REVIEW OF AUTOMATED TIME SERIES FORECASTING PIPELINES

Stefan Meisenbacher
Institute for Automation and Applied Informatics
Karlsruhe Institute of Technology
Eggenstein-Leopoldshafen, 76344, Germany
stefan.meisenbacher@kit.edu

Marian Turowski
Institute for Automation and Applied Informatics
Karlsruhe Institute of Technology
Eggenstein-Leopoldshafen, 76344, Germany

Kaleb Phipps
Institute for Automation and Applied Informatics
Karlsruhe Institute of Technology
Eggenstein-Leopoldshafen, 76344, Germany

Martin Rätz
Institute for Energy Efficient Buildings and Indoor Climate
RWTH Aachen University
Aachen, 52062, Germany

Dirk Müller
Institute for Energy Efficient Buildings and Indoor Climate
RWTH Aachen University
Aachen, 52062, Germany
Forschungszentrum Jülich GmbH
Institute of Energy and Climate Research
Energy Systems Engineering (IEK-10)
Jülich, 52425, Germany

Veit Hagenmeyer
Institute for Automation and Applied Informatics
Karlsruhe Institute of Technology
Eggenstein-Leopoldshafen, 76344, Germany

Ralf Mikut
Institute for Automation and Applied Informatics
Karlsruhe Institute of Technology
Eggenstein-Leopoldshafen, 76344, Germany

ABSTRACT

Time series forecasting is fundamental for various use cases in different domains such as energy systems and economics. Creating a forecasting model for a specific use case requires an iterative and complex design process. The typical design process includes the five sections (1) data pre-processing, (2) feature engineering, (3) hyperparameter optimization, (4) forecasting method selection, and (5) forecast ensembling, which are commonly organized in a pipeline structure. One promising approach to handle the ever-growing demand for time series forecasts is automating this design process. The present paper, thus, analyzes the existing literature on automated time series forecasting pipelines to investigate how to automate the design process of forecasting models. Thereby, we consider both Automated Machine Learning (AutoML) and automated statistical forecasting methods in a single forecasting pipeline. For this purpose, we firstly present and compare the proposed automation methods for each pipeline section. Secondly, we analyze the automation methods regarding their interaction, combination, and coverage of the five pipeline sections. For both, we discuss the literature, identify problems, give recommendations, and suggest future research. This review reveals that the majority of papers only cover two or three of the five pipeline sections. We conclude that future research has to holistically consider the automation of the forecasting pipeline to enable the large-scale application of time series forecasting.

Keywords Automated Machine Learning · Time Series Forecasting · AutoML · Pipeline · Pre-processing · Feature Engineering · Hyperparameter Optimization · Forecasting Method Selection · Ensemble

1   Introduction

One of the most prominent forms of collected data are time series. In a time series, the data is arranged sequentially, and each value is explicitly time-stamped, with information such as date and time (i. e, value and time stamp constitute an observation). The progression of a time series over a certain period of time in the future, also known as forecast horizon, is the subject of time series forecasting. Time series forecasting is applied with various forecast horizons at different temporal scales and aggregation levels in various domains [1]. Exemplary use cases from different domains are the following: sales forecasting for inventory optimization (*just-in-time supply chain*, [2]), forecasting the generation of renewable energy and the electricity demand in an area to balance the power grid load (*smart grids*, [3]), and forecasting the spread of the novel coronavirus COVID-19 (*pandemic control*, [4]). As the number and importance of use cases grow, the demand for time series forecasts is increasing steadily.

Designing a time series forecast for a particular use case typically incorporates five sections. The first section of the design process is the data pre-processing to transform the raw data into a desirable form for the forecasting method [5, 6]. The second section is the feature engineering, which aims to extract hidden characteristics of the considered time series or to identify useful exogenous information for the forecasting method [7]. Each forecasting method contains hyperparameters that have to be set by the data scientist. Therefore, the third section, the HyperParameter Optimization (HPO), intends to improve the forecast accuracy over the default hyperparameter configuration [8]. Apart from the HPO, selecting the most suitable forecasting method is crucial for the forecast accuracy and is addressed in the fourth section [9]. The fifth section aims to increase the robustness of the forecast by forecast ensembling [10], i. e., bundling multiple forecasts of different forecasting models to avoid occasional poor forecasts [11].

The above sections of the design process are commonly organized in a pipeline structure as shown in Figure 1. Manually tailoring the forecasting pipeline to a specific use case is time-consuming and challenging because selecting appropriate methods for the pipeline sections is iterative and requires expert knowledge. This expert knowledge is particularly crucial, as the forecast accuracy is sensitive to various design decisions [8]. It is also foreseeable that the number of knowledgeable data scientists cannot handle the ever-growing demand for time series forecasts in the future. Therefore, increasing the efficiency of the design process by automation is required [12].

To automate design decisions and remove the data scientist from the iterative design process, a variety of automation methods are available for each section of the forecasting pipeline.[1] The sequential organization of these automation methods and the management of the data flow can be realized by creating a pipeline [14, 15]. Running the created forecasting pipeline trains a forecasting method – e. g. a Linear Regression (LR) – with historical time series data and results in a parameterized forecasting model. Thereby, the pipeline automates the design process.
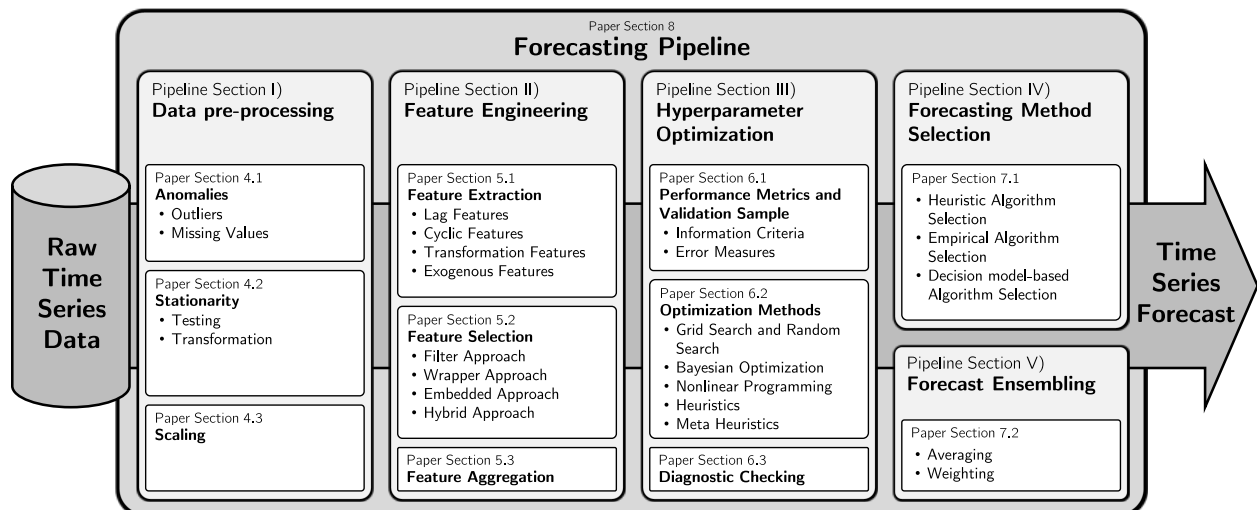

Figure 1: The forecasting pipeline systematizes the design process for time series forecasting using five pipeline sections.

---

[1]In addition to automating the design process, automating the operation of forecasts is proposed by [13], which includes self-monitoring and automatic model adaption as forecast accuracy decreases.

In this context, the long-term objective towards full automation has motivated numerous researchers and led to promising research results in the fields related to this review study, shown in Figure 2. Several surveys and review studies analyze the automation of single forecasting pipeline sections such as pre-processing [5, 6], feature engineering [7], HPO and forecasting method selection [8, 9], and forecast ensembling [10]. Moreover, rather than focusing on the automated design of forecasting pipelines, existing studies on time series forecasting only consider the statistical or machine learning forecasting methods themselves [16–18]. However, a comprehensive review study on the entire automated time series forecasting pipeline that considers the families of both statistical and machine learning methods is lacking.

HPO, HyperParameter Optimization;
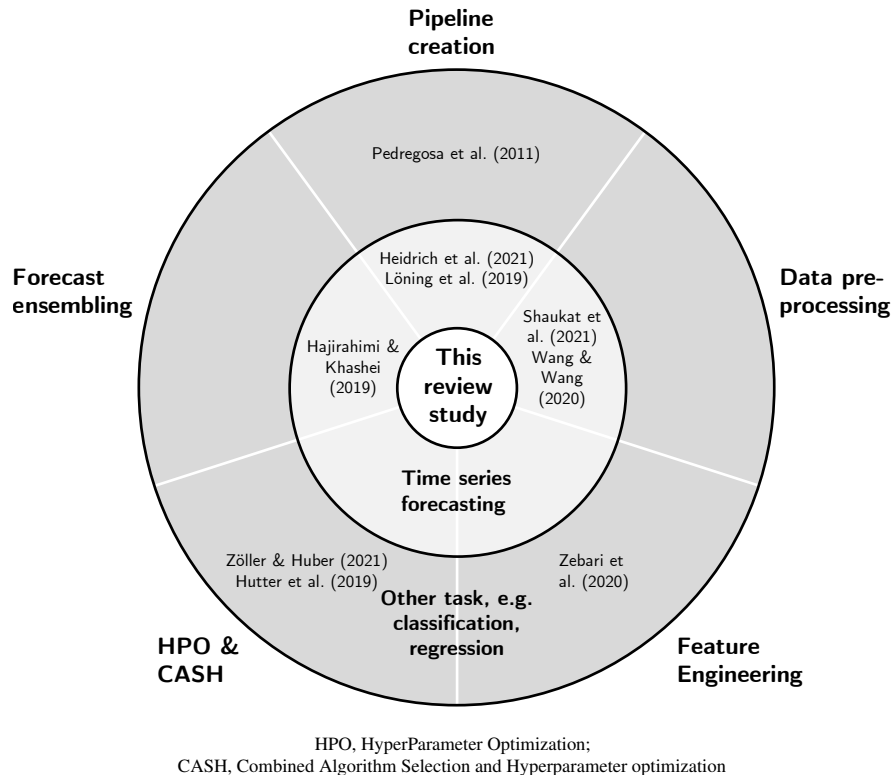CASH, Combined Algorithm Selection and Hyperparameter optimization

Figure 2: Related to this review are the fields of pipeline creation, data pre-processing, feature engineering, HyperParameter Optimization (HPO) & Combined Algorithm Selection and Hyperparameter optimization (CASH), and forecast ensembling.

Therefore, the present paper analyzes the literature on automated time series forecasting pipelines to investigate how to automate the design process of forecasting models. We consider literature from various research directions, including statistical forecasting, machine learning and deep learning.[2] More specifically, we focus on the interaction and combination of automation methods within the pipeline sections considering both Automated Machine Learning (AutoML) and automated statistical forecasting methods. For this purpose, we firstly present and systematically compare existing automation methods used in each pipeline section. Secondly, we analyze the complete automated forecasting pipeline considering how many pipeline sections are automated in the literature and highlighting the interaction and dependencies between pipeline sections. For both, we discuss the existing literature, identify potential problems, give recommendations, and suggest future research.

After describing the methodology in Section 2 and a brief introduction to time series forecasting in Section 3, the paper is organized by respective sections following the forecasting pipeline shown in Figure 1 and concludes in Section 9.

---

[2]In the following, we consider deep learning methods as part of the broader family of machine learning.

2 Methodology

The methodology of this literature review applies the following fundamental steps suggested by Webster and Watson [19] for the identification of major contributions, their origin, and evolution.

Literature Search: A search-term-based exploration of research articles considering title, abstract, and keywords is conducted using *Scopus*[3], resulting in 359 hits.[4] Potential predatory journals and publishers, and vanity press listed in *Beall's List*[5] are excluded.

Literature Screening: We screen the abstracts for relevance with the following criteria: i) the task type must be time series forecasting, and ii) at least one iterative element in the forecasting pipeline that previously required human intervention must be automated.

Backward-Forward Search: We identify additional articles that cite or are cited by the articles passing the screening. The obtained candidates also undergo the screening procedure defined above. Backward-forward search yield a pool of 2152 additional papers, which the keyword filtering reduces to 242 articles.[4]

Review: We present automation methods for each section of the forecasting pipeline and review the articles that pass the screening and the full text analysis. After the abstract screening, 144 of the 601 papers remain, and 71 after analyzing the full text. If there is similar work from a research group, we cite the article with the greatest methodological scope and articles that propose improvements, which reduces the included articles from 71 to 63 articles.

Discussion: We discuss the contributions towards design automation individually for each section of the forecasting pipeline. Afterward, we identify gaps and highlight future research directions by analyzing the coverage of the forecasting pipeline.

3 Time Series Forecasting

A time series $\{\mathbf{y}\,[k]\,;k=1,2\ldots,K\}$ reflects a set of $K\in\mathbb{N}^{>0}$ observations typically measured at equidistant points in time [20]. A time series forecasting model $f(\cdot)$ estimates future values $\hat{y}$ for one or more time points – the forecast horizon $H\in\mathbb{N}^{>0}$ – using current and past values [21]. It is defined as

$$
\begin{aligned}
\hat{y}[k + H] = f(&y[k],\ldots,y\,[k-H_1]\,,\\
&\mathbf{u}^T[k],\ldots,\mathbf{u}^T\,[k-H_1]\,,\\
&\hat{\mathbf{u}}^T[k],\ldots,\hat{\mathbf{u}}^T\,[k-H_1]\,,\\
&\mathbf{w});k,H,H_1\in\mathbb{N}^{>0};k>H_1,
\end{aligned}
\tag{1}
$$

where $H_1\in\mathbb{N}^{>0}$ indicates the horizon for past values $k-H_1$, the vector $\mathbf{w}$ contains the model's parameters, the vector $\mathbf{u}^\top$ denotes values from exogenous time series, the vector $\hat{\mathbf{u}}^\top$ indicates that the exogenous values originate from another forecast, and y represents values of the target time series [21].[6]

In time series forecasting, the following three families of methods exist: naïve methods, statistical methods, and machine learning methods. The families and their main representatives are briefly introduced in the following.

Naïve Forecasting Methods   The simplest method family of forecasting are naïve methods. Common representatives are the averaging method, where the forecasts of all future values are equal to the average of historical data, and the Random Walk (RW) method, where the value of the last observation is used as forecast [1]. For RW, modifications are available for data with drift (dRW) and seasonality (sRW).

Statistical Forecasting Methods   More sophisticated than naïve methods are statistical methods that use statistics based on historical data to forecast future time series values. The first representatives are AutoRegession (AR) methods. The AutoRegressive Moving Average (ARMA) method [22] assumes a linear relationship between the lagged inputs and is applicable if the time series is stationary. If trends and seasonal characteristics are present, the time series is

---

[3]https://www.scopus.com/

[4] The applied search strings and keywords are documented in the Supporting Information.

[5]https://beallslist.net/

[6]The vectors that include past values can be sparse, i. e., only certain time points from k to H_1 are included.

non-stationary, and the requirements for applying ARMA are not fulfilled. To address this problem, the AutoRegressive Integrated Moving Average (ARIMA) method [22] removes time series trends through differencing and the seasonal ARIMA (sARIMA) method eliminates the seasonality by seasonal differencing [23].

