Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 52 (2021) 171–178

Transportation Research Procedia

www.elsevier.com/locate/procedia

23rd Euro Working Group on Transportation Meeting (EWGT 2020), Pafos, Cyprus, 16-18 September 2020

Discerning Primary and Secondary Delays in Railway Networks using Explainable AI

David Rößler[a,*], Julian Reisch[b,**], Florian Hauck[a], Natalia Kliewer[a]

[a]*Freie Universität Berlin, Garystraße 21, 14195 Berlin, Germany*
[b]*Synoptics GmbH, Chemnitzer Str. 48b, 01187 Dresden, Germany*

Abstract

In this paper, we study the problem of discerning different reasons for which train delays occur. Given the total amount of delay a specific train builds up at a specific station, we discern the primary delays that would have occurred if there was no other train in the network, such as vehicle problems, from secondary delays which are knock-on delays. Our approach is to train an ML model that predicts the additional delay of a train, given a set of primary features such as weather and secondary features such as the delays of nearby other trains. Methods from explainable AI help to classify to which amount the primary features and to which amount the secondary features contribute to a specific prediction of the model. In particular, we apply SHAP values for the use case of delay management. In addition, we propose a novel method for the use case of railway simulations. In both cases, we use the classification to discern the different reasons for the specific delay.

Keywords: Machine Learning; Feature Engineering; Explainable AI (XAI); Railways; Delay Management; Simulation

1. Introduction

The occurrence of train delays in railway network can have different reasons. Normally, it is known which train builds up how much delay at which station. Unknown, however, is the reason why this delay build-up occurs. Hence, there is much interest in the causes of delays, as different causes imply different ways to prevent these delays from occurring or from propagation. For instance, if it is known that a train *A* often suffers from delay propagated from train *B*, then it would seem prudent to make train *B* punctual so that train *A* will in turn be punctual, as well. Another context where the reasons for delays are of interest is in railway simulations. Here, delays that are train-specific such as rolling stock problems, weather conditions, or signal malfunctions, the so-called **primary delays**, are the inputs. The simulation then calculates knock-on delays, the so-called **secondary delays**. In order for a simulation model to yield results that closely match real-world delays, given that the simulation implements realistic patterns of dispatch

* David Christian Daniel Rößler. Mail: david.roessler@fu-berlin.de; Tel.: +49-30-838-60652
** Julian Reisch. Mail: julian.reisch@synoptics.de; Tel.: +49-176-46073561

2352-1465
10.1016/j.trpro.2021.01.018

and disposition, one must input original real-world information on primary delays. Therefore, a decomposition of the full predicted delay data is needed into primary and secondary delays. The problem we are addressing in our paper is how we can infer from historical delay data the amount of primary and secondary causes contributed to the observed delays, when only the total amount of delay per train and station is known and when this distinction is not reliably encoded in the data. Intuitively, the above example shows that each delay of train A that correlates with a delay of train B is likely to be due to a secondary cause. However, primary causes can be attributed regularly, too. Furthermore, the interplay of several trains and primary causes gives an additional insight.

Many factors may be relevant for delay propagation to occur. Researchers have started to employ machine learning (ML) and artificial intelligence (AI) techniques in all areas for data analysis and prescriptive modeling However, while these techniques offer strong predictive capabilities, researchers and practitioners alike often lack a clear understanding for the actual reasoning behind a specific prediction. For this reason, explainable AI (XAI) is a field of research with increasing importance. In this paper we illustrate a way to make use XAI concepts to get a better understanding of the delay transmission and propagation. For that, we apply SHAP values introduced by Lundberg and Lee (2017) for explaining the deviation of the expected value to the prediction, as can be useful in delay management. Furthermore, we propose an adaptation of the SHAP values in order to decompose the full prediction.

This paper is structured as follows: In section 2, the contribution of this work is illustrated by a short introduction into existing related work. We then introduce a small example in section 3 upon which our methodology is presented. In section 4, we explain how our ML approach works and present computational results for selected stylized and real-world data. Finally, we draw conclusions for further research in section 5.

