VEST: automatic feature engineering for forecasting

Vitor Cerqueira[1] · Nuno Moniz[1,2] · Carlos Soares[1,2,3]

Abstract

Time series forecasting is a challenging task with applications in a wide range of domains. Auto-regression is one of the most common approaches to address these problems. Accordingly, observations are modelled by multiple regression using their past lags as predictor variables. We investigate the extension of auto-regressive processes using statistics which summarise the recent past dynamics of time series. The result of our research is a novel framework called VEST, designed to perform feature engineering using univariate and numeric time series automatically. The proposed approach works in three main steps. First, recent observations are mapped onto different representations. Second, each representation is summarised by statistical functions. Finally, a filter is applied for feature selection. We discovered that combining the features generated by VEST with auto-regression significantly improves forecasting performance in a database composed by 90 time series with high sampling frequency. However, we also found that there are no improvements when the framework is applied for multi-step forecasting or in time series with low sample size. VEST is publicly available online.

1 Introduction

Analysing and learning from time series is one of the most active topics in scientific research. One of the tasks related to this topic is forecasting, which denotes the process of predicting the value of future observations given some historical data. This task is relevant

✉ Vitor Cerqueira
cerqueira.vitormanuel@gmail.com

Nuno Moniz
nmmoniz@inesctec.pt

Carlos Soares
csoares@fe.up.pt

[1] INESC TEC, Porto, Portugal

[2] University of Porto, Porto, Portugal

[3] Fraunhofer AICOS Portugal, Porto, Portugal

Ⓐ Springer

to organisations across a wide range of domains of application. In many of these organisations, forecasting processes can have a significant financial impact (Kahn, 2003).

In the machine learning literature, it is widely accepted that the feature set used to represent a data set is a crucial component for building accurate predictive models (Guyon & Elisseeff, 2006). Hence, feature engineering is regarded as a critical step in machine learning projects. However, feature engineering is often an *ad-hoc* process. Practitioners design new features based on their domain knowledge and expertise. The limitations of current approaches to feature engineering are particularly relevant in time series forecasting, where, although evidence exists that it improves forecasting performance (Oliveira & Torgo, 2014), it is often overlooked.

Time series forecasting tasks are typically formalised using an auto-regressive approach. Accordingly, observations are modelled using multiple regression; the future observations we want to predict represent the target variable, and the past lags of these observations are used as explanatory variables. Auto-regression is at the core of many forecasting models in the literature, such as, for example, Auto-Regressive Integrated Moving Average (ARIMA) (Box et al., 2015). This approach presents an opportunity to approach feature engineering in a systematic way. Statistical features can be extracted from recent observations of time series. These features can help summarise the dynamics of the data and enrich its representation. For example, a simple feature such as the average of the past recent observations can help capture the overall level of the time series in a given point in time. If the process is adequately done, new features will lead to more accurate predictive models and better forecasting performance.

1.1 Our approach and paper organisation

Despite having domain expertise, professionals often lack proper time series analysis skills (Taylor & Letham, 2018). Getting the most out of state of the art time series forecasting methods requires significant experience, and it is a complex and time-consuming task. In this context, developing a framework for automatically extracting an optimal representation from an input time series is beneficial for practitioners with minimal technical expertise.

This paper presents and describes an automatic feature engineering approach called VEST (**Ve**ctor of **S**tatistics from **T**ime series). VEST is specifically designed to address forecasting problems.

VEST works according to the following three main steps:

- Transformation: We transform the time series into several distinct representations. This process may be beneficial for describing from different perspectives the underlying process causing the time series. For example, a simple moving average transformation can be useful to remove the spurious behaviour of time series;
- Summarisation: Each representation is summarised using statistics (e.g. mean, standard deviation);
- Selection: The first two steps can lead to a high dimensional problem. We apply a feature selection procedure to cope with this issue. The final set of features is concatenated with the original representation (before any transformation) to learn a regression model for forecasting.

In this paper, we show significant forecasting performance gains when applying VEST, based on a case study comprised of 90 univariate time series from several domains of

application. These improvements are evidenced for predicting the next point of the time series using two different learning algorithms: a variant of the model tree (Quinlan, 1993), and lasso (Tibshirani, 1996). Following our experimental design, we found that the time series sample size is important for the improvements in performance. The approach is also limited when applied to multi-step forecasting. Specifically, a pure auto-regressive approach is more adequate for multi-step forecasting or for time series with low sample size.

VEST is available online at https://github.com/vcerqueira/vest. Additionally, the code necessary to reproduce the experimental evaluation presented in the paper is made available to encourage reproducible research.

The organisation of the paper is as follows. In Sect. 2, we formalise the time series forecasting problems as a predictive regression task. We present a literature review in Sect. 3, including topics related to the feature engineering process and its automation, feature selection, and time series representation and feature extraction. Section 4 presents VEST, formalising its main steps: transforming the original representation, summarisation using statistics, and feature selection. We show the usefulness of this framework using empirical evidence presented in Sect. 5. We present a discussion on the results from such experiments, challenges, and future directions in Sect. 6. The paper is concluded in Sect. 7.

2 Problem definition

A univariate time series represents a temporal sequence of values $Y = \{y_1, y_2, \ldots, y_n\}$, where $y_i \in \mathcal{Y} \subset \mathbb{R}$ is the value of Y at time i and n is the length of Y. Forecasting denotes the process of predicting the value of the upcoming observations of the time series, $y_{n+1}, \ldots, y_{n+h}$, given the historical past observations, where h denotes the forecasting horizon. In this work, we focus on $h = 1$, which means we attempt to forecast the next value of the time series. We adopt an auto-regressive approach to address the problem of time series forecasting. Accordingly, observations of a time series are regressed on their past lags.

We start by reconstructing the time series as a geometric object by applying a time delay embedding using the Takens theorem (Takens, 1981). Then, the predictive task is framed as a multiple regression problem. We construct a set of observations of the form (X, y). In each observation, the value of y_i is modelled based on the past p values before it: $X_i = \{y_{i-1}, y_{i-2}, \ldots, y_{i-p}\}$, where $y_i \in \mathcal{Y} \subset \mathbb{R}$, which represents the observation we want to predict, and $X_i \in \mathcal{X} \subset \mathbb{R}^p$ represents the i-th embedding vector. Effectively, the time series is transformed into the data set $\mathcal{D}(X, y) = \{X_i, y_i\}_{p+1}^n$.

The learning objective is to build a regression model that provides an approximation to an unknown function $f : \mathcal{X} \to \mathcal{Y}$. The principle behind this approach is to model the conditional distribution of the i-th value of the time series given its p past values: $f(y_i|X_i)$.

3 Related research

This section provides an overview of the literature related to the topic of this work. First, we describe the importance of feature engineering and outline automatic procedures to address this task (Sect. 3.1). Afterwards, we focus on time series data. We describe approaches for changing the representation of time series (Sect. 3.2). Finally, we overview approaches for extracting features from time series with the objective of predictive modelling (Sect. 3.3).

3.1 Feature engineering

The goal of feature engineering is to enrich the representation of a data set with additional explanatory variables. The expectation is that such new predictors contain useful information and lead to more accurate predictive models (Guyon & Elisseeff, 2006).

In order to clarify the scope of our work, we split the generated variables through feature engineering into two classes: exogenous and endogenous. *Exogenous variables* are those derived from an external source. Consider an example of a time series representing the number of rooms occupied per day in a hotel. Forecasting the values of such a time series is interesting to the business for different reasons (e.g. pricing). In this scenario, a simple binary variable containing the information regarding whether or not the observation to be predicted occurs during the weekend may be useful to the predictive model. Since information is not contained within the original observations (each $y \in \mathcal{Y}$) we call it exogenous.

