Machine Learning Interpretability Techniques for Outage Prediction: A Comparative Study

Anahid Jalali[1], Alexander Schindler[2], Bernhard Haslhofer[3], Andreas Rauber[4]

[1,2,3] *Austrian Institute of Technology, Giefinggasse 4, Vienna, 1210, Austria*
name.lastname@ait.ac.at

[4] *Vienna University of Technology, Karlsplatz 13, Vienna, 1040, Austria*
rauber@ifs.tuwien.ac.at

ABSTRACT

Interpretable machine learning has recently attracted a lot of interest in the community. Currently, it mainly focuses on models trained on non-time series data. LIME and SHAP are well-known examples and provide visual-explanations of feature contributions to model decisions on an instance basis. Other post-hoc approaches, such as attribute-wise interpretations, also focus on tabular data only. Little research has been done so far on the interpretability of predictive models trained on time series data. Therefore, this work focuses on explaining decisions made by black-box models such as Deep Neural Networks trained on sensor data. In this paper, we present the results of a qualitative study, in which we systematically compare the types of explanations and the properties (e.g., method, computational complexity) of existing interpretability approaches for models trained on the PHM08-CMAPSS dataset. We compare shallow models such as regression trees (with limited depth) and black-box models such as Long-Short Term Memories (LSTMs) and Support Vector Regression (SVR). We train models on processed sensor data and explain their output using LIME, SHAP, and attribute-wise methods. Throughout our experiments, we point out the advantages and disadvantages of using these approaches for interpreting models trained on time series data. Our investigation results can serve as a guideline for selecting a suitable explainability method for black-box predictive models trained on time-series data.

1. INTRODUCTION

The opacity of machine learning models is increasingly seen as a problem in application areas, where transparency and accountability play an essential role. Examples of such application areas are health care systems, financial services, and industrial applications (Varshney & Alemzadeh, 2017). Due to missing model transparency, it is often impossible to explain how models derived their predictions from input data and how subsequent decisions were made. However, this is a fundamental requirement in application domains that rely on transparent and reproducible predictions and decision-making.

Explainable Artificial Intelligence (XAI) denotes ongoing research activities that concentrate on finding explainability approaches for such black-box models. At the moment, we can distinguish two different branches of studies: the first focuses on providing interpretable models through examining the prediction results and internal learning processes of (model-specific) algorithms. The other focuses on prediction interpretation and justification and produces (model-agnostic) explanations of prediction outcomes (Biran & Cotton, 2017). One approach to ensure the interpretability of a model's decisions is to use interpretable models such as decision trees, linear or logistic regression, where through meaningful features of these models, explanations can be extracted. However, models such as Deep Neural Networks (DNN), also known as Artificial Neural Networks (ANNs), have proven to be more robust at modeling complex data, such as multimedia analysis. Explanation methods used for such models are often model-specific, and most of them are focused on Computer vision (Carvalho, Pereira, & Cardoso, 2019). Recent studies focused on the interpretation of models trained on time-series data derived from text (Ribeiro, Singh, & Guestrin, 2016; Lundberg & Lee, 2016). However, we are not aware of any studies that examined the explanations provided by these interpretability approaches for black-box algorithms trained on time series sensor data in the industrial application context. Therefore, our work's focus is to investigate the applicability of existing interpretability techniques on the PHM08-CMAPSS dataset, which represents a well-known and widely studied Predictive Maintenance (PdM) use case. To point out their advantages and disadvantages, we apply these methods on Support Vector Regression (SVR) and Recurrent Neu-

ral Networks (RNNs) architectures such as Long-Short Term Memories (LSTMs) and Gated Recurrent Networks (GRUs) trained on sensor data. Furthermore, we use Bayesian linear regression and a decision tree to compare the output of interpretability techniques with the feature importance and coefficients of these interpretable models.

Our study aims to investigate how well-known interpretable approaches, such as LIME, SHAP, attribute-wise, and surrogate models, can be used to explain industrial time series models' decisions. Our contribution is a systematic and qualitative analysis of these interpretability approaches on black-box models. We will commence by providing further background on PdM and machine learning interpretability in section 2. Next, in section 3, we introduce the PHM08-CMAPSS dataset, results from our exploratory analysis, applied data preprocessing steps, as well as our methodology to compare the interpretability techniques. In section 4, we demonstrate the experimental results of our models and cover the analysis of explanations of the predictions produced by these techniques. Finally, we discuss the advantages and disadvantages of these approaches for industrial time series and their effectiveness.

2. BACKGROUND

Machine learning is becoming increasingly popular in industrial research. However, its opaqueness challenges the deployment of such systems in practice. Therefore, industry professionals are cautious in deploying such technologies.
Yet, they are still tackling to optimize their maintenance plans, reduce the costs of unplanned breakdowns, and unnecessary maintenance actions (W. Zhang, Yang, & Wang, 2019).

2.1. Predictive models for outage prediction

Remaining Useful Life (RUL) and Time to Failure (TTF) estimation are essential tasks of Predictive Maintenance. Machine Learning has proven robust in these tasks;
(Mutunga, Kimotho, & Muchiri, 2019) uses an ensemble of machine learning algorithms to estimate the remaining number of cycles to breakdown on the C-MAPSS dataset. (Guo, Li, Jia, Lei, & Lin, 2017) has developed a Recurrent Neural Network to produce a health indicator for RUL estimation of bearings. (Ren, Cui, Sun, & Cheng, 2017) uses a fully connected neural network to predict the RUL of a group of bearings, using a simulated dataset generated by AS2M department of FEMTO-ST Institute. (Y. Zhang, Xiong, He, & Pecht, 2018) uses LSTM architecture to estimate the RUL of lithium-ion batteries. However, there have been very few studies on interpreting these models in this domain (Gade, Geyik, Kenthapadi, Mithal, & Taly, 2019). An example of such studies is the work of (Kraus & Feuerriegel, 2019), where they interpreted an LSTM model trained on the C-MAPSS dataset by variational Bayesian inferences. Therefore, we have provided a comparative study on interpretable methods and applied them to explain the decision of black-box models trained on sensor data.