2. Related Work & Contribution

The approach of analysing correlations in order to understand the process of delay propagation is a widely studied subject. While progress has been made with respect to delay prediction, little work exists which tries to decompose delays into its primary and secondary causes. In Rößler et al. (2019) we studied different correlation coefficients between train arrivals and a delay increase. Kono et al. (2016) employ implication rules for secondary delays similar to the example above when delays of trains correlate. Furthermore, there are numerous works on computing delay propagation, both analytical approaches such as Goverde (2010), and stochastic ones such as Carey and Kwieciński (1994) or Yuan (2006). However, these approaches only try to explain the propagation of delay from one train to another. They are not able to determine to which amount a specific delay in the data is due to secondary or primary effects. Therefore, we use XAI to explain our model's mechanics. Up to our knowledge, there is no example in the literature that applies methods from explainable AI to understanding for causes of railway delays. We are aware that XAI can only be used to understand the ML model and not necessarily the ground truth. However, we do not see how classical econometric approaches for causality learning such as Thistlethwaite and Campbell (1960) can adequately discern primary from secondary delays as desired.

3. Methods

In our ML/XAI approach, we split the feature set F into primary features F_{prime} (such as vehicle type, season or weather) and secondary ones F_{second} (such as the delay of other trains at the station within a time window). The resulting multivariate regression model f predicts the amount of additional delay $f(x)$ that the train in question builds up at a given station for a feature configuration x. Assuming that the model has a reasoning for the prediction similar to the real-world, we deploy a feature analysis to explain which features contributed and to which extent to the prediction. In our approach, we try to understand how a specific model reaches a prediction for the delay difference. In the case of linear regression, it is easy to see that the regression coefficients in the model equation suffice to explain a certain prediction.

$$f(x) = \sum_{i \in F} c_i x_i = \sum_{i \in F_{prime}} c_i x_i + \sum_{i \in F_{second}} c_i x_i \tag{1}$$

Thus, if applied to our use case, discerning primary from secondary causes for delay is straight forward. Note that the intercept plays a special role which will be discussed below. A drawback of this well-explainable type of regression model is that it is limited to explaining additive effects. Hence, such type of model is a not well suited as a general approach to identifying non-linear or piece-wise linear relationships. When models become more complex (in the sense of parameter estimates) such that they operate within a so-called black box, other, more general methods for explaining the reasoning of the model are needed.

Example Model. Let the features f_{RF} for *rainfall* and $f_{Y'}$ for the delay of train Y' denote the only primary and secondary feature, respectively. Then, f constitutes a prediction model for the increase of delay of train Y where

$$f(x) = \begin{cases} 90 \text{ sec}, & \text{if } x_{RF} \geq 100 \text{ sec} \\ 90 \text{ sec}, & \text{if } x_{Y'} \geq 100 \text{ sec} \\ 0 \text{ sec}, & \text{otherwise} \end{cases} \tag{2}$$

If the data consists of $1/3$ of values where $f(x) = 90$ sec and $2/3$ where $f(x) = 0$ sec, the the expected value of f is $\mathbb{E}[f] = 1/3 * 90 + 2/3 * 0$ sec $= 30$ sec. If there is no data in x, then f will predict the expected value 30 sec.

3.1. Explaining Deviations from Expected Value with SHAP Values

SHapley Additive exPlanations (SHAP) values, introduced by Lundberg and Lee (2017), are based on a concept from game theory to determine the fair allocation of an outcome (e.g. a collective reward) to participants in a game of collaboration reciprocally to their individual contributions to achieving the outcome. They can be approximated with SHAP values, which serve as measurements for the contributions of features to the predictions made by an ML model. Given a set of features F, these values are a means to estimating the marginal contribution of a single feature $i \in F$ to a specific prediction result. They are obtained by performing the prediction for all possible subsets of F containing i, and then comparing the results for when the feature i was omitted or included.

$$\Phi_i(f, x) = \sum_{\substack{F' \subseteq F \\ i \in F'}} \frac{|F'|!(|F| - |F'| - 1)!}{|F|!} (f_{F'}(x) - f_{F' \setminus \{i\}}(x)) \tag{3}$$

where $f_{F'}(x)$ denotes the prediction of the model trained when only feature from F' were given. In their paper, Lundberg and Lee describe methods to avoid that for each subset F' of F, a separate model has to be trained.

Lemma 1. *Summed over all features, the SHAP values add up to the difference between the expected value and the prediction, i.e.* $\sum_{i \in F} \Phi_i(f, x) = f(x) - \mathbb{E}[f]$.