Regarding *endogenous variables*, X_i represents the embedding vector (Sect. 2) of a time series in a given point in time i. We can try to derive more information from X_i by applying some transformations or summary statistics. For example, the average of the values of X_i in a specific period may be a useful indicator for describing the overall level of the time series at that point. As such, a new explanatory variable is generated based on the time series itself, i.e. an endogeneous feature.

In this work, we focus on the automatic discovery of endogenous features in numeric time series. The goal is to augment the representation of the embedding vectors and improve forecasting performance of predictive models.

3.1.1 Automatic feature engineering

Typically, feature engineering is an *ad-hoc* process (Guyon & Elisseeff, 2006; Pinto et al., 2016). Practitioners design features based on their domain knowledge and expertise. However, this process is time-consuming, and requires both domain expertise and imagination.[1]

Recent approaches have been developed to systematise the feature engineering process. This research line is designated in the literature as automatic feature engineering. Examples include the following: Deep Feature Synthesis (Kanter & Veeramachaneni, 2015), ExploreKit (Katz et al., 2016), AutoLearn (Kaul et al., 2017), Cognito (Khurana et al., 2016), or OneBM (Lam et al., 2017). These frameworks focus on discovering relevant features from data sets comprised of several attributes, which can be either categorical or numeric. Moreover, most of these focus solely on classification problems and i.i.d. data. In this work, we focus on univariate time series data and the problem of forecasting. As such, many of the operations we develop to discover new relevant features intend to leverage the temporal dependency among the observations in the data set.

Deep learning methods can also be regarded as having an internal automatic feature engineering component. These approaches are able to learn higher-order representations based on the raw input data. Still, there are important factors which make standard feature engineering relevant. Deep learning models require a large amount of data, which is often not readily available. The internal representations of neural networks are opaque, while standard feature engineering is typically based on interpretable operations. This transparency may be important in sensitive applications. Besides, the two approaches are not

---

[1] https://www.kdnuggets.com/2019/03/why-automl-wont-replace-data-scientists.html.

incompatible, as neural networks can potentially leverage standard feature engineering, for example, to improve their learning efficiency.

3.2 Time series representation

Sometimes, time series are analysed using a representation that is different from the original one. Changing the representation of a time series can be beneficial for (1) reducing the dimensionality of the data, which leads to more efficient storage and processing; (2) implicit handling of noise; and (3) focusing on fundamental characteristics of the data (Esling & Agon, 2012). We refer to the work by Esling and Agon (2012) for a complete read on time series transformations.

Keogh et al. (2004) splits time series representation methods into three main types: non-data adaptive, data-adaptive, and model-based. In non-data adaptive approaches, the parameters of the transformation are independent of the underlying data. Examples of this approach are the discrete wavelet transformation (Percival & Walden, 2006) or the simple moving average. Conversely, the parameters of data-adaptive methods depend to some extent on the time series. Symbolic aggregate approximation (Lin et al., 2003) is a well-known approach of this sort.

Model-based approaches work on the assumption that some underlying model generates the time series. As such, parameters of the model represent the time series. Auto-regressive moving average (ARMA) (Chatfield, 2000) models are an example of this type.

3.3 Time series feature extraction

Extracting features from time series has been shown to improve performance in different tasks such as forecasting and classification (Prudêncio & Ludermir, 2004; Christ et al., 2016). We split the literature on this topic into two dimensions: sequence descriptions and sub-sequence descriptions. The former denotes approaches which summarise the complete set of observations available in a time series. The latter extract features from sub-sequences of time series, i.e. the embedding vectors.

3.3.1 Sequence descriptions

There are several approaches which extract features from the complete time series to improve forecasting performance. Examples of this are the works of Prudêncio and Ludermir (2004), Lemke and Gabrys (2010), Barandas et al. (2020), or Kang et al. (2017). Often, the goal of these approaches is to use meta-learning for model selection or combination. Essentially, they extract features from each time series in a given database. Then, a predictive model is created which relates the features of a time series with the most appropriate forecasting model in that data. In effect, for a new given time series, such a meta-learning model can make predictions regarding which model, or set of models, is more appropriate. Recently, Montero-Manso et al. (2020) applied this type of approach and ranked second in the well-known forecasting M4-competition.

In the context of time series classification, Christ et al. (2016) proposed the method FRESH for feature engineering. This method automatically extracts a large number of features from each time series in the database and selects the most relevant ones for building the classifier. Fulcher and Jones (2017) presented a feature extraction framework called *hctsa* for time series analysis. This tool extracts over 7000 features. Lubba

et al. (2019) selected a a subset of 22 *hctsa* features, leading to a 1000-fold reduction in computation time for feature extraction and only a small reduction in time series classification performance.

3.3.2 Sub-sequence descriptions

Compared to feature extraction from complete time series, few works are leveraging these processes for time series sub-sequences. Paras et al. (2009) used a set of statistics, such as simple and exponential moving averages, to improve a neural network model for weather forecasting. Oliveira and Torgo (2014) show that the average and standard deviation of recent values improve the performance of a bagging ensemble. Cerqueira et al. (2017) later corroborated these results using heterogeneous ensembles. However, the potential of the systematic application of approaches deriving new features from sub-sequences of time series has never been explored.

We follow this research line and derive new features from sub-sequences of time series. To accomplish this, we develop VEST, a novel framework for automatic feature engineering using univariate time series. VEST extracts new features from embedding vectors representing a time series and selects the most important ones for combination with the original vector.

4 VEST: vector of statistics from time series

In this section, we propose and formalise VEST (for **Ve**ctor of **S**tatistics from **T**ime series), an automatic feature engineering process for univariate and numeric time series. VEST is specifically designed for forecasting problems. Given a time series $Y = \{y_1, \ldots, y_n\}$, the goal is to predict the value of the next observation, y_{n+1}. Following the formalisation presented in Sect. 2, we address time series forecasting as an auto-regressive task. The i-th observation of a time series, y_i, is modelled according to the i-th embedding vector $X_i = \{y_{i-1}, y_{i-2}, \ldots, y_{i-p}\}$, which represents the p previous observations.

VEST is based on the manipulation of the embedding vectors representing each observation of a time series. Particularly, the method contains three main steps, addressing feature generation (steps 1 and 2) and selection (3):

1. Transforming each embedding vector X into different representations (Sect. 4.1.1);
2. Summarising each representation into features using statistical functions (Sect. 4.1.2);
3. Selection of relevant features (Sect. 4.2).

We adopt an expand–reduce approach for feature engineering (do Nascimento Reis, 2019). In the first two steps of the methodology, we expand the representation of the data with a large set of features. In the final step, we reduce this representation and keep only the most relevant variables.

In the next subsections, we formalise these steps in more detail. The workflow for a given instance X_i is exemplified in Fig. 1.
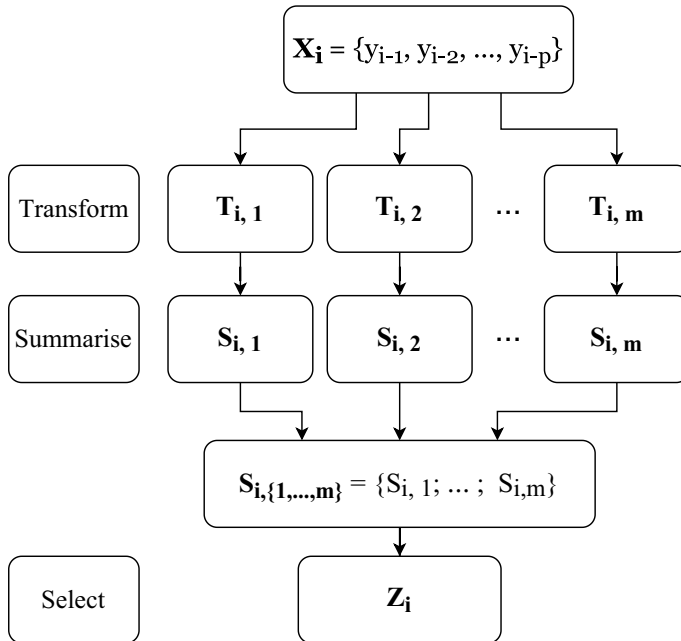
Fig. 1 Feature engineering workflow for a given embedding vector X_i. X_i is mapped into m different representations, $\{T_{i,1}, \ldots, T_{i,m}\}$. Each representation is summarised into a set of statistics $S_{i,m}$. For example, $S_{i,1}$ denotes a set of features constructed from the representation $T_{i,1}$. Finally, feature selection is carried out, leading to the final set of features Z_i

4.1 Feature generation

We base our feature generation process on the manipulation of each embedding vector $X \in \mathcal{X}$. In this sense, our approach is entirely endogenous. Exogenous features (e.g. holiday information) can also be essential to improve forecasting performance. However, such analysis is out of the scope of this work.

