MASTFM: Meta-learning and Data Augmentation to Stress Test Forecasting Models

Ricardo Inácio (✉)[1,2], Vítor Cerqueira[1,2], Marília Barandas[3], and Carlos Soares[1,2,3]

[1] Faculdade de Engenharia da Universidade do Porto, Portugal
{rcinacio,vcerqueira,csoares}@fe.up.pt
[2] Laboratory for Artificial Intelligence and Computer Science (LIACC), Portugal
[3] Fraunhofer Portugal AICOS, Portugal marilia.barandas@aicos.fraunhofer.pt

Abstract. Time series forecasting is pivotal across industries, as it fosters data-driven decision-making, increasing the chances of successful outcomes. Yet, certain instances that feature adverse characteristics, may lead models to manifest stress through decreases in performance (e.g., large errors). Hence, the ability to preemptively identify such cases, while establishing their root causes, would be advantageous to elevate the understanding of forecasting processes, informing users about the trustworthiness of predictions. Hence, we propose `MASTFM`, a method based on meta-learning that leverages statistical characteristics of input time series, and estimations of forecasting performance from model outputs, to build a metamodel that learns conditions for stress. Given that such occurrences are naturally rare, data augmentation is employed to ensure balance during training. Moreover, SHapley Additive exPlanations (SHAP) are used to explain how features impact forecasting behaviour.

Keywords: Time Series Forecasting · Meta-learning · Data Augmentation · Stress Testing

1 Introduction

Time series forecasting remains highly practical for decision-makers, as it enables statistically-based procedures [2], increasing the chances of success. Forecasts can be carried out with considerable accuracy and certainty, by leveraging patterns found in past data. Nonetheless, data difficulties, such as missing values or outliers, are prone to arise [10]. Difficult instances, which can be described as stress-inducing, may impact the underlying model negatively, resulting in abnormal behaviours such as large errors, high uncertainty, or hubris (i.e., large errors and low uncertainty). Inevitably, these are only made apparent after the fact, which leads users to distrust predictions. Hence, being able to preemptively identify those cases, while establishing contributing factors, would substantially elevate the understanding of forecasting mechanisms. It could also foster responsible practices, for example, by informing users about poor forecasts to dismiss.

To this end, we present `MASTFM`, a Python package that leverages meta-learning to explain which time series might induce model stress. It relies on

patterns derived from feature extraction methods, and estimations of forecasting performance based on model outputs. These are used to fit a metamodel, which learns to classify new instances as stress-inducing or not. Given that such cases are rare by nature, training a balanced, unbiased classifier might be challenging. Thus, resampling techniques, focused on data augmentation, are employed. The probabilities predicted by the metamodel are then used to explain the behaviour of the forecasting model. Moreover, SHAP [7] values are used to explain how features affect the metamodel, and consequently forecasting performance. A video demonstration [1] and the package [2], are available online.

2 MASTFM Specification

2.1 Forecasting Model

Our solution operates as a wrapper around forecasting models, as a way to identify which time series (and what specific feature values), might lead them to manifest stress. Any supervised regression algorithm that is compatible with the `scikit-learn` [8] API is also compatible with `MASTFM`. This implies that even those not belonging to the `scikit-learn` [8] library, are also supported, as long as compatibility with its API is ensured, such as `LightGBM` [6] or `XGBoost` [1].

2.2 Metamodel

A metamodel, in the form of a binary classifier based on meta-learning, is the central component of `MASTFM`. It leverages statistical features extracted from time series via `tsfeatures` [3], and data augmentation methods, either via oversampling, or synthetic time series generation, to mitigate the effects of target imbalance. Forecasting performance is estimated via SMAPE by default.