2.2. Machine learning model interpretability

Most ML models, such as ensemble models, random forest, SVM, and ANNs, have a complex inner structure and act as a black box. Therefore, ML-experts cannot provide explanations of the decisions made by these models. Even though ML techniques have proven their robustness in many different domains, the transparency of their decisions is still missing. Questions of *trusting the model vs. the adequateness of its performance arise*; can we trust a robust model without knowing why it is robust. Or, more critically, is the model truly robust? An example of such models was pointed out by Ribeiro, where he addressed the Husky or Wolf problem (Ribeiro et al., 2016). The classifier proved robust, but when investigated further, it was found that the model never learned the features of either wolves or huskies and instead focused on the background. Whenever the background was white (snow), the image was more likely to be predicted as a wolf. ML-Interpretability (also known as Explainable Artificial Intelligence or XAI) focuses on approaches to provide more transparency for these opaque algorithms. The explanations of these models can be grouped into Global or Local explanations.

- Global explanations justify all outputs of a model. Example of such approaches are attribute-wise approaches such as Partial Dependence Plots (PDP) (Friedman, 2001), Individual Conditional Expectation (ICE) (Goldstein, Kapelner, Bleich, & Pitkin, 2015) and Accumulated Local Effects (ALE) (Apley, 2016).
- Local explainability provides justifications for an instance and has local fidelity. Examples of such approaches are LIME(Ribeiro et al., 2016) and SHAP(Lundberg & Lee, 2016) (Given a trained model and the full test set, SHAP can also provide global explanations).

Interpretable approaches can further be structured into pre-model (before training), in-model or intrinsic (while training), and post-model or post-hoc (trained model). Techniques used for model interpretation often focus on explaining the model by looking at the contribution of a feature (or set of features) on a model's output. Besides attribute-wise techniques, the surrogate model is also a popular choice, where a transparent model such as linear regression or a decision tree approximates the decision boundary of a complex model. For a thorough reading, we recommend the survey (Carvalho et al., 2019).
These approaches enable us to answer the questions: *why was a specific prediction chosen? when does the model fail?* and *when does the model succeed?*.

2.3. Existing methods and techniques

Attribute-wise techniques, as their name implies, provide a feature summary and their contributions to predictions. For example, by calculating Partial Dependence of a feature on the predictions, we can observe the feature's effect on the average predictions by changing values of all attributes at each observation while freezing the value of targeted features for that observation. However, this approach does not take the linear dependency of the target feature with other features into account.

A similar approach to PDP is ALE, where the interaction between the features is taken into account. Using ALE plots to interpret a model's prediction, we observe the changes of the predictions within an interval where the values of other features change around the value of the target feature.

Another popular technique to interpret a model's outputs is to estimate the predictions made by a black-box model using a shallow model. A recently proposed method, Local Interpretable Model-Agnostic Explanations (LIME), combines both attribute-wise and surrogate models to explain the model's predictions locally. LIME is a post-hoc approach, proposed by (Ribeiro et al., 2016) and explains the trained model's decision for an individual instance. Initially, they only focused on classifiers for tabular and image data. Later on, they extended their approaches for models accepting time series as input, such as Recurrent Neural Networks. The goal of this approach is to identify an interpretable model g that locally explains the decision of the complex model f. LIME applies a trade-off between local fidelity and interpretability of the set of potentially interpretable models and chooses the one with minimum complexity as well as minimum unfaithfulness of g. For this purpose, they follow these consecutive steps;

1. Perturb the training sample and create a new dataset.
2. Measure the distance between the new samples and original samples.
3. Predict the new samples using the complex model f.
4. Select the feature set that is describing the predictions of the complex model best.
5. Train the interpretable model g on the permuted data on the selected feature set and use similarity scores as its weights.
6. Weights of the interpretable model g interpret the complex model f.

Shapley Additive explanations (Lundberg & Lee, 2016), also known as SHAP, use Shapley values, a game theory technique, to provide its explanations. A Shapley value is the average marginal contribution of a feature value over all possible combinations with other features (Molnar, 2019). SHAP uses these values to measure the effect of an individual feature on an individual prediction. It permutes overall options of removing the feature and the effect of it on the prediction, as well as its contribution. SHAP has multiple explainers. Examples of such explainers are;

- Kernel Explainer - very similar to LIME, where a weighted linear regression calculates the feature contributions.
- Tree Explainer - interprets ensemble tree models e.g. XGBoost.
- Deep Explainer - an enhanced version of DeepLift (Shrikumar, Greenside, & Kundaje, 2017) to explain deep neural networks (but does not support Tensorflow 2.0).
- Gradient Explainer - motivated by the integrated gradients method (Sundararajan, Taly, & Yan, 2017), interprets the models by calculating the expected gradients.

In section 4, we examine the performance and applicability of these approaches on our black box models to explain the RUL prediction on the PHM08 C-MAPSS dataset. Throughout our examination, we point out the advantages and disadvantages of these techniques. We compare their explanations to each other to provide a correlation between their explanations and the model outputs. We also provide extra information on these approaches in the following table 1. In column "Computational Complexity" N is the number of instances, F is the number of features and τ is a constant referring to time complexity of a decision made by complex model f. Note that LIME computational complexity is in addition to its sample perturbation.

