

TsSHAP: Robust model agnostic feature-based explainability for univariate time series forecasting

Vikas C. Raykar*

Arindam Jati

Sumanta Mukherjee

Nupur Aggarwal*

vikasraykar@gmail.com

arindam.jati@ibm.com

sumanm03@in.ibm.com

nupur.wal@gmail.com

IBM Research

Bangalore, India

Kanthy Sarpatwar*

Giridhar Ganapavarapu

Roman Vaculin

sarpatwa@us.ibm.com

giridhar.ganapavarapu@ibm.com

vaculin@us.ibm.com

IBM Research

Yorktown, USA

ABSTRACT

A trustworthy machine learning model should be accurate as well as explainable. Understanding why a model makes a certain decision defines the notion of explainability. While various flavors of explainability have been well-studied in supervised learning paradigms like classification and regression, literature on explainability for time series forecasting is relatively scarce. In this paper, we propose a feature-based explainability algorithm, TsSHAP, that can explain the forecast of any black-box forecasting model. The method is agnostic of the forecasting model being explained, and can provide explanations for a forecast in terms of interpretable features defined by the user *a priori*. The explanations are in terms of the SHAP values which are obtained by applying the TreeSHAP algorithm on a surrogate model that learns a mapping between the interpretable feature space and the forecast of the black-box model. Moreover, we formalize the notion of local, semi-local, and global explanations in the context of time series forecasting, which can be useful in several scenarios. We validate the efficacy and robustness of TsSHAP through extensive experiments on multiple datasets.

CCS CONCEPTS

- Applied computing → Forecasting; • Computing methodologies → Machine learning.

KEYWORDS

time series forecasting, explainability, interpretability

ACM Reference Format:

Vikas C. Raykar, Arindam Jati, Sumanta Mukherjee, Nupur Aggarwal, Kanthy Sarpatwar, Giridhar Ganapavarapu, and Roman Vaculin. 2023. TsSHAP: A Robust Model-Agnostic Feature-Based Explainability Method for Univariate Time Series Forecasting. In *Proceedings of ACM Conference (Conference'22)*. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/nnnnnnnn.nnnnnnn>

*The authors were associated with IBM Research during the work.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'22, August 2022, Washington, DC, USA

© 2023 Association for Computing Machinery.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...\$15.00

<https://doi.org/10.1145/nnnnnnnn.nnnnnnn>

Robust model agnostic feature-based explainability for univariate time series forecasting. In *Proceedings of ACM Conference (Conference'22)*. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/nnnnnnnn.nnnnnnn>

1 INTRODUCTION

The goal of time series forecasting is to predict the future observations of a temporally ordered sequence from its history. Forecasting is central to many domains, including retail, manufacturing, supply chain, workforce, weather, finance, sensor networks, and health sciences. An accurate forecast is key to successful business planning. There exist several time-series forecasting methods in the literature [8]. They are broadly classified into **statistical models** (SARIMA, Exponential Smoothing, Prophet, Theta, etc.), **modern machine learning models** (tree-based ensembles like XGBoost, CatBoost, LightGBM, Gaussian Process Regression, etc.), and more recent **deep learning models** including DeepAR [25], N-BEATS [21], Informer [31], and SCINet [13].

Although the classical statistical models are well-motivated and inherently explainable, recent research demonstrates that modern machine learning and deep learning approaches outperform them in several benchmarks. For example, in the latest M competition (M5) [17], gradient boosted trees and several combinations of boosted trees and deep neural networks were among the top performers. In the M4 competition, a hybrid modeling technique between statistical and machine learning models was the winner [29]. More recently, a pure deep learning model N-BEATS has outperformed the M4 winner's performance [21]. While the more complex machine learning and deep learning techniques outperform the statistical methods, particularly due to the availability of large-scale training datasets; the former models are partially or fully black-box, and hence, it is hard or impossible to explain a certain forecasting prediction made by them. This engenders a lack of trust in deploying these models for sensitive real-world tasks.

Since performance is an important factor along with explainability, effort in devising novel and complex machine learning models for time series forecasting is actively ongoing. Researchers are recently focusing on building algorithms that can explain the predictions of a complex black-box model. Although research on explainability is well-matured for supervised classification and regression tasks, the forecasting domain still lacks it (see § 3 for details).

1.1 Our contribution

In this paper, we propose a model agnostic feature-based explainability method, named TsSHAP, that can explain the prediction of *any* black-box forecasting model in terms of SHAP value-based feature importance scores [15]. The novelties of the proposed method are the following.

- TsSHAP can explain the prediction(s) of any black-box forecaster model from a set of interpretable features defined by the user *a priori*. It only needs to access the outputs of the `fit()` and `predict()` functions of the model being explained¹. It always needs to access the `predict()` function, and `fit()` is required only for a specific set of forecasters (see § 4.2 for details).
- It amalgamates the notion of model surrogacy, and the strength of the fast TreeSHAP algorithm into a suitable and ready-to-use post-hoc explainability method for time series forecasting.
- The novel TsSHAP methodology uniquely reduces the surrogate forecasting task into a regression problem where the surrogate model learns a mapping between the interpretable feature space and the output of the forecaster model.
- It introduces multiple scopes of explanation, *viz.* local, semi-local, and global, which can be useful to obtain useful explanations in several real-world scenarios.

2 BACKGROUND AND NOTATIONS

2.1 Time series forecasting

Let $f(t) : \mathbb{Z} \rightarrow \mathbb{R}^1$ represent a latent **univariate** time series for any discrete time index $t \in \mathbb{Z}$. We observe a sequence of historical *noisy* (and potentially missing) values $y(t)$ for $t \in [1, \dots, T]$ such that in expectation $\mathbb{E}[y(t)] = \mathbb{E}[f(t)]$. For example, in the retail domain $y(t)$ could represent the daily sales of a product and $f(t)$ the true latent demand for the product.

The task of **time series forecasting** is to estimate $f(t)$ for all $t > T$ based on the observed historical time series $y(t)$ for $t \in [1, \dots, T]$. The time series forecast is typically done for a fixed number of periods in the future, referred to as the **forecast horizon**, H . We denote

$$\hat{f}(T+h|y(1), \dots, y(T)) \triangleq \hat{f}(T+h|T) \text{ for } h \in [1, \dots, H] \quad (1)$$

as the projected forecast over time horizon H based on the historical observed time series $y(1), \dots, y(T)$ of length T .

2.1.1 Forecasting with external regressors. In many domains, the value of the target time series potentially depends on several other external time series, called **related external regressors**. For example, in the retail domain, the sales are potentially influenced by discounts, promotions, events, and weather. Let $z_1(t), \dots, z_K(t)$ be the K external regressors available that can potentially influence the target time series $f(t)$. Let²

$$\hat{f}(T+h|y(1), \dots, y(T), (z_k(1), \dots, z_k(T+h))_{k=1}^K) \text{ for } h \in [1, \dots, H] \quad (2)$$

¹Following conventional machine learning terminology, `fit()` and `predict()` refer to the training and inference functions of the forecasting algorithm respectively.

²Equation 2 assumes that the future value of the external regressor is also available, which is true in many situations.

be the forecasted time series for a forecast horizon H based on the historical observed time series $y(1), \dots, y(T)$ of length T and the K external regressors $(z_k(1), \dots, z_k(T+h))_{k=1}^K$ each of length $T+h$.

2.2 Explainability

Explainability is the degree to which a human can understand the cause of a decision (or prediction) made by a prediction model [18]. An explanation can be based on an instance, a set of features, or (for a time series) inherent components (e.g., trend, seasonality *etc.*). In this paper, our focus is feature-based explanation.

A feature-based explanation $\phi \in \mathbb{R}^d$ is a d -dimensional vector where each element corresponds to the importance /contribution score for the corresponding feature. Several feature-based explanation methods have been proposed in the supervised classification and regression literature (see § 3). We will adopt SHAP for explanation because of its intuitive nature [15] and state-of-the-art performance [14].