4.1.1 Transform operations

The first step of VEST is a transformation procedure. This procedure generates new representations from the original embedding vectors X. As we mentioned in Sect. 3.2, changing the representation of time series embedding vectors is beneficial for handling noise, and to focus on essential characteristics of the data (Esling & Agon, 2012). We hypothesize that different representations obtained by distinct transformation operations generate new, complementary information regarding the dynamics of the time series. Therefore, this combination of these new types of information can lead to improvements in forecasting performance which cannot be obtained by using each one of them separately. Formally, a transform operation maps an embedding vector X_i into another q-dimensional vector T:

$$\text{Transform}_j : \mathbb{R}^p \to \mathbb{R}^q$$
$$X_i \mapsto T_{i,j}$$

Essentially, X_i is mapped onto $T_{i,j}$, $\forall\ j \in \{1, \dots, m\}$. Hence, $T_{i,j}$ is a vector which denotes the j-th representation of the i-th embedding vector.

An example of a possible transform operation is differencing, which means the differences between consecutive observations. This transformation is often applied to time series to remove the trend component.

4.1.2 Summary operations

By applying distinct transform operations, an embedding vector has several representations (one for each such operation). The next step in the methodology is the application of summary operations. These operations compress each of the m representations of X_i ($\{T_{i,1}, T_{i,2}, \dots, T_{i,m}\}$) into a set of features through the application of statistical functions. A summary operation can be defined as follows:

$$\text{Summary}_k : \mathbb{R}^q \to \mathbb{R}$$
$$T_{i,j} \mapsto s_{i,j,k}$$

where Summary_k denotes the k-th summary operation, and $s_{i,j,k}$ denotes the feature obtained when applying the k-th summary operation to the j-th representation of the i-th embedding vector. Each $s_{i,j,k}$ is part of the set $S_{i,j}$, which represents the features describing $T_{i,j}$.

Essentially, each summary operation compresses a numeric vector into a scalar which summarises the current state of the time series in some way. A simple example is the arithmetic mean, which describes the central tendency.

4.2 Feature selection

The feature generation process described above produces a large number of features. This procedure may lead to a problem of high dimensionality and, consequently, overfitting. We introduce a feature selection procedure to cope with this issue.

Depending on the nature of the time series, the features extracted may have three problems: (1) they may not be applicable, which leads to missing values; (2) they may not vary enough across the observations and do not provide any information for forecasting; or (3) they may be highly correlated with each other. Concerning the first problem, we remove any feature with more than a certain percentage of missing values, *na_perc*. Features with a lower percentage of missing values are imputed using the median function. The second issue is dealt with by removing features with a low number of unique values. Specifically, we remove any feature whose number of unique values relative to the total number of observations is below *u_perc*. Finally, we apply a filter for removing correlated features. If a pair of features shows a level of correlation above *corr_perc*, one of them is discarded.

This process leads to the final set of features, Z_i. We concatenate this set with the original embedding vector X_i.

Table 1 Transform operations used in `VEST`

Operation	Description
I	The Identity transformation, in which each *X* is mapped onto itself
SMA	We apply a one-sided simple moving average which can be beneficial to smooth out spurious fluctuations and highlight the general trend. The number of periods is equal to the square root of the length of *X*, rounded to the nearest unit
DIFF	First differences are applied to transform the original embedding vector into one without trend. This transformation can help with the modelling of time series with a strong trend component
DIFF2	Second differences, which is equivalent to applying the **DIFF** operation twice to X_i. This transformation is useful for describing the curvature of the data
BC	Box-Cox transformation, for stabilising the variance of the time series. The transformation parameter is optimised using all the available observations according to Guerrero (1993) (minimizing the coefficient of variation)
SIN	Sine terms of order 1 of the Fourier series. This transformation captures the seasonality of the time series. We remark that the frequency of the time series must be available to compute these terms
COS	Similar and complementary to SIN, COS captures the cosine terms of order 1 of the Fourier series.
DWT	We apply a 1-level discrete wavelet transform using the Daubechies wavelet (Percival & Walden, 2006), and retrieve the coefficients of the respective detail signal

5 Experiments

This section presents the empirical experiments carried out to validate the proposed approach. First, we detail the transform and summary operations used in the feature generation process of `VEST` (Sect. 5.1). Then, we present the experimental setup (Sect. 5.2), describing the research question, case study, methods and respective hyper-parameters, and evaluation approach. We compare the proposed approach with state of the art approaches in Sect. 5.3. We perform a feature important analysis in Sect. 5.4. Finally, we analyse the impact of sample size in the results obtained (Sect. 5.5).

5.1 `VEST` setup

5.1.1 Transform and summary operations

The transform operations applied to each embedding vector X_i are described in Table 1 and the summary operations applied to each representation $T_{i,j}$ are described in Table 2.

The set of transform operations used try to capture the dynamics of the time series from distinct perspectives. Moreover, the list of summary operations contains several statistics which try to capture different components; from centrality and dispersion to chaos and stochastic randomness.

Overall, we apply eight different transformations and 32 different summary operations, leading to 256 different features before feature selection. Henceforth, we will employ the following notation to refer to a feature generated by `VEST`: *TransformFunction.SummaryFunction*. For example **DIFF.MEAN** represents the average value of the embedding vector representation when transformed with the differencing operation.

Table 2 Summary operations used in `VEST`

Operation	Description
MEAN	Arithmetic mean, which is used to estimate the average level of the vector
MDN	Median: similar to the mean, but more robust to outliers
SD	Standard deviation, as a measure of the overall dispersion in the vector
VAR	Variance of the vector, which also measures dispersion
IQR	Inter-quartile range, which is another measure of dispersion of the data, but more robust to outliers
RD	Relative Dispersion, which is estimated according to the ratio between the standard deviation of the vector and the standard deviation of the differenced vector (Wang et al., 2006)
MIN	Minimum value of the vector
MAX	Maximum value of the vector
LP	Last known point of the vector
SK	Skewness of the distribution of the vector, which is a measure of its asymmetry (Wang et al., 2006)
KRT	Kurtosis for describing the flatness of the data with respect to a normal distribution (Wang et al., 2006)
P05, **P95**	The 5th and 95th percentiles of the vector
ACC_1, ACC_2	Average (**ACC_1**) and standard deviation (**ACC_2**) of the acceleration of the vector, estimated according to the ratio between the simple moving average and the exponential moving average of equal period. In our experiments, the period for computing the moving averages was set to the squared root of the length of the vector, rounded to units
BP	Level of auto-correlation, which is estimated using a Box-Pierce test statistic (Box & Pierce, 1970; Wang et al., 2006)
PACF	Average value of the partial auto-correlation function of the vector up to 10 lags
ACF	Average value of the auto-correlation function of the vector up to 10 lags
LRD1 LRD2	Long-range dependence, estimated using the Hurst exponent approach with wavelet transform with 1 (**LRD1**) and 2 moments (**LRD2**) (Wang et al., 2006)
SLP	Slope of the vector which describes its overall steepness (Prudêncio & Ludermir, 2004)
NORM	Euclidean norm of the vector, which captures its total energy
NO	Number of outliers, estimated according to the number of observations above or below 1.5 times the inter-quartile range
AMP	Average amplitude of the fast Fourier transform of the vector
STEP	Binary random variable which denotes the presence of a step change (Lemke & Gabrys, 2010). This statistic detects structural breaks in the data
PEAK_I, PEAK_D	Number of local maxima (**PEAK_I**) and local minima (**PEAK_D**) in the vector (Lemke & Gabrys, 2010). These statistics describe the level of oscillation of the data
OD	Overall direction of the vector, estimated by the difference between the number of times the vector increases and the number of times the vector decreases
PV_ST, PV_LT	Short-term and long-term variability, respectively, estimated using the Poincaré plot (Brennan et al., 2001)
MLE	Maximum Lyapunov exponent, which quantifies the chaotic level of a time series (Wang et al., 2006)

