.96**	**20.03**	31.52	428.25	33.97	**30.70**	1360.75	40.91	44.22
	sub3	640.82	23.68	46.30	392.31	31.50	34.09	**898.56**	**35.35**	**42.45**
	full	467.24	22.40	45.71	430.47	33.19	35.37	974.64	36.18	42.63
GPR	sub1	683.74	27.35	45.72	543.26	40.44	44.74	1657.81	56.60	67.56
	sub2	493.58	23.61	30.45	424.50	31.88	34.42	1358.78	54.06	56.27
	sub3	655.74	26.43	51.25	448.09	36.00	39.30	978.78	36.36	54.20
	full	625.42	26.79	51.97	458.47	38.05	37.49	922.46	36.03	55.08

Table D.5
Win counts for each featureset on every algorithm and dataset. A win is defined as a featureset leading to the best result for one of the evaluation metrics. The wins of every featureset are summed up. The leading featuresets for every algorithm on OwnDoc and CashierData are printed in bold and highlighted in green.

		OwnDoc			CashierData			
		SoldTulips_short	SoldTulips_long	summary	CutFlowers	PotTotal_short	PotTotal_long	summary
algorithm	featureset	wins	wins	wins	wins	wins	wins	wins
SARIMAX	sub1	0	1	1	0	1	1	2
	sub2	0	2	2	3	2	2	**7**
	sub3	0	0	0	0	0	0	0
	full	3	0	**3**	0	0	0	0
LSTM	sub1	0	0	0	2	1	0	3
	sub2	0	0	0	1	0	3	**4**
	sub3	2	0	2	0	1	0	1
	full	1	3	**4**	0	1	0	1
XGB	sub1	0	0	0	1	0	0	1
	sub2	0	1	1	2	1	0	3
	sub3	1	1	2	0	2	3	**5**
	full	2	1	**3**	0	0	0	0
GPR	sub1	0	0	0	0	0	0	0
	sub2	1	1	**2**	3	3	0	**6**
	sub3	1	1	**2**	0	0	1	1
	full	1	1	**2**	0	0	2	2