2.2.1 SHapley Additive exPlanations (SHAP). SHAP is a game theoretic approach to explain the output of any feature-based machine learning model. It connects optimal credit allocation with local explanations using the Shapley values from game theory and their related extensions. SHAP computes the contribution of a feature towards the conditional expectation of the model's output $g_x(S) = \mathbb{E}(g(x)|\text{do}(X_S = x_S))$, by estimating the expected difference in the predicted result in the absence of the corresponding feature [14]³.

$$\phi_i(g, x) = \sum_{R \in \mathcal{R}} \frac{1}{M!} \left[g_x(P_i^R \cup i) - g_x(P_i^R) \right] \quad (3)$$

where, \mathcal{R} is the set of all feature orderings, M is the number of features, P_i^R is the subset of feature that come before feature i in ordering R . In our scenario, $g(x)$ will denote the surrogate model (see § 4.1).

3 PRIOR ART

Several research efforts have been devoted toward formulating model agnostic explainability for classification and regression methods that work with image and tabular data. Recent developments include LIME [24], DeepLIFT [28], Layer-Wise Relevance Propagation (LRP) [2], Classical Shapley values [12], SHAP [15] and its several variants like KernelSHAP [15], TreeSHAP [14], and DeepSHAP [15]. A detailed survey can be found in [20].

Research on explainability for time series data mainly started with time series classification problems. Mokhtari *et. al.* [19] employed SHAP to derive global feature-based explanations for several time series classifiers. Madhikermi *et. al.* [16] utilized LIME for explaining the predictions of SVM and neural network classifiers designed to detect faults in an air handling unit. Schlegel *et. al.* [27] performed an extensive evaluation of multiple explainers like LIME, LRP, and SHAP to interpret multiple binary and multi-class time series classifiers. The authors found SHAP to be the most robust model agnostic explainer, while the other algorithms tend to be biased towards specific model architectures. Assaf *et. al.* [1] proposed

³We follow causal do-notation here [14].

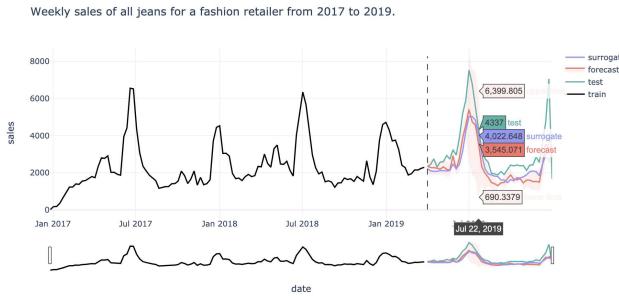


Figure 1: The forecasts from the surrogate model and the forecasts from the forecaster. The surrogate model is trained to mimic the forecasts of the corresponding forecaster.

a white-box explainability method to generate two-dimensional saliency maps for time series classification.

The researchers in [7] employed SHAP to analyze a neural network-based forecasting model that predicts the nitrogen dioxide concentration in a city. However, the local and global SHAP explanations were not evaluated. Saluja *et. al.* [26] demonstrated the explanations derived with SHAP and LIME for a sales time series forecasting model. The work also performed a human study to find the usefulness of the explanations. However, the work of [26] employed only a support vector regressor as the forecasting model, while the proposed TsSHAP method supports any forecasting model and it only needs to access the `fit()` and `predict()` methods of the model (we experiment with six different forecasting models, see § 6). The tree-based surrogate modeling of TsSHAP helps to achieve this. Moreover, we evaluate TsSHAP on five different datasets as compared to [26] which focused on a specific case study. Furthermore, we incorporate a more generic and much larger set of *interpretable features* that is well-suited for time series forecasting. Recently, Rajapaksha *et. al.* [23] proposed the LoMEF framework to provide local explanations for global time series forecasting models by training explainable statistical models at the local level. Our proposed method is inherently different than [23] since the explanations are in terms of the interpretable features that the user can define *a priori*. We provide explanations in terms of the SHAP values which satisfy all three desirable properties (local accuracy, missingness, and consistency) of additive feature attribution methods, and thus, was found to be have more consistency with human intuition [15]. Moreover, the TsSHAP can provide explanations at multiple scopes *viz.* local, semi-local, and global.

4 TSSHAP METHODOLOGY

4.1 Surrogate model

TsSHAP is model agnostic. Since we do not have access to the actual internals of the forecaster model, we first learn a surrogate model g .

$$g(T + h|T) = g(T + h|y(1), \dots, y(T)) \text{ for } h = 1, \dots, H. \quad (4)$$

The surrogate model g produces a point-wise forecast approximation **of the forecaster**. While the original forecaster learns to predict $y(T + h)$ based on $y(1), \dots, y(T)$ the surrogate model is

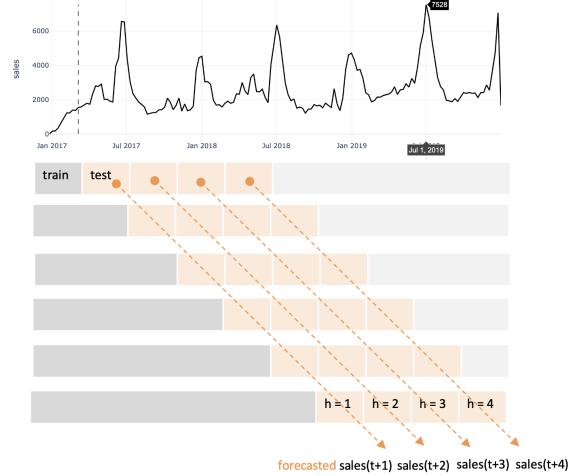


Figure 2: Backtested historical forecasts using an expanding window splitter to train the surrogate forecasting model.

trained to predict the forecasts $\hat{f}(T + h)$ made by the forecaster. Essentially we want to mimic the forecaster using a surrogate model since we aim to explain the forecasts made by the forecaster and not the original time series. Figure 1 illustrates this concept with an example time series from a real-world dataset.

4.2 Backtested historical forecasts

To generate the training data for the surrogate model, we use **back-testing**. For a given univariate time-series, we produce a sequence of train and test partitions using an *expanding window* strategy (see Figure 2). The expanding window partitioning strategy uses more and more training data while keeping the test window size fixed to the forecast horizon. The forecaster is trained (`fit()` is accessed) on the train partition and evaluated (`predict()` is accessed) on the test split. All the test split predictions are concatenated to get the backtested historical forecasts for each step of the forecast horizon.

Note that we need to access the `fit()` function only for a set of classical forecasters like SARIMA or exponential smoothing, where we train the forecaster for every expanding window. For most of the machine learning and deep learning forecaster models, we do not need to access their `fit()` functions because a single model is generally trained (beforehand) on the entire data.

4.3 Regressor reduction

We now have an original time series and the corresponding backtested forecast time series for each step of the forecast horizon. The goal of the surrogate model is to learn to predict the backtested forecast time series based on the original time series. We construct a surrogate time series forecasting task by reducing it to a supervised regression problem. A standard supervised regression task takes a d -dimensional feature vector as input ($\mathbf{x} \in \mathbb{R}^d$) and predicts a scalar $y \in \mathbb{R}$. The regressor $y = f(\mathbf{x})$ is learnt based on a labelled training dataset (\mathbf{x}_i, y_i) , for $i = 1, \dots, n$. The input time series sequence

Table 1: Interpretable features based on lag, seasonal lag, expanding and rolling window and trend based features. An example is provided with each feature description.

| feature name | feature description |
|------------------------------------|---|
| LagFeatures(lags=3) | Value at previous time steps. |
| sales(t-3) | The value of the time series at (t-3) previous time step. |
| SeasonalLagFeatures(lags=2, m=365) | Value at time steps for the previous seasons. |
| sales(t-2*365) | The value of time series at (t-2*365) previous time step. |
| RollingWindowFeatures(window=3) | Rolling window statistics (mean,max,min). |
| sales-max(t-1, t-3) | The max of the past 3 values. |
| ExpandingWindowFeatures() | Expanding window statistics (mean,max,min). |
| sales-mean(0, t-1) | The mean of all the values so far. |
| TrendFeatures(degree=2) | Features to model simple polynomial trend. |
| t2 | Feature to model polynomial (of degree 2) trend. |

does not satisfy classical feature definitions. Instead, we process the original time series to compute various interpretable features.