The second representatives are Exponential Smoothing (ES) methods, where the forecast is determined by a weighted average of past observations, with the weights decaying exponentially with their age [1]. Simple Exponential Smoothing (SES) is a valid forecasting method for time series data without a trend or seasonal pattern. For time series with a trend, SES is adapted to Double Exponential Smoothing (DES), and Triple Exponential Smoothing (TES) is suitable for time series with seasonality. An extensive discussion of statistical forecasting methods can be found in reference [16].

Machine Learning Forecasting Methods   While most statistical forecasting methods are based on assumptions about the distribution of the time series data, machine learning methods have fewer restrictions in terms of linearity and stationarity [23]. In addition to statistical forecasting methods, which are specifically developed for time series forecasting, one can use regression methods based on machine learning to forecast multiple time points ahead using the following strategies, either solely or in combination [18]:

Recursive strategy: One trains a single regression model $f(\cdot)$ to forecast one time point ahead. In the operation, one recursively feedbacks the output value to the input for the next time point.

Direct strategy: One trains multiple independent regression models $f_h(\cdot), h = 1, \ldots H$, each to forecast the value at time $k + h$. The input is similar for each model.

Multiple output strategy: One trains a single regression model to forecast the whole horizon H at once. Consequently, the output is not a single value but a vector.

Representative machine learning methods are the Support Vector Regression (SVR), Decision Tree (DT)-based methods like the Gradient Boosting Machine (GBM), and Artificial Neural Networks (ANNs). Reference [17] gives an overview of machine learning and deep learning techniques applied to time series forecasting.

4   Data Pre-processing

Since most forecasting methods rely on assumptions about data properties, data pre-processing is of crucial importance. Data pre-processing includes anomaly detection and handling, transforming the time series to make it stationary, and scaling the time series. In the following sub-sections, automated methods for data pre-processing are introduced, and their utilization in forecasting pipelines is exemplified with the reviewed literature.

4.1   Anomalies

An anomaly is a value that significantly deviates from the rest of the time series [24]. Anomalies are induced by rare events or by errors in the data. Apart from anomalous existing values, which we call outliers, anomalies also comprise missing values in the time series. Both outliers and missing values can degrade the performance of forecasting methods or cause the training to fail. Therefore, appropriate anomaly detection and handling are necessary. Table 1 shows the summary of automated anomaly detection and handling methods used in the literature for time series forecasting pipelines.

Outlier Detection and Handling   Automated outlier detection and handling aim to identify abnormal values and replace them with plausible values without human intervention. Liu et al. [25] define an interval based on the global mean and the variance of the time series

$$[\text{mean}(y) - \alpha_t \cdot \text{var}(y), \ \text{mean}(y) + \alpha_t \cdot \text{var}(y)], \tag{2}$$

with the threshold α_t and the anomaly detection method considers values outside the interval as outliers. Detected abnormal values are automatically substituted with the arithmetical averages of the nearest previous and posterior normal values. However, anomalous values themselves bias the estimation of the mean and the variance, and the method is only valid for stationary time series. Other authors tackle this weakness by calculating the local median instead of the global mean. Martínez et al. [26], Yan [27], and Fan et al. [28] consider an observation as outlier if its absolute value is four times greater than the absolute medians of the three consecutive points before and after the observation. However, only extreme values above the absolute medians are detected, extreme values below the absolute medians are not identified. Widodo et al. [29] apply the Hampel method, which automatically replaces any value that deviates from the median of its neighbors by more than three Median Absolute Deviations (MAD) with that median value. Unlike previous methods, which use statistical measures, the anomaly detection and handling method of Maravall et al. [30] is based on a forecasting model. The method fits an ARIMA model, evaluates the MAD of the estimation residuals, and automatically replaces detected outliers with the forecast of the ARIMA model.

Table 1: Summary of automated anomaly detection and handling methods for data pre-processing in time series forecasting pipelines.

Ref.	Outlier Detection	Outlier Handling	Missing Value Handling
[25]	glob. mean-var. thres.	average nearest	
[26, 27]	loc. median thres.	average nearest	
[28]	loc. median thres.	average nearest	median imput.
[29]	loc. median-MAD thres.	median nearest	
[30]	ARIMA-MAD thres.	ARIMA values	
[31]	glob. median-robust-var. thres.	linear interpolation	copy-paste imput.

glob. global, imput. imputatio, loc. local, thresh. threshold, var. variance

Missing Value Handling   Automated missing value handling aims to reconstruct absent observations without human assistance. The method of Fan et al. [28] automatically replaces missing values in the time series with the median of 12 consecutive points before and after the observation. Yet, this method is prone to larger gaps of missing data. The method of Züfle and Kounev [31] automatically imputes missing values by multiplying the known value one season before or after the missing value by the trend factor estimated between the day of the missing values and the day copied. Using this procedure in chronological order allows the imputed values for the imputation of subsequent missing values. After the missing value imputation, the authors apply a similar outlier detection method like (2) with $\alpha_t = 3$. Unlike [25], Züfle and Kounev [31] use the robust standard deviation between the 1st and 99th percentile of the data and replace the outliers by linearly interpolating between the two nearest non-anomalous values.[7]

4.2 Stationarity

In a stationary time series $y[k]$, the statistical properties do not depend on the time of observation k, i. e., the distribution of $y[k, \ldots, k+s]$ is independent of k for all s [1]. Therefore, a time series with trends or seasonal patterns is not stationary because either the mean of the time series, its variance, or both change over time. Since some statistical forecasting methods assume a stationary time series, their application to non-stationary time series requires an appropriate transformation. Stationarity tests help to identify the type of non-stationarity and support the automatic selection of the appropriate transformation. Table 2 shows the summary of automated stationarity testing and transformation methods used in the literature for time series forecasting pipelines, introduced in the following.

Autocorrelation and Differencing Transformations   A first approach to automatically identify non-stationarities in time series is proposed by Tran and Reed [32] based on the AutoCorrelation Function (ACF) and the Partial AutoCorrelation Function (PACF). The ACF and the PACF visualize the correlation of a time series with a delayed copy of itself. The authors automatically detect decay patterns by calculating the average rate of change in the magnitude of high frequencies in the ACF and PACF, and consider rates of less than 10 percent as slow decay. The slow decay patterns indicate trends in the time series. To remove the trends, the time series is differenced by subtracting successive observations d times. After differencing, the ACF and PACF show significant peaks at regular intervals s, if the time series is seasonal. To remove the seasonality, the time series is seasonally differenced D times by subtracting observations separated by s. Note that the ARIMA forecasting method explicitly include differencing as hyperparameter d in the model structure and the sARIMA method additionally considers seasonal differencing with D and s.

Frequency Filters   Apart from the ACF and PACF, methods based on frequency filters are used to identify seasonality. Bauer et al. [33] use the periodogram to automatically retrieve all frequencies within the time series, iterate over the found frequencies, and match each frequency with reasonable frequencies (e. g. daily, hourly, and yearly) with tolerance to determine seasonal frequencies. Kourentzes and Crone [34] propose the Iterative Neural Filter (INF) to automatically identify seasonal frequencies. The filter distinguishes between stochastic and deterministic components and iteratively removes seasonalities, trends, and irregularities in the time series.

---

[7]The authors disaggregated the time series into the seasonal, trend, and residual components and applied anomaly detection and handling on the stationary residual.

Table 2: Summary of automated stationarity testing and transformation methods for data pre-processing in time series forecasting pipelines.

Ref.	Forecasting Method(s)	Stationarity Testing	Stationarity Transformation
[32]	sARIMA	ACF, PACF pattern analysis	diff., s. diff.
[33]	Telescope	periodogram	Box-Cox, STL
[34]	MLP, TES	INF	INF
[35]	MLP	ADF, INF	diff., INF
[36]	ARIMA	KPSS	diff.
[37]	AR, VAR	KPSS	diff.
[38]	autoARIMA	KPSS, Canova-Hansen	diff., s. diff.
[39]	sARIMA	ACF, PACF t-test	log, diff., s. diff.
[30]	ARMA, sARIMA	log-level, Kendall-Ord, Pierce, Lytras, seasonal frequency peaks	log, diff., s. diff.
[25]	ARIMA	ADF, log-level	diff., log
[40]	sARIMA	OSCB, KPSS, correlation, t-test, ADF	diff., s. diff., log
[41]	ARIMA, SETARMA	KS, KPSS	log, Box-Cox
[26]	kNN		diff., Box-Cox, STL
[42]	autoARIMA, Theta, Damped, RW, sRW, SES, DES, TES	Cox-Stuart	multiplicative decomposition
[29]	MKL		additive decomposition
[27]	GRNN	heuristic autocorrelation	diff., s. diff.
[43]	LSTM		diff., log, STL

s. seasonal, diff. differencing

Unit Root Tests   Statistical unit root tests are used to identify non-stationarities before applying transformation methods. The unit root tests used in the literature include the Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test [44] or the Cox-Stuart test [45] for testing if the time series is stationary around a deterministic trend, the Augmented Dickey-Fuller (ADF) test [46] for the existence of stochastic trends in the time series, and the Canova-Hansen test [47] or Osborn–Chui–Smith–Birchenhall (OCSN) test [48] for the existence of seasonal patterns over time. According to the trends and seasonalities detected in the unit root tests, the time series is automatically differenced or seasonally differenced, respectively.

Logarithm and Normality Transformations   Apart from unit root testing, Maravall et al. [30] and Liu et al. [25] propose log-level tests to automatically evaluate if a log-transformation of the time series is beneficial. Amin et al. [41] use the Kolmogorov–Smirnov (KS) test [49] to determine whether a time series is normally distributed – if not, the log-transformation is applied. Other authors apply the log-transformation without testing to reduce the variance [40, 43] or apply various transformations until critical values are satisfied in t-tests of ACF and PACF [39]. For achieving normality and stabilizing the variance, the Box-Cox transformation [50] is often applied without preceding testing, like in references [26, 31, 33, 41].

Time Series Decomposition   In addition to transformation operations to achieve stationarity, time series can be decomposed into the components trend, season, and irregular (i. e. the residual). Afterward, each component can be handled by an individual forecasting model. Prominent decomposition methods are the Seasonal and Trend decomposition using Loess (STL) [51] used in the references [26, 33, 43], and the additive or multiplicative decomposition applied by the authors of references [29, 42].

4.3 Scaling

The scale of the time series influences the performance of many forecasting methods based on machine learning. If the range of values is large, learning methods based on gradient descent may converge much slower or fail due to instability. Additionally, in the case of multiple inputs with different scales, the inputs with larger variance dominate the others in the calculation of many distance measures [52].

The general formalization for time series scaling is

$$y' = \frac{y - a}{b}. \tag{3}$$

Table 3 shows the scaling methods used in the literature on automated forecasting pipelines compared to the family of the forecasting method. A common approach is min-max scaling, where the time series are scaled in the range $[0, 1]$ with $a = \min(y)$, $b = \max(y) - \min(y)$. While min-max scaling guarantees that all time series are scaled to the same range $[0, 1]$, the Z-score normalization scales the time series to zero-mean and unit-variance with $a = \text{mean}(y)$, $b = \text{var}(y)$. Both scaling methods are sensitive to outliers and thus require a preceding anomaly detection and handling. Alternatively, robust scaling methods like *scikit-learn's RobustScaler*[8] can be used, which removes the median and scales the data according to the Inter-Quartile Range (IQR).

Table 3: Summary of scaling methods for data pre-processing applied in time series forecasting pipelines.

	Min.-max.	Zero-mean	Z-score
Statistical	[37]	[25]	[53]
Machine learning	[27, 34, 35, 54–63]		[29, 62, 64–66]

4.4 Discussion

We discuss data pre-processing as the first section of the automated forecasting pipeline, show possible problems, give recommendations, and suggest future research.

As shown in Table 2, automated stationarity testing and transformation methods are predominantly applied to pipelines using methods of the statistical forecasting family. In contrast, Table 3 shows that time series scaling is mainly used in pipelines with methods of the machine learning family. Both forecasting families require methods for automated anomaly detection and handling, shown in Table 1.

Regardless of their predominant use for automation, both transformation methods – for achieving stationarity and scaling – can be adversely affected by anomalies. Therefore, it is essential that automated anomaly detection and handling is performed before time series transformations. In automated outlier detection and handling, the main concern is that methods applied in the literature on automated forecasting pipelines focus on single outliers, although methods for consecutive outliers are widely available, e. g., [67]. Due to this focus, these methods may have a limited performance if several consecutive outliers or abnormal patterns are present. For automated forecasting pipelines, we thus recommend the first step of pre-processing to be detecting and removing isolated outliers, followed by detecting consecutive anomalous values – outliers or missing values – and handling them with an appropriate method. Ideally, the handling also considers domain-specific knowledge (e. g. [68]).
The second step of pre-processing, automated testing for stationarity and time series transformation, is only crucial if the pipeline uses a forecasting method that assumes a stationary time series. For each condition – deterministic and stochastic trends, seasonalities, and normality – only one test must be applied because the use of multiple tests may lead to conflicting answers [1]. For machine learning forecasting methods, however, these transformations are not mandatory. Yet, a stationary time series may reduce the required complexity of the machine learning method.
In the third step of pre-processing, we recommend using an appropriate time series scaling method to facilitate the training process of the respective forecasting method.

Given these recommendations, future work on automated forecasting pipelines should also consider domain knowledge to identify anomalies, e. g., if only positive values are valid or domain-adapted imputation to improve the reconstruction. Additionally, it should be systematically evaluated whether stationarity transformations improve the forecast accuracy of machine learning methods if applied in the automated forecasting pipeline.

5 Feature Engineering

A time series feature reflects the observations of an explanatory variable in the process being forecast (i. e. the target variable). Figure 1 shows feature engineering as a sub-section of the time series forecasting pipeline, consisting of extraction and selection of features. Table 4 guides the following literature analysis.

---

[8]https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html

Table 4: Summary of feature engineering methods in automated time series forecasting pipelines.

Ref.	Forecasting Method(s)	Data Domain	Feature Extraction lag	cyc.	tran.	exo.	Feature Selection filter	wrapper	embedded	Feature Aggregation
[69]	AR, autoARIMA, ETS, TBATS	several (5)	X	-	X	-				
[31]	GBM, RF	human access	X	X	-	X				
[33]	Telescope	several (5)	-	X	X	-	spec. value			
[70]	kNN	nature	X	-	-	-	frac. dim.			
[58]	GRNN	economics	X	-	-	-	PACF			
[27]	GRNN	energy	X	-	-	-		grid search		
[28]	ELM	several (3)	X	-	-	-		random search		
[71]	ANN, AR, RW	economics	X	-	-	-		FS		
[26]	kNN	economics	X	-	-	-		FS		
[72]	sARIMA	human access	-	-	-	X		MIQP		
[73]	SVR	energy	-	-	-	X		PSO		
[55, 56]	MLP	several (5)	X	-	-	-		EDA, DEA, GA		
[59]	MLP	several (5)	X	-	-	-			DEA	
[74]	SVR	energy	X	-	-	X			FS	
[57]	SVR	energy	X	-	-	X			BE	
[65]	GBM, LASSO, MLP, SVR, RF	energy	X	-	X	X	low var.	BO		
[29]	MKL	several (5)	X	-	-	-	ACF		kernel comb.	
[34]	MLP	human access	X	X	-	-	INF			
[66]	LASSO	economics	X	-	-	X		FS	shrinkage	
[75]	sARIMA	economics	-	-	-	X				PCA

cyc. cyc; tran. transformation, exo. exogenous, spec. spectral, frac. fractal, dim. dimension, var. variance, comb. combination

5.1 Feature Extraction

The feature space of the training data set contains for each explanatory variable a time series of the same resolution and length as the target variable.[9] Feature extraction aims at automatically enriching the feature space with additional explanatory variables. In the following, time series features are introduced, and their usage in automated forecasting pipelines is explored with literature references.