3. DATA AND METHODOLOGY

We first apply data pre-processing steps such as noise handling and normalization on the PHM08 C-MAPSS dataset. Afterward, we select a set of models, both shallow and black-box models such as SVR and ANNs, to estimate the engine's remaining useful life. To evaluate our model's performance, we have chosen the RMSE (Root Mean Squared Error). Subsequently, we apply state of the art interpretability approaches such as LIME, SHAP, and attribute-wise explanations of PDP, ALE plots.

3.1. Dataset

C-MAPSS aircraft engine data is a benchmark dataset that contains run-to-failure data, including labeled breakdowns. It represents 218 engines of a similar type, which all start from a healthy state. Faults are injected throughout the entire engine's life span until it goes to a breakdown state. The maximum and the minimum number of cycles to failure in the training set are 357 and 128, respectively, with a mean of 210. The engine data's attributes consist of three operational settings, vibration data collected from vibration sensors, and two binary attributes. We have named the columns as follows: the first two columns represent the engine number and cycles, columns 3 to 5 represent the operational settings and

Table 1. A General Comparison of Interpretability Approaches

Method	License	Output	Multiple Features per Plot	Explanatory Approach	Scope	Computational Complexity	Post -Hoc
LIME	BSD 2-Clause	Bar-plot	True	Feature Contribution & Surrogate Model	Local	$\mathcal{O}(N_\tau + F^3 + F^2N)$	True
SHAP	MIT	Scatter-Plot	True	Feature Contribution with SHapley values	Local & Global	$\mathcal{O}(2^F)$	True
PDP	sk-learn 3-Clause BSD	Line-plot	False	Feature Contribution via calculating prediction over marginal distribution	Global	$\mathcal{O}(2NF)$	True
ALE	ALEPython Apache License 2.0	Line-plot	False	Feature Contribution via calculating prediction over interval conditional distribution	Global	$\mathcal{O}(2NF)$	True

from column six, and every attribute is named as $s1$ to $s21$ (sensor 1, sensor two and so on).

3.2. Exploratory analysis

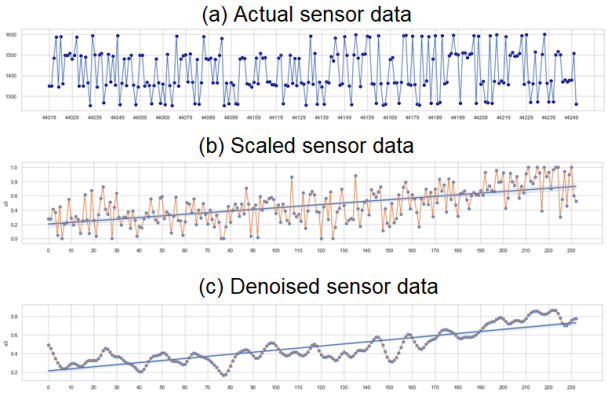
As mentioned in the provided documentation for this dataset, the operational settings have a direct effect on the engine's health degradation. Analyzing these settings results in 6 groups of values caused by six different operational modes in the data (Peel, 2008; Wang, Yu, Siegel, & Lee, 2008). After scaling the data into a range between 0 and 1 within each operational mode, we have noticed the degradation trend in the given stationary time series. Figure 1 illustrates the differences between the actual signal and the noise-removed scaled one. Further analyzing the trends, we notice a sudden curve after a specific number of cycles is passed. Visibility of these trends and their strong correlations with target variables make this dataset appropriate for our study comparing interpretability models applied to the algorithms trained on this data. In this way, we have a sense of when and why a breakdown is approaching. We see that these curves often occur after roughly 120 cycles, and we use this information as the knee point to build our second target value as the remaining useful life of the machine. Scaled raw data already demonstrates the degradation trend over time. Therefore, we apply no further signal processing on the data.

3.3. Data pre-processing

We use k-means clustering to label those six operational clusters. Next, we min-max scale the vibration data (no columns with binary value or operational settings) within their corresponding units and labeled operational modes. Furthermore, we have used a Gaussian filter to remove the noise in the data 1.

To create the target variable, we have experimented with two different target values;

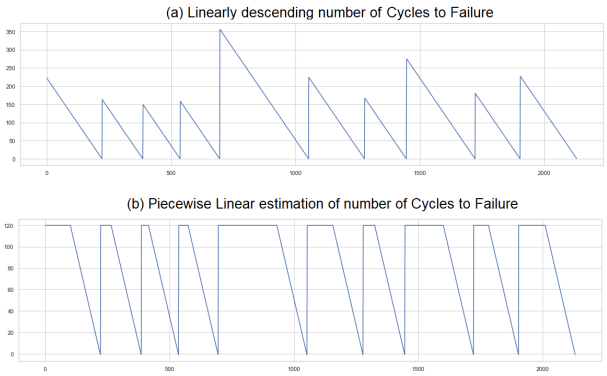
Figure 1. Illustrating the visible trend in the data after preprocessing; Sub-figure (a) demonstrates the actual raw sensor data. In sub-figure (b) we have scaled the signal within its unit and labeled operational cluster. Sub-figure (c) is the scaled signal after passing it through a Gaussian filter ($\sigma = 2$) for noise removal.



1. Target value is a descending line derived from the number of cycles, creating a zig-zag of the number of cycles to failure. This target value was used and motivated by (Peel, 2008).