4.4 Interpretable features

For each time point t , we compute features $\mathbf{x}(t) \in \mathbb{R}^d$ based on the original time series values observed so far. The surrogate model uses these d features to predict the backtested forecast for each step in the forecast horizon. We focus only on **interpretable features** such that the explanations produced on the features are easily comprehensible by the end-user. The set of interpretable features can be defined by the (expert) user as well. As a starting point, we define a set of seven feature families, viz., lag, seasonal lag, rolling window, expanding window, date and time encoding, holidays, and trend-related features. Table 1 and 2 list the features.

4.5 Multi-step forecasting

Recall that we have H backtested time series forecasts which are to be used as targets to learn the surrogate regressor model. A **single regressor** is trained for a one-step-ahead forecast horizon and then called recursively to predict multiple steps. Let, $\mathcal{G}(y(1), \dots, y(T))$ represent the one-step-ahead forecast by the surrogate model trained on the training data. The surrogate produces a one-step-ahead forecast using features based on the time-series till T , that is, $y(1), \dots, y(T)$. The final forecasts from the surrogate model are made recursively as follows,

$$g(T+1|T) = \mathcal{G}(y(1), \dots, y(T)). \quad (5)$$

For $h = 2, 3, \dots, H$

$$g(T+h|T) = \mathcal{G}(y(1), \dots, y(T), g(T+1|T), \dots, g(T+h-1|T)). \quad (6)$$

Table 2: Interpretable features based on date and time encoding and holidays. Some features like `fashion-season` might be more relevant for a fashion sales time series forecasting.

| feature name | feature description |
|---|---|
| DateFeatures() | Date related features. |
| month | The month name from January to December. |
| day-of-year | The ordinal day of the year from 1 to 365. |
| day-of-month | The ordinal day of the month from 1 to 31. |
| week-of-year | The ordinal week of the year from 1 to 52. |
| week-of-month | The ordinal week of the month from 1 to 4. |
| day-of-week | The day of the week from Monday to Sunday. |
| is-weekend | Indicates whether the date is a weekend or not. |
| quarter | The ordinal quarter of the date from 1 to 4. |
| season | The season Spring/Summer/Fall/Winter. |
| fashion-season | The fashion season Spring/Summer(January to June) or Fall/Winter(July to December). |
| is-month-start | Whether the date is the first day of the month. |
| is-month-end | Whether the date is the last day of the month. |
| is-quarter-start | Whether the date is the first day of the quarter. |
| is-quarter-end | Whether the date is the last day of the quarter. |
| is-year-start | Whether the date is the first day of the year. |
| is-year-end | Whether the date is the last day of the year. |
| is-leap-year | Whether the date belongs to a leap year. |
| TimeFeatures() | Time related features. |
| hour | The hours of the day. |
| minute | The minutes of the hour. |
| second | The seconds of the minute. |
| HolidayFeatures(country="IN", buffer=2) | Encode country specific holidays as features. |
| holiday-IN | Indicates whether the date is a IN holiday or not. |

4.6 Tree ensemble regressor and SHAP explanation

In this work, we have used a tree-based ensemble regression model, XGBoost [5]⁴, as the surrogate model⁵. The selection of tree-based model is biased by its reasonably accurate forecast capability and, the support of the fast (polynomial time) TreeSHAP algorithm which we rely on for the various TsSHAP explanations as described below. [14]⁶.

5 TSSHAP EXPLANATIONS

TsSHAP provides a wide range of explanations based on SHAP values.

5.1 Scope of explanations

We introduce the notion of **scope of explanation** in the context of time series forecasting. An explanation answers to either a **why**-question or a **what if**-scenario [20]. We generalize three scopes of explanations.

⁴<https://github.com/dmlc/xgboost>

⁵Other models like CatBoost [22] and LightGBM [10] can also be employed.

⁶<https://github.com/slundberg/shap>

- (1) A **local explanation** explains the forecast made by the forecaster at a certain point in time. For example,
 - *Why is the sale forecast for July 1, 2019 much higher than the average sales?*
 - *What is the impact on the sale forecast for July 1, 2019 if I offer a discount of 60% that week?*
- (2) A **global explanation** explains the forecaster trained on the historical time series.
 - *What are the most important attributes the forecaster algorithms rely on to make the forecast?*
 - *What is the overall impact of discount on the sales forecast?*
- (3) A **semi-local explanation** explains the overall forecast made by a forecaster in a certain time interval.
 - *Why is the sales forecast over the next 4 weeks much higher?*
 - *What is the impact of a particular markdown schedule on a 4 week planning horizon sales forecast?*

Notions of local and global explanations exist in the supervised learning paradigms. In addition, we introduce the concept of semi-local explanations in the context of time series forecasting, which returns one (semi-local) explanation aggregated over many successive time steps in the forecast horizon. TsSHAP is able to provide explanations suitable for all the scopes.

5.2 Type of explantions

For each of the scopes, TsSHAP provides multiple types of explanations.

- 5.2.1 *For local explanation.* TsSHAP provides the following explanations.
 - (1) **SHAP explanation** shows features contributing to push up or down a certain forecast from the base value (the average) to the forecaster model output (as explained in § 2.2). Since this is a local explanation, the explanation for the forecast at a different forecast horizon can be quite different.
 - (2) **Partial dependence plot (PDP)** for a given feature shows how the actual forecast (from the surrogate model) varies as the feature value changes.
 - (3) **SHAP dependence plot (SDP)** for a given feature shows how the corresponding SHAP value varies as the feature value changes. SDP is an example of a **what-if** explanation.
- 5.2.2 *For global explanation.* TsSHAP provides the following explanations.
 - (1) **SHAP explanation** for a global scope means the absolute value of the SHAP scores for each feature across the entire dataset.
 - (2) **Partial dependence plot (PDP)** in a global scope shows how the average prediction in your dataset changes when a particular feature is changed.
 - (3) **SHAP dependence plot (SDP)** for each feature shows the mean SHAP value for a particular feature across the entire dataset. This shows how the model depends on the given feature, and is a richer extension of the classical PDP.

- 5.2.3 *For semi-local explanation.* TsSHAP provides SHAP feature importance, PDP, and SDP aggregated over a certain time interval in the forecast horizon.

Table 3: Datasets Univariate time series forecasting datasets (with available external regressors) used for experimental validation.

| dataset | periodicity | size | external regressors |
|---------------------------|-------------|------|---|
| jeans-sales-daily | daily | 1095 | discount Daily sales of all jeans for a fashion retailer from 2017 to 2019. |
| jeans-sales-weekly | weekly | 158 | discount Weekly sales of all jeans for a fashion retailer from 2017 to 2019. |
| us-unemployment | monthly | 872 | N/A Monthly US unemployment rate since 1948. |
| peyton-manning | daily | 2905 | N/A Log of daily page views for Wikipedia page of Peyton Manning. |
| bike-sharing | daily | 731 | weather, temperature, humidity, wind speed Daily count of rental bikes from 2011 to 2012 in Capital bikeshare system. |

Table 4: Forecasters Univariate time series forecasting algorithms used for experimental validation. Only Prophet and XGBoost can handle external regressors. Note that some of the forecasters are easily interpretable to validate the correctness of the output of TsSHAP.

| forecaster | description |
|----------------------------------|--|
| Naive [8] | The forecast is the value of last observation. |
| SeasonalNaive [8] | The forecast is the value of the last observation from the same season of the year. |
| MovingAverage [8] | A moving average forecast of order k , or, $MA(k)$, is the mean of the last k observations of the time series. |
| Exponential Smoothing [8] | The forecast is the exponentially weighted average of its past values. It can be interpreted as a weighted average between the most recent observation and the previous forecast. |
| Prophet [30] | Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. |
| XGBoost [5] | A machine learning based forecasting to regression reduction algorithm using XGBoost. |

6 EXPERIMENTS

6.1 Datasets

We experimentally validate the proposed explainer algorithm and the corresponding various explanations on five univariate time series forecasting datasets (with any available external regressors) listed in Table 3. For all datasets, we use a forecast horizon of 10% of the actual data. Availability of data is provided in the Reproducibility section (§ 9).