Lag Features   In autocorrelated time series, past observations can be valuable explanatory variables to forecast the target variable. Lag features provide values from prior time points for the forecasting method, i. e., at time point k, the model also processes values that date back a certain time horizon H_1. Lag features are useful if the target variable has inertia that is significantly reflected in the resolution of the time series or if exogenous influences affect the target variable in periodic patterns [65].

Table 4 reveals that lag features are the most applied type of features.

Cyclic Features   In the training data, the time stamps (e. g. (`YYYY-MM-DD hh:mm:ss`)) of the target variable are unique and specify the sequence of the observations. Cyclical patterns that humans can detect in this format, like the hour of the day, the day of the week, weekday and weekend, or month and year, cannot be processed by machine learning methods without encoding. Cyclic features provide this cyclic relationship by ordinal, interval, or categorical encoding. In the following, we exemplify these encodings for the day of the week (see Table 5). Ordinal encoding assigns an integer numerical value to individual days. For an interval encoding, one may utilize a periodical sine-cosine encoding to establish similarities between related observations, e. g., for encoding the day of the week, one adds two time series features that encode each weekday.[10] A categorical encoding is achieved through so-called one-hot encoding. In this example, we create a time series feature for each day of the week, being one on that day and zero otherwise. Since the last category – the Sunday – is already implicitly represented if each other categorical feature is zero, one may omit an explicit feature.

In the literature reviewed, cyclic features are not widely applied. Züfle and Kounev [31] apply Random Forest (RF) and GBM methods, using lag features and the cyclic features hour of the day, day of the week, and the exogenous feature holiday. However, the encoding of the cyclic features is not described. Sin-cos encoded cyclic features are used by Kourentzes and Crone [34] as input for a MultiLayer Perceptron (MLP) forecasting method.

---

[9]If the features are originally in a different resolution or length, they must be transformed accordingly.

[10]Two time series are necessary because, otherwise, the encoding would be ambiguous for several days.

Table 5: Exemplification of cyclical encoding methods for the day of the week of a time series.

	Ordinal	Sin-cos Interval		One-hot Categorical					
	x_dow	x_sin	x_cos	x_mon	x_tue	x_wed	x_thu	x_fri	x_sat
Mon	0	1.000	0.000	1	0	0	0	0	0
Tue	1	0.623	0.782	0	1	0	0	0	0
Wed	2	-0.223	0.975	0	0	1	0	0	0
Thu	3	-0.901	0.434	0	0	0	1	0	0
Fri	4	-0.901	-0.434	0	0	0	0	1	0
Sat	5	-0.223	-0.975	0	0	0	0	0	1
Sun	6	0.623	-0.782	0	0	0	0	0	0

Transformation Features   Time series transformations are widely used to make time series stationary, and can also be applied to extract explanatory variables that we term transformation features. These features include calculating time derivatives, the decomposition into trend, seasonality and residual, moving averages [69], as well as applying mathematical operations on existing features, e. g., multiplication of exogenous features [65].

Exogenous Features   In addition to features that are endogenously derived from the target variable, the forecast can be improved by using exogenous features if the target variable is subject to exogenous influences. In addition, lag, cyclical, and transformation features can also be extracted from exogenous features. Whether and which exogenous influences exist depends on the data domain.[11]

For example, energy data often depends on exogenous weather measures [57, 65, 73, 74], human access data underlies the influences of public holidays and weather [31, 72], and sales data often correlates with economic indicators [66, 75].[12]

5.2   Feature Selection

After automatically extracting several features, they typically undergo a selection to remove features that provide no additional information value, e. g., because of redundancy. In the following, methods for automated feature selection are presented, and their application in forecasting pipelines is explored based on the reviewed literature.

Filter Methods   Filter methods use metrics to rank single features or feature combinations and automatically select a promising feature set based on a threshold [78]. Hence, the filters rely on the general characteristics of the training data and are independent of the forecasting method and other subsequent sections in the forecasting pipeline.

The characteristics used for filtering are manifold. Chakrabarti and Faloutsos [70] propose an automated filtering method to determine the optimal lag features based on a threshold of the time series' fractal dimension. Martínez et al. [58] automatically select features by identifying significant lags in the PACF. Kourentzes and Crone [34] propose the INF method to identify seasonal frequencies in time series, which automatically selects lag and sin-cos encoded cyclic features. The method of Bauer et al. [33] automatically filters frequencies of the time series using the periodogram with the threshold of a spectral value greater than $50\,\%$ of the most dominant frequency. Additionally, cyclic features are extracted by calculating Fourier terms of the dominant frequencies. The most dominant frequency is used to decompose the time series into trend, season, and residual components by STL. Finally, the cyclic features and the seasonal component as a transformation feature are used to forecast the de-trended time series.

Wrapper Methods   In contrast to filter methods, the wrapper methods assess candidate features based on the forecasting performance on a validation data set. The best-performing feature set is selected by a search method, which tailors the feature set to the forecasting method or the entire forecasting pipeline, respectively. The search methods and performance metrics used in the literature are diverse. Independent from the chosen search method and metric, wrapper methods commonly take more computing time than filter methods, as training and validation require considerably more computing effort than calculating statistical measures. However, empirical studies show that the computing effort pays off in a better performance of the wrapper approach compared to filter methods [78].

---

[11]We classify the data domain according to the following categories: economics & finance, energy, nature & demographics, human access, and other or unknown. The categories are adapted from the references [76, 77] and extended to cover the literature reviewed.

[12]Authors that apply forecasting methods on data from several domains mainly use univariate time series from forecasting competitions, where no exogenous time series are provided.

In the literature reviewed, automated feature selection based on the performance of the forecasting pipeline is addressed by different methods. The method of Yan [27] and Fan et al. [28] automatically determines the optimal lag features based on the forecast error by searching over candidate lags. The candidate lags are either predefined individually (grid search) or drawn randomly between specified boundaries (random search). Balkin and Ord [71] and Martínez et al. [26] apply Forward Selection (FS) to automatically identify the optimal lag features. Firstly, several forecasting pipelines are trained for each individual lag feature. Secondly, the FS selects the best-performing lag feature and repeats the first procedure by adding another lag feature. It retains the combination with the highest improvement and repeats the procedure until the forecasting performance stops increasing. Besides search heuristics, optimization can be applied for feature selection. For automatically selecting exogenous features, Lowther et al. [72] adopt a Mixed Integer Quadratic Programming (MIQP) problem [79] and Son and Kim [73] apply Particle Swarm Optimization (PSO). Three evolutionary search strategies for automatically selecting the optimal lag features of MLPs are evaluated by Donate et al. [55, 56], where the Estimation Distribution Algorithm (EDA) yields the best convergence speed and the lowest forecast error.

Embedded Methods   Embedded methods integrate the feature selection into the training process of the forecasting method [78].[13] During the training, the embedded feature selection commonly estimates the feature importance and weights the features accordingly. Embedded methods require less computing effort than wrapper methods, but more than filter methods [65].

The embedded methods in the literature are specific to the forecasting method. The approach of Panigrahi and Behera [59] automatically determines the optimal lag features by a Differential Evolution Algorithm (DEA). Instead of using gradient-based methods for training an MLP, the authors integrate the weight estimation into the DEA, aiming to increase the convergence speed. Valente and Maldonado [74] consider the automated selection of lag and exogenous features by an FS embedded into the SVR training process. The FS is based on a contribution metric that takes into account lags whose inclusion minimizes the metric. An automated Backward Elimination (BE) for SVR using embedded kernel penalization is described by Maldonado et al. [57]. Lag and exogenous features that are irrelevant for the forecasting performance are successively removed during training.

Hybrid Methods   Hybrid methods aim to combine the advantages of the above methods.

Rätz et al. [65] evaluate several filter, wrapper, embedded, and hybrid feature selection methods. Based on their experiments, they propose to use a filter method to automatically remove all features with low variance in the first step. Afterward, they apply Bayesian Optimization (BO) to assess the remaining feature candidate combinations, consisting of lag, transformation, and exogenous features based on the forecasting performance. The approach of Widodo et al. [29] filters significant lags using ACF, associates the remaining lag features with a kernel and automatically assigns appropriate weights to the kernels during training of the Multiple Kernel Learning (MKL) method. Sagaert et al. [66] firstly filter candidate lag features using automated FS, similar to the stepwise search for the automated ARIMA design of Hyndman and Khandakar [38]. Then, the identified set of lag features together with exogenous features are used as inputs of the embedded Least Absolute Shrinkage and Selection Operator (LASSO) method that automatically selects features by shrinking the coefficients of irrelevant ones.

5.3   Feature Aggregation

Feature aggregation aims to transform the feature space into a low-dimensional representation while retaining the primary properties of the time series. The transformation is advantageous in high-dimensional feature spaces to reduce processing time and avoid the curse of dimensionality.[14] The Principal Component Analysis (PCA), e. g., maps the data to a lower-dimensional space, maximizing the variance in the lower-dimensional representation.[15]

Dellino et al. [75] pre-whiten the time series data with a PCA, i. e., the PCA aggregates exogenous features and discards the principal components with a low variance that are assumed as noise.

5.4   Discussion

We discuss feature engineering as the second section of the automated forecasting pipeline, highlight a potential issue, provide recommendations, and suggest future work.

---

[13]Since the embedded feature selection is specifically designed for the training algorithm of the respective forecasting method, the transfer to other forecasting methods is not straightforward.

[14]As the dimensionality of the feature space increases, the available training data becomes sparse.

[15]Since the PCA assumes a normal distribution, anomalies must be identified and handled beforehand and appropriate transformations need to be performed to obtain a stationary time series [80].

As shown in Table 4, most automated pipelines that apply forecasting methods based on machine learning rely on lag features because – unlike statistical forecasting methods – they do not consider time lags implicitly.[16] The sparse use of cyclic and transformation features can be explained because most of the analyzed forecasting pipelines already consider this information by automatically selecting appropriate lags.

In terms of automation, the majority of the literature combines the extraction of predefined features with an automated selection. Regarding feature extraction, the primary concerns remain in the human-defined feature extraction since it requires experience and may be biased. For this reason, extracting a large set of default features – including lag, cyclical, and transformation features – for the automated forecasting pipeline can be valuable since the subsequent automated feature selection removes irrelevant ones. Additionally, exogenous features are a powerful opportunity to integrate domain-specific knowledge into the forecasts.[17]

In the automated feature selection, all four methods – filter, wrapper, embedded, and hybrid – are useful for removing irrelevant features. While an embedded method is computationally beneficial by integrating the feature selection in the training process, it is specifically designed for a forecasting method. Because of its independence from the forecasting method, we recommend combining a filter with a wrapper method (hybrid) to initially reduce the candidate features through filtering and then tailoring the feature selection to the forecasting method or pipeline, respectively.

Since aggregated features often correlate with other features and the number of features can be reduced by an automated feature selection method, we do not consider automated feature aggregation necessary to reach a high forecast accuracy. Moreover, the aggregation of features limits their interpretability regarding their information value for the forecast.

Based on these challenges, future work towards automated forecasting pipelines should consider the automated extraction of default endogenous features[18], extended by a domain-specific extraction to include exogenous features.

6   Hyperparameter Optimization

Forecasting methods incorporate a wide range of hyperparameters about the forecasting model's structure, training regularization, and algorithm setup; parameters whose values are not directly derived from the data and must be selected by the data scientist. The hyperparameter configuration λ includes all considered hyperparameters and their selected values [82]. By tailoring λ to the specific problem using HPO, one may improve the model performance over the default setting of common forecasting libraries [65].

6.1   Performance Metrics and Validation Sample

Most HPO methods assume that the performance of the model is quantifiable. To evaluate the model performance in time series forecasting, measures from information theory and error measures are used. Information Criteria (IC) measure the amount of information lost by a statistical model, taking into account the Goodness-Of-Fit (GOF) and the model complexity. The less information a model loses, the higher the model performance. One IC is the Akaike Information Criterion (AIC)

$$\text{AIC} = 2w - 2\ln\left(\hat{L}\right) \tag{4}$$

with the number of estimated parameters w and the model's maximum value of the likelihood function $\hat{L}$. It rewards GOF but penalizes high numbers of model parameters. The penalty is required since adding more parameters may increase the likelihood without being justified by the data (overfitting). Another popular IC is the Bayesian Information Criterion (BIC)

$$\text{BIC} = w\ln\left(K\right) - 2\ln\left(\hat{L}\right) \tag{5}$$

that additionally considers the number of data points K.

Popular error measures are the Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{N}\sum_{n=1}^{N}\left(y[n] - \hat{y}[n]\right)^2, \tag{6}$$

---

[16]A well-known exception is Recurrent Neural Networks (RNNs) that feedback input values within the neural structure to capture sequential relationships.

[17]For multiple-point-ahead forecasts, future values of the exogenous time series are required if lag, cyclical, and transformation features are extracted from them; either through simultaneous forecasting or the integration of exogenous forecast results.

[18]Initial work on the automated extraction of endogenous features exists, e. g., [69, 81].

and the Mean Average Error (MAE)

$$\text{MAE} = \frac{1}{N} \sum_{n=1}^{N} |y[n] - \hat{y}[n]|, \tag{7}$$

where $\hat{y}_i$ is the forecast and y_i the realized value.[19] The smaller the error of the model, the higher the model performance. Error measures can be calculated *in-sample* or *out-of-sample*. In-sample means that the forecast error is determined on data points that are part of the training data sample, while out-of-sample uses unseen points from the validation data sample. To increase the robustness, cross-validation can be performed by splitting the data several times into different training and validation sub-sets and averaging the validation error across the folds. The out-of-sample error validation is generally considered to be a more trustworthy empirical evidence, as in-sample error validation is prone to overfitting.[20]

6.2   Optimization Methods

Given validation data and metrics, the hyperparameter configuration of forecasting methods can be optimized using different optimization methods. Table 6 shows the summary of HPO methods for time series forecasting. In the following sub-sections, automated HPO methods are introduced that are applied in the literature on forecasting pipelines.

Table 6: Summary of HyperParameter Optimization (HPO) methods in automated time series forecasting pipelines.