2. Target value is a piece-wise linear function of the aforementioned target. It uses 120 cycles as maximum life expectancy, achieved during our exploratory analysis, and motivated by the winner of the challenge (Heimes, 2008). These target values are depicted in figure 2. The argument for this shape of the target value is to demonstrate the health state of the equipment, where over a period of time, it remains healthy and then starts to degrade linearly.

Figure 2. Comparison of two target variables; Linearly descending number of cycles to failure are depicted in sub-figure (a) and the piece wise estimation of this target is depicted in sub-figure (b).



3.4. Methodology

In subsection 2.3, we provide an overview of interpretability approaches and their differences. We do this by investigating their output types, the content of the plot and supported feature dimension, e.g., plotting the best features describing the black-box model over target value, explanation techniques (whether it is feature contribution or surrogate model or both), the scale of the approach (whether it explains all of the decisions made by the model or per instance), as well as their time complexity. This general comparison can be seen in table 1. Our results comparing these techniques are provided in section 4.2, where we explain how we derive an explanation from the outputs of interpretability methods, the advantages and disadvantages of their explanations, and extra information can achieve from their outputs. Furthermore, we need to adapt our models to be able to use some of these approaches. Interpretable approaches such as PDP and ALE do not support an array as an input. Therefore, we modify our SVR, Bayesian linear regression, and decision tree by accepting one descriptive value of a time-window. The descriptive value of time-windows is the mean of the window. This is acceptable for this dataset as the fluctuations in processed data do not affect the visible trend of the data. Moreover, denoising the signal removes the anomalies, thereby allowing for the replacement of the mean with the median, as it no longer significantly changes the data.

4. EXPERIMENTAL RESULTS

We divide our experiments into two sections: first, where the target value is linearly descending (motivated by (Peel, 2008)) and second, where the target value is a representation of the equipment's remaining useful life and is estimated by a piece-wise linear function (motivated by (Heimes, 2008)). We use a Bayesian linear model as well as a decision tree with limited depth for our shallow models. Our complex (black-box) models are neural networks (LSTMs, GRUs) and Support Vector Regressors with a nonlinear kernel (Radial Basis Function). We applied multiple experiments on different combinations of attributes and chose the set of combinations, which are only 13 vibration signals $s2$, $s3$, $s4$, $s7$, $s8$, $s9$, $s11$, $s12$, $s13$, $s14$, $s15$, $s17$, $s20$ from 21 inputs and 3 operational settings. We chose this set because of their strong correlation with the target variable. The removed features are two binary features and operational settings. Our study focuses on interpreting the decisions made by the predictive models, and therefore, we omit those two binary features to focus only on the vibration data. Since we use the operational settings in scaling the features, we removed these settings as well. Furthermore, the length of our time window is 30 observations over time.

4.1. Prediction results

In this section, we demonstrate the prediction results of our trained models. The column "Mean of Series" refers to the mean of each time window. We train our models on 70% of the dataset and evaluate the remaining 30%. This evaluation set was chosen at the beginning (using group shuffle split function with the randomness value set to 42 from the scikit-learn library) and always remained the same throughout the experiments. For our training process, we use a four-fold cross-validation technique to make sure the model is not overfitting on a portion of the data, and the evaluation results are reliable. For the cross-fold validation, at each fold, we use 85% of training data, and the remaining 15% of this set was for evaluation within the folds. The following tables demonstrate results on the out of sample dataset. We do not carry our analysis on the test set for the apparent reason that the R2F data is not provided.

Table 2. Linearly descending number of cycles to failure as target value

	RMSE	
Model	**Series**	**Mean of series**
LSTM	32.63	-
GRU	33.16	-
SVR	29.69	35.28
Lin-Bayesian	32.67	36.61
Tree	33.23	46.85

Table 3. Piece-wise linear estimation of number of cycles to failure as target value

	RMSE	
Model	**Series**	**Mean of series**
LSTM	15.17	-
GRU	14.11	-
SVR	13.29	13.88
Lin-Bayesian	13.56	14.67
Tree	17.78	19.44

For both experiments, we see that SVR achieved the lowest RMSE and Decision Tree, the highest RMSE. Linear Bayesian and GRUs also achieved RMSE close to SVR. We further investigate the predictions of trained models by looking at the data from engines with successful predictions (lower RMSE) and failed predictions. Looking at the predictions, we notice almost all the engines have more accurate predictions close to the breakdown (last quarter of the engine's life).

For the linear target variable, the higher error in prediction often occurs in the first half of the engine's life span. This is due to the model's inability to estimate the maximum number of runs when the machine is still in a healthy state. This maximum number is often estimated by the average maximum number of cycles of all the engines in the train data.

Changing the target variable from linear to piece-wise linear reduced the RMSE (second phase of the experiments). Failed cases in piece-wise linear are the same as a linear target, due to the change in the first half of the engine's life. Often engines run "stable" over longer intervals at hardly changing sensor values indicating no significant degradation of the engine. That leads to the model mistakenly predicting higher remaining life cycles for that engine than the actual life cycles remaining. This shows the difficulty of estimating the remaining number of cycles to failure at the beginning of the equipment's healthy life.

We compare the failed predictions of LSTMs (and GRUs) with SVR (as this model achieved the lowest error rate). SVR had the same failures (same engines with high RMSE) as LSTMs and GRUs. Figures 3 and 4 illustrate failed predictions of LSTMs and SVR on both target variables. Figures 5 and 6, illustrate the successful predictions for both target variables. Our analysis and comparison of failed and successful cases among the trained model showed that they mostly fail on the same engines and successfully predict similar engines.