Proof. The efficiency property of the SHAP values reads as follows: $\sum_{i \in F} \Phi_i(f, x) = f_F(x) - \Phi_\emptyset(f, x)$. If we set $\Phi_\emptyset(f, x) = \mathbb{E}[f]$, then we have $f_F(x) = f(x)$ which concludes the proof. □

Use Case 1: Delay Management. Consider the example model and assume that $f_\emptyset(x) = \mathbb{E}[f]$ and that f_{RF} and $f_{Y'}$ predict just as f when $x_{Y'} = 0$ and $x_{RF} = 0$, respectively. The expected increase of delay of train Y in a station is 30 seconds. Given the feature values $x = (x_{RF}, x_{Y'}) = (0, 120)$ of a specific day, the ML model f predicts an increase of 90 seconds. This is due to train Y' that is extraordinarily delayed. The SHAP value $\Phi_{Y'}(f, x) = 75$ and $\Phi_{RF}(f, x) = -15$ reflect this trend. Hence, an operations manager can now decide that train Y will not wait for passengers from train Y' in order to avoid the extra delay.

By Lemma 1, we can decompose $f(x)$ into $\mathbb{E}[f] + \sum_{i \in F_{prime}} \Phi_i(f, x) + \sum_{i \in F_{second}} \Phi_i(f, x)$.

3.2. Explaining the Entire Prediction

The standard SHAP values set $\Phi_\emptyset(f, x) = \mathbb{E}[f]$ and are hence only suitable for computing the marginal contribution of a feature to the deviation of the prediction from the expected value. This definition of $\Phi_\emptyset(f, x) = \mathbb{E}[f]$, however, is arbitrary. If we want to decompose the full amount of increase of delay into a primary and a secondary part, we set $\Phi_\emptyset(f, x) = 0$, then Lemma 1 implies $f(x) = \sum_{i \in F_{prime}} \Phi_i(f, x) + \sum_{i \in F_{second}} \Phi_i(f, x)$.

Use Case 2: Railway Simulations. Consider the example model and x from Use Case 1. The model predicts an increase of $f(0, 120) = 90$ seconds which decomposes into $\Phi_{Y'} + \Phi_{RF} = 90 + 0$. We conclude that the whole 90 seconds are due to a secondary delay. In a simulation, we would set no (primary) delay for train Y because the only delay Y build-up is due to train Y' and this knock-on delay will be computed within the simulation itself.

Note of caution: Despite its potential explainability, the adapted SHAP values do not necessarily satisfy the null player property anymore, that is, features with no contribution in any coalition can have a positive adapted SHAP value.

4. Computational Experiments

We conduct our experiments by following the following steps: First, we apply a feature generation procedure that provides us with features suited to account for irregular patterns, as well as threats to continuity and linearity in the data, e.g. structural jumps. Then, we train the ML models on various combinations of trains and stations – the cases – and try to improve generalization and goodness of fit by means of hyper-parameter tuning and feature selection. For the resulting models that yield predictions of acceptable quality, we use SHAP values in order to measure the amount to which primary and secondary features contribute to the prediction.

Data Model. Following the transmission model presented in Rößler et al. (2019), we use the difference between the arrival and departure delay of a train at a station as the response variable of our model. The following facts are used as input variables:

- **Arrival delay** of the focal train. If there is buffer time scheduled for the train, a train will be able to recover from some delay, which leads to a negative delay build-up. However, if the buffer is exhausted, additional delay may result upon departure. (Secondary)
- **Arrival delays of other trains** within the time window. These are supposedly connected with the focal train via passenger transfers. (Secondary)
- **Departure delays of other trains** within the time window, as delayed departures into the next section might block the focal train. (Secondary)
- **The day of the week.** A distinct weekly pattern can be observed. (Primary)
- **The season.** During the year, delays vary. This is a promising candidate for the delay analysis and prediction of individual trains. (Primary)
- **Weather data** (i.e. daily total rainfall and average temperature on the observed day and in the area of the station (here the city)). (Primary)

Test cases. To test our approach, we conduct experiments on stylized and real-world data. The train constellations in the real-world data are selected based on expert knowledge, giving us the ability to compare further verify the ML reasoning. We use several stylized data instances to assert the models ability to pick up certain feature relationships and some rather simple disposition rules, resulting in primary and secondary delay increase.

The **stylized instances** are artificially created and implement sets of rules as presented in table 1. Note that while cases 1 and 2 exhibit more obvious dependencies, we have introduced some amount of noise within case 3 to make it harder for models to pick up the relevant underlying process.