The binary label in each task ($\delta \in \{\delta^E, \delta^U, \delta^H\} \rightarrow$ errors, uncertainty, and hubris, respectively), takes its corresponding threshold(s) in consideration: $\tau \rightarrow E$, $\beta \rightarrow U$, $(\tau, \beta) \rightarrow H$. These are defined by percentiles of forecasting performance estimates e_i, from a model f in a time series Y_i, comprising information of both errors (e_i^e) and uncertainty (e_i^u): $e_i = (e_i^e, e_i^u)$, to classify a time series $Y_i \in \mathcal{Y}$ as stress-inducing ($\hat{\delta}_i = 1$) as follows: $\delta_i^E = e_i^e > \tau$, $\delta_i^U = e_i^u > \beta$, and $\delta_i^H = (e_i^e > \tau) \wedge (e_i^u < \beta)$. Stress-inducing time series are identified using the above schemes for each task, and used as ground truths for the metamodel.

2.3 Performance

The quality of the metamodel, in terms of the trustworthiness of its predictions is measured in ROC AUC, which quantifies its ability to discern from the two established classes of instances: stress-inducing or not. A set of experiments which showcases the results of several variants of the metamodel, each leveraging

[1] https://www.youtube.com/watch?v=0bm99xHWBrs
[2] https://pypi.org/project/mastfm/

a different augmentation technique, across six distinct datasets, is presented in the paper that introduces the theoretical foundations behind this work. [3]

Furthermore, analyses that compare how each metamodel variant performs both on average and on each dataset individually, across increasingly stricter stress settings, are also available on the paper. The reported outcomes indicate that the method is generally able to identify and characterise conditions for forecasting model stress, and that it performs more favourably when paired with data augmentation, mainly with methods that directly generate synthetic time series data, rather than generating features.

2.4 Explanations

The metamodel can then be applied to learn patterns present in time series features, which might be correlated to stress. The predicted probabilities can be used to explain forecasting behaviour, by employing explainability methods. `MASTFM` uses SHAP [7], to indicate which are the most important features in each meta-classification task (i.e., $E, U,$ or H), and how each contributes to the outcomes. Visual explanations are made available to the user, as shown in Figure 1.

3 Applications

This package targets users who seek to identify conditions for stress in a time series dataset, which might lead a forecasting model to behave abnormally. Therefore, given a model f, a set of time series $\mathcal{Y}$, and the kind of stress to quantify ($E, U,$ or H), `MASTFM` can automatically determine which series might cause it, explaining it via statistical features. Although many methods incorporate the modules that comprise this work, as far as we are aware, this is the first that integrates them in the context of stress testing based on meta-learning, to model the characteristics of challenging scenarios. This leads to a practical understanding of forecasting mechanisms, via state-of-the-art explainability approaches [7].

One use case is shown in Algorithm 1, where `XGBoost` is put to forecast time series captured in a monthly frequency, with seasonal periods of length 12. The user is interested in stress that manifests as large errors, and it considers those above the $80th$ percentile as significant. Besides point forecasts, the associated prediction intervals are computed, with a confidence level of 90%, via Conformal Prediction [9], quantifying uncertainty. Imbalance is mitigated by generating synthetic time series using `Scaling` [5], which adjusts data magnitude. It is also possible to apply data transformations (e.g., first differences), to ease modelling. The subsequent methods produce the explanations shown in Figure 1, which not only illustrate the distribution of series across differing manifestations of stress, but also how feature values affect the outcomes of the metamodel, and consequently forecasting performance. In this example, time series showcasing low `trend` and `linearity` values, lead the metamodel to classify them as stress-inducing, meaning that the forecasting model struggles with that kind of data.

[3] Meta-learning and Data Augmentation for Stress Testing Forecasting Models [4]

A practical example of how this method can be used, as showcased in the previously mentioned video demonstration [4], can also be found in the open repository of this project, in the form of a simple test notebook [5], which allows the use of diverse augmentation methods from the two aforementioned categories.