6.2 Forecasters

In order to validate the explanations we present results on six different univariate forecasting algorithms listed in Table 4. Only some forecasters (Prophet [30] and XGBoost [5]) can explicitly incorporate external regressors.

6.3 Robustness evaluation via time series perturbation

Robustness of TsSHAP explanations is evaluated by perturbing the original time series (in § 6.4, we will describe the evaluation metrics that will use the perturbed time series). To generate bootstrapped samples (i.e., multiple perturbed versions) of a time series $y(t)$, we employ the **block bootstrap** algorithm [4, 11]. The block bootstrap algorithm extends Efron’s bootstrap [6] setup for independent observations to time series where the observations ($y(t)$ for different values of t) might not be independent.

To preserve the trend-cycle of the time series [8], we first decompose $y(t)$ into trend-cycle and residual components using a moving average model of order m ,

$$y_{\text{trend-cycle}}(t) = \frac{1}{m} \sum_{j=-k}^{j=k} y(t+j) \quad \text{where, } m = 2k + 1 \quad (7)$$

$$y_{\text{residual}}(t) = y(t) - y_{\text{trend-cycle}}(t). \quad (8)$$

The residual series $y_{\text{residual}}(t)$ is then utilized by the block bootstrap algorithm. It samples contiguous blocks of $y_{\text{residual}}(t)$ at random (with replacement) and concatenates them together to produce a bootstrapped residual,

$$\tilde{y}_{\text{residual}}(t) = \text{Block-Bootstrap}(y_{\text{residual}}(t), L_b), \quad (9)$$

where, L_b is the block-length. The final perturbed series is then obtained by summing the bootstrapped residual to the trend-cycle component of the original series,

$$\tilde{y}(t) = y_{\text{trend-cycle}}(t) + \tilde{y}_{\text{residual}}(t). \quad (10)$$

6.4 Metrics to evaluate explanations

We have extended three well-adopted evaluation metrics in the context of time series, viz. faithfulness, sensitivity, and complexity. Faithfulness measures the consistency of an explanation, sensitivity measures the robustness, and complexity measures the sparsity of an explanation [3].

6.4.1 Faithfulness. It is high when the change in predicted values due to the change in feature values is correlated to the change in the feature importance in the explanations. Let $\hat{f}_X(T+h|T)$ represents predictions for $h = 1, \dots, H$ when \hat{f} was trained on data X . Let $N(X)$ represents a neighbourhood around X , obtained through perturbation (§ 6.3). $\phi_j(\hat{f}(T+h|T))$ represents the feature importance for the j^{th} feature. Then faithfulness is given by

$$\mu_F = \text{correlation}(\delta_{\hat{f}}, \delta_{\phi}) \quad (11)$$

where

$$\delta_{\hat{f}} = \hat{f}_X(T+h|T) - \hat{f}_{N_X}(T+h|T)$$

$$\delta_{\phi} = \sum_j (\phi_j(\hat{f}_X(T+h|T)) - \phi_j(\hat{f}_{N_X}(T+h|T)))$$

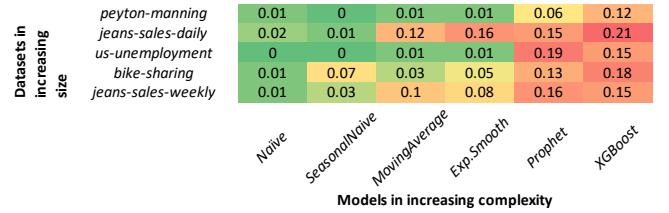


Figure 3: MAPE for different models and datasets. In the vertical axis, the dataset size increases from the bottom to the top. In the horizontal axis, the model complexity increases from left to right.

A higher value of faithfulness is preferred in an explanation.

6.4.2 Sensitivity. It measures the change in explanations when model inputs and outputs vary by an insignificant amount. A lower value of sensitivity is preferred in an explanation. We define average sensitivity as follows.

$$\mu_S = \int_z \mathcal{D}(\phi(f_X(x)), \phi(f_{N_X}(z))) \mathbb{P}_X(z) dz \quad (12)$$

where \mathcal{D} is a distance measure (Euclidean in this paper) and z is a neighborhood of x .

6.4.3 Complexity. A simple explanation is easy to understand. To measure the complexity of an explanation, we calculate the entropy of the fractional distribution of feature importance scores.

$$\mu_C = - \sum_{i=1}^d p_i \ln p_i \quad (13)$$

where, $p_i = \phi_i / \sum_{j=1}^d \phi_j$ and d denotes the total number of features.

7 RESULTS

7.1 Accuracy of the surrogate model

We first validate the accuracy of the surrogate model forecasts from the explainer via backtesting (temporal cross-validation). Using an expanding window approach, we partition the univariate time series into a sequence of train and test datasets. The forecaster and the explainer are trained on the train partition and then evaluated on the test dataset. We report various error metrics between the **forecasts from the original forecaster** and the **forecasts made by the surrogate model of the explainer**, viz. MAE (Mean Absolute Error), RMSE (Root Mean Squared Error), MAPE (Mean Absolute Percentage Error), and MASE (Mean Absolute Scaled Error) [9]. Table 5 (in § 9 Reproducibility) shows the accuracy scores between the forecaster and the surrogate model for the six forecasters and five datasets.

Figure 3 shows the variation in MAPE with increasing size of the datasets and increasing complexity of the models. It is evident that the MAPE increases with an increase in the model complexity. This might be because it is harder for the surrogate model to mimic a complex forecaster (e.g. XGBoost) than a simpler one (e.g., Naive). However, there is no such pattern in the variation of MAPE with increase in the dataset size. This is particularly beneficial since it

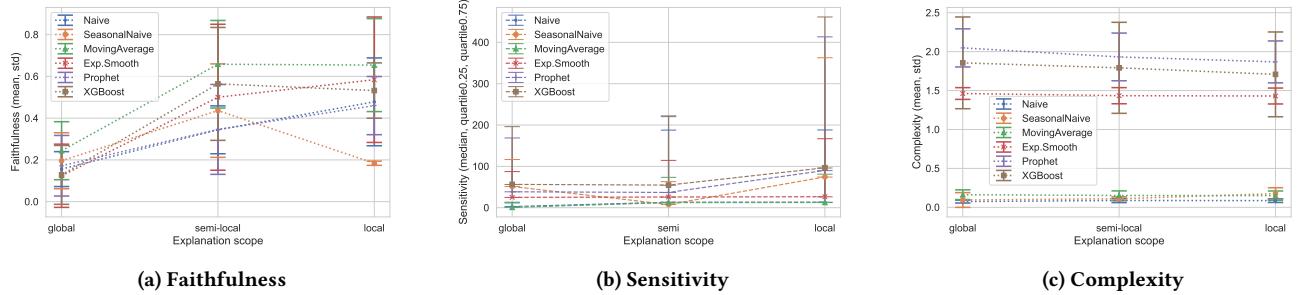


Figure 4: Summary of the evaluation metrics aggregated across all datasets. For faithfulness and complexity, mean is shown with standard deviation as the error bar. For sensitivity, median is shown with 1st and 3rd quartiles as the error bars. Best viewed in color.