Figure 3. Comparison of failed predictions on linear target variable and piece-wise linear target; Sub-figure (a) demonstrates the actual raw sensor data, actual target values and predictions. In sub-figure (b) another unit's data with piece-wise linear target variables and the corresponding prediction produced by LSTMs.

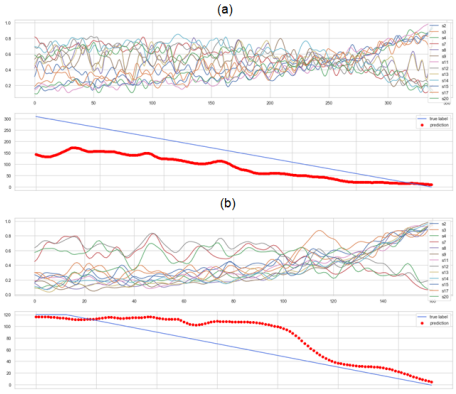


Figure 4. Comparison of failed predictions on linear target variables and piece-wise linear target; Sub-figure (a) demonstrates the actual raw sensor data, actual target values and predictions. In sub-figure (b) another unit's data with piece wise linear target variable and the corresponding prediction produced by SVR.

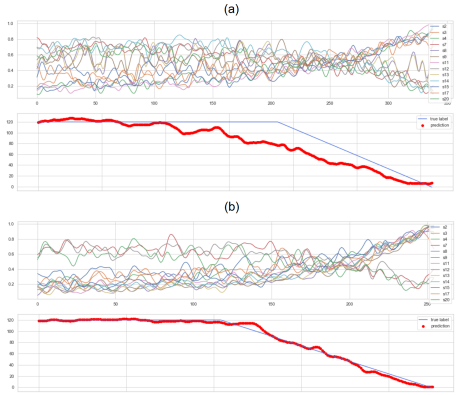
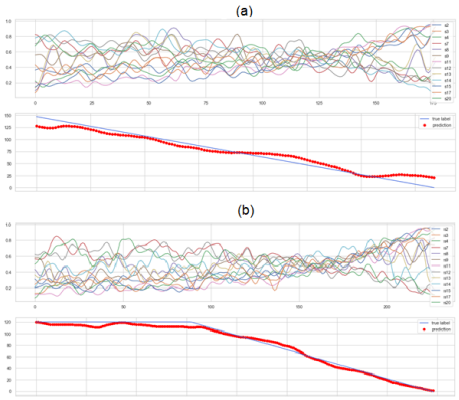


Figure 5. Comparison of successful predictions on linear target variable and piece-wise linear target; Sub-figure (a) demonstrates the actual raw sensor data, actual target values and predictions. In sub-figure (b) another unit's data with piece wise linear target variable and the corresponding prediction produced by SV.



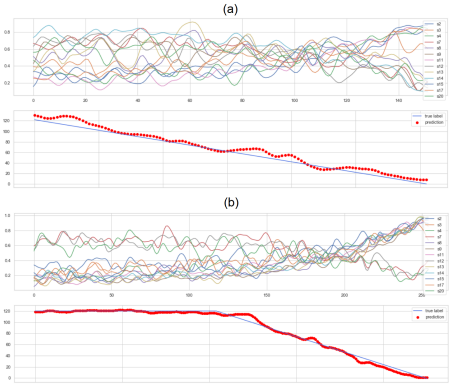
4.2. Interpretability results

In this section, we analyze the interpretability techniques and derive explanations on the model's outputs to justify its predictions. The first analysis we provide in prediction results help us to understand and check the validity of interpretability approaches. To compare their outputs, we extract the feature coefficients from the Bayesian linear regression model, as well as a decision tree, where they both point to features $s12$, $s9$ and $s7$ with positive coefficients and features $s15$, $s11$, $s17$, $s3$, $s2$, $s4$ with negative coefficients.

4.2.1. LIME

We use linear regression for LIME's approximation model to explain our trained LSTMs and GRUs as well as SVR. We

Figure 6. Comparison of successful predictions on linear target variable and piece-wise linear target; Sub-figure (a) demonstrates the actual raw sensor data, actual target values and predictions. In sub-figure (b) another unit's data with piece wise linear target variable and the corresponding prediction produced by SVR.



explain the train set instances from the first quarter of the machine's life, third quarter and breakdown point.
Analyzing LIME's output for all samples in different quarters of the engine's life span, we notice that the model has a strong focus on the behavior of 3 sensors. LIME approximates a range for different timesteps and checks if the sensor value of the individual sample violates the range.
For sensors with a positive correlation with the target variable, the higher value of the sensor results in higher predictions of the number of Cycles to Failure (CTF) and for lower CTF model predicts the cycles as they linearly degrade. This means based on LIME's approximation, $s12$ and s7 are best describing both LSTMs and GRUs.
For instance, at the beginning of the timesteps, min-max scaled $s12$ should have a range smaller than 0.39 and towards the end of the timesteps should be smaller than 0.37 for the model to predict a lower number of cycles to breakdown. Close to the breakdown, this value was changed to 0.70 and 0.72, respectively. The same is observed for GRUs, and SVRs.
Contrary to LSTMs and GRUs, where approximations mostly were weighted by positively correlated features, SVR is approximated by negatively correlated features with the target value. The chosen range for $s11$ and 15 (which are highly correlated) approaching breakdown was often a sensor value larger than 0.57 and 0.59, respectively. We further apply LIME for SVR and noticed that LIME's approximation gives more weight to sensors that are negatively correlated with the target variable and the positively correlated sensor $s12$ has many fewer coefficients than $s15$ and $s11$.
Figure 7 is a crop of LIME prediction at the beginning of the engine's life, and as it approaches the breakdown point, explaining LSTMs predictions.