Ref.	Optimized Forecasting Method(s)	Optimized Hyper-parameters	Performance Metric	Validation Sample in-sample	out-of-sample	Optimization Method
[32]	sARIMA	p, q		-	-	ACF & PACF
[41]	ARIMA, SETARMA	p, q		-	-	testing, ACF & PACF, checking
[37]	AR, VAR	p, s	BIC	X	-	grid search, checking
[83]	MA	q	user-defined	X	-	grid search
[36]	ARIMA	p, d, q	user-defined	X	-	testing, ACF & PACF, grid-search
[84]	ARIMA	p, d, q	GOF	X	-	grid search, checking
[38]	sARIMA	p, d, q, P, D, Q, s	AIC	X	-	testing, two-stage grid search
[85]	sARIMA	p, d, q, P, D, Q	user-defined	X	X	testing, grid search, checking
[86]	sARIMA	p, d, q, P, D, Q, s	AIC	X	-	var. minimization, two-stage grid search
[72]	sARIMA	p, d, q, P, D, Q, s	user-defined	X	X	two-stage MIQP & grid search
[53]	sARIMA	p, d, q, P, D, Q	MAE	-	X	BO GP
[87]	ETS	E, T, S	AIC	X	-	grid search
[88]	TES	$\alpha, \beta, \gamma, \varphi, p$	RMSE, MAD MAPE	X	-	multiobjective NLP
[89]	Theta	φ T, S	MAE	X	-	Brent testing, grid search
[90]	UC	T, S, I	BIC	X	-	grid search
[70]	kNN	k		-	-	heuristic
[66]	LASSO	λ	MAPE	-	X	grid search
[73]	SVR	C, γ	RMSE	-	X	grid search
[57, 74]	SVR	C, γ	MAPE	-	X	grid search
[29, 62]	MKL	C, ϵ kernel, γ	sMAPE	-	X	grid search embedded
[27]	GRNN	spread		-	-	heuristic
[59]	MLP	N_i, N_h	RMSE	X	X	DEA
[56]	MLP	$N_i, N_h, \alpha, \Delta_{max}$	MSE	-	X	EDA
[55]	MLP	N_i, N_h, α, s	MSE	-	X	GA, DEA, EDA
[28]	ELM	N_i, N_h	sMAPE	-	X	random search
[43]	LSTM	epoch-size, batch-size, N_h, α, epochs, noise, L2	MASE	-	X	BO GP
[60]	ANN, DBN, SVR	method-dependent	MSE	-	X	PSO
[65]	GBM, LASSO, MLP, RF, SVR	method-dependent	MAE	-	X	BO TPE

---

[19]There are further error measures that are derived from the MSE and the MAE, such as the Root MSE (RMSE), the Mean Absolute Scaled Error (MASE), the Mean Absolute Percentage Error (MAPE), and the symmetric MAPE (sMAPE). For their definitions, we refer to reference [1].

[20]Since ICs try to prevent overfitting implicitly with the penalty term, they are computed in-sample.

6.2.1   Grid Search and Random Search

The most elementary method for HPO is the exhaustive *grid search*, where the data scientist defines a finite set of $\pi = 1, \ldots, \Pi$ hyperparameter values to be evaluated, resulting in a full factorial configuration space $\mathbf{\Lambda} \in \mathbb{R}^{\Pi}$. As the grid search evaluates the Cartesian product of these sets, the number of computations B grows exponentially with the dimensionality of $\mathbb{R}^{\Pi}$. Hence, increasing the discretization resolution increases the computing effort substantially [82]. The *random search* [91] is an alternative to the grid search. It irregularly samples the hyperparameter set until a certain number of computations B is exhausted. The random search may performs better than the grid search if some hyperparameters are much more important than others.[21] Figure 3 shows a comparison of both methods with two hyperparameters and an equal number of computations B.
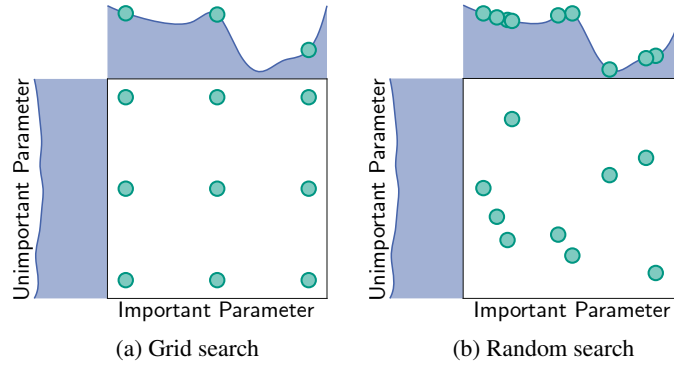

Figure 3: Comparison of a *grid search* and a *random search* for minimizing a function with one important and one unimportant parameter [82, 91].

Hyndman et al. [87] generalize the formulation of ES forecasting methods with the Error Trend Seasonality (ETS) method and suggest a grid search that automatically selects the hyperparameter configuration with the lowest in-sample AIC. Utilizing grid search for HPO is also applied to statistical forecasting methods based on AR and Moving Averages (MAs) averages. Sekma et al. [37] optimize the hyperparameters of an AR (p, s) and a Vector AutoRegression (VAR) (p, s) method, respectively, using grid search. The hyperparameter configuration resulting in the minimal BIC, calculated in-sample, is automatically selected. A similar HPO method for MA (q) methods is proposed by Svetunkov and Petropoulos [83], where the data scientist needs to define the in-sample optimization criterion. An advanced method, combining pre-processing and HPO of ARIMA (p, d, q) methods is proposed by Alzyout et al. [36]. Firstly, a stationarity test determines the differencing order d. Secondly, the maximal AR-lag order p_{max} and the maximal MA-lag order q_{max} are determined by automatically evaluating the PACF and the ACF, respectively. Finally, a grid search from 0 to p_{max} and 0 to q_{max} selects the optimal hyperparameter configuration based on in-sample AIC or BIC. Combining pre-processing and HPO is also proposed by Hyndman and Khandakar [38]. Firstly, the authors propose to automatically determine the differencing and seasonal differencing order d and D and the seasonal period s with a stationarity test. Secondly, instead of an exhaustive grid search over the hyperparameters p, q, P, and Q of the sARIMA $(p, d, q)(P, D, Q)_s$, they propose a two-stage grid search, reducing the number of evaluations. In the first stage, four candidate hyperparameter configurations are evaluated, whose hyperparameters depend on the previously determined s. The hyperparameter configuration with the smallest in-sample AIC value proceeds to the second stage, where p, q, P, and Q are varied ± 1. If a configuration with a lower AIC value is found, it becomes the active configuration of stage two and the variation is repeated until the AIC stops improving. Pedregal et al. [86] adopt this method except for the pre-processing step. Instead of stationarity tests, the variance of the time series is minimized to identify the differencing orders d and D.

Grid search is also applied for HPO of forecasting methods based on machine learning. Sagaert et al. [66] determine the optimal shrinkage factor λ of the LASSO method for embedded feature selection in terms of the mean MAPE of a 10-fold cross-validation (out-of-sample). Several authors apply a grid search to determine the optimal hyperparameter configuration of an SVR. Son and Kim [73] optimize the hyperparameters C and γ based on the RMSE, Maldonado et al. [57] and Valente and Maldonado [74] based on the MAPE. The grid search used by Widodo et al. [29, 62] is based on a 5-fold cross-validation, using the mean sMAPE value across folds as the forecast error measure. They optimize the hyperparameters C and ε of the SVRs used in a MKL approach with embedded feature selection. Combining automated feature selection and HPO is also suggested by Fan et al. [28]. The authors identify the optimal lag features using random search and link the number of hidden neurons N_h to the number of input neurons N_i.

---

[21]The greater importance of a few hyperparameters over others applies in many cases, e. g., [65, 91].

6.2.2 Bayesian Optimization

Rather than evaluating a finite search grid, BO explores and exploits the configuration space $\mathbf{\Lambda} = \Lambda_1 \times \Lambda_2, \dots, \Lambda_H$ of H hyperparameters Λ. BO uses a probabilistic surrogate model to approximate the objective function $\mathcal{Q}$ that maps the forecasting pipeline's performance Q on $\mathbf{\Lambda}$. More specifically, an observation $Q(\lambda)$ of the objective function reflects the pipeline's performance with a particular hyperparameter configuration $\lambda \in \mathbf{\Lambda}$ of the forecasting methods used. In each iteration, the optimization updates the surrogate model with the new observation and uses an acquisition function to decide on the next hyperparameter configuration $\lambda \in \mathbf{\Lambda}$ to be explored (see Figure 4). The acquisition function trades off the exploration against the exploitation of $\mathbf{\Lambda}$ by determining the expected benefit of different hyperparameter configurations using the probabilistic distribution of the surrogate model [82].
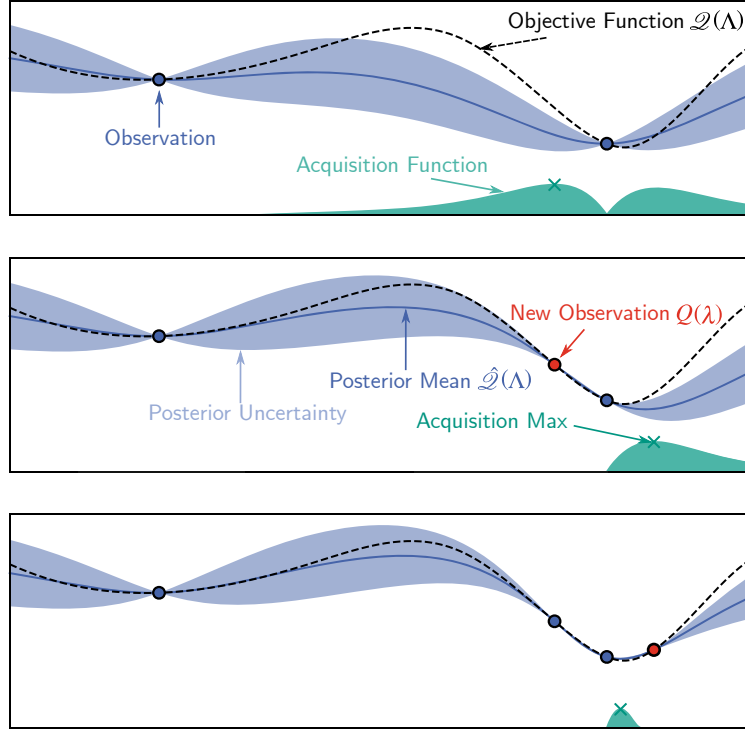

Figure 4: Two exemplary iterations of a Bayesian Optimization (BO) on a 1D function. The BO minimizes the predicted objective function $\hat{\mathcal{Q}}(\lambda)$ (blue line) by maximizing the acquisition function $\mathbb{E}\left[I(\lambda)\right]$ (green surface). The acquisition value is high where the value of $\hat{\mathcal{Q}}(\lambda)$ is low, and its predictive uncertainty (light blue interval) is high [82]. The true objective function (dashed line) might lie outside of the predicted uncertainty interval.

For surrogate modeling, various approaches exist, ranging from Gaussian Processes (GPs) and their modifications to machine learning approaches, e. g., RFs or Tree-structured Parzen Estimators (TPEs). Feurer and Hutter [82] recommend using a GP-based BO for configuration spaces with real-valued hyperparameters and computationally expensive training, and an RF or TPE-based BO for configuration spaces with categorical hyperparameters and conditions, e. g., the choice of a forecasting method and its conditional (sub-)configuration space.

Dellino et al. [53] apply a BO based on GP surrogate modeling to optimize the hyperparameters of the sARIMA $(p, d, q)(P, D, Q)$ using an out-of-sample validation data set and the MAE. They compare the BO to an exhaustive grid search, where the BO achieves lower forecast errors but requires more computing time. In their experiment, however, the comparison is unequal as the configuration space of the BO is larger, and the best-performing model of the BO results in a hyperparameter configuration that the grid search does not evaluate.
Bandara et al. [43] use a BO with a GP surrogate model to optimize the recurrent neural architecture and training hyperparameters of a Long Short-Term Memory (LSTM). The performance of each hyperparameter configuration is evaluated out-of-sample based on the MASE. Rätz et al. [65] optimize multiple forecasting methods, combining feature selection, HPO, and forecasting method selection using a BO based on a TPE surrogate model. The hyperparameter configurations' performances are estimated using the mean MAE of a 5-fold out-of-sample cross-validation.

6.2.3 Non-linear Programming

Mathematical programming can be applied for HPO if calculating the performance metric is solvable in closed-form. Non-Linear Programming (NLP) solves an optimization problem, where at least one of the objective functions or constraints is a non-linear function of the decision variables. The objective function may be convex or non-convex, where non-convex NLP incorporates multiple feasible regions and multiple locally optimal solutions within them [92]. Depending on the formulation of the objective function and its constraints, different solving methods are appropriate.

Bermúdez et al. [88] apply the generalized reduced gradient method to solve a multiobjective NLP. They jointly minimize the in-sample RMSE, MAPE, and MAD to determine the smoothing parameters $\alpha, \beta, \gamma, \varphi$ and the number of periods of the seasonal cycle p of the TES method. Lowther et al. [72] use MIQP to select suitable exogenous features for ARIMA. They combine this selection with a grid search over the sparsity parameter k of the MIQP formulation and the sARIMA hyperparameters p, d, q, P, D, Q.

6.2.4 Heuristics

A heuristic is an informed search technique that systematically explores a configuration space $\mathbf{\Lambda}$ subject to a constant search rule [93].

Tran and Reed [32] propose heuristics based on the ACF and PACF to determine the AR and MA lag order p and q. A similar approach is published by Amin et al. [41]. To determine the transformation parameter θ of the Theta forecasting method, Spiliotis et al. [89] apply the Brent–Dekker method – a root-finding method. They determine the optimal θ for eight different trend T and season S configurations, and select the configuration that minimizes the in-sample MAE. Chakrabarti and Faloutsos [70] propose a heuristic to specify the number of nearest neighbors k of the k-Nearest Neighbors (kNN) forecasting method after selecting the optimal lag features. A training sample-related heuristic for determining the spread factor of the General Regression Neural Network (GRNN) is introduced by Yan [27].

6.2.5 Metaheuristics

Metaheuristics are strategies for guiding a search according to feedback from the objective function, previous decisions, and prior performance [94], i. e., the searching behavior changes while exploring the configuration space $\mathbf{\Lambda}$. Metaheuristics do not require assumptions about the objective function and can solve optimization problems where gradient-based methods fail.

Evolutionary Optimization   Evolutionary Algorithms (EAs) comprise a wide range of population-based metaheuristics inspired by biological evolution [95]. A population of candidate hyperparameter configurations is evaluated using a fitness function to determine the performance of solutions. Weak solutions drop out, while well-performing solutions evolve. The mechanisms of selection and evolution differ between algorithms.

Genetic Algorithms (GAs) evolve a population of candidate hyperparameter configurations to explore and exploit the configuration space $\mathbf{\Lambda}$. The hyperparameters of a candidate solution are encoded as genes in a chromosome. In each generation, the fitness of the population is evaluated, and the chromosomes of individual candidates are modified to create a new generation – the offspring. The modification includes recombination and mutation and depends on an individual candidate's fitness. A part of the population is retained and forms with the offspring the next generation. DEAs differ from GAs in the mechanism of generating the offspring. While in GAs an individual acts as parent to generate an offspring, the DEA adds the weighted difference between two chromosomes to create a new individual. In this way, no separate probability distribution is required, making the algorithm self-organizing. In EDAs, the population is replaced by a probability distribution over the choices available at each position in the chromosome of the individuals. A new generation is obtained by sampling this distribution, avoiding premature convergence and making the representation of the population more compact.

