Best Practices for Time Series Forecasting (Tutorial)

André Bauer, Marwin Züfle, Nikolas Herbst, and Samuel Kounev
University of Würzburg, Germany
Email: andre.bauer@uni-wuerzburg.de, marwin.zuefle@uni-wuerzburg.de, {firstname}.{lastname}@uni-wuerzburg.de

***Abstract*—In a fast-paced world, software systems require autonomic management. To enable accurate and proactive autonomic systems, reliable time series forecasting methods are needed. In this tutorial paper, we guide the reader step-by-step through different forecasting steps. In each step, we highlight best practices and present available approaches. That is, we explain how to pre-process the data and retrieve features. Then, the model selection and fitting steps are shown. Finally, we discuss the forecasting itself and its evaluation. For the individual steps, we provide some basic code snippets in the language R.**

***Index Terms*—Forecast, Pre-processing, R, Benchmarking, Metrics, Time series**

I. Introduction

Nowadays, software systems are pushed to their limits, on the one hand, by the fast living and changing requirements of their users and, on the other hand, by a massive amount of data that they create or have to process. Although cloud computing is a paradigm that allows facing the increasing size and complexity of modern software, the management is so demanding that automatic systems are required. Based on observations, systems plan and execute actions either to adjust to environment or to adapt to environment. However, when triggering actions based on observations, the system only reacts to past events. These actions have an inherent delay that may lead to severe problems. To upgrade the systems to be proactive, the systems require accurate and reliable forecasting methods.

A forecasting task can usually be divided into eight basic steps: (i) problem definition, (ii) data analysis, (iii) data pre-processing, (iv) feature engineering, (v) method selection, (vi) model fitting, (vii) forecasting, and (viii) evaluation. We guide the reader through the steps (iii) to (viii).

II. Prerequisite for the R snippets

For the following R code example snippets, we use the time series AirPassengers [1], which shows the monthly totals of international airline passengers from 1949 to 1960. Further, the following R packages are required: stats, datasets, forecast, and AnomalyDetection[1]. Due to the simplicity and limited space, we show only some basic excerpts to highlight each forecasting task.

```
# Set variables
air <- AirPassengers
freq <- 12 # monthly values
len <- length(air)
```

[1]AnomalyDetection package: https://github.com/twitter/AnomalyDetection

III. Data Pre-processing

Before the forecast can be conducted, the data have to be analysed and pre-processed. As most forecasting methods cannot handle missing values, these values have to be reconstructed, e.g., using interpolation. Also, the data should be checked if it contains seasonal patterns since there are forecasting methods that cannot handle seasonal data. If the data is seasonal, the suitable forecasting methods require the frequency of the time series. In the case that the frequency is not known, the most dominant frequency (e.g., daily, hourly, and yearly) can be estimated, for example, with periodograms [2]. We also recommend removing anomalies from the data set. One way is to use a generalised extreme studentized deviate test [3] to detect anomalies and replace these values with surrounding values.

```
# Find anomalies
AnomalyDetectionVec(x=air, period=freq)
```

IV. Feature Engineering

One possibility, which may lead to a better forecast, is transforming the time series to gain a simpler model. To this end, the data is adjusted by applying a Box-Cox transformation [4] as it reduces both variance and multiplicative effects. The Box-Cox transformation offers logarithms and power transformations. The choice of the function depends on the transformation parameter, which can be estimated by the method proposed by Guerrero [5].

```
# Box-Cox transformation
lambda <- BoxCox.lambda(x=air)
air <- BoxCox(x=air,lambda=lambda)
```

Another possibility is to decompose the time series into components and use the parts either for modifying the data (e.g., removing the trend or seasonality) or as additional features. Many different decomposition methods lead to diverse components. In this tutorial, STL (Seasonal and Trend decomposition using Loess) [6] is considered for time series decomposition. STL is a commonly used method that decomposes the time series into the components trend, season, and irregular, often called noise.

```
# Decompose time series
astl <- stl(x=air,s.window="periodic",t.window=len/2)
```

To add further information to the forecasting method, various features can be extracted from a time series such as further seasonal patterns, categorical information, Fourier terms, and many more.