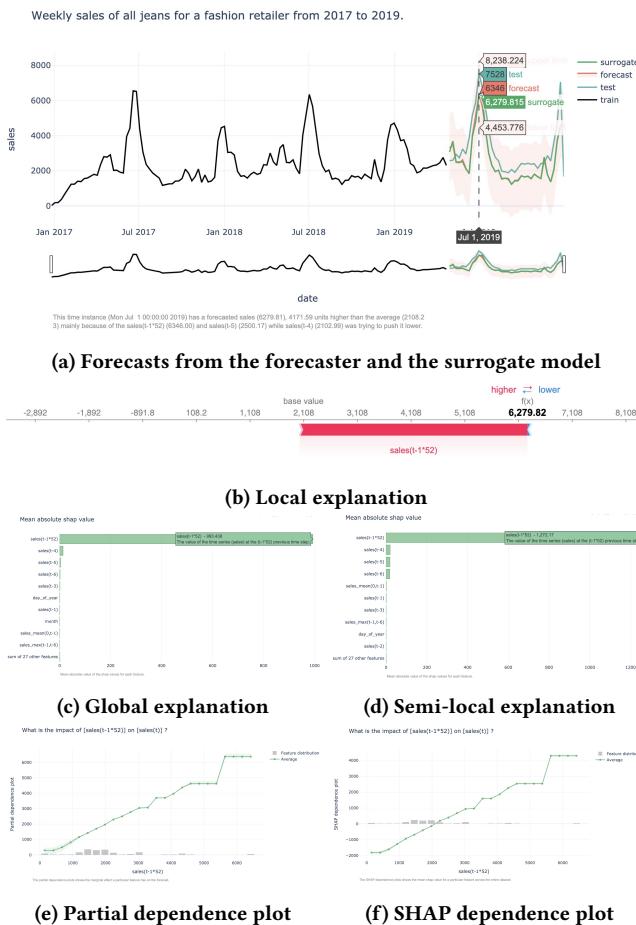


Figure 5: Explanations for SeasonalNaive($m=52$) forecaster on the jeans-sales-weekly dataset.

conveys that the dataset size does not affect the surrogate model training.

7.2 Evaluating explanations

We evaluate the feature-based explanations in terms of the (high) faithfulness, (low) sensitivity and (low) complexity metrics defined in § 6.4. In the Reproducibility section, § 9, we provide various metrics for global, semi-local and local explanations for all the forecasters and the datasets in Table 6, 8 and 7. The metrics for the local explanations were averaged over the entire forecast horizon.

Figure 4 provides a graphical summary of Table 6, 8 and 7 by aggregating the metric values across all the datasets⁷. We can observe that, for all the forecasters, the faithfulness of semi-local and local explanations are higher than that of global explanations. A closer look reveals that the faithfulness of a semi-local explanation is always higher than the faithfulness of a global explanation. However, when we move to local explanations (from semi-local), there is no definite pattern in change in the faithfulness metric. The complexity of the explanations do not vary much with the scope of explanation, as can be seen in Figure 4c. This might refer to an implicit relationship between the complexity of the forecast explanation and that of the forecaster. The median sensitivity curves stay (approximately) flat for all the forecasters.

Moreover, we can notice that a particular forecaster might not achieve the best performance in all the metrics, and this highlights the complementary nature of the metrics defined in § 6.4. For example, in Table 6, for “jeans-sales-weekly” dataset, MovingAverage forecaster achieves the highest faithfulness and the lowest sensitivity, but the lowest complexity is obtained by the Naive forecaster. This type of behaviors is also observed in local and semi-local explanations (Table 7 and Table 8).

A closer look on Figure 4 (and on Table 6, 7 and 8) reveals that the MovingAverage forecaster achieves the average highest faithfulness (0.244 for global, 0.654 for local, and 0.658 for semi-local explanations) across all datasets. The Naive forecaster obtains the average lowest complexity (0.074 for global, 0.086 for local, and 0.088 for semi-local explanations) across all datasets. This might be because the inherent simplicity of the Naive forecaster results in the fractional distribution of the feature importance scores to have low entropy (see § 6.4).

⁷For sensitivity, we do not show mean (std) because its value depends on the absolute value of the data (see § 6.4). Instead, we show median with 1st and 3rd quartiles as the error bars.

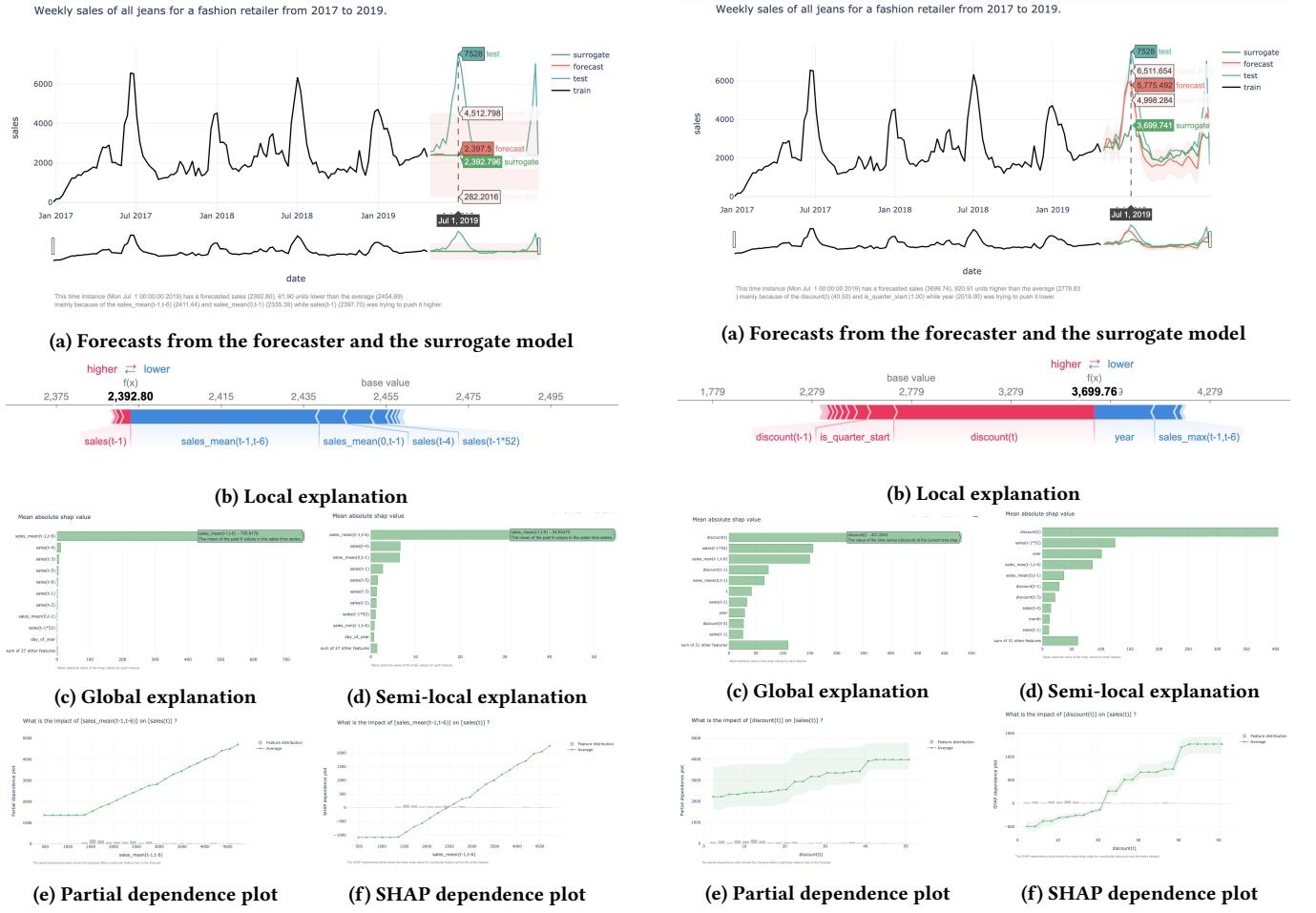


Figure 6: Explanations for MovingAverage($k=6$) forecaster on the jeans-sales-weekly dataset.