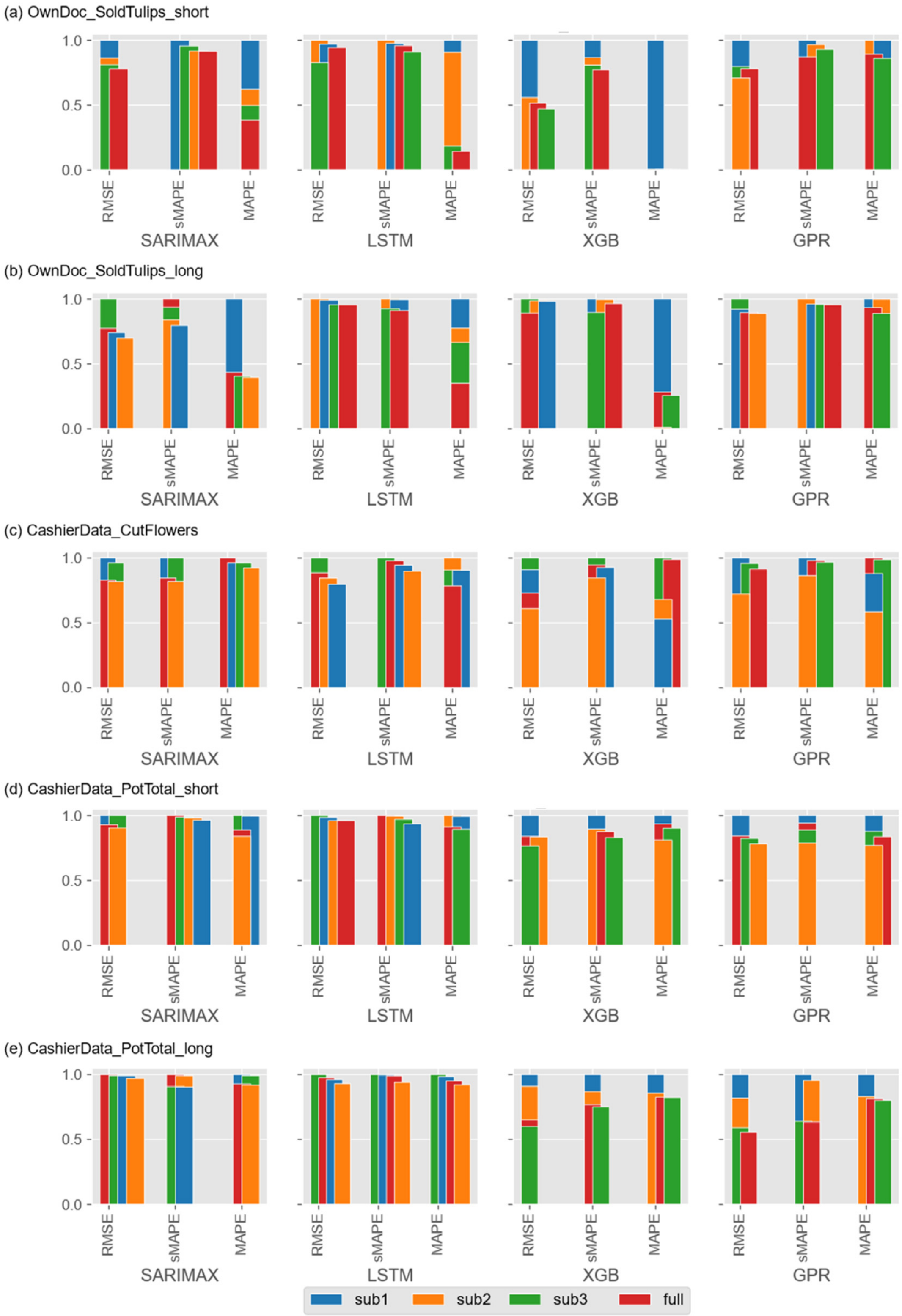

Fig. D.1. Top results per featureset for SARIMAX, LSTM, XGB, GPR (columns) on every dataset (rows). The values achieved with every featureset are scaled by the worst result. Smaller bars represent a better outcome with similar ones plotted with a small offset on the x-axis to ensure visibility. Bars might not be visible in case of a major difference to the worst result, e.g. XGB's MAPE for OwnDoc_Sold_Tulips_short.

References

Arunraj, N. S., & Ahrens, D. (2015). A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting. *International Journal of Production Economics*, *170*, 321–335. http://dx.doi.org/10.1016/j.ijpe.2015.09.039.

Arunraj, N. S., Ahrens, D., & Fernandes, M. (2016). Application of SARIMAX model to forecast daily sales in food retail industry. *International Journal of Operations Research and Information Systems*, *7*(2), 1–21. http://dx.doi.org/10.4018/IJORIS.2016040101.

Arunraj, N. S., Ahrens, D., Fernandes, M., & Müller, M. (2014). Time series sales forecasting to reduce food waste in retail industry. *Rotterdam*, http://dx.doi.org/10.13140/RG.2.1.4829.1607.

Athanasopoulos, G., Hyndman, R. J., Song, H., & Wu, D. C. (2011). The tourism forecasting competition. *International Journal of Forecasting*, *27*(3), 822–844. http://dx.doi.org/10.1016/j.ijforecast.2010.04.009.

Behe, B. K., Getter, K. L., & Yue, C. (2012). Should you blame the weather? The influence of weather parameters, month, and day of the week on spring herbaceous plant sales in the U.S. midwest. *HortScience*, *47*(1), 71–73. http://dx.doi.org/10.21273/HORTSCI.47.1.71.

Bishop, C. M. (2009). *Pattern Recognition and Machine Learning* (8th ed.). Springer.

Bojer, C. S., & Meldgaard, J. P. (2021). Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, *37*(2), 587–603. http://dx.doi.org/10.1016/j.ijforecast.2020.07.007.

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2016). *Wiley Series in Probability and Statistics*, *Time Series Analysis: Forecasting and Control* (5th ed.). Wiley.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In B. Krishnapuram, M. Shah, A. Smola, C. Aggarwal, D. Shen, & R. Rastogi (Eds.), *Kdd 2016: 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 785–794). ACM Association for Computing Machinery, http://dx.doi.org/10.1145/2939672.2939785.

Duan, Y., Li, G., Tien, J. M., & Huo, J. (2012). Inventory models for perishable items with inventory level dependent demand rate. *Applied Mathematical Modelling*, *36*(10), 5015–5028. http://dx.doi.org/10.1016/j.apm.2011.12.039.

Gardner, E. S. (2006). Exponential smoothing: The state of the art—Part II. *International Journal of Forecasting*, *22*(4), 637–666. http://dx.doi.org/10.1016/j.ijforecast.2006.03.005.

Grande, R. C., Walsh, T. J., Chowdhary, G., Ferguson, S., & How, J. P. (2017). Online regression for data with changepoints using Gaussian processes and reusable models. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(9), 2115–2128. http://dx.doi.org/10.1109/TNNLS.2016.2574565.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Del Río, J. F., Wiebe, M., Peterson …, P., & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. http://dx.doi.org/10.1038/s41586-020-2649-2.

Haselbeck, F., & Grimm, D. G. (2021). Evars-GPR: Event-triggered augmented refitting of Gaussian process regression for seasonal data. In S. Edelkamp, R. Möller, & E. Rueckert (Eds.), *Lecture Notes in Computer Science*: *vol. 12873*, *Ki 2021: 44th German Conference on AI, Virtual Event, September 27 - October 1, 2021, Proceedings (Vol. 12873)* (pp. 135–157). Springer, http://dx.doi.org/10.1007/978-3-030-87626-5_11.