Setup. We set the *na_perc* value to 70. As such, we remove features which have more than 70% of its values missing. Also, we set the *u_perc* threshold to 1. Therefore, we

remove features where the percentage of unique values is below 1% of the total number of observations. The feature correlation threshold (*corr_perc*) was set to 95.

5.2 Experimental design

The main research question addressed in this paper is the following:

Does VEST, *an automatic feature engineering procedure, improve forecasting performance relative to a pure auto-regressive approach?*

Our experiments to answer this question can be split into the following items:

- **RQ1**: Effect of VEST on the predictive performance of the state of the art pure auto-regressive approach. We assess the significance of results according to Bayesian methods;
- **RQ2**: Comparison of the forecasting performance relative to state of the art approaches, such as ARIMA and exponential smoothing;
- **RQ3**: Sensitivity to different learning algorithms;
- **RQ4**: Analysis of the different feature selection approaches;
- **RQ5**: Analysis of the importance scores of each transformation and summary operations (Sect. 5.1);
- **RQ6**: Sensitivity to different time series sample size.

5.2.1 Data

We used time series from two sources. From the *tsdl* benchmark library (Hyndman & Yang, 2019), we selected all the univariate time series with at least 1000 observations and which have no missing values. This represents 55 time series. These show a varying sampling frequency (e.g. daily) and are from different domains of application. For a complete description of these time series, we refer to the their source (Hyndman & Yang, 2019). We also included 35 time series used by Cerqueira et al. (2019). Essentially, from the set of 62 series used by the authors, we selected those with at least 1000 observations and that were not originally from the *tsdl* database (since these were already retrieved as described above). We refer to the work by Cerqueira et al. (2019) for a description of those time series.

5.2.2 Parameter setting

For each time series, we optimise the embedding dimension using validation data, testing values from 10 to 30. The chosen embedding dimension p is the one minimising the error (Sect. 5.2.4 describes the evaluation metric). In this analysis, we train a model according to pure auto-regressive forecasting models (i.e., no feature engineering is involved at this point). We set the minimum value for searching the embedding dimension to 10 to guarantee a reasonable number of observations for computing the transform and summary operations of VEST.

We focus on two learning algorithms. One is the cubist method (Kuhn et al., 2014), which is a variant of the model tree proposed by Quinlan (1993); This method is competitive in time-dependent data (Ikonomovska et al., 2011; Cerqueira et al., 2019). We also use the lasso (Tibshirani 1996) regression algorithm. Each one of the methods was optimised according to a grid search using validation data.

We will present results that quantify the importance of each feature across the 90 problems. We resort to the RReliefF (Robnik-Šikonja & Kononenko, 1997) method for this task. RReliefF (for Regressional ReliefF) extends ReliefF for numerical prediction problems. It estimates the importance of each feature in a data set by measuring the variability of the values of the features in the neighbourhood the observations. This method has been shown to have a connection to impurity measures (Robnik-Šikonja & Kononenko, 1997).

5.2.3 Methods

The learning algorithms indicated above were trained according to the following procedures:

– AR: A pure auto-regressive process, where the value of the next set of observations is predicted according to the most recent *p* values. This is the typical approach to tackle time series forecasting problems;
– AR+VEST: The proposed approach – the combination of AR with the features obtained with VEST;
– VEST: A baseline which discards the AR component and models the future behaviour of the time series using only the features obtained with VEST;
– AR+BT: Variant of AR+VEST, in which the feature selection approach is different: We use the feature from only a single representation. For each time series, we pick the transformation which maximizes feature importance (according to RReliefF). The importance scores are average across the available summary operations. In other words, this variant contains all the summary operations detailed on Table 2, but are computed only for the best estimated transformation;
– AR+BF: Another variant of AR+VEST, in which we select a single transformation for summary operation. This is similar to the variant AR+BT described above. The difference is that, in this case, the single transformation is picked for each summary operation. This selection is also based on feature importance. To be precise, for each time series and for each summary operation, we select the transformation that maximizes feature importance.

Additionally, we also include ARIMA, ETS, and TBATS in the experimental setup. These methods are state-of-the-art approaches for time series forecasting. They establish a reference to assess whether the results obtained here are acceptable or not. We resort to the implementations provided by the forecast R package (Hyndman, 2014), which automatically tunes these methods to an optimal parameter setting.

5.2.4 Evaluation

We use a holdout repeated in multiple testing periods as the estimation method according to Cerqueira et al. (2019). We perform 10 repetitions of this procedure. The training size in each repetition was set to 60% of the total number of observations, while the subsequent 10% of observations are used for testing. In each repetition, part of the training data (also 10% of it) was used as a validation set to optimise parameters, such as the embedding dimension or the parameters of the learning algorithms.

Regarding the evaluation metric, we use the mean absolute scaled error (MASE), which is a typical measure of forecasting performance (Hyndman, 2006). We average the loss of
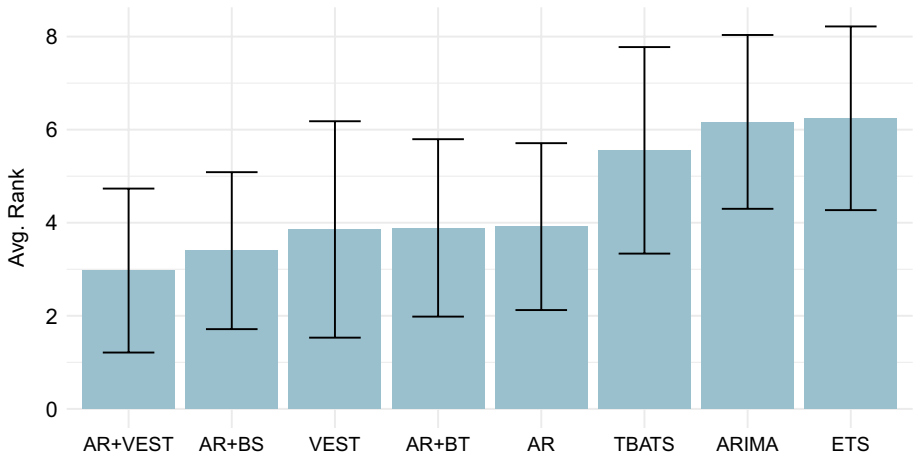
Fig. 2 The average rank, and respective standard deviation, of each method across the 90 time series when using cubist as learning algorithm

each method across the repetitions of the holdout procedure described above. We evaluate the statistical significance of the results according to a Bayesian analysis (Benavoli et al., 2017). In particular, we applied the Bayes sign test to compare pairs of methods across multiple problems. In the next section, we specify the setup of the test. For a thorough read on Bayesian analysis for comparing predictive models, we refer to the work by Benavoli et al. (2017).