For the **real-world instances**, we are provided operational delay data by *DB Netze AG* with a time period of one year. We perform extensive data preparation, i.e. outlier removal and imputation of missing values. We focus on specific trains at specific stations. For each such combination of train and station, we find the set of other trains that might interact. These are trains which run within a time difference of at most 30 minutes and not on less than

150 days within the considered year. For the experiments we have selected three structurally different examples from the real-world data (s. Table 1). Note that real-world train numbers and station names are anonymized. Thus, in all presented cases, trains are designated with a letter from the alphabet. The train whose delay increase is modelled – the focal train – always carries the letter *a*. Departure and arrival delays are named with prefix *dep.* or *arr.* followed by the letter identifying the respective train.

Stylized data	**Real-world data**
St1. Train a exhibits positive change in delay, if: (1) Train b is delayed; 50 % of train b's delay are transmitted. (2) The day is a Monday. (3) It rains.	**Rw1.** This is a very long-distance IC/EC train, without a priori known passenger connections or dependencies.
St2. Train a exhibits positive fixed change in delay, if: (1) Train b is delayed (2) There was strong rain.	**Rw2.** This is an example of a regular train with frequent arrival delay, large planned buffer time and defined passenger connections (waiting for the arrival of other trains).
St3. This data set is created by randomly drawing arrival delays of trains b, c, and d from a gamma distribution using the formula $f_r(\alpha = 1, \beta = 2) * 100 - 60$, resulting in realistic delays. The trains transmit 80 % (b), 200 % (c) and 50 % (d) of delay to train a. Furthermore, days of week and seasons are drawn uniformly. Seasons from winter to autumn create delays of 90, 120, 150 and 180 seconds. The days of week in order starting on Monday contribute 25, 30, 50, 45, 40, 15, 20, 90 seconds.	**Rw3.** Regular late evening IC train. without any known dependencies.

Table 1: Short description of stylized and real-world data instances.

4.1. Model Evaluation & Selection

A prerequisite for model interpretation, a reasonable goodness of fit has to be achieved. We employ a nested cross-validation (CV) scheme to evaluate a) the selection of relevant features by means of the reverse feature elimination procedure and b) the tuning of the hyper-parameters of the ML models. To attain comparable results between splits, we use a stratified k-fold scheme (with $k = 5$), balancing the folds by season for the outer loop. For the inner loop, regular k-fold CV is used, however with $k = 10$ folds and the hyper-parameter tuning is performed with a randomized grid search. We measure the goodness of fit by means of R^2 – the coefficient of determination – and the mean absolute error (MAE). R^2 is a widely used measure for the model fit, as it can be interpreted as the relative portion of variance in the dependent variable that can be explained by the model. The MAE, on the other hand, is a measure of the absolute prediction error. This is of importance to be observed in addition to R^2, since a model with little explanatory power may still yield valuable insights, if it has a relatively small average absolute error.

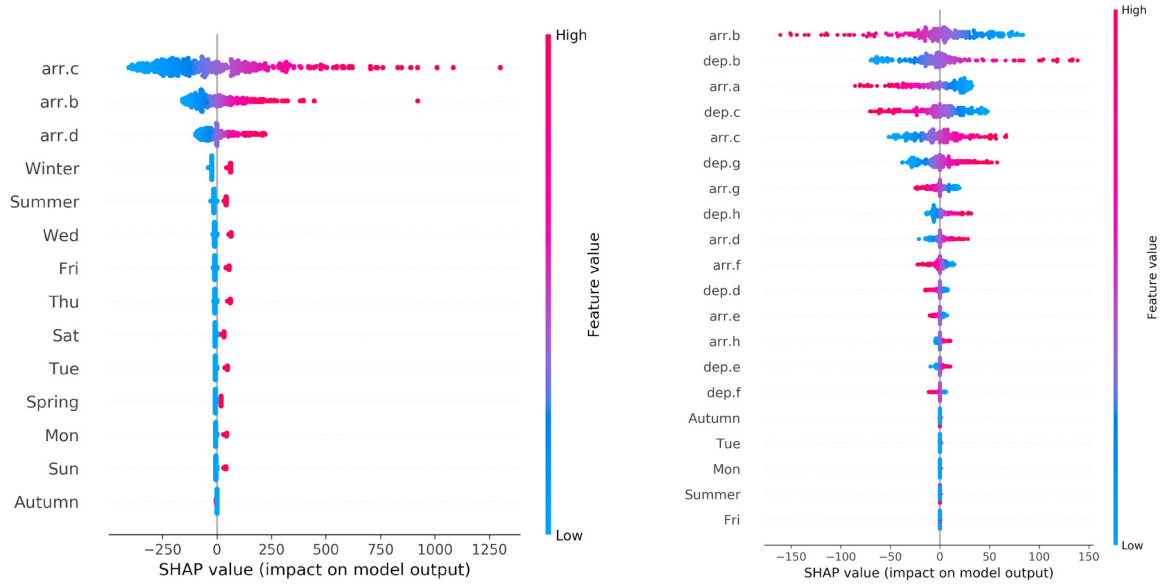
		Stylized data		**Real-world data**	
Case	**Model**	**⊘R^2**	**⊘MAE**	**⊘R^2**	**⊘ MAE**
1	RF	0.586	22.145	0.201	42.270
1	MLP	0.597	22.685	0.607	30.083
2	RF	1.000	0.000	0.810	56.560
2	MLP	1.000	0.147	0.933	34.485
3	RF	0.948	63.124	0.358	48.818
3	MLP	1.000	0.059	0.641	41.797