However, the median lowest sensitivity is shared among MovingAverage and SeasonalNaive forecasters across different scopes of explanations. The MovingAverage forecaster has the lowest median sensitivity for global (0.8) and local (12.64) explanations, but the SeasonalNaive attains the lowest value (7.7) for semi-local explanation.

Another noticeable observation from our experiments is that the complex (and possibly non-linear) models (like Prophet and XGBoost) have high complexity, which is expected. However, they might achieve better faithfulness and sensitivity than a simpler model. For example, in Table 6, Prophet attains higher faithfulness than a Naive forecaster, and in Table 8, XGBoost achieves lower sensitivity and higher faithfulness than a Naive forecaster in multiple datasets.

7.3 TsSHAP Illustrations

Figure 5, 6, and 7 illustrate a few local, semi-local, and global explanations for the jeans-sales-daily dataset for all different forecasters. Note that the PDP and SDP plots are only shown for a local scope. It

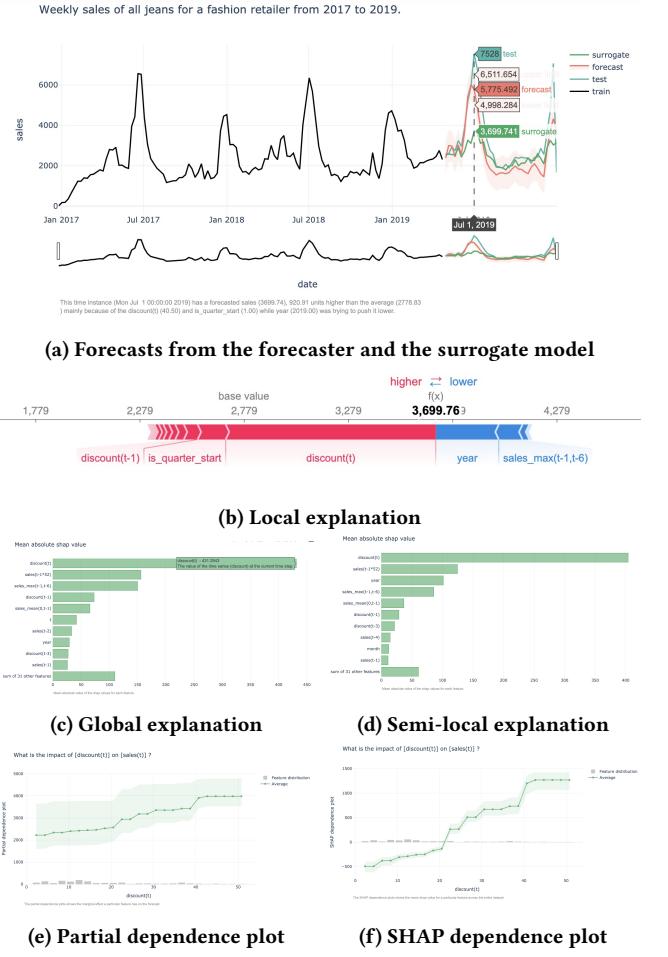


Figure 7: Explanations for Prophet forecaster on the jeans-sales-weekly dataset.

can be seen that the explanations reflect the underlying aspects the forecaster is able to learn from the data. From the local, semi-local, and global explanations of Figure 5, we can see that the main contributing factor for the forecast is the seasonal history at ($t - 52$) weeks time, which is expected from a seasonal naive forecaster. The partial dependence plot also depicts a linear relationship between the sales($t - 52$) and forecaster output. From the local explanation of Figure 6, we can see that the mean value of the past 6 observations is primarily responsible for bringing the forecast down. Other explanations depict the same story. From the local explanation of Figure 7, we can see that, mainly, discount(t) and is_quarters_start are pushing the forecast up, but year and sales_max($t-1, t-6$) are pushing the forecast down. The Prophet forecaster supports external regressors, and here, it is picking discount as the primary explanatory factor. The global explanation clearly shows the main contributing features responsible for the forecast across the entire time series dataset. Some other illustrations are provided in § 9.

8 CONCLUSIONS AND FUTURE WORK

We proposed a robust interpretable feature-based explainability algorithm for time-series forecasting. The method is model agnostic and does not require access to the model internals. Moreover, we formalized the notion of local, semi-local, and global explanations in the context of time series forecasting. We validated the correctness of the explanations by applying TsSHAP on multiple forecasters. We demonstrated the robustness of TsSHAP by evaluating the explanations with faithfulness and sensitivity metrics that work on multiple perturbed versions of a time series. Following are the main observations from the experiments.

- The TsSHAP explanations for the inherently interpretable forecasters (like Naive, Seasonal Naive, Moving Average, and Exponential Smoothing) were found to be correct, i.e., they follow the modeling strategy of the forecaster.
- The surrogate model prediction accuracy decreases with increasing model complexity.
- The TsSHAP explanations are *more faithful* in the local and semi-local scopes.
- The *complexity* of an explanation is stable over different explanation scopes.
- Although TsSHAP tends to generate complex explanations for a complex model, the explanations might have better faithfulness and sensitivity scores than a simpler model.

TsSHAP can also be applied to derive interpretable insights of complex neural network-based models. TsSHAP can be extended to explain the prediction interval. Instead of regressing on the mean forecast from the forecaster, we regress on the prediction interval. We also plan to extend the above algorithm to multivariate time series forecasting models.

REFERENCES

- [1] Roy Assaf and Anika Schumann. 2019. Explainable Deep Neural Networks for Multivariate Time Series Predictions. In *IJCAI*. 6488–6490.
- [2] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS one* 10, 7 (2015), e0130140.
- [3] Umang Bhatt, Adrian Weller, and José M. F. Moura. 2020. Evaluating and Aggregating Feature-based Model Explanations. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, Christian Bessiere (Ed.). International Joint Conferences on Artificial Intelligence Organization, 3016–3022. <https://doi.org/10.24963/ijcai.2020/417> Main track.
- [4] Peter Bühlmann. 2002. Bootstraps for time series. *Statistical science* (2002), 52–72.
- [5] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 785–794. <https://doi.org/10.1145/2939672.2939785>
- [6] Bradley Efron. 1992. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*. Springer, 569–593.
- [7] María Vega García and José L. Aznarte. 2020. Shapley additive explanations for NO₂ forecasting. *Ecological Informatics* 56 (2020), 101039.
- [8] Robin John Hyndman and George Athanasopoulos. 2018. *Forecasting: Principles and Practice* (2nd ed.). OTexts, Australia.
- [9] Rob J. Hyndman and Anne B. Koehler. 2006. Another look at measures of forecast accuracy. *International Journal of Forecasting* 22 (2006).
- [10] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 3149–3157.
- [11] Jens-Peter Kreiss and Soumendra Nath Lahiri. 2012. Bootstrap methods for time series. In *Handbook of statistics*. Vol. 30. Elsevier, 3–26.
- [12] Stan Lipovetsky and Michael Conklin. 2001. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry* 17, 4 (2001), 319–330.
- [13] Minhao Liu, Ailing Zeng, Zhijian Xu, Qiuxia Lai, and Qiang Xu. 2021. Time Series is a Special Sequence: Forecasting with Sample Convolution and Interaction. *arXiv preprint arXiv:2106.09305* (2021).
- [14] Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence* 2 (2020), 56–67. <https://doi.org/10.1038/s42256-019-0138-9>
- [15] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*. 4768–4777.
- [16] Manik Madhikermi, Avleen Kaur Malhi, and Kary Främling. 2019. Explainable artificial intelligence based heat recycler fault detection in air handling unit. In *International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*. Springer, 110–125.
- [17] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2022. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting* (2022).
- [18] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (2019), 1–38. <https://doi.org/10.1016/j.artint.2018.07.007>
- [19] Karim El Mokhtari, Ben Peachey Higdon, and Ayşe Başar. 2019. Interpreting financial time series with SHAP values. In *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering*. 166–172.
- [20] Christoph Molnar. 2019. *Interpretable Machine Learning*. <https://christophm.github.io/interpretable-ml-book>
- [21] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2019. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*.
- [22] Liudmila Prokhorenko, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorgush, and Andrey Gulin. 2018. CatBoost: Unbiased Boosting with Categorical Features. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 6639–6649.
- [23] Dilini Rajapaksha, Christoph Bergmeir, and Rob J. Hyndman. 2021. LoMEF: A Framework to Produce Local Explanations for Global Model Time Series Forecasts. *CoRR* abs/2111.07001 (2021). arXiv:2111.07001 <https://arxiv.org/abs/2111.07001>
- [24] Marcus Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. “Why should i trust you?” Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.
- [25] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.
- [26] Rohit Saluja, Avleen Malhi, Samanta Knapić, Kary Främling, and Cicek Cavdar. 2021. Towards a Rigorous Evaluation of Explainability for Multivariate Time Series. *arXiv preprint arXiv:2104.04075* (2021).
- [27] Udo Schlegel, Hiba Arnout, Mennatallah El-Assady, Daniela Oelke, and Daniel A Keim. 2019. Towards a rigorous evaluation of xai methods on time series. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 4197–4201.
- [28] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *International Conference on Machine Learning*. PMLR, 3145–3153.
- [29] Sławek Smyl. 2020. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting* 36, 1 (2020), 75–85.
- [30] Sean J Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician* 72, 1 (2018), 37–45.
- [31] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of AAAI*.