V. Method Selection

After preparing the data and selecting proper features, the forecasting method needs to be chosen. According to the "*No-Free-Lunch Theorem*" [7], there is no forecasting method that performs best for all time series. Instead, the operator has to select a forecasting method from the domains: (i) statistical models, (ii) machine learning, and (iii) hybrid methods. Again, each of these categories contains various different forecasting methods. Hybrid models combine multiple individual forecasting methods to build a new method. Forecasting methods of this type typically show a lower variance in their forecasting accuracy, but also less precise results. Although there is no best forecasting method, a suitable forecasting method can be selected on the basis of time series characteristics.

VI. Model Fitting

The choice of the method influences the features, which can be used for the forecast. That is, some forecasting methods can only handle one feature; some can handle multiple features. Note that the selection of the features has an impact on the resulting forecast error. In other words, the selection has to be done carefully. There are different methods like (i) correlation analysis also known as filter, (ii) wrapper (i.e., adding or removing features iterative), and (iii) embedded (i.e., the selection is already part of the forecasting method). During the training of the model, the risk of over-fitting may occur, as the best model does not always lead to the best forecast. To counteract the over-fitting problem, the historical data can be split into train and test data and internal validations can be conducted. For estimating the forecasting error, the structural error can be estimated to get an upper-bound for the forecast error, or the internal model error can be taken into account.

```
# Model fitting
season <- astl$time.series[, 1]
air <- air - season
fit <- ets(y=ts(air))
```

VII. Forecasting

The forecast step itself can be distinguished into one-step-ahead or multi-step-ahead forecast. While performing a one-step-ahead forecast, only the next value is forecast. Scientific papers mostly do this approach. However, using the naïve forecast (i.e., using the last value of the history as forecast) can be more accurate than using another method. In contrast, a horizon of values is forecast at one time while using multi-step-ahead forecasts. Another approach is to perform multiple, subsequent one-step-ahead forecasts and use the last forecast as the most recent value. A useful way to report the uncertainty of the forecast is to display the prediction intervals in addition to the forecast values. The prediction intervals are based on the model error.

```
# Forecasting
sfc <- tail(rep(x=season, length.out=len+10),10)
airfc <- forecast(object=fit,h=10)$mean
airfc <- airfc + sfc
airfc <- InvBoxCox(x=airfc,lambda=lambda)
```

VIII. Evaluation

To evaluate the performed forecast, there are three types of forecast/accuracy error measures (i) scale-dependent error measures, (ii) percentage error measures, and (iii) scaled error measures. The first group has errors on the same scale as the data. Thus, comparisons among different time series with different scales are not possible. Examples are mean absolute error (MAE), or root mean squared error (RMSE). The second group is scale-free due to reporting in percentage and allows the comparison between different time series. However, the error can be zero or infinity. Examples are mean absolute percentage error (MAPE) or the symmetric mean absolute percentage error (sMAPE). The last group is an alternative to the percentage error and uses the training mean absolute error for normalisation. Although this group is recommended [8], it is rarely used. An example is the mean absolute scaled error (MASE). While forecasting, for example, the number of enrollments at a university has no time constraints, the time-to-result in the context of autonomic computing has strict requirements. Thus, we recommend considering the time-to-result in addition to the error measurements. For testing a forecasting method, there are numerous data sets available online: competitions (e.g., NN3[2], M3[3], and M4[4]), kaggle, `R` packages, and many more.

IX. Conclusion

In this tutorial paper, we briefly summarise the most important steps for forecasting of time series while guiding the reader through each step by providing best practices and some basic `R` code snippets.

Acknowledgments

This work is co-funded by the German Research Foundation (DFG) under grant No. KO 3445/11-1.

References

[1] G. E. Box, G. Jenkins, and G. C. Reinsel, "Time series analysis, prediction and control," 1970.

[2] M. Züfle, A. Bauer, N. Herbst, V. Curtef, and S. Kounev, "Telescope: A Hybrid Forecast Method for Univariate Time Series," in *Proceedings of the International work-conference on Time Series (ITISE 2017)*, September 2017.

[3] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning," *arXiv preprint arXiv:1704.07706*, 2017.

[4] G. E. Box and D. R. Cox, "An analysis of transformations," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 211–252, 1964.

[5] V. M. Guerrero, "Time-series analysis supported by power transformations," *Journal of Forecasting*, vol. 12, no. 1, pp. 37–48, 1993.

[6] 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.

[7] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[8] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[2]NN3 competition: http://www.neural-forecasting-competition.com/NN3/

[3]M3 competition: https://forecasters.org/resources/time-series-data/m3-competition/

[4]M4 competition: https://www.mcompetitions.unic.ac.cy/the-dataset/