5.3 Results

In order to have a commensurable metric across data sets, we compute the percentage difference between the MASE of each approach and a benchmark model. We use `AR+VEST` as the benchmark as it represents the proposed approach that combines auto-regression with automatic feature engineering. We formalise the percentage of difference computation as follows:

$$\frac{L_a - L_{\text{AR+VEST}}}{L_{\text{AR+VEST}}} * 100 \tag{1}$$

where $L_{\text{AR+VEST}}$ and L_a represent the loss of the model `AR+VEST` and the loss of model a (the one is under comparison), respectively. We perform a Bayesian analysis of the results using the Bayes sign test (Benavoli et al., 2017). We define the *region of practical equivalence* (Benavoli et al., 2017) (ROPE) to be the interval $[-2.5, 2.5]$. Essentially, this means that the performance level of these two methods are nearly indistinguishable if the percentage difference in predictive performance between them falls within this interval.

We start by analysing the average rank, and respective standard deviation, of each method. This is reported in Fig. 2 using cubist as learning algorithm. A method with rank 1 in a task means that it was the best performing one in that task. The average rank describes the average position of each method relative to the remaining ones. `AR+VEST` shows the best average rank score, which shows the usefulness of the proposed approach.
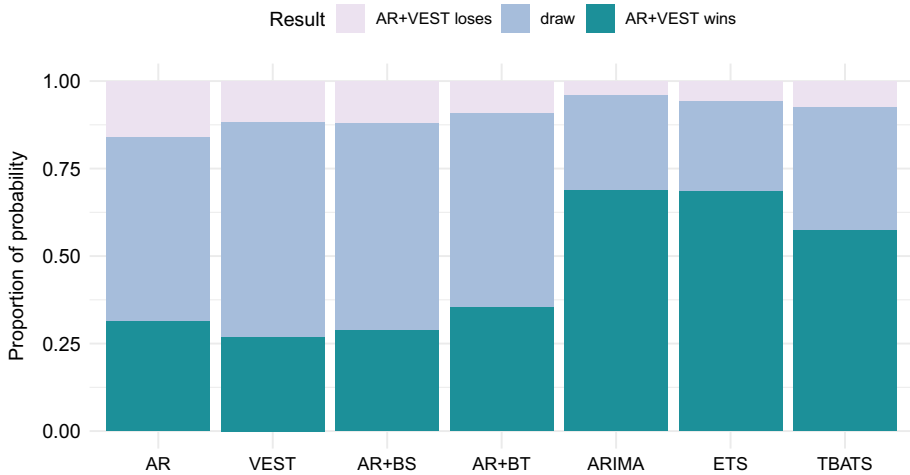
Fig. 3 Probability of each method winning, drawing, or losing significantly against AR+VEST (the proposed method) according to the Bayes sign test. Results shown for the cubist method

In terms of significance analysis, Fig. 3 shows the probability that the respective method wins, draws (result within the ROPE), or loses significantly, against the proposed model (AR+VEST) also when using the cubist learning algorithm. AR+VEST significantly outperforms the standard auto-regressive model (AR) with around 30% probability (**RQ1**). The opposite scenario occurs with around 17% probability. In the remaining cases, the results are within the ROPE, which means the approaches are statistically equivalent. AR+VEST is also significantly better relative to state of the art forecasting approaches, including ARIMA, TBATS, and ETS (**RQ2**). This corroborates previous experiments shown by Cerqueira et al. (2019).

These results show important evidence that feature-based forecasting is worthwhile, and may be important to improve forecasting performance.

Figures 4 and 5 are similar to Figs. 2 and 3, but the analysis is carried out using the lasso learning algorithm. Although not identical, the illustration shows that performance gains are also obtained when using this method (**RQ3**).

Besides state of the art forecasting approaches, the results also indicate that AR+VEST outperforms three variants: VEST, AR+BT and AR+BS. VEST denotes the approach that discards the auto-regressive attributes and uses only the features derived from the proposed framework to forecast the next value of the time series. However, the results show that combining AR with VEST is critical to the performance obtained. By itself, VEST shows a competitive performance, but does not provide a consistent advantage.

Regarding AR+BT and AR+BS, these variants provide a different approach for selecting the features from VEST. We devised AR+BT (the approach which selects a single transformation for each time series) to show the usefulness of multiple representations in a given problem. On the other hand, by outperforming AR+BS (the approach that uses a single transformation per summary operation), we show that multiple transformations are useful for a given summary operation even within the same problem. We dealt with eventual redundancies with a simple correlation filter, as described in the experimental design (**RQ4**).
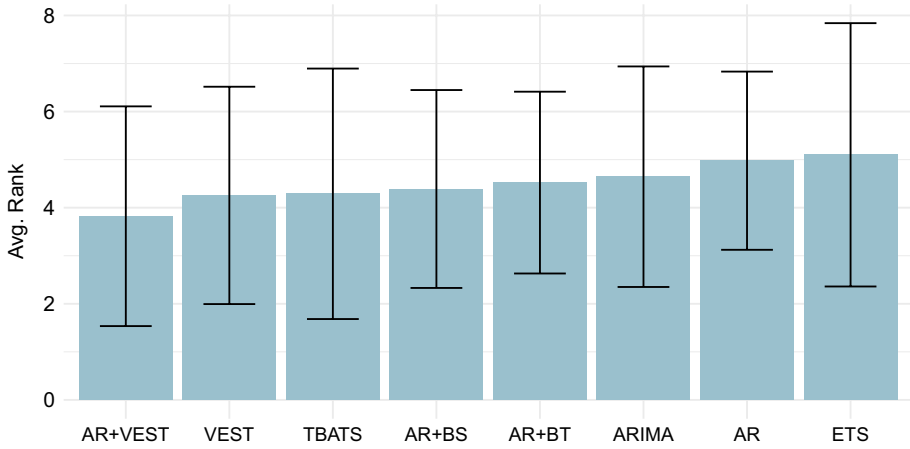
Fig. 4 The average rank, and respective standard deviation, of each method across the 90 time series
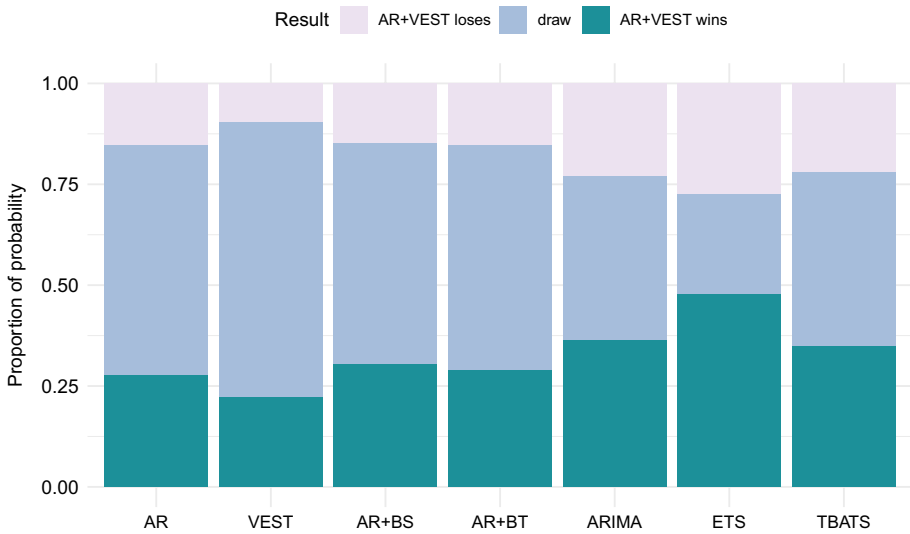

Fig. 5 Probability of each method winning, drawing, or losing significantly against AR+VEST (the proposed method) according to the Bayes sign test. Results show for the lasso method