Donate et al. [55] evaluate a GA, a DEA, and an EDA for optimizing the hyperparameter configuration of an MLP, including the number of input and hidden neurons N_i and N_h, as well as the training hyperparameters learning rate α and the weight initialization seed s. The results of the experiments show that the DEA and the EDA require more than 100 generations to improve significantly over GA. After 200 generations, the EDA achieves the lowest forecast error, followed by the DEA and the GA. In a later publication [56], the authors adapt the chromosome encoding and replace s with the hyperparameter Δ_{max} of the used training algorithm. In both publications, the fitness of each individual is evaluated by calculating the MSE on an out-of-sample validation data set. Panigrahi and Behera [59] apply a DEA to optimize N_i and N_h of an MLP, combining in-sample and out-of-sample validations. The fitness of each individual (RMSE) is calculated in-sample, and the DEA is terminated when the RMSE on the validation data set increases[22], indicating the beginning of overfitting.

---

[22]This regularization is also called *early stopping*.

Particle Swarm Optimization   In PSO, a population of hyperparameter candidate configurations – the swarm – is evaluated. The candidates move through the configuration space Λ, where the movement of the swarm is guided by the best-performing candidates so far.

Sergio et al. [60] apply PSO to optimize the hyperparameter configuration of multiple forecasting methods, combining the best hyperparameter configurations afterward to an ensemble.

6.3   Diagnostic Checking

Diagnostic checking evaluates the fitted forecasting model against criteria that indicate an adequate forecasting method configuration. An adequate forecasting method configuration yields residuals $r[k] = y[k] - \hat{y}[k]$ with properties of white noise, i. e., the residuals are not autocorrelated having zero mean and finite variance [1]. For statistical forecasting methods, the relationship between dependent and independent variables should be statistically significant. That is, the parameters estimated in the fitting process describing this relationship are significantly different from zero. If the forecasting method makes assumptions about the time series characteristics (e. g. stationarity), it is advisable to check whether the assumptions hold [1].

Amin et al. [41] check the randomness of the residuals with a Box-Pierce test and the significance of the parameters of the ARIMA or Self Exciting Threshold AutoRegressive Moving Average (SETARMA) models with a t-test. Furthermore, they test the invertibility and the stationarity, i. e., both the sum of the AR parameters and the sum of the MA parameters have to be smaller than one. Analogously, Hwang et al. [84] verify stationarity and invertibility. For residual diagnostics, they additionally check the residual's correlations with the Ljung-Box test. Similar to the above authors, Sekma et al. [37] test the parameters' significance and check if the residuals are white noise with the Ljung-Box test [96].

6.4   Discussion

We discuss HPO as the third section of the automated forecasting pipeline, identify a possible problem, give recommendations, and suggest future research.

For HPO, most automated forecasting pipelines apply grid search as shown in Table 6. Directed search methods, e. g., evolutionary optimization and BO, are, however, used for HPO of forecasting methods with high computational training complexity, primarily including machine learning methods. After an HPO, diagnostic checking is only applied sporadically and only for statistical forecasting methods.

With regard to automation, several authors link HPO and the preceding automated steps of pre-processing (Section 4) and feature engineering (Section 5). One potential problem is the optimization of the differencing orders d and D of the ARIMA (p, d, q) and sARIMA $(p, d, q)(P, D, Q)_s$ methods using IC metrics. The differencing transformation affects the likelihood in IC metrics, making the metrics between different values of d and D not comparable [1]. Therefore, d and D should be determined in the pre-processing section. For the subsequent HPO, we assume a straightforward grid search to be sufficient since the computational training complexity of ARIMA and sARIMA methods is rather low, and the configuration space is small. In contrast, for HPO of forecasting methods with high computational training complexity, e. g., ANNs, and large categorical and conditional configuration spaces, we recommend a BO based on TPE.

Given these recommendations, for automated forecasting pipelines in future work, we suggest that the automated analysis of residuals is also integrated into the pipeline using machine learning-based forecasting methods.

7   Forecasting Method Selection and Ensembling

Not only optimizing hyperparameters of a forecasting method but also selecting the appropriate method is crucial for the forecast accuracy. Consequently, the forecasting method selection is often combined with an individual HPO.[23] Forecast ensembling aims to bundle the forecasts of several methods, thereby reducing the impact of occasional poor forecasts – which can even occur with the best-selected and optimally configured forecasting method.

7.1   Forecasting Method Selection

For automatically selecting the best-performing forecasting method, there are several approaches that we divide into heuristic, empirical, and decision model-based selection. The selection is based on experience, a determined

---

[23]Selecting the optimal forecasting method and finding the optimal hyperparameter configuration is also called the Combined Algorithm Selection and Hyperparameter optimization (CASH) problem.

Table 7: Summary of method selection and ensembling methods in automated time series forecasting pipelines.

Ref.	Forecasting Method(s)	Forecasting Method Selection	Selection Basis		Forecast Ensembling	
			pool	ensemble		
[26]	kNN		-	-	use all	avg.
[97]	ANN, MA, sRW	heuristic	-	-		
[98]	autoARIMA, ETS, LR, RW, sRW, Theta	heuristic	-	-	use all	wgt.
[71]	ANN, AR, RW	empirical	X	-		
[41]	ARIMA, SETARMA	empirical	X	-		
[37]	AR, VAR	empirical	X	-		
[99]	ARIMA DES, LR, dRW, SES, TES	empirical	X	-		
[31]	GBM, RF, Telescope	empirical	X	-		
[65]	GBM, LASSO, MLP, RF, SVR	empirical	X	-		
[42]	autoARIMA, Damped, RW, sRW, SES, DES, TES, Theta	empirical	X			
[35]	MLP	empirical	X	-	k-best	avg.
[100]	ETS	heuristic, empirical	X	-	k-best	avg.
[101]	autoARIMA, ETS, NNetAR, TBATS	empirical	X	-	k-best	wgt.
[102]	ELM, ENN, FNN, GRNN, MLP, RBFNN	empirical	X	-	k-best	wgt.
[60]	DBN, MLP, SVR	heuristic, empirical	X	-	dyn. best	avg., wgt.
[103]	autoARIMA, MA, SES, DES, TES, Theta	decision model	X	-		
[104]	SES, DES, TES	decision model	X	X		
[62]	autoARIMA, MKL, PR, SVR	decision model	-	X		
[105]	ANN, autoARIMA, ETS, LC, LL, RW	decision model	-	X		
[106]	DT, GBM, LR, RF	decision model	-	X		
[107]	ANN, GP, MARS, PR, RBFNN, SVR	decision model	-	X		
[108]	autoARIMA, BSTS, ETS, GBM, Prophet, TBATS, Theta	decision model	-	X		
[54]	Telescope: Cubist, Evtree, GBM, NNetAR, RPaRT, SVR	decision model	-	X		
[43]	LSTM	decision model	-	X		
[109]	AR, autoArima, ETS, NNetAR, RW, dRW, sRW, TBATS, Theta	decision model	-	X	use all	wgt.
[110]	autoARIMA, ETS, NNetAR, RW, dRW, sRW, STL-AR, TBATS, Theta	decision model	-	X	use all	wgt.
[64]	autoARIMA, ELM, ETS, GBM, LR, RF, SVR	decision model	-	X	use all	wgt.
[63]	autoARIMA, ETS, NNetAR, RW	decision model	-	X	k-best	wgt.
[111]	MA, IMA, White Noise	decision model	X	X		

avg. averaging, dyn. dynamic, wgt. weighting

performance, or meta-features. The determined performance reflects IC or error measures. Meta-features describe properties of the time series to be forecast and provide meta-information, including statistical characteristics of the target time series, such as the skewness, kurtosis, and self-similarity; and domain information, such as physical properties of the system and environmental characteristics. In the following, methods for automated forecasting method selection are presented, and their application in forecasting pipelines is examined based on the reviewed literature, guided by the summary in Table 7.

Heuristic Forecasting Method Selection   The heuristic selection of the forecasting method relies on fixed rules. The basis of these rules is experience and statistical tests that examine the time series for certain characteristics. Therefore, heuristic selection requires neither a determined performance nor meta-features.

Shcherbakov et al. [97] propose decision rules that consider the amount of available training data. For small amounts of training data (i. e. less than 672 observations), a naïve method is selected that uses the previous day's value as the forecast. For moderate amounts of training data (i. e. 672-2688 observations), a MA method is applied, whereas an ANN is selected for greater amounts of training data (i. e. more than 2688 observations). Besides the amount of training data, the availability of calendar information and exogenous time series also determines the chosen forecasting method.

Empirical Forecasting Method Selection   The empirical forecasting method selection determines the performance of several forecasting methods during training (in-sample) or on a validation data set (out-of-sample) and automatically selects the best-performing forecasting method.[24]

---

[24]For a detailed description of the in-sample and out-of-sample validation, refer to Section 6.

Balkin and Ord [71] train an AR, an MLP, and a naïve RW method, and select the forecasting method with the smallest in-sample BIC. Similarly, Sekma et al. [37] decide between AR and VAR forecasting methods based on the smallest in-sample BIC; Amin et al. [41] use the AIC to choose between ARIMA and SETARMA.

An out-of-sample validation is used by Pereira et al. [99]. They calculate the MAE of six forecasting methods on a validation data set and select the method with the lowest MAE. A similar selection strategy is used by Züfle and Kounev [31]. They select the best-performing candidate forecasting method in terms of the R^2 score on validation data. Robustness can be increased by cross-validation. Rätz et al. [65] combine feature selection, HPO, and the selection of the optimal forecasting method using BO with the mean MAE over five folds.

The effectiveness of in-sample and out-of-sample empirical forecasting method selection is evaluated by Fildes and Petropoulos [42]. They assess the following four empirical selection methods: i) minimal one-point-ahead in-sample MSE, ii) minimal one-point-ahead out-of-sample MAPE, iii) minimal h-point-ahead out-of-sample MAPE, and iv) minimal 1-18-points-ahead out-of-sample MAPE. The latter method proves to be better than the 1-point-ahead validation and even than adjusting the validation to the corresponding forecast horizon H.

Decision Model-based Forecasting Method Selection   Instead of a heuristic or empirical forecasting method selection, one may train a decision model to automatically select the optimal forecasting method. As decision models, regression, classification, and clustering methods can be used to establish a relationship between performance information or meta-features and the optimal forecasting method.

In a decision model, the selection and aggregation of meta-features can improve decision accuracy. The principle of feature selection and aggregation methods corresponds to the descriptions in Section 5. Similar to the optimization of forecasts, the decision model can also be improved by HPO. Table 8 gives an overview of meta-feature engineering and decision models applied in the literature.

Table 8: Summary of meta-feature selection and aggregation methods for decision models to select forecasting methods.

Ref.	Forecasting Method(s)	Meta-Feature		Decision Model		
		selection	aggregation	method	type	HPO
[103]	autoARIMA, MA, SES, DES, TES, Theta			LogR, SVM, DT	class.	
[104]	SES, DES, TES	grid search		MLP	class.	grid search
[62]	autoARIMA, MKL, PR, SVR	FS		kNN	class.	
[105]	ANN, autoARIMA, ETS, LC, LL, RW		PCA	LDA	class.	
[106]	DT, GBM, LR, RF	BE	autoencoder	MLP, RF	class.	grid search
[107]	ANN, GP, MARS, PR, RBFNN, SVR	PCA, Pearson	SVD	MLP	class.	grid search
[108]	autoARIMA, BSTS, ETS, GBM, Prophet, TBATS, Theta	use all		RF	class.	grid search
[54]	Telescope: Cubist, Evtree, GBM, NNetAR, RPaRT, SVR	use all		RF	reg., class.	
[43]	LSTM	use all		k-means, DBSCAN, Snob	clust.	embedded
[109]	AR, autoArima, ETS, NNetAR, RW, dRW, sRW, TBATS, Theta	use all		GBM	class.	BO
[110]	autoARIMA, ETS, NNetAR, RW, dRW, sRW, STL-AR, TBATS, Theta	learning		GBM	class.	not described
[64]	autoARIMA, ELM, ETS, GBM, LR, RF, SVR	learning		CNN, FCNN	class.	grid search
[63]	autoARIMA, ETS, NNetAR, RW	use all		LR	reg.	
[111]	MA, IMA, White Noise	grid search		SVM	class.	grid search

class. classification, clust. clustering, reg. regression

Taghiyeh et al. [103] propose a decision model-based selection method that relies on a classification method. It selects the optimal forecasting method from a pool of candidates based on their in-sample and out-of-sample MSE. None of the three classification methods, including Logistic Regression (LogR), DT, and Support Vector Machine (SVM), outperforms the others. Kück et al. [104] propose a decision model-based selection based on the out-of-sample sMAPE

and meta-features. They apply an MLP classifier as decision model and select its inputs using a grid search over 127 feature sets that include error measures and meta-features.

The approaches above have the disadvantage that all candidate forecasting methods must be trained on the target data set to determine the applied error measures. Computing meta-features, in contrast, does not require training. Hence, relying only on meta-features for decision model-based selection saves computing time. Widodo and Budi [62] propose a kNN classifier to select the forecasting method for a target time series based on the meta-features introduced in reference [112]. In the design of the classifier, the authors apply FS for meta-feature selection. Another approach based on the same meta-features is introduced by Scholz-Reiter et al. [105]. They use a meta-feature aggregation instead of meta-feature selection, and a Linear Discriminant Analysis (LDA) as classification method to select the optimal forecasting method. Shahoud et al. [106] introduce statistical meta-features for different aggregation levels of the time series to extract characteristics at different time scales. They select suitable meta-features by a BE and aggregate them using an autoencoder. RF and ANN classifiers are compared for decision-making, both optimized with a grid search, where the ANN classifier achieves better performance in terms of selecting the best forecasting method. In addition to statistical and time series meta-features, Cui et al. [107] include domain information. The domain-based meta-features describe physical properties of buildings for which energy consumption is to be forecast. Bauer et al. [54] evaluate three types of decision models, i. e., classification, regression, and hybrid. In the classification decision model, an RF is trained to map meta-features to the forecasting method with the lowest forecast error. In the regression decision model, an RF learns how much worse each forecasting method is compared to the best method (i. e. forecast accuracy degradation). The hybrid decision model combines this RF regression with an RF classifier that maps the RF regression prediction to the best method. In the evaluation, the hybrid approach achieves the best performance in terms of forecast accuracy degradation. Instead of training an individual model for each new time series, Bandara et al. [43] suggest clustering time series using meta-features and training only one model for each cluster, which is applied to all time series in the cluster.

7.2   Forecast Ensembling

Forecast ensembling aims to improve the forecast robustness by bundling multiple forecasts of different models. We differentiate forecast ensembling from ensemble learning methods that build an ensemble of weak models[25], such as RF and GBM. Ensembling the forecasts from a pool of different forecasting models aims to avoid occasional poor forecasts, rather than outperforming the best individual forecasting model [11]. In the following, methods for ensembling in automated forecasting pipelines are introduced and exemplified using the reviewed literature, guided by the summary in Table 7.