We explain the predictions of LSTMs and GRUs by analyzing LIME's output that sudden jumps in the most influential features cause the failed prediction. More importantly, we notice that LSTMs, GRUs, and SVRs are modeling the noise in the signals as well, and therefore they fail at predicting correct CTF.

4.2.2. SHAP

We first apply SHAP on SVR to explain its prediction. For this purpose, we use the kernel explainer and locally explain the instance of the engine with the lowest RMSE (successful prediction) for all 30 instances of it. Local explanation plots are provided from SHAP's force plot and are presented in figure 8. This plot shows that for instance 1, increasing the value of $s11$ with each of its timesteps is positively affecting the model's prediction (raising the number of CTF) and 30 instances later, the decreasing value of $s13$ is the reason of lowering the number of CTF. Moving closer to the breakdown, we see that SHAP values for $s12$ and $s11$ have higher values and push the CTF value lower than their previous time steps.

We further use SHAP's summary plot to have an overview of all the calculated SHAP values for all the features. Figure 9 is the output of SHAP's summary plot, where the x-axis is the predicted target value (CTF) and each line of the y-axis is the calculated SHAP value for each feature. Each point of this plot is a SHAP value for a feature and an instance. The color blue represents the lower SHAP values, and color red represents the higher SHAP values. The dense area of the plots shows the distribution of the SHAP values for that feature. In this figure, we see that a lower value of $s15$, $s11$ have a high influence on predictions with a higher value of CTF. We also see that four sensors $s15$, $s11$, $s12$, and $s3$ are describing the SVR model throughout the CTF predictions and the other features are only considered in the mid-life of the engine.

We also used SHAP's dependence plot to visualize (see figure 10) the dependence of the feature values (x-axis) over SHAP values (y-axis). This plot is an alternative to a partial dependence plot. We see that sensors $s15$, $s11$, $s3$ and $s2$ have the most linear dependency with their SHAP values. We further see that this plot is very similar to the output produced by the ALE plot.

As kernel explainer of SHAP does not support LSTMs and GRUs (models trained on 3-dimensional data) and Deep Explainer is also not supporting RNN models, we could not carry out our analysis of justifying LSTMs and GRUs decisions with SHAP. We can apply gradient explainer on these models, but the expected gradient is currently only supported for image data.

Figure 7. Comparison of LIME's explanation for LSTMs predictions at the beginning and end of an engine's life span.

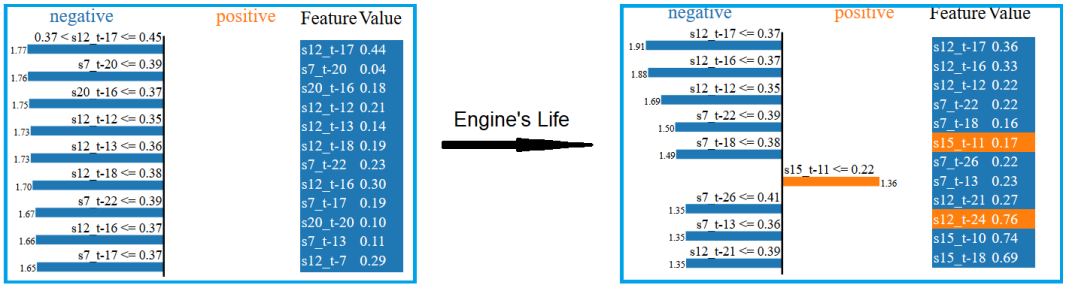


Figure 8. Local explanations of SHAP on SVR for a successful prediction. Each force-plot represents an instance and each instance is 30 instances apart from its previous.

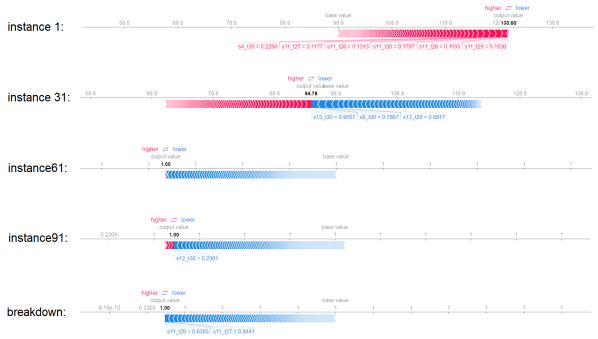
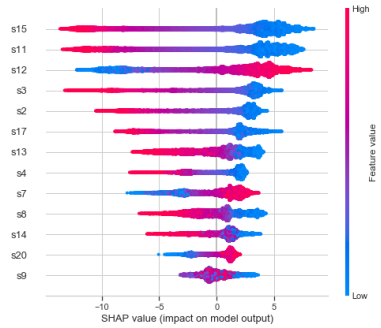


Figure 9. Global explanation of SHAP on SVR (Featuere Summary Plot)



4.2.3. Attribute-Wise

For attribute wise cases such as Partial dependency plot (PDP) and Accumulated Local Effect (ALE), we obtain more global explanations of each input signal. These approaches explain only tabular data and do not accept series as input. To use these approaches and compare their outputs with LIME and SHAP, we describe the temporal behavior of signals as the mean value of each window and transform the series into tabular data. We have interpreted SVR using PD plots, as this is the only black-box model with appropriate input to PD function. The output of this approach is illustrated in 11, where each subplot demonstrates the partial dependency of each feature over the model's predictions. Here we will see all the sensors with the highest correlation with the target variable are having the highest influence on the predictions. The slope of the partial dependency of all features, except $s9$ and $s14$ are almost the same. We further see that for s3, s4, s11, s13, and s15, partial dependency linearly degrade as the time to engine's failure approaches to break down. We see that PDP agrees with LIME (except that here negatively correlated features are also contributing as LIME mostly mentioned the Positively correlated ones). We can argue that by looking at the beginning and end of the line plot for the negatively correlated features calculated for SVR, they are not influencing the start and end of the engine's life as high as the positively correlated features.