Table 2: Average R^2 and MAE scores of Random Forest (RF) and Multi-Layer Perceptron (MLP) on unseen test data from the outer CV-loop for both data sets.

For the selected real-world and stylized data, we achieve reasonable model fits using random forests (RF) and multi layer perceptrons (MLP). This is en par with expected characteristics within the data, i.e. the occurrence of multiple piece-wise linearities or non-linear relationships, and the selected binary predictors (i.e. seasonality and day of the week). Table 2 gives an overview over the cross-validation results. In most cases, the average R^2 for both models lies well above 0.5. For the stylized data cases 2 and 3, the models even seem capable to pick-up almost all the variance in the data while avoiding to learn any noise. Yet, the magnitude of the MAE achieved by the RF in case three is enormous by comparison with the MLP. The RF model for the real-world cases 1 and 3 scores low regarding R^2, however, the average MAE is relatively low, as compared to the MLP result. Thus, even the lower scoring RF models may be of some value. In general, with the exception of the stylized case 2, but especially on the real-world data, the MLP outperforms the RF[1].

4.2. SHAP values

In this following section, we analyse the calculated SHAP values and try to match their meaning with our interpretations from section 3. We also present results for the transformed SHAP values as proposed in section 3.2[2].



(a) Stylized data: Case 3 (MLP model)

(b) Real-world data: Case 1 (MLP model)

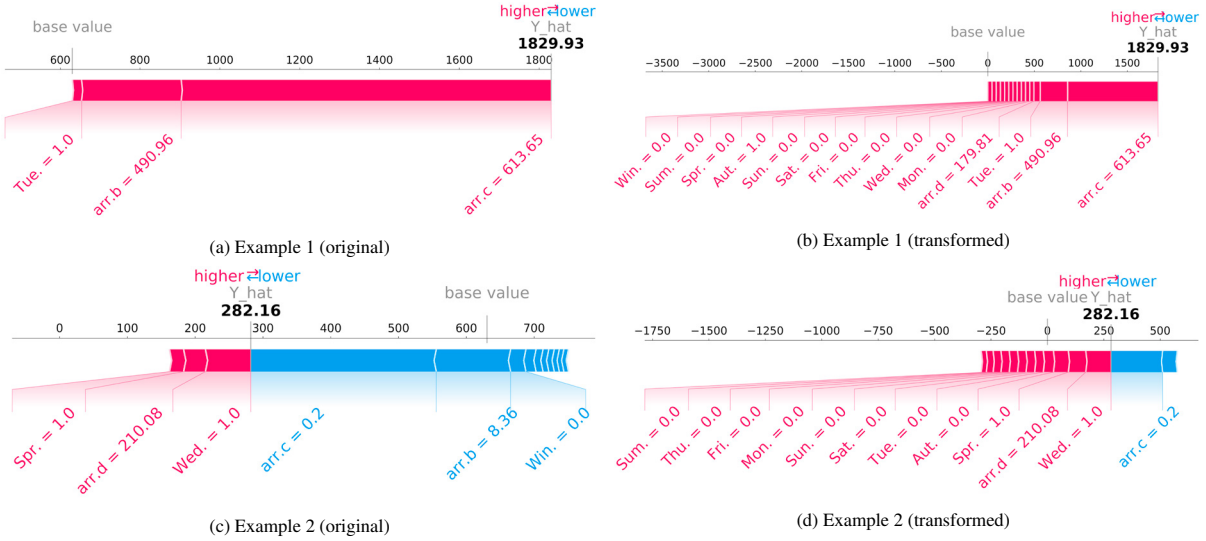
Fig. 1: Summary plots for the MLP model for the three cases. The estimated SHAP values are plotted in lanes for each feature; Features are ordered by the absolute cumulative feature importance. Additionally, the data points are colored according to the observed feature values (pink: high value; blue: low value).