5.4 Feature importance

In the previous section, we presented significant empirical evidence that the use of VEST can significantly improve forecasting performance. In this section, we dive deeper into this matter by analysing the importance of the features used in the development of models. This covers research question **RQ5**.
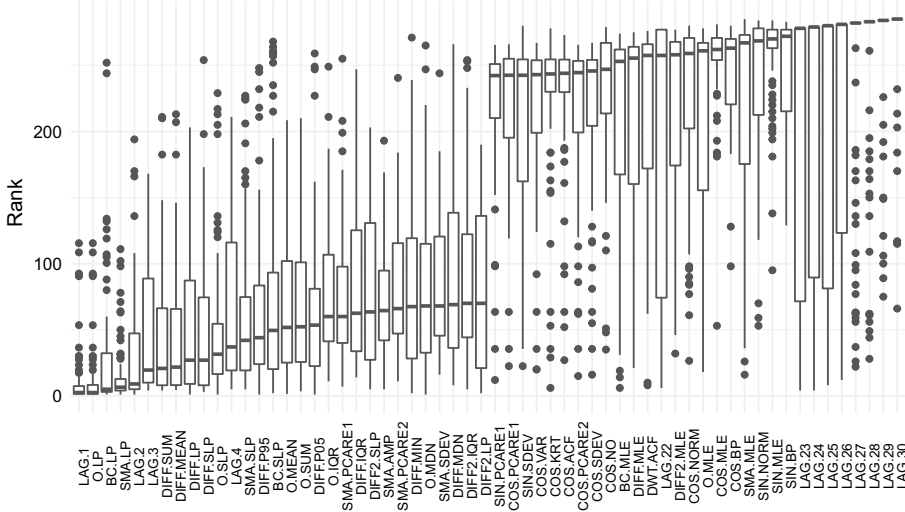
Fig. 6 Distribution of rank importance of the top 30 and bottom 30 features. The rank of a particular feature in a given problem is computed according to its importance

5.4.1 Rank distributions

We start by analysing the distribution of the rank of the feature importance across the 90 time series. We proceed as follows.

1. We measure the RReliefF of each feature. This score is averaged across the repetitions of the repeated holdout procedure;
2. We compute the rank of each feature according to its score of importance across the 90 time series. A feature with rank 1 in a given time series has the best score of importance in that problem. We split the computation of ranks into three parts according to the following criteria:

   - All operations: We compute the rank of all features irrespective of the underlying representation;
   - Representation: We compute the rank of each transformation. Specifically, we average the score of the importance of the features for each representation. For example, the average importance of all features using the DIFF representation. In this analysis, we include the importance of the past lags of the series (denoted as LAG variables);
   - Summary function: We also compute the rank of each summary operation. Similarly as above, we average the score of importance across summary operation to obtain the overall importance of the respective function.

Overall rank Figure 6 shows the results of the overall rank as a set of boxplots (one for each feature), which are ordered by median importance rank (lower values are better). The names of the features (x-axis) follows the convention described before. In the interest of conciseness, we only show the top and bottom 30 features in terms of median rank.

The feature with the best median rank is **LAG.1**, which represents the last known value of the time series in a given point in time. Figure 6 clearly shows the advantage of methods
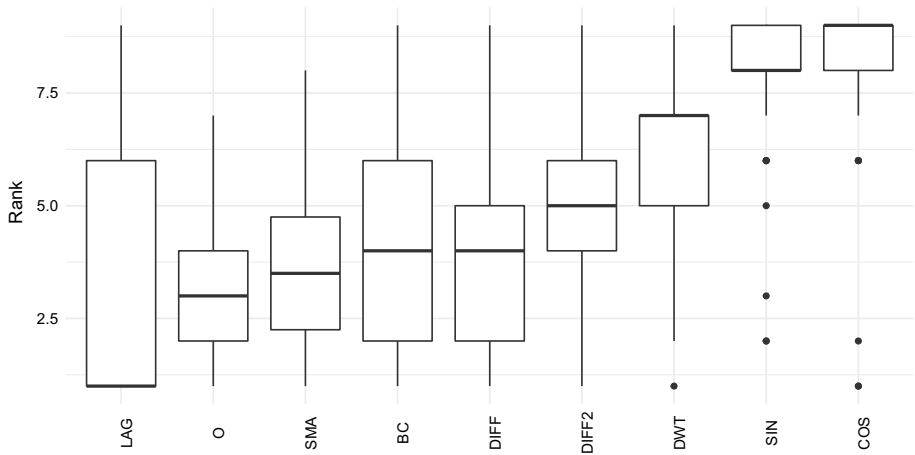
Fig. 7 Distribution of rank importance for each type of feature

to systematically generate large numbers of new features, when compared to, typically, a few features generated manually, based on domain knowledge. We observe that, overall, there is a great dispersion in the rank importance, showing that different features are more important in different time series. In fact, even those features with high median rank are among the most important in some of the problems.

Rank by representation Figure 7 provides a similar analysis as Fig. 6, but combines the results by representation, as explained above. The boxplots provide additional evidence for two observations made previously. Firstly, it shows that, although the obtained features from VEST improve predictive performance, the previous points of the time series (LAG features) provide useful information. In fact, they obtain the best median rank. Secondly, the high dispersion of the rank distributions shows that there is no particular representation which is the most appropriate for all time series. This provides additional evidence of the usefulness and complementarity of the different representations, as observed earlier.

Rank by summary operation Figure 8 shows a similar analysis but referring to each summary operation. Again, the boxplots show high dispersion suggesting that different statistics are more valuable in different tasks. The statistic with the best median rank is LP, which denotes the last point of the respective transformation.

5.5 Impact of sample size

We focused the experimental setup on high frequency time series. This type of data sets is increasingly relevant in many practical applications due to the widespread adoption of sensors. High frequency time series are typically associated with larger sample sizes relative to lower frequency ones. We hypothesise that the sample size is important when performing feature engineering with a method such as VEST. In a small data set additional attribute variables may lead to over-fitting issues due to the *curse of dimensionality*. Therefore, it is important to collect a reasonable amount of observations for feature engineering.

We test the hypothesis above by repeating the experiments with increasing sample size values, similarly to Cerqueira et al. (2019) (**RQ6**). To be more precise, we start by truncating the sample size of the time series to 3000 observations. Only 42 of the 90 time series
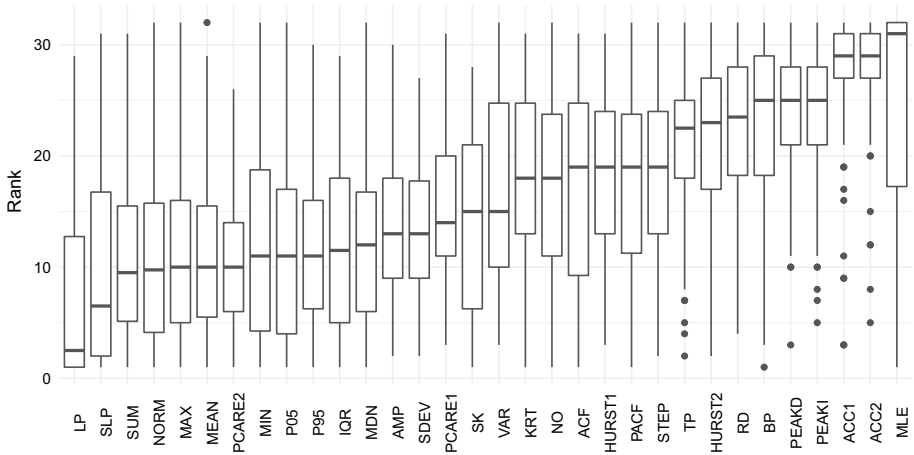
Fig. 8 Distribution of rank importance for each summary operation

had enough sample size, and we focus this analysis on that subset of problems. Afterwards, we repeated the experiments (as described in Sect. 5.2) in 30 different samples sizes from 100 to 3000 in steps of 100 observations ({100, 200, … , 3000}). We remark that, in this particular experiment, we focus on a simple holdout estimation method in which 80% of the initial observations are used for training and the subsequent 20% data points are used for testing. Accordingly, we evaluate the performance of AR+VEST relative to other approaches which do not use VEST. In this experiment, we remove the variants of the proposed method, and keep only **AR**, **ETS**, **TBATS**, and **ARIMA**. The performance is evaluated as follows: for each problem as for each sample size we compute the MASE error of each approach. Then, each method is ranked according to this error (lower error gives lower rank). We average the rank of each method across the 42 time series in each sample size experiment. This allows us to describe how the average rank of each method evolves as the sample size increases. We remark that we resort to the rank, as opposed to the MASE loss, because it is non-parametric and robust to outliers. Finally, we remark that we focus on the cubist learning algorithm for this analysis in the interest of conciseness. The results are similar when using the lasso algorithm.