As the partial dependence approach does not take the cross-influence of the features into account, we have used the ALE plot to interpret SVR. As mentioned for PDP, ALE is also developed only for tabular data, and therefore, LSTMs and GRUs cannot be interpreted by these approaches. We have used our trained SVR on the mean of each time series window to interpret the decision of this model. Generated plots by this approach are depicted below in Figure 12. ALE plot shows us the S3, s11, s15, and then s12 has the highest and most constant effect on the prediction value. Therefore, we see that ALE agrees more With LIME on SVR's predictions. Almost in all sensors (some more than others), we notice the noise at the beginning and end of engine lifetime where explains why often at the beginning, models had difficulties estimating CTF and also in some cases at the breakdown phase.

5. DISCUSSION

Our experiments on these four approaches to explain the output of TTF-predictive models show that LIME is the most appropriate. Explanations provided by their bar plots are relatively understandable. Feature coefficients provided by LIME are similar to the coefficients extracted from Bayesian linear regression (as well as feature importance from decision trees)

Figure 10. Global explanation of SHAP on SVR (Dependence Plot)

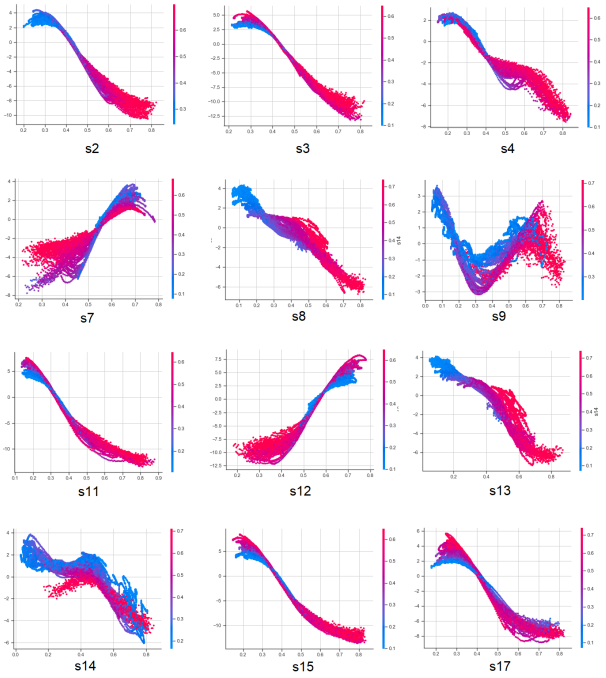
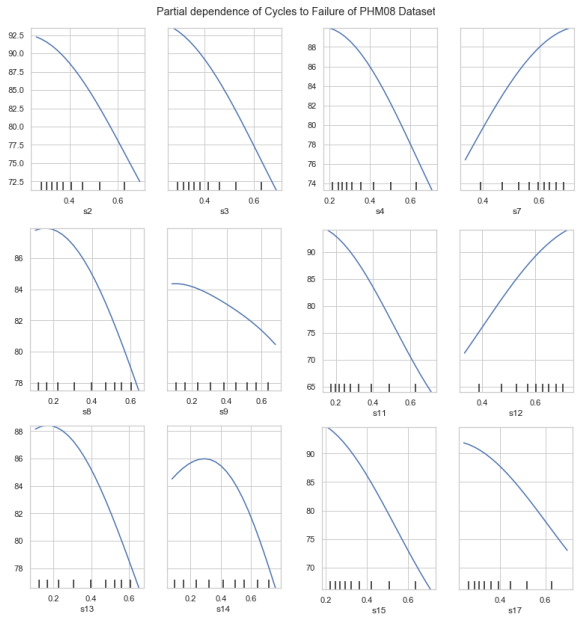


Figure 11. Partial Dependency plots for SVR



and, therefore, consistent in explanations. Contrary to LIME, SHAP's explanations are fairly distributed across all possible cases of the instance for one target feature. However, the computational complexity of this approach is a significant drawback. Furthermore, the kernel explainer does not support models on 3-dimensional data (such as LSTMs and GRUs), and the deep explainer does not support all the latest versions of TensorFlow. However, explaining SVR's decisions was a similar process as using LIME, Bayesian regression or decision trees.