Figure 1a shows the feature importance plot for the stylized data case 3. It captures the expected effect: Higher arrival delays of train c lead to an increase in the predicted value while lower delay (blue dots) reduce the prediction value. This is true for trains b and d as well, but by a much smaller magnitude. Furthermore, the primary features reflect the pattern, we have created in the data. In figure 1b, the feature importance measures for the more complex model for real-world case 1 – the IC/EC train – are plotted. Higher arrival delays lead to negative deviations of the predicted from expected values. Moreover, it becomes obvious, how departure and arrival delays of other trains seem

[1] The proposed ML and XAI methods are implemented in *Python* using algorithms provided as part of the packages *scikit-learn* (https://scikit-learn.org/stable/) and *shap* (https://github.com/slundberg/shap) for the calculation of the SHAP values

[2] The transformation is achieved by distributing the expected value of the prediction $\mathbb{E}[f]$ to the predictors. Thus the SHAP values are obtained as: $\Phi_i(f,x) = \Phi'_i(f,x) + \mathbb{E}[f] * (n-1)!/n!$.

to cancel out one another. Another property, not present in the stylized data is the use of buffer times which the model seems to capture. The arrival delay of train a (*arr.a*) lowers the predicted delay build-up while very small values lead to an increase only up to a limited degree. In general, the primary features have a far smaller impact on the prediction than secondary ones. We can further investigate differences between the two SHAP values and better understand the presented approach by analyzing individual data points as shown in figures 2 for the stylized data and figures 3 for the real-world data, respectively. In the former, example 1 exhibits a very high delay increase and, in example 2, we can observe a medium estimated delay build-up. In example 1, it is obvious that the secondary features – the arrival delay



(a) Example 1 (original)

(b) Example 1 (transformed)

(c) Example 2 (original)

(d) Example 2 (transformed)

Fig. 2: Force plots for selected MLP predictions in the stylized data case 3. Plots in the left column use the original SHAP values, plots in the right are based on the transformed SHAP values. Bars show the effects ob select features on the predicted value. The *base value* marks the expected value and the **black** number is the estimated delay build-up for the data object.

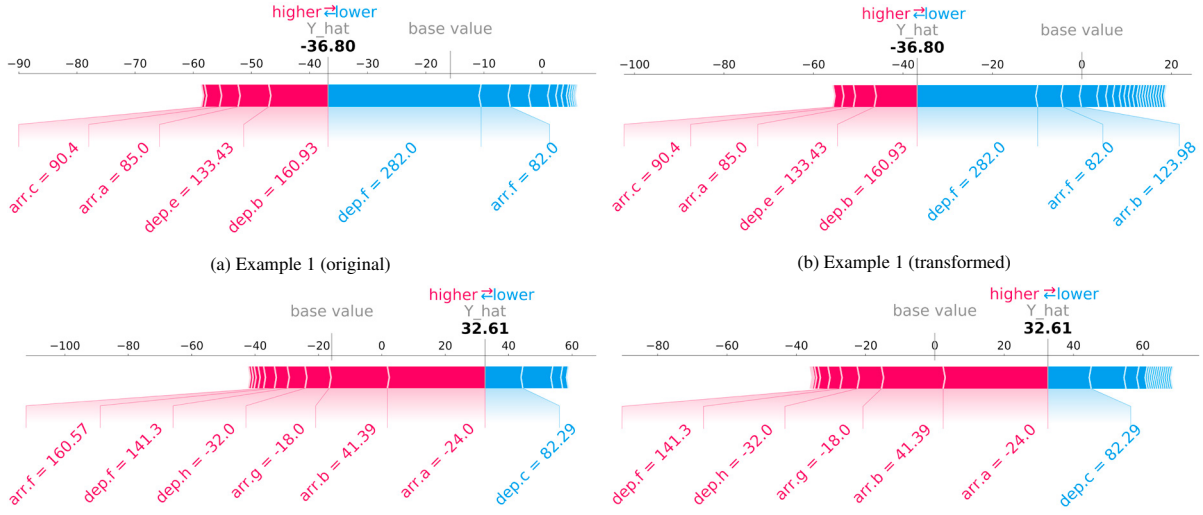
of train b (490 seconds), but even more importantly, of train c (613 seconds) – have a major positive impact on the predicted delay difference. There are no diminishing factors involved. In the example 2, primary features (*Spring* and *Wednesday*) have a distinct upward impact on the delay build-up, while secondary features, i.e. the arrival delays of trains c, b, and d push the prediction downward, as these exhibit relatively small values. Figures 3 show two examples of small delay differences. In example 1, the MLP predicts some delay recovery; in example two, a subtle increase is predicted. We can observe only very small effects of the primary features. These are overcast by the strongly estimated secondary effects.