The benefit of forecast ensembling is empirically demonstrated in many cases. For example, in the analysis of the so-called M3 forecasting competition [77], averaging the model output of all submitted forecasting methods performs better than each individual method itself. Simple forecast ensembling through averaging is used by Martínez et al. [26]. They average the output of three kNN models with $k \in \{3, 5, 7\}$ after the identification of optimal lag features with FS.

To improve averaging, one may weight the forecasting methods according to the expected individual performance. An ensemble can also be improved by only considering the k-best candidate methods, ranked by a forecasting method selection beforehand (i. e. heuristic, empirical, and decision model-based methods).

Selecting the candidate methods based on a heuristic forecasting method selection is proposed by Pawlikowski and Chorowska [98]. They categorize the time series data of the M4 forecasting competition [113] in terms of their frequency, and the existence of a trend and seasonality. Depending on the category, they select a distinct pool of candidate forecasting methods. The hyperparameters of the candidate methods are optimized and the weight for each candidate is determined based on the sMAPE error, validated out-of-sample with a rolling origin evaluation.

The following authors use empirical forecasting method selection to consider only the k-best candidate methods in the ensemble. Crone and Kourentzes [35] empirically determine the forecasting performance of candidate MLP architectures in a grid search (N_{h}, activation) with a rolling origin evaluation (out-of-sample) and average the outputs of the ten best candidates to reduce the impact of overfitting. Kourentzes et al. [100] propose an FS heuristic to decide on the number of ranked candidates to be considered for averaging. They calculate the performance metric's rate of increase C' assigned to each forecast and include all candidates until the first steep increase $C' > T$. To detect the increase, they use the same approach used for detecting outliers in boxplots, i. e., $T = \mathrm{Q}3 + 1.5\mathrm{IQR}$, where Q3 is the 3rd quartile.

---

[25]The forecast of a weak model, e. g., a DT, is only slightly superior to a random estimate. Ensemble learning aims to combine many weak models to achieve a good estimate.

Instead of averaging the k-best candidates, Shetty and Shobha [101] assign weights to the filtered candidates before averaging. Candidates with low forecast errors Q receive more weight, i.e.,

$$w_i = \frac{\prod_{j=1}^{k} Q_j}{Q_i \sum_{j=1}^{k} Q_j}, \quad \sum_{i=1}^{k} w_i = 1. \tag{8}$$

Wu et al. [102] propose a multi-objective optimization to determine the optimal weights for averaging candidates. They apply the flower pollination metaheuristic to minimize

$$\min \sum_{i=1}^{4} w_i Q_i, \quad \sum_{i=1}^{4} w_i = 1 \tag{9}$$

with the error metrics Q_i including the MAE, RMSE, and their relative formulations.

In the decision model-based method selection, one can also include weighted averaging directly in the decision model. Montero-Manso et al. [109] and Li et al. [110] train a GBM classifier with softmax-transformed outputs corresponding to the weights of the candidates for averaging. Similarly, Ma and Fildes [64] compare a Convolutional Neural Network (CNN) and a Fully Connected Neural Network (FCNN) classifier using output neurons with softmax activation to predict the weights for averaging. Instead of softmax outputs, Züfle et al. [63] use an LR as a decision model. Its output reflects the probability of how well the forecasting method fits the time series and determines the weight for averaging. For filtering the k-best candidate methods, they post-process the LR output.

Above mentioned forecast ensembling methods with weighted average determine the weights statically, i.e., the weights do not change as the time series evolves. Sergio et al. [60] propose a dynamic weighting method for the ensembling of forecasts. For every single forecast, the method searches for the k-nearest patterns in the training data similar to the given input data. Three ensemble functions – average, median, and softmax – are evaluated on the found similar patterns, and the method with the best performance is chosen for the forecast. A dynamic decision model-based approach is also introduced by Villegas et al. [111]. In their work, a binary SVM classifier is trained on performance metrics and meta-features to predict the best forecasting method from a pool of candidates for each forecast origin. In their experiment, the dynamic selection achieves the best performance compared to the mean and the median ensembling of all candidate forecasting methods.

7.3   Discussion

We discuss the automated selection of the optimal forecasting methods as the fourth and forecast ensembling as the fifth section of the automated forecasting pipeline. For both pipeline sections, we highlight potential problems, give recommendations, and suggest future work.

The automated selection includes heuristic, empirical, and decision model-based methods. As shown in Table 7, most empirical selection methods for the forecasting method are based on performance metrics, whereas decision models base their selection mainly on meta-features. Considering automation, the selection of the forecasting method is often combined with an individual HPO of the candidate forecasting methods. If multiple forecasting methods are evaluated in the selection, about half of the reviewed literature combines the selection with forecast ensembling.

In the automated selection, we notice different potential problems. The heuristic forecasting method selection is based on straightforward decision rules. To strengthen the evidence of the selection, however, we consider that performance metrics or meta-features are needed. The empirical selection based on performance metrics requires a high computing effort as every candidate forecasting method needs to be fitted to the data. The selection with a decision model based on meta-features reduces the computing effort but requires a sufficiently large and diverse data set for training the decision model. Based on our analysis, a comprehensive benchmark comparing the computing effort and the forecast accuracy of heuristic, empirical, and decision model-based forecasting method selection is missing, and therefore no recommendation can be made. Empirical evidence, however, exists for the forecast ensembling. We recommend combining forecast ensembling with the forecasting method selection by determining the pool of candidate methods and ensemble weights.

Based on the challenges of automated forecasting method selection, future work could tailor decision models for specific domains using HPO. Furthermore, the research could analyze if the selection of pre-trained forecasting methods is beneficial – either for application to the new time series without adaptation or after re-training on the new data.

8   Automated Forecasting Pipeline

The forecasting pipeline consists of the sections pre-processing, feature engineering, HPO, and forecasting method selection and ensembling (see Figure 1).

8.1   Status Quo

Table 9 shows that the analyzed literature covers different sections of the forecasting pipeline.[26] Clustering the analyzed papers that cover the same sections of the forecasting pipeline yields 17 clusters. Most of the 63 papers reviewed cover only two or three sections of the forecasting pipeline. Only two papers cover four sections, and none of the papers reviewed covers all five sections of the forecasting pipeline.

27 papers consider only statistical forecasting methods, 22 papers only machine learning methods, and 14 papers both. A majority of the papers that include both families of forecasting methods consider forecasting method selection in their pipeline (12 of 14 papers).

Of all the papers analyzed, most papers focus on pre-processing (35) and HPO (32), followed by the forecasting method selection (30). Since statistical methods consider lag features implicitly in the model structure, only 21 papers explicitly deal with feature engineering. The least attention is paid to the ensembling of forecasting methods, i.e., in only 11 papers.

Table 9: Overview of the forecasting pipeline sections covered in the reviewed literature.

Ref.	Pre-processing	Feature Engineering	Hyperparameter Optimization	Forecasting Method Selection	Forecast Ensembling	Covered Sections	Cluster
[25, 39, 40, 69]	X	-	-	-	-	●	A
[83–88, 90]	-	-	X	-	-	●	B
[97, 99, 103–108, 111]	-	-	-	X	-	●	C
[33, 34, 58]	X	X	-	-	-	● ●	D
[30, 32, 36, 38, 53]	X	-	X	-	-	● ●	E
[42, 61]	X	-	-	X	-	● ●	F
[70, 72–74]	-	X	X	-	-	● ●	G
[71]	-	X	-	X	-	● ●	H
[43, 89]	-	-	X	X	-	● ●	I
[98, 100–102, 109, 110]	-	-	-	X	X	● ●	J
[27–29, 55–57, 59, 66, 75]	X	X	X	-	-	● ● ●	K
[37, 41, 62]	X	-	X	X	-	● ● ●	L
[35, 63, 64]	X	-	-	X	X	● ● ●	M
[26]	X	X	-	-	X	● ● ●	N
[31, 54]	X	X	-	X	-	● ● ●	O
[65]	X	X	X	X	-	● ● ● ●	P
[60]	X	-	X	X	X	● ● ● ●	Q

8.2   Discussion

Based on the analysis of the status quo, we give the following recommendations to holistically automate the complete forecasting pipeline. The optimization of the hyperparameters should be combined with the automated feature selection (clusters G, K, and P in Table 9) because it is expected that each feature set candidate requires an individual hyperparameter configuration. Moreover, combining an HPO and the automated selection of forecasting methods is advantageous as the best method is selected from a pool of candidates whose hyperparameters, in turn, are already optimized by an HPO (clusters I, L, P, and Q in Table 9). Also, the forecast ensembling should be combined with the automated selection of appropriate forecasting methods to eliminate poor candidates (clusters J, M, and Q in Table 9). Hereby, both statistical and machine learning-based forecasting methods should be considered, as the diversity of forecasting methods has the potential to increase the robustness of the results.

In addition to these recommendations, we discuss how automated forecasting pipelines are related to two other current research directions. The first research direction is hybrid modeling, combining well-studied statistical methods with deep learning in time series forecasting. More specifically, deep neural networks encode time-varying parameters for non-probabilistic forecasting methods or produce distribution parameters for probabilistic methods [114]. The second

---

[26]The scope and complexity with which the analyzed literature addresses the sections of the forecasting pipeline are analyzed in the previous Sections 4-7 of this paper.

research direction is End-to-End (E2E) learning. It aims to include all five sections of the forecasting model's design process into the training using gradient-based learning, e.g., embedded feature learning in a deep neural network [64]. While hybrid modeling only considers specific sections of the design process using deep learning, E2E learning uses one deep neural network to address several sections with appropriate layers. As in automated forecasting pipelines, both hybrid modeling and E2E learning need to consider all five sections of the design process to obtain high-performing and robust forecasts. Regarding automating the design process, emerging approaches such as neural architecture search and meta-learning [8] are promising for both research directions in the future.

Finally, we note that manual tailoring of forecasting models is still a common practice in time series forecasting competitions despite first successful applications of automated forecasting pipelines. For example, in the most recent M5 forecasting competition [115], Liang and Lu [116] provide evidence that their AutoML pipeline achieves competitive performance with moderate computing effort. Therefore, we assume that the potential of the automated design is not fully utilized yet in time series competitions.

9   Conclusions

Time series are collected in many domains, and forecasting their progression over a certain future period is becoming increasingly important for many use cases. This rapidly growing demand requires making the design process of time series forecasts more efficient by automation; that is, automating design decisions within each section of the forecasting pipeline or automatically combining methods across pipeline sections. Although the first aspect of design automation has already been considered by various researchers, understanding how various automation methods interact within the pipeline and how they can be combined is a critical open question. Therefore, the present paper considers the automation of each of the five sections in the time series forecasting pipeline. It also investigates the corresponding literature in terms of the interaction and combination of automation methods within the five pipeline sections, incorporating both Automated Machine Learning (AutoML) and automated statistical forecasting methods.

Besides various specific insights related to each pipeline section that we discuss throughout the corresponding sections, we find on a general level that the majority of the 63 papers only cover two or three of the five forecasting pipeline sections. Therefore, we conclude that there is a research gap regarding approaches that holistically consider the automation of the forecasting pipeline, enabling the large-scale application to use cases without manual and time-consuming tailoring.

Besides the holistic automation, future work should research the adaption of the presented automation methods for probabilistic time series forecasts. Concurrently, the automated forecasting pipelines should be validated and tailored to particular use cases as an initial starting point before generalizing them to universal automated forecasting pipelines. Furthermore, given the insights on the combination and interaction of automation methods in the pipeline, future work should examine their performance (e. g. in forecasting competitions). This performance evaluation would benefit if future work on automated forecasting pipelines would be open source – both the implementation of the evaluated methods and the data sets used for the evaluation. Moreover, open-source publishing should promote the adoption of automated pipelines for time series forecasting, thus leveraging the great potential of improving the design efficiency through automation, achieving a high forecasting performance and a robust operation.
Overall, the present paper focuses on the automated design of forecasting pipelines. Therefore, to fully automate the entire forecasting process, future work should also consider the automated application of the resulting forecasting models, including performance monitoring during operation and model adaption.[27]

---

[27]For the fusion of automated design and automated application, automation levels are defined by Meisenbacher et al. [13].

Conflict of Interest

The authors declare no conflict of interest.

Supporting Information

Search-term for title, abstract, and keywords:

(
    (
        automl OR
        "automat* model*" OR
        "automat* deep learning" OR
        "automat* machine learning" OR
        "automat* forecast*" OR
        "automat* predict*"
)
    AND ("time-series")
)
OR "automat* time-series"

The search-term based exploration was conducted on 4/29/2021 using *Scopus* and yielded 359 papers. The forward search was conducted on 6/9/2021 and yielded 1572 additional papers. The backward search was conducted on 8/17/2021 and yielded 580 additional papers.

Acronyms

ACF	AutoCorrelation Function
ADF	Augmented Dickey-Fuller
AIC	Akaike Information Criterion
ANN	Artificial Neural Network
AR	AutoRegression
ARMA	AutoRegressive Moving Average
ARIMA	AutoRegressive Integrated Moving Average
AutoML	Automated Machine Learning
BE	Backward Elimination
BIC	Bayesian Information Criterion
BO	Bayesian Optimization
BSTS	Bayesian Structural Time Series
CASH	Combined Algorithm Selection and Hyperparameter Optimization
CNN	Convolutional Neural Network
DBN	Deep Belief Network
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DEA	Differential Evolution Algorithm
DES	Double Exponential Smoothing
DT	Decision Tree
E2E	End-to-End
EA	Evolutionary Algorithm
EDA	Estimation Distribution Algorithm
ELM	Extreme Learning Machine
ENN	Elman Neural Network
ES	Exponential Smoothing
ETS	Error Trend Seasonality
FCNN	Fully Connected Neural Network
FNN	Fuzzy Neural Network
FS	Forward Selection

GA	Genetic Algorithm
GBM	Gradient Boosting Machine
GOF	Goodness-Of-Fit
GP	Gaussian Process
GRNN	General Regression Neural Network
HPO	HyperParameter Optimization
IC	Information Criteria
IMA	Integrated Moving Average
INF	Iterative Neural Filter
IQR	Inter-Quartile Range
kNN	k-Nearest Neighbors
KPSS	Kwiatkowski–Phillips–Schmidt–Shin
KS	Kolmogorov–Smirnov
LASSO	Least Absolute Shrinkage and Selection Operator
LC	Locally Constant
LDA	Linear Discriminant Analysis
LL	Locally Linear
LogR	Logistic Regression
LR	Linear Regression
LSTM	Long Short-Term Memory
MA	Moving Average
MAD	Median Absolute Deviation
MAE	Mean Average Error
MAPE	Mean Absolute Percentage Error
MARS	Multivariate Adaptive Regression Splines
MASE	Mean Absolute Scaled Error
MIQP	Mixed Integer Quadratic Programming
MKL	Multiple Kernel Learning
MLP	MultiLayer Perceptron
MSE	Mean Squared Error
NLP	Non-Linear Programming
NNetAR	Neural Network AutoRegression
OSCB	Osborn–Chui–Smith–Birchenhall
PACF	Partial Auto-Correlation Function
PCA	Principal Component Analysis
PR	Polynomial Regression
PSO	Particle Swarm Optimization
RBFNN	Radial Basis Function Neural Network
RF	Random Forest
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
RPaRT	Recursive Partitioning and Regression Trees
RW	Random Walk
sARIMA	seasonal AutoRegressive Integrated Moving Average
SES	Simple Exponential Smoothing
SETARMA	Self Exciting Threshold AutoRegressive Moving Average
sMAPE	symmetric Mean Absolute Percentage Error
STL	Seasonal and Trend decomposition using Loess
SVD	Singular Value Decomposition
SVM	Support Vector Machine
SVR	Support Vector Regression
TBATS	Trigonometric Box-Cox transform, ARMA errors & Trend and Seasonal components
TES	Triple Exponential Smoothing
TPE	Tree Parzen Estimator
UC	Unexplored Component
VAR	Vector AutoRegression

Implementations of Forecasting Methods

autoARIMA	[R] [Python]
autoTheta	[R]
AR, MA, ARMA, (s)ARIMA	[R] [Python]
BSTS	[R]
ELM	[R]
ETS	[R] [Python]
GRNN	[R]
kNN	[R]
MLP INF	[R]
NNetAR	[R]
Prophet	[R] [Python]
RPaRT	[R]
RW, dRW, sRW	[R] [Python]
SES, DES, TES	[R] [Python]
TBATS	[R] [Python]
Telescope	[R]
Theta	[R] [Python]
VAR	[R] [Python]

References

[1] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*, Third. Melbourne, Australia: OTexts, 2021.