Hewamalage, H., Bergmeir, C., & Bandara, K. (2021). Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, *37*(1), http://dx.doi.org/10.1016/j.ijforecast.2020.06.008.

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

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

Holt, C. C. (1957). *Forecasting Seasonals and Trends by Exponentially Weighted Moving Averages*. Office of Naval Research Memorandum.

Hong, T., Xie, J., & Black, J. (2019). Global energy forecasting competition 2017: Hierarchical probabilistic load forecasting. *International Journal of Forecasting*, *35*(4), 1389–1399. http://dx.doi.org/10.1016/j.ijforecast.2019.02.006.

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

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. http://dx.doi.org/10.1109/MCSE.2007.55.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice (2. Auflage)*. OTexts.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. http://dx.doi.org/10.1016/j.ijforecast.2006.03.001.

Ivanov, D., Tsipoulanidis, A., & Schönberger, J. (2019). Demand forecasting. In D. Ivanov, A. Tsipoulanidis, & J. Schönberger (Eds.), *Springer Texts in Business and Economics. Global Supply Chain and Operations Management: A Decision-Oriented Introduction to the Creation of Value* (pp. 319–333). Springer, http://dx.doi.org/10.1007/978-3-319-94313-8_11.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). *An Introduction to Statistical Learning: With Applications in R* (8th ed.). Springer, http://dx.doi.org/10.1007/978-1-4614-7138-7.

Jiao, E. X., & Chen, J. L. (2019). Tourism forecasting: A review of methodological developments over the last decade. *Tourism Economics*, *25*(3), 469–492. http://dx.doi.org/10.1177/1354816618812588.

Jolliffe, Ian T., & Cadima, Jorge (2016). Principal component analysis: A review and recent developments. *Phil. Trans. Ser. A Math. Phys. Eng. Sci.*, *374*(2065), http://dx.doi.org/10.1098/rsta.2015.0202.

Kolassa, Stephan (2021). Commentary on the M5 forecasting competition. *International Journal of Forecasting*, http://dx.doi.org/10.1016/j.ijforecast.2021.08.006, Advance online publication.

Liu, X., & Ichise, R. (2017). Food sales prediction with meteorological data — A case study of a Japanese chain supermarket. In Y. Tan, H. Takagi, & Y. Shi (Eds.), *Lecture Notes in Computer Science*: *vol. 10387*, *Data Mining and Big Data: Second International Conference, DMBD 2017, Fukuoka, Japan, July 27 - August 1, 2017, Proceedings (Vol. 10387)* (pp. 93–104). Springer International Publishing, http://dx.doi.org/10.1007/978-3-319-61845-6_10.

Liu, B., Qi, Y., & Chen, K.-J. (2020). Sequential online prediction in the presence of outliers and change points: An instant temporal structure learning approach. *Neurocomputing*, *413*, 240–258. http://dx.doi.org/10.1016/j.neucom.2020.07.011.

Lloyd, J. R. (2014). Gefcom2012 hierarchical load forecasting: Gradient boosting machines and Gaussian processes. *International Journal of Forecasting*, *30*(2), 369–374. http://dx.doi.org/10.1016/j.ijforecast.2013.07.002.

Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Newton, J., Parzen, E., & Winkler, R. (1982). The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of Forecasting*, *1*(2), 111–153. http://dx.doi.org/10.1002/for.3980010202.

Makridakis, S., Chatfield, C., Hibon, M., Lawrence, M., Mills, T., Ord, K., & Simmons, L. F. (1993). The M2-competition: A real-time judgmentally based forecasting study. *International Journal of Forecasting*, *9*(1), 5–22. http://dx.doi.org/10.1016/0169-2070(93)90044.

Makridakis, S., & Hibon, M. (2000). The M3-competition: results, conclusions and implications. *International Journal of Forecasting*, *16*(4), 451–476. http://dx.doi.org/10.1016/S0169-2070(00)00057-1.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). The M4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, *34*(4), 802–808. http://dx.doi.org/10.1016/j.ijforecast.2018.06.001.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 competition: 100, 000 time series and 61 forecasting methods. *International Journal of Forecasting*, *36*(1), 54–74. http://dx.doi.org/10.1016/j.ijforecast.2019.04.014.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2021). The M5 competition: Background, organization, and implementation. *International Journal of Forecasting*, http://dx.doi.org/10.1016/j.ijforecast.2021.07.007, Advance online publication.

Matthews, A. G. de. G., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., & Hensman, J. (2017). Gpflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*, *18*(40), 1299–1304.