9 REPRODUCIBILITY

9.1 Data availability

- (1) jeans-sales-daily: Proprietary dataset.
- (2) jeans-sales-weekly: Proprietary dataset.
- (3) us-unemployment:
<https://www.bls.gov/data/#unemployment>
- (4) peyton-manning:
https://en.wikipedia.org/wiki/Peyton_Manning
- (5) bike-sharing: <http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset>

9.2 Detailed results

We have provided a summarized observation from our experiments in the main text (§ 7). Here, we demonstrate the detailed results obtained in the experiments.

9.2.1 Detailed Accuracy of the surrogate model. Table 5 show the errors of the surrogate model obtained through time series cross validation as explained in § 6.

Table 5: Backtested error metrics for the surrogate model for different forecasters and datasets.

| forecaster | MAE | RMSE | MAPE | MASE |
|--|--------|--------|------|------|
| Naive | | | | |
| jeans-sales-weekly | 43.69 | 44.78 | 0.01 | 0.10 |
| jeans-sales-daily | 17.09 | 17.17 | 0.02 | 0.15 |
| us-unemployment | 0.00 | 0.00 | 0.00 | 0.02 |
| peyton-manning | 0.06 | 0.06 | 0.01 | 0.19 |
| bike-sharing | 52.43 | 56.78 | 0.01 | 0.08 |
| SeasonalNaive | | | | |
| jeans-sales-weekly | 75.65 | 116.52 | 0.03 | 0.18 |
| jeans-sales-daily | 4.45 | 14.55 | 0.01 | 0.04 |
| us-unemployment | 0.01 | 0.02 | 0.00 | 0.08 |
| peyton-manning | 0.01 | 0.02 | 0.00 | 0.04 |
| bike-sharing | 262.34 | 363.34 | 0.07 | 0.42 |
| MovingAverage(k=6) | | | | |
| jeans-sales-weekly | 347.26 | 359.37 | 0.10 | 0.78 |
| jeans-sales-daily | 61.74 | 62.42 | 0.12 | 0.54 |
| us-unemployment | 0.11 | 0.11 | 0.01 | 0.66 |
| peyton-manning | 0.05 | 0.05 | 0.01 | 0.16 |
| bike-sharing | 158.67 | 161.28 | 0.03 | 0.24 |
| SimpleExponentialSmoothing(0.5) | | | | |
| jeans-sales-weekly | 239.29 | 261.30 | 0.08 | 0.55 |
| jeans-sales-daily | 101.76 | 104.07 | 0.16 | 0.85 |
| us-unemployment | 0.09 | 0.09 | 0.01 | 0.54 |
| peyton-manning | 0.11 | 0.11 | 0.01 | 0.35 |
| bike-sharing | 256.76 | 259.95 | 0.05 | 0.39 |
| Prophet | | | | |
| jeans-sales-weekly | 463.60 | 586.44 | 0.16 | 1.07 |
| jeans-sales-daily | 100.64 | 119.25 | 0.15 | 0.87 |
| us-unemployment | 0.95 | 1.04 | 0.19 | 5.88 |
| peyton-manning | 0.46 | 0.53 | 0.06 | 1.50 |
| bike-sharing | 650.72 | 739.13 | 0.13 | 1.01 |
| XGBoost | | | | |
| jeans-sales-weekly | 398.31 | 485.52 | 0.15 | 0.91 |
| jeans-sales-daily | 97.39 | 123.92 | 0.21 | 0.86 |
| us-unemployment | 0.96 | 1.06 | 0.15 | 6.04 |
| peyton-manning | 0.96 | 1.16 | 0.12 | 3.11 |
| bike-sharing | 879.15 | 981.73 | 0.18 | 1.36 |

9.2.2 Evaluation of TsSHAP explanations. Table 6, Table 8, and Table 7 provide the detailed results of the evaluation of TsSHAP

explanations using faithfulness, sensitivity, and complexity metrics for global, semi-local, and local explanations respectively.

9.3 Some more TsSHAP illustrations

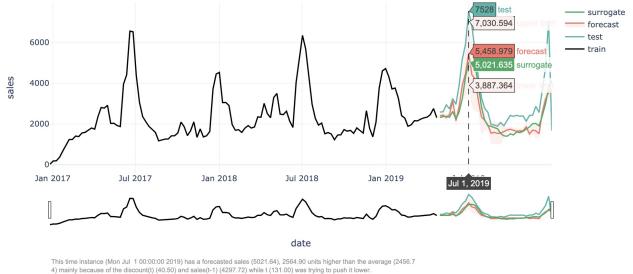
Figure 8 shows the TsSHAP explanations for XGBoost forecaster on the jeans-sales-weekly data. We can see that the surrogate model accurately approximates the forecaster in this case. Moreover, a closer look into the local explanation says that sales(t-1), last year’s value sales(t-52), and the external regressor discount(t) are primarily bringing the forecast up from the average sales. The global explanation shows that sales(t-1) and discount(t) are the main explanatory factors for the forecast made by the forecaster across the entire time series data. The semi-local explanation depicts similar picture for the entire forecast horizon interval. The specific PDP shows that after a certain threshold, increasing discount is helpful for increasing sales, but it can also saturate at some point. The specific SDP shows sales(t) is positively correlated with sales(t-1).

TsSHAP also provides correct explanations for the Naive forecasters, i.e., it picks the last observation as the most important feature in all the scopes. We skip the illustration for the Naive forecaster due to space limitation.