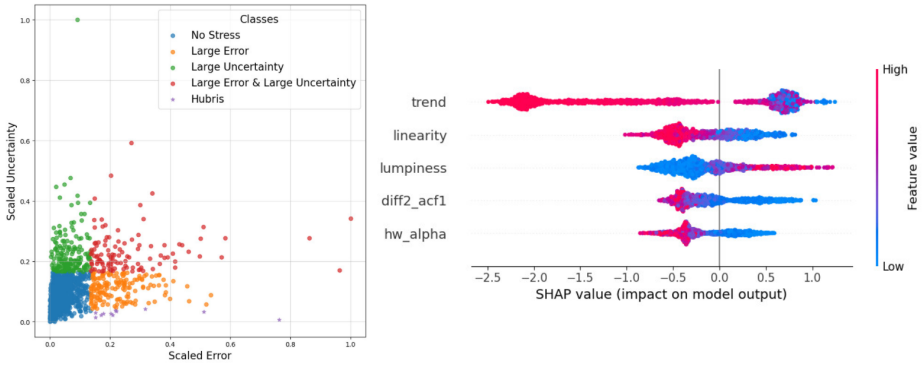

Fig. 1. Visual explanations. On the left, a scatter plot shows each series coloured by the respective stress class. On the right, SHAP values of the 5 most important features for the metamodel are shown. Values to the right of the vertical line contribute to a positive classification, and colour denotes the feature value (red = high, blue = low).

Algorithm 1 Example of usage for the `MASTFM` package

```
mast = MASTFM(                                          ▷ Initialize MASTFM class
  forecasting_model=XGBoost(),      ▷ Provide the forecasting regression model to wrap around
  seasonality=12,                          ▷ Set the seasonality (e.g., 12 to monthly data)
  frequency="M",                                        ▷ Set the time series frequency
  horizon=12,                                           ▷ Set the forecast horizon
  target="errors",                                      ▷ Set type of stress to gauge
  level=90,                  ▷ Set confidence level between 0 and 100 for prediction intervals
  quantile=80,                                ▷ Set quantile for the threshold of stress
  augmentation_method="Scaling"               ▷ Set a valid data augmentation method
)
mast.fit(df=df, target_differences=1)           ▷ Fit the method and apply first differences
mast.plot_stress()            ▷ Plot each series, across the error and uncertainty dimensions
mast.explanations()                ▷ Explain how features affect the metamodel using SHAP
mast.show_large_errors_ids()                          ▷ List series that lead to large errors
mast.show_large_uncertainty_ids()                ▷ List series that lead to high uncertainty
mast.show_hubris_ids()                ▷ List series that lead to overconfident predictions
```

Acknowledgments. This work was partially funded by projects AISym4Med (n.º 101095387) supported by Horizon Europe Cluster 1: Health, ConnectedHealth (n.º 46858), supported by Competitiveness and Internationalisation Operational Programme

[4] c.f. footnote 1

[5] https://github.com/ricardoinaciopt/mastfm/tree/main/test

(POCI) and Lisbon Regional Operational Programme (LISBOA 2020), under the PORTUGAL 2020 Partnership Agreement, through the European Regional Development Fund (ERDF) and Agenda "Center for Responsible AI", nr. C645008882-00000055, investment project nr. 62, financed by the Recovery and Resilience Plan (PRR) and by European Union - NextGeneration EU, and also by FCT plurianual funding for 2020-2023 of LIACC (UIDB/00027/2020 UIDP/00027/2020).

References

1. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD (2016)
2. Hewamalage, H., Bergmeir, C., Bandara, K.: Global models for time series forecasting: A simulation study. Pattern Recognition **124**, 108441 (2022)
3. Hyndman, R., Kang, Y., Montero-Manso, P., O'Hara-Wild, M., Talagala, T., Wang, E., Yang, Y.: tsfeatures: Time Series Feature Extraction (2024)
4. Inácio, R., Cerqueira, V., Barandas, M., Soares, C.: Meta-learning and data augmentation for stress testing forecasting models. In: International Symposium on Intelligent Data Analysis. pp. 343–357. Springer (2025)
5. Iwana, B.K., Uchida, S.: An empirical survey of data augmentation for time series classification with neural networks. Plos one **16**(7), e0254841 (2021)
6. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems **30** (2017)
7. Lundberg, S.: A unified approach to interpreting model predictions (2017)
8. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. Journal of machine Learning research **12** (2011)
9. Vovk, V., Gammerman, A., Shafer, G.: Algorithmic learning in a random world, vol. 29. Springer (2005)
10. Wang, Y.: Robustness and reliability of machine learning systems: a comprehensive review. Eng Open **1**(2), 90–95 (2023)