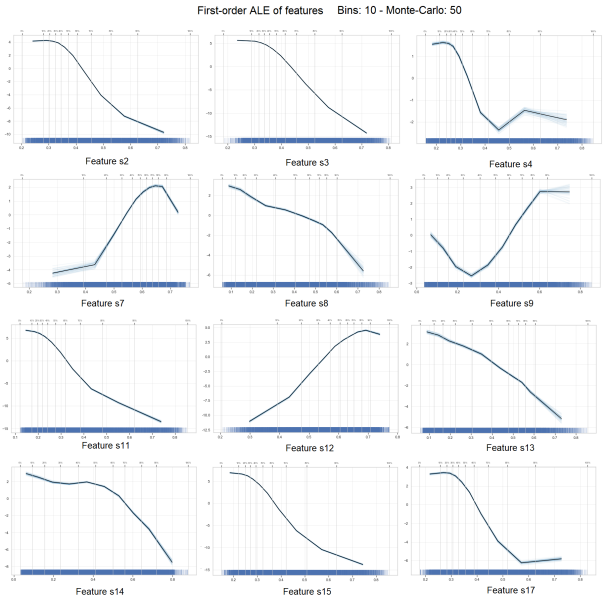
To use attribute-wise approaches, we have represented the temporal behavior of windowed signals with their mean values and could interpret the models. We stress that this approach can be applied to this dataset as the sensor's values tend to increase (or decrease) over time continuously. We applied both PDP and ALE on SVR (as these approaches also only support 2d arrays with static values) and compared their outputs with SHAP, Bayesian linear regression, and decision tree feature summaries. PDP does not take interactions between the features into account, and therefore its plot pointed out the influence of almost all the sensor inputs except $s9$ and $s14$. ALE's output extracts more details from feature interactions and their dependency on the target value, and therefore more accurate feature influence on the predictions. A comparison of these four approaches shows that at the moment, only LIME can effectively provide justifications on black-box model outputs trained on sensor data. SHAP's deep explainer does not support recent TensorFlow versions and has a very high computational complexity. ALE and PDP plots provide global explanations on a model's output but only supports tabular data and therefore are not suitable for RNN architectures such as LSTMs and GRUs.

6. CONCLUSION

In this work, we systematically examined the interpretability approaches such as LIME, SHAP, PDP, and ALE to explain the decisions made by the black-box ML models on the PHM08 C-MAPSS dataset. Moreover, we provided a comparative study on these approaches and highlighted their positives and drawbacks in explaining time series data.

In the beginning phase of our analysis, we compared the predictions of trained models and noticed that often models failed cases are similar to each other. This holds for their successful predictions. We examined the outputs and components of these approaches by investigating properties such as their art of explanation bar plot on feature contribution), supported number of feature dimensions per output, the scope of their explainability, and their computational complexity. By interpreting model decisions using LIME, SHAP, and attribute-wise methods, we notice that the models all consider the same sensors to estimate the countdown to machine failure. Moreover, these approaches all agree with each other on the essential features explaining each model. However, only LIME could provide justifications for LSTMs and GRUs, and the other methods do not support such TTF-predictive models.

Figure 12. Accumulated Local Effect plots for SVR



ACKNOWLEDGMENT

Thanks to Austrian Research Promotion Agency (FFG) for funding this work, which is a part of industrial project under the name DeepRUL, project ID 871357.

REFERENCES

Apley, D. W. (2016). Visualizing the effects of predictor variables in black box supervised learning models. *arXiv preprint arXiv:1612.08468*.

Biran, O., & Cotton, C. (2017). Explanation and justification in machine learning: A survey. In *Ijcai-17 workshop on explainable ai (xai)* (Vol. 8).

Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, *8*(8), 832.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Gade, K., Geyik, S. C., Kenthapadi, K., Mithal, V., & Taly, A. (2019). Explainable ai in industry. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining* (pp. 3203–3204).

Goldstein, A., Kapelner, A., Bleich, J., & Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, *24*(1), 44–65.

Guo, L., Li, N., Jia, F., Lei, Y., & Lin, J. (2017). A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing*, *240*, 98–109.

Heimes, F. O. (2008). Recurrent neural networks for remaining useful life estimation. In *2008 international conference on prognostics and health management* (pp. 1–6).

Kraus, M., & Feuerriegel, S. (2019). Forecasting remaining useful life: Interpretable deep learning approach via variational bayesian inferences. *Decision Support Systems*, *125*, 113100.

Lundberg, S. M., & Lee, S. (2016). A unified approach to interpreting model predictions. *Conference on Neural Information Processing Systems (NIPS 2017)*.

Molnar, C. (2019). *Interpretable machine learning*. Lulu. com.

Mutunga, J. M., Kimotho, J. K., & Muchiri, P. (2019). Estimating the remaining useful lifetime of a turbofan engine using ensemble of machine learning algorithms. In *Proceedings of sustainable research and innovation conference* (pp. 236–240).

Peel, L. (2008). Data driven prognostics using a kalman filter ensemble of neural network models. In *2008 international conference on prognostics and health management* (pp. 1–6).

Ren, L., Cui, J., Sun, Y., & Cheng, X. (2017). Multi-bearing remaining useful life collaborative prediction: A deep learning approach. *Journal of Manufacturing Systems*, *43*, 248–256.

Ribeiro, M., Singh, S., & Guestrin, C. (2016). *Arxiv*(1602.04938v1).

Shrikumar, A., Greenside, P., & Kundaje, A. (2017). Learning important features through propagating activation differences. In *Proceedings of the 34th international conference on machine learning-volume 70* (pp. 3145–3153).

Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic attribution for deep networks. In *Proceedings of the 34th international conference on machine learning-volume 70* (pp. 3319–3328).

Varshney, K. R., & Alemzadeh, H. (2017). On the safety of machine learning: Cyber-physical systems, decision sciences, and data products. *Big data*, *5*(3), 246–255.

Wang, T., Yu, J., Siegel, D., & Lee, J. (2008). A similarity-based prognostics approach for remaining useful life estimation of engineered systems. In *2008 international conference on prognostics and health management* (pp. 1–6).

Zhang, W., Yang, D., & Wang, H. (2019). Data-driven methods for predictive maintenance of industrial equipment: A survey. *IEEE Systems Journal*, *13*(3), 2213–2227.

Zhang, Y., Xiong, R., He, H., & Pecht, M. G. (2018). Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Transactions on Vehicular Technology*, *67*(7), 5695–5705.