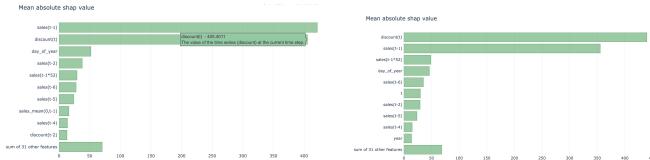
Table 6: Metrics for global explanations.

| forecaster | explanation metrics | | |
|--|---------------------|-------------|------------|
| | faithfulness | sensitivity | complexity |
| Naive | | | |
| jeans-sales-weekly | 0.05 | 43.99 | 0.08 |
| jeans-sales-daily | 0.24 | 2.68 | 0.07 |
| us-unemployment | 0.26 | 0.01 | 0.04 |
| peyton-manning | 0.08 | 0.00 | 0.09 |
| bike-sharing | 0.15 | 10.47 | 0.09 |
| SeasonalNaive | | | |
| jeans-sales-weekly | 0.42 | 65.22 | 0.28 |
| jeans-sales-daily | 0.23 | 51.34 | 0.06 |
| us-unemployment | 0.20 | 0.01 | 0.02 |
| peyton-manning | 0.02 | 0.01 | 0.07 |
| bike-sharing | 0.11 | 448.40 | 0.04 |
| MovingAverage(k=6) | | | |
| jeans-sales-weekly | 0.45 | 25.58 | 0.23 |
| jeans-sales-daily | 0.20 | 0.80 | 0.08 |
| us-unemployment | 0.35 | 0.02 | 0.24 |
| peyton-manning | 0.06 | 0.00 | 0.13 |
| bike-sharing | 0.16 | 10.68 | 0.13 |
| SimpleExponentialSmoothing(0.5) | | | |
| jeans-sales-weekly | 0.02 | 62.34 | 1.57 |
| jeans-sales-daily | 0.08 | 25.01 | 1.50 |
| us-unemployment | 0.01 | 0.11 | 1.44 |
| peyton-manning | 0.42 | 0.04 | 1.46 |
| bike-sharing | 0.09 | 79.14 | 1.34 |
| Prophet | | | |
| jeans-sales-weekly | 0.14 | 130.16 | 2.01 |
| jeans-sales-daily | 0.36 | 38.52 | 2.10 |
| us-unemployment | 0.00 | 0.05 | 1.62 |
| peyton-manning | 0.04 | 0.06 | 2.37 |
| bike-sharing | 0.32 | 154.21 | 2.14 |
| XGBoost | | | |
| jeans-sales-weekly | 0.00 | 217.22 | 1.75 |
| jeans-sales-daily | 0.36 | 56.39 | 2.06 |
| us-unemployment | 0.22 | 0.07 | 0.85 |
| peyton-manning | 0.06 | 0.20 | 1.95 |
| bike-sharing | 0.001 | 139.71 | 2.67 |

Weekly sales of all jeans for a fashion retailer from 2017 to 2019.



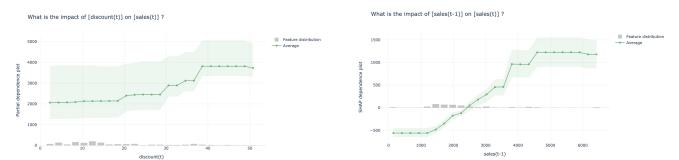
(a) Forecasts from the forecaster and the surrogate model



(c) Global explanation

(d) Semi-local explanation

(b) Local explanation



(e) Partial dependence plot

(f) SHAP dependence plot

Figure 8: Explanations for XGBoost forecaster on the jeans-sales-weekly dataset.

Table 8: Metrics for semi-local explanations.

| forecaster | explanation metrics | | |
|--|---------------------|-------------|------------|
| | faithfulness | sensitivity | complexity |
| Naive | | | |
| jeans-sales-weekly | 0.27 | 174.43 | 0.14 |
| jeans-sales-daily | 0.18 | 13.30 | 0.08 |
| us-unemployment | 0.51 | 0.08 | 0.06 |
| peyton-manning | 0.42 | 0.17 | 0.08 |
| bike-sharing | 0.34 | 434.51 | 0.08 |
| SeasonalNaive | | | |
| jeans-sales-weekly | 0.18 | 71.85 | 0.13 |
| jeans-sales-daily | 0.75 | 7.70 | 0.08 |
| us-unemployment | 0.18 | 0.04 | 0.09 |
| peyton-manning | 0.56 | 0.01 | 0.10 |
| bike-sharing | 0.51 | 55.32 | 0.15 |
| MovingAverage(k=6) | | | |
| jeans-sales-weekly | 0.42 | 61.11 | 0.23 |
| jeans-sales-daily | 0.89 | 12.32 | 0.08 |
| us-unemployment | 0.93 | 0.10 | 0.21 |
| peyton-manning | 0.53 | 0.05 | 0.10 |
| bike-sharing | 0.52 | 186.45 | 0.14 |
| SimpleExponentialSmoothing(0.5) | | | |
| jeans-sales-weekly | 0.10 | 88.58 | 1.58 |
| jeans-sales-daily | 0.32 | 25.89 | 1.50 |
| us-unemployment | 0.99 | 0.22 | 1.44 |
| peyton-manning | 0.84 | 0.11 | 1.37 |
| bike-sharing | 0.25 | 187.61 | 1.28 |
| Prophet | | | |
| jeans-sales-weekly | 0.26 | 183.38 | 1.88 |
| jeans-sales-daily | 0.60 | 36.88 | 2.12 |
| us-unemployment | 0.08 | 0.13 | 1.41 |
| peyton-manning | 0.60 | 0.09 | 2.33 |
| bike-sharing | 0.19 | 282.65 | 1.92 |
| XGBoost | | | |
| jeans-sales-weekly | 0.35 | 167.43 | 1.67 |
| jeans-sales-daily | 0.33 | 54.68 | 1.96 |
| us-unemployment | 0.35 | 0.30 | 0.79 |
| peyton-manning | 0.90 | 0.20 | 1.95 |
| bike-sharing | 0.89 | 249.49 | 2.59 |

Table 7: Metrics for local explanations.

| forecaster | explanation metrics | | |
|--|---------------------|-------------|------------|
| | faithfulness | sensitivity | complexity |
| Naive | | | |
| jeans-sales-weekly | 0.27 | 174.68 | 0.13 |
| jeans-sales-daily | 0.87 | 13.38 | 0.08 |
| us-unemployment | 0.50 | 0.09 | 0.06 |
| peyton-manning | 0.41 | 0.17 | 0.08 |
| bike-sharing | 0.34 | 435.12 | 0.08 |
| SeasonalNaive | | | |
| jeans-sales-weekly | 0.18 | 288.36 | 0.16 |
| jeans-sales-daily | 0.20 | 74.42 | 0.12 |
| us-unemployment | 0.17 | 0.11 | 0.08 |
| peyton-manning | 0.18 | 0.25 | 0.23 |
| bike-sharing | 0.19 | 549.87 | 0.29 |
| MovingAverage(k=6) | | | |
| jeans-sales-weekly | 0.40 | 68.44 | 0.23 |
| jeans-sales-daily | 0.89 | 12.64 | 0.08 |
| us-unemployment | 0.95 | 0.11 | 0.21 |
| peyton-manning | 0.54 | 0.05 | 0.10 |
| bike-sharing | 0.49 | 191.81 | 0.13 |
| SimpleExponentialSmoothing(0.5) | | | |
| jeans-sales-weekly | 0.20 | 140.37 | 1.58 |
| jeans-sales-daily | 0.66 | 26.50 | 1.49 |
| us-unemployment | 0.95 | 0.22 | 1.43 |
| peyton-manning | 0.84 | 0.11 | 1.37 |
| bike-sharing | 0.27 | 195.08 | 1.28 |
| Prophet | | | |
| jeans-sales-weekly | 0.25 | 323.18 | 1.82 |
| jeans-sales-daily | 0.36 | 90.15 | 2.03 |
| us-unemployment | 0.62 | 0.18 | 1.41 |
| peyton-manning | 0.59 | 0.37 | 2.22 |
| bike-sharing | 0.48 | 508.43 | 1.86 |
| XGBoost | | | |
| jeans-sales-weekly | 0.44 | 364.76 | 1.59 |
| jeans-sales-daily | 0.62 | 96.66 | 1.85 |
| us-unemployment | 0.32 | 0.36 | 0.77 |
| peyton-manning | 0.68 | 0.59 | 1.89 |
| bike-sharing | 0.60 | 921.83 | 2.44 |