The results are presented in Fig. 9, which shows the average rank of each method across the 42 time series with an increasing sample size. The results show that, when the sample size is small, AR+VEST shows worse results than all other methods, including AR and the state of the art forecasting methods ARIMA, ETS, and TBATS. However, as the sample size increases, AR+VEST becomes the approach with best average rank. We remark that the average rank scores may be slightly different from the analysis shown previously as there are only 42 time series under analysis in this scenario.

6 Discussion

The experiments carried out in the previous section show the benefits of using VEST for time series forecasting tasks. In this section, we discuss the results obtained and point future directions of research.
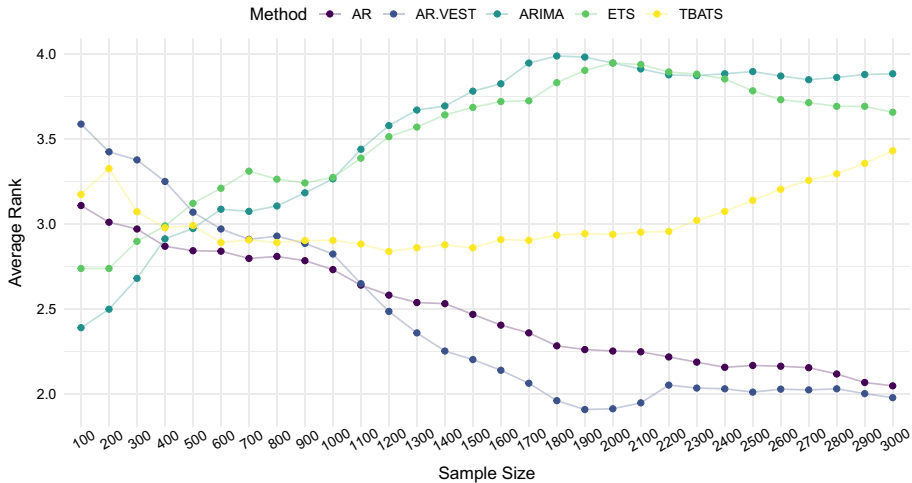
Fig. 9 The average rank (computed across the 42 time series) of each method as the sample size of the time series increases. Lower values are better

6.1 Main results

VEST is a framework for automatically extracting relevant features from the embedding vectors representing the time series. We showed the usefulness of VEST to tackle time series forecasting tasks based on an extensive set of experiments. When the features generated by VEST are combined with a state of the art auto-regressive model (AR), forecasting performance significantly improves relative to only using AR.

We explored these results from different perspectives. Particularly, we presented an analysis which suggests that there is no specific representation or summary statistics which is more appropriate for all time series problems. Even within a single time series, the results suggest that applying summary operations to different representations is important for forecasting performance. This outcome shows the potential benefit of using an automatic approach to extract meaningful features for this type of data. Rather than finding a single feature that improves results across multiple problems, VEST obtains a set of features, each one of which is very important for a small, particular subset of the problems, although not very relevant on the remaining ones.

We believe our work is relevant for automated machine learning frameworks, especially to enable professional with low technical skill to develop accurate forecasting models efficiently (Taylor & Letham, 2018).

6.2 Points for improvement

Despite significant gains in forecasting performance, we believe it is possible to improve the proposed feature engineering process.

VEST is designed as a brute force approach. It works by testing different representations, which are then summarised using different statistics. Those with low feature importance are removed using a feature selection filter. The key factor for the gains in performance is the predictive quality of the features that are tested. In this context, a potentially interesting research line is to develop a method for selecting apriori which transform

and summary operations should be computed. For example, do Nascimento Reis (2019) attempts to use meta-learning to predict whether a given feature is going to improve predictive performance. A similar approach could be developed for extending VEST. Other possible interesting solutions are landmarkers (Pfahringer & Giraud-Carrier, 2000) or bayesian optimisation (Rasmussen, 2003). Notwithstanding, we remark that, in the proposed framework, the different transform operations are independent of each other, and so are summary ones. Therefore, the processes within each step can run in parallel.

Another point of improvement for VEST is long-term analysis. VEST focuses on extracting information from the past p lags. In other words, feature extraction is self-contained within each embedding vector. In future work, we plan to extend the approach to include a longer-term analysis and extract information across embedding vectors. Such analysis enables a more long-term perspective on the dynamics of the time series. An example of a long-term feature is one attempting to capture a "number of observations since an outlier occurred".

Although the pool of operations applied cover many properties of time series, the set of transform and summary operations can be increased. Other transformations could be carried out, for example, seasonal adjustment or discrete cosine transform. One could also combine available transform operations, e.g. a transformation which applies the operations BC (Box-Cox transformation) and DIFF (first differences transformation) sequentially (c.f. Sect. 5.1.1).

As we mentioned previously, VEST is designed to extract endogenous features from time series. Notwithstanding, external information may be crucial for building accurate predictive models. With additional time series as explanatory variables, the number of operations to be carried out may be too high for a brute force approach. Thus, the ideas outlined in the second paragraph of this section may be important in these scenarios.

Our experiments are based on 90 time series with a high sampling frequency (daily or higher). Research is still necessary to show the impact of feature engineering in time series with lower sampling frequency. In Sect. 5.5, we showed that time series sample size is important for the proposed feature engineering solution.

Another point for improvement for VEST is multi-step forecasting. During our experiments, we did not find enough evidence that VEST may be better than AR for multi-step forecasting. We intend to explore this topic in future research.

Finally, another interesting research line is that of global forecasting models (Salinas et al., 2020). These approaches pool multiple time series and fit a single predictive model with them. This research direction may be helpful to overcome the low sample size issue of VEST.

7 Summary

Time series forecasting is a relevant predictive task in many domains of application. Data-driven organisations rely on forecasting systems to cope with future uncertainty and support their decision-making process.

One of the most important tasks in machine learning is feature engineering. However, this task still requires considerable manual effort and expertise from practitioners (Kaul et al., 2017). This lead to increasing demand for approaches that automate this part of machine learning projects.

In this work, we present a novel automatic procedure for feature engineering using time series data. The proposed approach, called VEST, is specifically designed to tackle time series forecasting problems. VEST is based on the manipulation of the embedding vectors, which represent the past recent observations used to predict the future ones. It works by transforming time series sub-sequences into distinct representations. Describing a time series using multiple representations may be useful for capturing its dynamics from different perspectives. Each representation is summarised using statistical functions, such as the mean. After a feature selection process, the final set of features across the available representations is coupled with an auto-regressive model.

We validated the proposed approach using an extensive set of experiments, comprised of 90 time series from several domains of application. The results show that the features provided by VEST, along with auto-regression, lead to significant gains in forecasting performance.

In future work, we will extend the approach to other forecasting scenarios, for example, multivariate time series or multi-step forecasting. VEST is publicly available online.