Comparing the basic SHAP and transformed SHAP values, we can observe that the latter maintain proportions, however, the negative impact of features with smaller observed values is reduced. Consequently, the impact of features - primary features among them -, appears more pronounced. Especially for the real-world cases, an identification of the impact of primary and secondary features becomes possible only after the transformation, which allows us to use these values in a simulation tool, as was detailed in use case 2 (Section 3).

5. Conclusion and Outlook

Our contribution was to apply XAI and SHAP values in particular to the use cases of railway delay management and simulations. Moreover, we proposed a method to overcome the obstacle of SHAP values to explain only the difference between expected value and prediction but the entire prediction which is relevant for the use case of railway simulations. In experiments with stylized and real-world data, we have shown the merits of the proposed methods.

However, we see limitations of our methods. On the one hand, both the SHAP values and our adapted SHAP values can be negative which is difficult to interpret. Then, a model might have features that are not merely primary



(a) Example 1 (original)

(b) Example 1 (transformed)

(c) Example 2 (original)

(d) Example 2 (transformed)

Fig. 3: Force plots for selected MLP predictions in the real world data case 1. Plots in the left column use the original SHAP values, plots in the right are based on the transformed SHAP values. Bars show the effects ob select features on the predicted value. The *base value* marks the expected value and the **black** number is the estimated delay build-up for the data object.

nor secondary. In particular, the intercept is such a feature. Also, we see the shortcoming of our methodology to explain only a prediction model and not the ground truth.

The presented approach is an endpoint to a more complete ML approach to analyzing historical operational railway data. The presented ML models, although exhibiting reasonable performance, suffer from the sole availability of small samples to learn from and a rather rudimentary feature selection and feature engineering procedure. We are currently conducting related research to improve the predictive power and thus the informative value of the obtained SHAP values.

Acknowledgements

We would like to thank *DB Netze AG* for providing us with historical delay data. Moreover, we thank Thorsten Schaer and his team for giving valuable insights to their operations and all the helpful comments and inspiring discussions which helped improve our work.

References

Carey, M., Kwieciński, A., 1994. Stochastic approximation to the effects of headways on knock-on delays of trains. Transportation Research Part B: Methodological 28, 251–267.

Goverde, R.M.P., 2010. A delay propagation algorithm for large-scale railway traffic networks. Transportation Research Part C: Emerging Technologies 18, 269–287.

Kono, A., Yakubi, H., Tomii, N., 2016. Identifying the cause of delays in urban railways using datamining technique, in: Asian conference on Railway infrastructure and Transportation, pp. 227–230.

Lundberg, S.M., Lee, S.I., 2017. A unified approach to interpreting model predictions, in: Advances in neural information processing systems, pp. 4765–4774.

Rößler, D., Kliewer, N., Reisch, J., 2019. Modeling delay propagation and transmission in railway networks, in: Proceedings of the 14th International Conference on Wirtschaftsinformatik, pp. 98–111.

Thistlethwaite, D.L., Campbell, D.T., 1960. Regression-discontinuity analysis: An alternative to the ex post facto experiment. Journal of Educational Psychology 51, 309—-317.

Yuan, J., 2006. Stochastic modelling of train delays and delay propagation in stations. volume 2006. Eburon Uitgeverij BV.