McKinney, W. (2010). Data structures for statistical computing in python. In proceedings of the python in science conference. In *Proceedings of the 9th Python in Science Conference* (pp. 56–61). SciPy, http://dx.doi.org/10.25080/Majora-92bf1922-00a.

Ni, W., Tan, S. K., Ng, W. J., & Brown, S. D. (2012). Moving-window GPR for nonlinear dynamic system modeling with dual updating and dual preprocessing. *Industrial and Engineering Chemistry Research*, *51*(18), 6416–6428. http://dx.doi.org/10.1021/ie201898a.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang ., L., & Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems, Vol. 32* (pp. 8024–8035). Curran Associates, Inc.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Petropoulos, F., Hyndman, R. J., & Bergmeir, C. (2018). Exploring the sources of uncertainty: Why does bagging for time series forecasting work? *European Journal of Operational Research*, *268*(2), 545–554. http://dx.doi.org/10.1016/j.ejor.2018.01.045.

Priyadarshi, R., Panigrahi, A., Routroy, S., & Garg, G. K. (2019). Demand forecasting at retail stage for selected vegetables: a performance analysis. *Journal of Modelling in Management*, *14*(4), 1042–1063. http://dx.doi.org/10.1108/JM2-11-2018-0192.

Rasmussen, C. E., & Williams, C. K. I. (2008). *Gaussian Processes for Machine Learning (3. Print). Adaptive Computation and Machine Learning*. MIT Press.

Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., & Aigrain, S. (2013). Gaussian processes for time-series modelling. *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences*, *371*(1984), Article 20110550. http://dx.doi.org/10.1098/rsta.2011.0550.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. http://dx.doi.org/10.1037/h0042519.

Sankaran, S. (2014). Demand forecasting of fresh vegetable product by seasonal ARIMA model. *International Journal of Operational Research*, *20*(3), 315. http://dx.doi.org/10.1504/IJOR.2014.062453.

Santosa, F., & Symes, W. W. (1986). Linear inversion of band-limited reflection seismograms. *SIAM Journal on Scientific and Statistical Computing*, *7*(4), 1307–1330. http://dx.doi.org/10.1137/0907087.

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*.

Seaman, B., & Bowman, J. (2021). Applicability of the M5 to forecasting at walmart. *International Journal of Forecasting*, http://dx.doi.org/10.1016/j.ijforecast.2021.06.002, Advance online publication.

Shukla, M., & Jharkharia, S. (2011). ARIMA models to forecast demand in fresh supply chains. *International Journal of Operational Research*, *11*(1), 1–18. http://dx.doi.org/10.1504/IJOR.2011.040325.

Shumway, R. H., & Stoffer, D. S. (2000). Time series regression and ARIMA models. In R. H. Shumway, & D. S. Stoffer (Eds.), *Springer Texts in Statistics. Time Series Analysis and Its Applications* (pp. 89–212). Springer, http://dx.doi.org/10.1007/978-1-4757-3261-0_2.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(56), 1929–1958.

Stepnicka, M., & Burda, M. (2017). On the results and observations of the time series forecasting competition CIF 2016. In *Fuzz-IEEE 2017: 2017 IEEE International Conference on Fuzzy Systems : 9-12 2017, Royal Continental Hotel, Naples, Italy* (pp. 1–6). IEEE, http://dx.doi.org/10.1109/FUZZ-IEEE.2017.8015455.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *58*(1), 267–288. http://dx.doi.org/10.1111/j.2517-6161.1996.tb02080.x.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, *1*, 211–244. http://dx.doi.org/10.1162/15324430152748236.

SciPy 1. 0 Contributors (2020). SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, *17*, 261–272. http://dx.doi.org/10.1038/s41592-019-0686-2.

Waskom, M. L. (2021). Seaborn: statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. http://dx.doi.org/10.21105/joss.03021.

Williams, C., & Rasmussen, C. (1996). Gaussian processes for regression. In D. Touretzky, M. C. Mozer, & M. Hasselmo (Eds.), *Advances in Neural Information Processing Systems (Vol. 8)*. MIT Press.

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. *Management Science*, *6*(3), 324–342. http://dx.doi.org/10.1287/mnsc.6.3.324.

Yeo, I.-K., & Johnson, R. A. (2000). A new family of power transformations to improve normality or symmetry. *Biometrika*, *87*(4), 954–959. http://dx.doi.org/10.1093/biomet/87.4.954.

Zentralverband Gartenbau e. V (2020). Jahresbericht 2020.

Zhang, G., Eddy Patuwo, B., & Y. Hu, M. (1998). Forecasting with artificial neural networks. *International Journal of Forecasting*, *14*(1), 35–62. http://dx.doi.org/10.1016/S0169-2070(97)00044-7.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *67*(2), 301–320. http://dx.doi.org/10.1111/j.1467-9868.2005.00503.x.