[2] T. Boone, R. Ganeshan, A. Jain, and N. R. Sanders, "Forecasting sales in the supply chain: Consumer analytics in the big data era," *International Journal of Forecasting*, vol. 35, no. 1, pp. 170–180, 2019. DOI: 10.1016/j.ijforecast.2018.09.003.

[3] T. Ahmad, H. Zhang, and B. Yan, "A review on renewable energy and electricity requirement forecasting models for smart grid and buildings," *Sustainable Cities and Society*, vol. 55, p. 102 052, 2020. DOI: 10.1016/j.scs.2020.102052.

[4] I. Rahimi, F. Chen, and A. H. Gandomi, "A review on covid-19 forecasting models," *Neural Computing and Applications*, 2021. DOI: 10.1007/s00521-020-05626-8.

[5] K. Shaukat, T. M. Alam, S. Luo, S. Shabbir, I. A. Hameed, J. Li, S. K. Abbas, and U. Javed, "A review of time-series anomaly detection techniques: A step to future perspectives," in *Advances in Information and Communication*, virtual event, 2021, pp. 865–877. DOI: 10.1007/978-3-030-73100-7_60.

[6] X. Wang and C. Wang, "Time series data cleaning: A survey," *IEEE Access*, vol. 8, pp. 1866–1881, 2020. DOI: 10.1109/ACCESS.2019.2962152.

[7] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56–70, 2020. DOI: 10.38094/jastt1224.

[8] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated machine learning: Methods, systems, challenges*, ser. The Springer Series on Challenges in Machine Learning. Cham, Switzerland: Springer International Publishing, 2019.

[9] M.-A. Zöller and M. F. Huber, "Benchmark and survey of automated machine learning frameworks," *Journal of Artificial Intelligence Research*, vol. 70, pp. 409–472, 2021. DOI: 10.1613/jair.1.11854.

[10] Z. Hajirahimi and M. Khashei, "Hybrid structures in time series modeling and forecasting: A review," *Engineering Applications of Artificial Intelligence*, vol. 86, pp. 83–106, 2019. DOI: 10.1016/j.engappai.2019.08.018.

[11] D. Shaub, "Fast and accurate yearly time series forecasting with forecast combinations," *International Journal of Forecasting*, vol. 36, no. 1, pp. 116–120, 2020. DOI: 10.1016/j.ijforecast.2019.03.032.

[12] L. Tuggener, M. Amirian, K. Rombach, S. Lorwald, A. Varlet, C. Westermann, and T. Stadelmann, "Automated machine learning in practice: State of the art and recent results," in *2019 6th Swiss Conference on Data Science (SDS)*, Bern, Switzerland, 2019, pp. 31–36. DOI: 10.1109/SDS.2019.00-11.

[13]   S. Meisenbacher, J. Pinter, T. Martin, V. Hagenmeyer, and R. Mikut, "Concepts for automated machine learning in smart grid applications," in *Proceedings - 31. Workshop Computational Intelligence: Berlin, 25. - 26. November 2021*, Berlin, Germany, 2021, pp. 11–35. DOI: 10.5445/KSP/1000138532.

[14]   B. Heidrich, A. Bartschat, M. Turowski, O. Neumann, K. Phipps, S. Meisenbacher, K. Schmieder, N. Ludwig, R. Mikut, and V. Hagenmeyer, "Pywatts: Python workflow automation tool for time series," *arXiv:2106.10157*, 2021.

[15]   M. Löning, A. Bagnall, S. Ganesh, V. Kazakov, J. Lines, and F. J. Király, "Sktime: A unified interface for machine learning with time series," in *2019 Workshop on Systems for ML at NeurIPS*, Vancouver, Canada, 2019.

[16]   J. G. De Gooijer and R. J. Hyndman, "25 years of time series forecasting," *International Journal of Forecasting*, vol. 22, no. 3, pp. 443–473, 2006. DOI: 10.1016/j.ijforecast.2006.01.001.

[17]   Z. Han, J. Zhao, H. Leung, K. F. Ma, and W. Wang, "A review of deep learning models for time series prediction," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 7833–7848, 2019. DOI: 10.1109/JSEN.2019.2923982.

[18]   S. B. Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa, "A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition," *Expert Systems with Applications*, vol. 39, no. 8, pp. 7067–7083, 2012. DOI: 10.1016/j.eswa.2012.01.039.

[19]   J. Webster and R. T. Watson, "Analyzing the past to prepare for the future: Writing a literature review," *MIS Quarterly*, vol. 26, no. 2, pp. xiii–xxiii, 2002.

[20]   P. J. Brockwell and R. A. Davis, *Introduction to time series and forecasting*, Third, ser. Springer Texts in Statistics. Cham, Switzerland: Springer International Publishing, 2016, ISBN: 978-3-319-29852-8.

[21]   J. Á. González Ordiano, S. Waczowicz, V. Hagenmeyer, and R. Mikut, "Energy forecasting tools and services," *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 2, e1235, 2018. DOI: 10.1002/widm.1235.

[22]   G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: Forecasting and control*, Fifth, ser. Wiley Series in Probability and Statistics. Hoboken, USA: Wiley, 2016.

[23]   C. Cheng, A. Sa-Ngasoongsong, O. Beyca, T. Le, H. Yang, Z. Kong, and S. T. Bukkapatnam, "Time series forecasting for nonlinear and non-stationary processes: A review and comparative study," *IIE Transactions*, vol. 47, no. 10, pp. 1053–1071, 2015. DOI: 10.1080/0740817X.2014.999180.

[24]   V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, 2009. DOI: 10.1145/1541880.1541882.

[25]   S. Liu, S. Gu, and T. Bao, "An automatic forecasting method for time series," *Chinese Journal of Electronics*, vol. 26, no. 3, pp. 445–452, 2017. DOI: 10.1049/cje.2017.01.011.

[26]   F. Martínez, M. P. Frías, M. D. Pérez, and A. J. Rivera, "A methodology for applying k-nearest neighbor to time series forecasting," *Artificial Intelligence Review*, vol. 52, no. 3, pp. 2019–2037, 2019. DOI: 10.1007/s10462-017-9593-z.

[27]   W. Yan, "Toward automatic time-series forecasting using neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 7, pp. 1028–1039, 2012. DOI: 10.1109/TNNLS.2012.2198074.

[28]   S. Fan, X. Qin, Z. Jia, X. Qi, and M. Lin, "Elm-based improved layered ensemble architecture for time series forecasting," *IEEE Access*, vol. 7, pp. 97 827–97 837, 2019. DOI: 10.1109/ACCESS.2019.2927047.

[29]   A. Widodo, I. Budi, and B. Widjaja, "Automatic lag selection in time series forecasting using multiple kernel learning," *International Journal of Machine Learning and Cybernetics*, vol. 7, no. 1, pp. 95–110, 2016. DOI: 10.1007/s13042-015-0409-7.

[30]   A. Maravall, R. López-Pavón, and D. Pérez-Cañete, "Reliability of the automatic identification of arima models in program tramo," in *Empirical Economic and Financial Research*, ser. Advanced Studies in Theoretical and Applied Econometrics, J. Beran, Y. Feng, and H. Hebbel, Eds., vol. 48, Springer International Publishing, 2015, pp. 105–122, ISBN: 978-3-319-03121-7.

[31]   M. Züfle and S. Kounev, "A framework for time series preprocessing and history-based forecasting method recommendation," in *Proceedings of the 2020 Federated Conference on Computer Science and Information Systems*, Sofia, Bulgaria, 2020, pp. 141–144. DOI: 10.15439/2020F101.

[32]   N. Tran and D. A. Reed, "Automatic arima time series modeling for adaptive i/o prefetching," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 4, pp. 362–377, 2004. DOI: 10.1109/TPDS.2004.1271185.

[33]   A. Bauer, M. Züfle, N. Herbst, S. Kounev, and V. Curtef, "Telescope: An automatic feature extraction and transformation approach for time series forecasting on a level-playing field," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, Dallas, USA, 2020, pp. 1902–1905. DOI: 10.1109/ICDE48307.2020.00199.

[34] N. Kourentzes and S. F. Crone, "Frequency independent automatic input variable selection for neural networks for forecasting," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*, Barcelona, Spain, 2010, pp. 1–8. DOI: 10.1109/IJCNN.2010.5596637.

[35] S. F. Crone and N. Kourentzes, "Feature selection for time series prediction: A combined filter and wrapper approach for neural networks," *Neurocomputing*, vol. 73, no. 10, pp. 1923–1936, 2010. DOI: 10.1016/j.neucom.2010.01.017.

[36] M. Alzyout, M. Alsmirat, and M. I. Al-Saleh, "Automated arima model construction for dynamic vehicle gps location prediction," in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, Granada, Spain, 2019, pp. 380–386. DOI: 10.1109/IOTSMS48152.2019.8939197.

[37] N. C. Sekma, A. Elleuch, and N. Dridi, "Automated forecasting approach minimizing prediction errors of cpu availability in distributed computing systems," *International Journal of Intelligent Systems and Applications*, vol. 8, no. 9, pp. 8–21, 2016. DOI: 10.5815/ijisa.2016.09.02.

[38] R. J. Hyndman and Y. Khandakar, "Automatic time series forecasting: The forecast package for r," *Journal of Statistical Software*, vol. 27, no. 3, pp. 1–22, 2008. DOI: 10.18637/jss.v027.i03.

[39] Y. Lu and S. M. AbouRizk, "Automated box-jenkins forecasting modelling," *Automation in Construction*, vol. 18, no. 5, pp. 547–558, 2009. DOI: 10.1016/j.autcon.2008.11.007.

[40] S. Anvari, S. Tuna, M. Canci, and M. Turkay, "Automated box-jenkins forecasting tool with an application for passenger demand in urban rail systems," *Journal of Advanced Transportation*, vol. 50, no. 1, pp. 25–49, 2016. DOI: 10.1002/atr.1332.

[41] A. Amin, L. Grunske, and A. Colman, "An automated approach to forecasting qos attributes based on linear and non-linear time series modeling," in *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*, Essen, Germany, 2012, pp. 130–139. DOI: 10.1145/2351676.2351695.

[42] R. Fildes and F. Petropoulos, "Simple versus complex selection rules for forecasting many time series," *Journal of Business Research*, vol. 68, no. 8, pp. 1692–1701, 2015. DOI: 10.1016/j.jbusres.2015.03.028.

[43] K. Bandara, C. Bergmeir, and S. Smyl, "Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach," *Expert Systems with Applications*, vol. 140, p. 112 896, 2020. DOI: 10.1016/j.eswa.2019.112896.

[44] D. Kwiatkowski, P. C. Phillips, P. Schmidt, and Y. Shin, "Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?" *Journal of Econometrics*, vol. 54, no. 1, pp. 159–178, 1992. DOI: 10.1016/0304-4076(92)90104-Y.

[45] D. R. Cox and A. Stuart, "Some quick sign tests for trend in location and dispersion," *Biometrika*, vol. 42, no. 1/2, pp. 80–95, 1955.

[46] Y.-W. Cheung and K. S. Lai, "Lag order and critical values of the augmented dickey-fuller test," *Journal of Business & Economic Statistics*, vol. 13, no. 3, pp. 277–280, 1995.

[47] F. Canova and B. E. Hansen, "Are seasonal patterns constant over time? a test for seasonal stability," *Journal of Business & Economic Statistics*, vol. 13, no. 3, pp. 237–252, 1995.

[48] D. R. Osborn, A. P. L. Chui, J. P. Smith, and C. R. Birchenhall, "Seasonality and the order of integration for consumption," *Oxford Bulletin of Economics and Statistics*, vol. 50, no. 4, pp. 361–377, 1988. DOI: 10.1111/j.1468-0084.1988.mp50004002.x.

[49] H. W. Lilliefors, "On the kolmogorov-smirnov test for normality with mean and variance unknown," *Journal of the American Statistical Association*, vol. 62, no. 318, pp. 399–402, 1967.

[50] G. E. P. Box and D. R. Cox, "An analysis of transformations," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 26, no. 2, pp. 211–243, 1964.

[51] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning, "Stl: A seasonal-trend decomposition procedure based on loess," *Journal of Official Statistics*, vol. 6, no. 1, pp. 3–73, 1990.

[52] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.

[53] G. Dellino, T. Laudadio, R. Mari, N. Mastronardi, and C. Meloni, "Microforecasting methods for fresh food supply chain management: A computational study," *Mathematics and Computers in Simulation*, vol. 147, pp. 100–120, 2018. DOI: 10.1016/j.matcom.2017.12.006.

[54] A. Bauer, M. Züfle, J. Grohmann, N. Schmitt, N. Herbst, and S. Kounev, "An automated forecasting framework based on method recommendation for seasonal time series," in *Proceedings of the ACM/SPEC International Conference on Performance Engineering*, Edmonton, Canada, 2020, pp. 48–55. DOI: 10.1145/3358960.3379123.

[55] J. P. Donate, X. Li, G. G. Sánchez, and A. S. de Miguel, "Time series forecasting by evolving artificial neural networks with genetic algorithms, differential evolution and estimation of distribution algorithm," *Neural Computing and Applications*, vol. 22, no. 1, pp. 11–20, 2013. DOI: 10.1007/s00521-011-0741-0.

[56] J. P. Donate and P. Cortez, "Evolutionary optimization of sparsely connected and time-lagged neural networks for time series forecasting," *Applied Soft Computing*, vol. 23, pp. 432–443, 2014. DOI: 10.1016/j.asoc.2014.06.041.