References

Barandas, M., Folgado, D., Fernandes, L., Santos, S., Abreu, M., Bota, P., et al. (2020). Tsfel: Time series feature extraction library. *SoftwareX, 11,* 100456.

Benavoli, A., Corani, G., Demšar, J., & Zaffalon, M. (2017). Time for a change: A tutorial for comparing multiple classifiers through bayesian analysis. *The Journal of Machine Learning Research, 18*(1), 2653–2688.

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: Forecasting and control*. Wiley.

Box, G. E., & Pierce, D. A. (1970). Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association, 65*(332), 1509–1526.

Brennan, M., Palaniswami, M., & Kamen, P. (2001). Do existing measures of poincare plot geometry reflect nonlinear features of heart rate variability? *IEEE Transactions on Biomedical Engineering, 48*(11), 1342–1347.

Cerqueira, V., Torgo, L., & Mozetic, I. (2019). *Evaluating time series forecasting models: An empirical study on performance estimation methods*. arXiv preprint arXiv:1905.11744.

Cerqueira, V., Torgo, L., Oliveira, M., & Pfahringer, B. (2017). Dynamic and heterogeneous ensembles for time series forecasting. In *2017 IEEE international conference on data science and advanced analytics (DSAA)* (pp. 242–251). https://doi.org/10.1109/DSAA.2017.26.

Cerqueira, V., Torgo, L., Pinto, F., & Soares, C. (2019). Arbitrage of forecasting experts. *Machine Learning, 108*(6), 913–944.

Cerqueira, V., Torgo, L., & Soares, C. (2019). *Machine learning vs statistical methods for time series forecasting: Size matters*. arXiv preprint arXiv:1909.13316.

Chatfield, C. (2000). *Time-series forecasting*. CRC Press.

Christ, M., Kempa-Liehr, A.W., & Feindt, M. (2016). *Distributed and parallel time series feature extraction for industrial big data applications*. arXiv preprint arXiv:1610.07717.

do Nascimento Reis, G. F. (2019). Automated feature engineering for classification problems.

Esling, P., & Agon, C. (2012). Time-series data mining. *ACM Computing Surveys (CSUR), 45*(1), 12.

Fulcher, B. D., & Jones, N. S. (2017). hctsa: A computational framework for automated time-series phenotyping using massive feature extraction. *Cell Systems, 5*(5), 527–531.

Guerrero, V. M. (1993). Time-series analysis supported by power transformations. *Journal of Forecasting, 12*(1), 37–48.

Guyon, I., & Elisseeff, A. (2006). An introduction to feature extraction. In I. Guyon, M. Nikravesh, S. Gunn, & L. A. Zadeh (Eds.), *Feature extraction* (pp. 1–25). Springer.

Hyndman, R., & Yang, Y. (2019). *tsdl: Time Series Data Library*. https://finyang.github.io/tsdl/, https://github.com/FinYang/tsdl.

Hyndman, R. J., & with contributions from George Athanasopoulos, Razbash, S., Schmidt, D., Zhou, Z., Khan, Y., Bergmeir, C., & Wang, E. (2014). forecast: Forecasting functions for time series and linear models. R package version 5.6.

Hyndman, R. J. (2006). Another look at forecast-accuracy metrics for intermittent demand. *Foresight: The International Journal of Applied Forecasting, 4*(4), 43–46.

Ikonomovska, E., Gama, J., & Džeroski, S. (2011). Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery, 23*(1), 128–168.

Kahn, K. B. (2003). How to measure the impact of a forecast error on an enterprise? *The Journal of Business Forecasting, 22*(1), 21.

Kang, Y., Hyndman, R. J., & Smith-Miles, K. (2017). Visualising forecasting algorithm performance using time series instance spaces. *International Journal of Forecasting, 33*(2), 345–358.

Kanter, J. M., & Veeramachaneni, K. (2015). Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE international conference on data science and advanced analytics (DSAA)* (pp. 1–10). IEEE.

Katz, G., Shin, E. C. R., & Song, D. (2016). Explorekit: Automatic feature generation and selection. In *2016 IEEE 16th international conference on data mining (ICDM)* (pp. 979–984). IEEE.

Kaul, A., Maheshwary, S., & Pudi, V. (2017). Autolearn—Automated feature generation and selection. In *2017 IEEE international conference on data mining (ICDM)* (pp. 217–226). IEEE

Keogh, E., Lonardi, S., & Ratanamahatana, C. A. (2004). Towards parameter-free data mining. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 206–215).

Khurana, U., Turaga, D., Samulowitz, H., & Parthasrathy, S. (2016). Cognito: Automated feature engineering for supervised learning. In *2016 IEEE 16th international conference on data mining workshops (ICDMW)* (pp. 1304–1307). IEEE.

Kuhn, M., Weston, S., Keefer, C., & Ross Quinlan, N. C. C. (2014). Cubist: Rule- and instance-based regression modeling. R package version 18.

Lam, H. T., Thiebaut, J. M., Sinn, M., Chen, B., Mai, T., & Alkan, O. (2017). *One button machine for automating feature engineering in relational databases*. arXiv preprint arXiv:1706.00327.

Lemke, C., & Gabrys, B. (2010). Meta-learning for time series forecasting and forecast combination. *Neurocomputing, 73*(10–12), 2006–2016.

Lin, J., Keogh, E., Lonardi, S., & Chiu, B. (2003). A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on research issues in data mining and knowledge discovery* (pp. 2–11).

Lubba, C. H., Sethi, S. S., Knaute, P., Schultz, S. R., Fulcher, B. D., & Jones, N. S. (2019). catch22: Canonical time-series characteristics. *Data Mining and Knowledge Discovery, 33*(6), 1821–1852.

Montero-Manso, P., Athanasopoulos, G., Hyndman, R. J., & Talagala, T. S. (2020). Fforma: Feature-based forecast model averaging. *International Journal of Forecasting, 36*(1), 86–92.

Oliveira, M., & Torgo, L. (2014). Ensembles for time series forecasting. In *ACML proceedings of Asian conference on machine learning. JMLR: Workshop and conference proceedings*.

Paras, S. M., Kumar, A., & Chandra, M. (2009). A feature based neural network model for weather forecasting. *International Journal of Computational Intelligence, 4*(3), 209–216.

Percival, D. B., & Walden, A. T. (2006). *Wavelet methods for time series analysis* (Vol. 4). Cambridge University Press.

Pfahringer, B., & Giraud-Carrier, C. (2000). Meta-learning by landmarking various learning algorithms. pp. 743–750.

Pinto, F., Soares, C., & Mendes-Moreira, J. (2016). Towards automatic generation of metafeatures. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 215–226). Springer.

Prudêncio, R. B., & Ludermir, T. B. (2004). Meta-learning approaches to selecting time series models. *Neurocomputing, 61,* 121–137.

Quinlan, J. R. (1993). Combining instance-based and model-based learning. In *Proceedings of the tenth international conference on machine learning* (pp. 236–243).

Rasmussen, C. E. (2003). Gaussian processes in machine learning. In *Summer school on machine learning* (pp 63–71). Springer

Robnik-Šikonja, M., & Kononenko, I. (1997). An adaptation of relief for attribute estimation in regression. In *Machine learning: Proceedings of the fourteenth international conference (ICML'97)* (Vol. 5, pp. 296–304).

Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. (2020). Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting, 36*(3), 1181–1191.

Takens, F. (1981). Dynamical systems and turbulence. In: *Warwick 1980: Proceedings of a symposium held at the University of Warwick 1979/80, chap. Detecting strange attractors in turbulence* (pp. 366–381). Springer

Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician, 72*(1), 37–45.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological), 58*(1), 267–288.

Wang, X., Smith, K., & Hyndman, R. (2006). Characteristic-based clustering for time series data. *Data Mining and Knowledge Discovery, 13*(3), 335–364.