[57] S. Maldonado, A. González, and S. Crone, "Automatic time series analysis for electric load forecasting via support vector regression," *Applied Soft Computing*, vol. 83, p. 105 616, 2019. DOI: 10.1016/j.asoc.2019.105616.

[58] F. Martínez, F. Charte, A. J. Rivera, and M. P. Frías, "Automatic time series forecasting with grnn: A comparison with other models," in *Advances in Computational Intelligence*, ser. Lecture Notes in Computer Science, I. Rojas, G. Joya, and A. Catala, Eds., vol. 11506, Springer International Publishing, 2019, pp. 198–209, ISBN: 978-3-030-20520-1. DOI: 10.1007/978-3-030-20521-8_17.

[59] S. Panigrahi and H. S. Behera, "Time series forecasting using differential evolution-based ann modelling scheme," *Arabian Journal for Science and Engineering*, vol. 45, no. 12, pp. 11 129–11 146, 2020. DOI: 10.1007/s13369-020-05004-5.

[60] A. T. Sergio, T. P. de Lima, and T. B. Ludermir, "Dynamic selection of forecast combiners," *Neurocomputing*, vol. 218, no. C, pp. 37–50, 2016. DOI: 10.1016/j.neucom.2016.08.072.

[61] J. Violos, S. Tsanakas, M. Androutsopoulou, G. Palaiokrassas, and T. Varvarigou, "Next position prediction using lstm neural networks," in *11th Hellenic Conference on Artificial Intelligence*, Athens, Greece, 2020, pp. 232–240. DOI: 10.1145/3411408.3411426.

[62] A. Widodo and I. Budi, "Feature enhancement for model selection in time series forecasting," in *2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, Bali, Indonesia, 2013, pp. 367–373. DOI: 10.1109/ICACSIS.2013.6761603.

[63] M. Züfle, A. Bauer, V. Lesch, C. Krupitzer, N. Herbst, S. Kounev, and V. Curtef, "Autonomic forecasting method selection: Examination and ways ahead," in *2019 IEEE International Conference on Autonomic Computing (ICAC)*, Umeå, Sweden, 2019, pp. 167–176. DOI: 10.1109/ICAC.2019.00028.

[64] S. Ma and R. Fildes, "Retail sales forecasting with meta-learning," *European Journal of Operational Research*, vol. 288, no. 1, pp. 111–128, 2021. DOI: 10.1016/j.ejor.2020.05.038.

[65] M. Rätz, A. P. Javadi, M. Baranski, K. Finkbeiner, and D. Müller, "Automated data-driven modeling of building energy systems via machine learning algorithms," *Energy and Buildings*, vol. 202, p. 109 384, 2019. DOI: 10.1016/j.enbuild.2019.109384.

[66] Y. R. Sagaert, E.-H. Aghezzaf, N. Kourentzes, and B. Desmet, "Tactical sales forecasting using a very large set of macroeconomic indicators," *European Journal of Operational Research*, vol. 264, no. 2, pp. 558–569, 2018. DOI: 10.1016/j.ejor.2017.06.054.

[67] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," *ACM Comput. Surv.*, vol. 54, no. 3, 2021. DOI: 10.1145/3444690.

[68] M. Weber, M. Turowski, H. K. Çakmak, R. Mikut, U. Kühnapfel, and V. Hagenmeyer, "Data-driven copy-paste imputation for energy time series," *IEEE Transactions on Smart Grid*, vol. 12, no. 6, pp. 5409–5419, 2021. DOI: 10.1109/TSG.2021.3101831.

[69] V. Cerqueira, N. Moniz, and C. Soares, "Vest: Automatic feature engineering for forecasting," *Machine Learning*, 2021. DOI: 10.1007/s10994-021-05959-y.

[70] D. Chakrabarti and C. Faloutsos, "F4: Large-scale automated forecasting using fractals," in *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, McLean, USA, 2002, pp. 2–9. DOI: 10.1145/584792.584797.

[71] S. D. Balkin and J. Ord, "Automatic neural network modeling for univariate time series," *International Journal of Forecasting*, vol. 16, no. 4, pp. 509–515, 2000. DOI: 10.1016/S0169-2070(00)00072-8.

[72] A. P. Lowther, P. Fearnhead, M. A. Nunes, and K. Jensen, "Semi-automated simultaneous predictor selection for regression-sarima models," *Statistics and Computing*, vol. 30, no. 6, pp. 1759–1778, 2020. DOI: 10.1007/s11222-020-09970-6.

[73] H. Son and C. Kim, "Forecasting short-term electricity demand in residential sector based on support vector regression and fuzzy-rough feature selection with particle swarm optimization," *Procedia Engineering*, vol. 118, pp. 1162–1168, 2015. DOI: 10.1016/j.proeng.2015.08.459.

[74] J. M. Valente and S. Maldonado, "Svr-ffs: A novel forward feature selection approach for high-frequency time series forecasting using support vector regression," *Expert Systems with Applications*, vol. 160, p. 113 729, 2020. DOI: 10.1016/j.eswa.2020.113729.

[75] G. Dellino, T. Laudadio, R. Mari, N. Mastronardi, and C. Meloni, "A reliable decision support system for fresh food supply chain management," *International Journal of Production Research*, vol. 56, no. 4, pp. 1458–1485, 2018. DOI: 10.1080/00207543.2017.1367106.

[76] A. Bauer, M. Züfle, S. Eismann, J. Grohmann, N. Herbst, and S. Kounev, "Libra: A benchmark for time series forecasting methods," in *Proceedings of the ACM/SPEC International Conference on Performance Engineering*, Virtual Event, France, 2021, pp. 189–200. DOI: 10.1145/3427921.3450241.

[77] S. Makridakis and M. Hibon, "The m3-competition: Results, conclusions and implications," *International Journal of Forecasting*, vol. 16, no. 4, pp. 451–476, 2000. DOI: 10.1016/S0169-2070(00)00057-1.

[78] A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications," in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, 2015, pp. 1200–1205. ISBN: 978-9-5323-3082-3. DOI: 10.1109/MIPRO.2015.7160458.

[79] D. Bertsimas, A. King, and R. Mazumder, "Best subset selection via a modern optimization lens," *The Annals of Statistics*, vol. 44, no. 2, pp. 813–852, 2016. DOI: 10.1214/15-AOS1388.

[80] K. Yang and C. Shahabi, "On the stationarity of multivariate time series for correlation-based data analysis," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*, Houston, USA, 2005, pp. 805–808. DOI: 10.1109/ICDM.2005.109.

[81] M. Barandas, D. Folgado, L. Fernandes, S. Santos, M. Abreu, P. Bota, H. Liu, T. Schultz, and H. Gamboa, "Tsfel: Time series feature extraction library," *SoftwareX*, vol. 11, p. 100456, 2020. DOI: 10.1016/j.softx.2020.100456.

[82] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated Machine Learning*, ser. The Springer Series on Challenges in Machine Learning, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., Cham, Switzerland: Springer International Publishing, 2019, pp. 3–33. ISBN: 978-3-030-05317-8.

[83] I. Svetunkov and F. Petropoulos, "Old dog, new tricks: A modelling view of simple moving averages," *International Journal of Production Research*, vol. 56, no. 18, pp. 6034–6047, 2018. DOI: 10.1080/00207543.2017.1380326.

[84] S. Hwang, M. Park, H.-S. Lee, and H. Kim, "Automated time-series cost forecasting system for construction materials," *Journal of Construction Engineering and Management*, vol. 138, no. 11, pp. 1259–1269, 2012. DOI: 10.1061/(ASCE)CO.1943-7862.0000536.

[85] J. Arlt and P. Trcka, "Automatic sarima modeling and forecast accuracy," *Communications in Statistics - Simulation and Computation*, vol. 50, no. 10, pp. 2949–2970, 2021. DOI: 10.1080/03610918.2019.1618471.

[86] D. J. Pedregal, "Time series analysis and forecasting with ecotool," *PLOS ONE*, vol. 14, no. 10, pp. 1–23, 2019. DOI: 10.1371/journal.pone.0221238.

[87] R. J. Hyndman, A. B. Koehler, R. D. Snyder, and S. Grose, "A state space framework for automatic forecasting using exponential smoothing methods," *International Journal of Forecasting*, vol. 18, no. 3, pp. 439–454, 2002. DOI: 10.1016/S0169-2070(01)00110-8.

[88] J. D. Bermúdez, J. V. Segura, and E. Vercher, "Siopred performance in a forecasting blind competition," in *2012 IEEE Conference on Evolving and Adaptive Intelligent Systems*, Madrid, Spain, 2012, pp. 192–197. DOI: 10.1109/EAIS.2012.6232828.

[89] E. Spiliotis, V. Assimakopoulos, and S. Makridakis, "Generalizing the theta method for automatic forecasting," *European Journal of Operational Research*, vol. 284, no. 2, pp. 550–558, 2020. DOI: 10.1016/j.ejor.2020.01.007.

[90] M. A. Villegas and D. J. Pedregal, "Automatic selection of unobserved components models for supply chain forecasting," *International Journal of Forecasting*, vol. 35, no. 1, pp. 157–169, 2019. DOI: 10.1016/j.ijforecast.2017.11.001.

[91] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[92] M. S. Bazaraa, H. D. Sherali, and S. C. M., *Nonlinear programming: Theory and algorithms*, Third. Hoboken, USA: John Wiley & Sons, 2006. DOI: 10.1002/0471787779.

[93] J. Pearl, *Heuristics: Intelligent search strategies for computer problem solving*, First. Boston, USA: Addison Wesley Publishing Company, 1984.

[94] T. Stützle, "Local search algorithms for combinatorial problems: Analysis, improvements, and new applications," Ph.D. dissertation, Darmstadt University of Technology, Germany, 1999.

[95] T. Bäck, *Evolutionary algorithms in theory and practice: Evolution strategies, evolutionary programming, genetic algorithms*, First. Oxford, UK: Oxford university press, 1996.

[96]   G. M. Ljung and G. E. P. Box, "On a measure of lack of fit in time series models," *Biometrika*, vol. 65, no. 2, pp. 297–303, 1978.

[97]   M. Shcherbakov, V. Kamaev, and N. Shcherbakova, "Automated electric energy consumption forecasting system based on decision tree approach," *IFAC Proceedings Volumes*, vol. 46, no. 9, pp. 1027–1032, 2013. DOI: 10.3182/20130619-3-RU-3018.00486.

[98]   M. Pawlikowski and A. Chorowska, "Weighted ensemble of statistical models," *International Journal of Forecasting*, vol. 36, no. 1, pp. 93–97, 2020. DOI: 10.1016/j.ijforecast.2019.03.019.

[99]   P. Pereira, J. Araujo, R. Matos, N. Preguica, and P. Maciel, "Software rejuvenation in computer systems: An automatic forecasting approach based on time series," in *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, Orlando, USA, 2018, pp. 1–8. DOI: 10.1109/PCCC.2018.8711347.

[100]   N. Kourentzes, D. Barrow, and F. Petropoulos, "Another look at forecast selection and combination: Evidence from forecast pooling," *International Journal of Production Economics*, vol. 209, pp. 226–235, 2019. DOI: 10.1016/j.ijpe.2018.05.019.

[101]   J. Shetty and G. Shobha, "An ensemble of automatic algorithms for forecasting resource utilization in cloud," in *2016 Future Technologies Conference (FTC)*, San Francisco, USA, 2016, pp. 301–306. DOI: 10.1109/FTC.2016.7821626.

[102]   Z. Wu, X. Xia, L. Xiao, and Y. Liu, "Combined model with secondary decomposition-model selection and sample selection for multi-step wind power forecasting," *Applied Energy*, vol. 261, p. 114 345, 2020. DOI: 10.1016/j.apenergy.2019.114345.

[103]   S. Taghiyeh, D. C. Lengacher, and R. B. Handfield, "Forecasting model selection using intermediate classification: Application to monarchfx corporation," *Expert Systems with Applications*, vol. 151, p. 113 371, 2020. DOI: 10.1016/j.eswa.2020.113371.

[104]   M. Kück, S. F. Crone, and M. Freitag, "Meta-learning with neural networks and landmarking for forecasting model selection an empirical evaluation of different feature sets applied to industry data," in *2016 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, Canada, 2016, pp. 1499–1506. DOI: 10.1109/IJCNN.2016.7727376.

[105]   B. Scholz-Reiter, M. Kück, and D. Lappe, "Prediction of customer demands for production planning: Automated selection and configuration of suitable prediction methods," *CIRP Annals*, vol. 63, no. 1, pp. 417–420, 2014. DOI: 10.1016/j.cirp.2014.03.106.

[106]   S. Shahoud, H. Khalloof, C. Duepmeier, and V. Hagenmeyer, "Incorporating unsupervised deep learning into meta learning for energy time series forecasting," in *Proceedings of the Future Technologies Conference (FTC) 2020, Volume 1*, vol. 1288, Vancouver, Canada, 2020, pp. 326–345. DOI: 10.1007/978-3-030-63128-4_25.

[107]   C. Cui, T. Wu, M. Hu, J. D. Weir, and X. Li, "Short-term building energy model recommendation system: A meta-learning approach," *Applied Energy*, vol. 172, pp. 251–263, 2016. DOI: 10.1016/j.apenergy.2016.03.112.

[108]   K. E. Baddour, A. Ghasemi, and H. Rutagemwa, "Spectrum occupancy prediction for land mobile radio bands using a recommender system," in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, Chicago, USA, 2018, pp. 1–6. DOI: 10.1109/VTCFall.2018.8690654.

[109]   P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman, and T. S. Talagala, "Fforma: Feature-based forecast model averaging," *International Journal of Forecasting*, vol. 36, no. 1, pp. 86–92, 2020. DOI: 10.1016/j.ijforecast.2019.02.011.

[110]   X. Li, Y. Kang, and F. Li, "Forecasting with time series imaging," *Expert Systems with Applications*, vol. 160, p. 113 680, 2020. DOI: 10.1016/j.eswa.2020.113680.

[111]   M. A. Villegas, D. J. Pedregal, and J. R. Trapero, "A support vector machine for model selection in demand forecasting applications," *Computers & Industrial Engineering*, vol. 121, pp. 1–7, 2018. DOI: 10.1016/j.cie.2018.04.042.

[112]   X. Wang, K. Smith-Miles, and R. J. Hyndman, "Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series," *Neurocomputing*, vol. 72, no. 10, pp. 2581–2594, 2009. DOI: 10.1016/j.neucom.2008.10.017.

[113]   S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The m4 competition: 100,000 time series and 61 forecasting methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020. DOI: 10.1016/j.ijforecast.2019.04.014.

[114]   B. Lim and S. Zohren, "Time-series forecasting with deep learning: A survey," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 379, no. 2194, p. 20 200 209, 2021. DOI: 10.1098/rsta.2020.0209.

[115] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The m5 competition: Background, organization, and implementation," *International Journal of Forecasting*, 2021. DOI: 10.1016/j.ijforecast.2021.07.007.

[116] C. Liang and Y. Lu, *Using AutoML for time series forecasting*, accessed: 2022-01-25, 2020. [Online]. Available: http://ai.googleblog.com/2020/12/using-automl-for-time-series-forecasting.html.