Explainable Process Prescriptive Analytics

Alessandro Padella[*], Massimiliano de Leoni[*], Onur Dogan[+], Riccardo Galanti [†*]
[*] University of Padua, Padua, Italy,[+]Izmir Bakiracy University, Izmir, Türkiye, [†] IBM, Bologna, Italy
Email : alessandro.padella@phd.unipd.it, deleoni@math.unipd.it,
onur.dogan@bakircay.edu.tr, riccardo.galanti@ibm.com

***Abstract*—Process-aware Recommender systems (PAR systems) are information systems that aim to monitor process executions, predict their outcome, and recommend effective interventions to have better ends. Recent literature puts forward proposals of PAR systems that return valuable, practical recommendations. However, recommendations without sensible explanations prevent process owners from feeling engaged in the decision process or understanding why these interventions should be carried out. Therefore, the risk of process owners do not trust the PAR system and overlook these recommendations is high. This paper proposes a framework to accompany recommendations with sensible explanations based on the process behavior, the intrinsic characteristics, and the context in which the process is carried on. The paper illustrates the potential relevance of these explanations for process owners in two use cases.**

***Index Terms*—Process Mining, Prescriptive Analytics, Recommender Systems, Explainability, Process Improvement**

I. Introduction

Process-aware Recommender systems (hereafter shortened as PAR systems) are a specific class of Information Systems that aim to *predict* how process instances are going to evolve, and to *recommend* the corrective activities to recover the instances with higher risk not to achieve the expected outcome [1]. Outcomes are defined in terms of process-specific Key Performance Indicators (KPIs), such as costs, execution times, or customer satisfaction.

Conceptually, a PAR system can be seen by two constituent blocks: Predictive Monitoring, and Prescriptive Analytics. The former aims at the prediction aspects, whereas the latter focuses on recommending the corrective activities.

Recent literature has put forward a number of proposals of prescriptive analytics. However, these do not provide explanations of the reasons that led the systems to propose the suggested recommendations [2], [3].

However, recommendations without sensible explanations prevent process actors from feeling engaged in the decision process of the recommendations, and from understanding the rationale behind. In this scenario, process actors tend to not follow recommendations, which are based on data, but rather to enact contingency actions that are subjective and, thus, can even potentially worsen the KPI outcome [1].

This papers proposes a framework to extend PAR systems with explanations that provide process actors with the rationale behind the choice of the recommended activities. Explanations are based on process-related characteristics, such as the values of process variables (e.g., the customer is requesting a loan of 70 K€), the activities performed in process (e.g., the loan assessment has already been repeated twice), or the resources that performed activities (e.g., the loan application was validated by Alex, who is a manager).

Our framework for the explanation of the selected recommendations leverages on current state of the art of Explainable AI, specifically on the game-theory approach of the Shapley Value (cf. Section III-C). The proposed framework is independent of the machine- or deep-learning technique that is employed to generate the recommendation. However, we aim to instantiate the framework to prove its effectiveness. Therefore, we extended the PAR system proposed in [2] with our explanation framework, using gradient boosting on decision trees as machine-learning model for generating predictions and recommendations.

Experiments were run on two real-life datasets, which referred to instances of one process in an Italian bank, and one at Volvo Belgium. The experiments first confirmed the quality of recommendations to improve the KPIs of interest in relevant process instances, then illustrated the typical shapes of the accordant explanations generated by our framework proposal.

II. Related Works

Literature has largely focused on predicting the future outcome of process instances [4], [5], [6]. A recent body of research is being focused on recommend which activities to work on as next, to improve process' KPIs of interest, or to suggest the most common continuations [2], [3], [7], [8].

In parallel, several research works focused on explaining the reasons of these predictions and the affecting factors, using approaches based on Shapley values [9], attention mechanisms [10], or based on counter facts [11].

However, no works exist that provide an explanation of the recommendations, in addition to the rationale of the predicted KPI values. Note that explaining predictions is certainly different than explaining recommendation: while explaining the predictions focuses on every aspect that significantly affects the expected process' outcome, explaining the recommendations should solely focus on those aspects whose negative impact on the KPI values is much more mitigated by the recommendations than by other actions. Last, we did consider to use attention mechanisms [12] because they require neural networks, while we employ random-forest models. Furthermore, attention mechanisms are disputed whether or not they are always correlated to feature importance [13].

2022 4th International Conference on Process Mining (ICPM 2022), ISBN 979-8-3503-9714-7

III. Preliminaries

A. Process Predictive Analytics

In this section, we discuss the typical techniques adopted to train a predictive and recommendation model starting with business processes.

The starting point for a prediction system is an *event log*. An event log is a multiset of *traces*. Each trace is a sequence of events, each describing the life-cycle of a particular *process instance* (i.e. a *case*) in terms of the *activities* executed and the process *attributes* manipulated. An attribute can be given a value $\perp$ indicating uncertainty whether or not a value was assigned to an attribute by an event and what this value was.

Definition III.1 (Events)**.** *Let $\mathcal{A}$ be the set of process' activities. Let $\mathcal{T}$ the set of possible timestamps and let $\mathcal{V}$ the set of process attributes. Let $\mathcal{W}_{\mathcal{V}}$ be a function that assigns a domain $\mathcal{W}_{\mathcal{V}}(x)$ to each process attribute $x \in \mathcal{V}$. Let $\overline{\mathcal{W}} = \cup_{x \in \mathcal{V}} \mathcal{W}_{\mathcal{V}}(x) \cup \perp$. An event is a tuple $(a, t, v) \in \mathcal{A} \times \mathcal{T} \times (\mathcal{V} \not\rightarrow \overline{\mathcal{W}})$ where a is the event activity, v is a partial function assigning values to process attributes with $v(x) \in \mathcal{W}_{\mathcal{V}}(x)$, and t its timestamp.*

A trace is a sequence of events, the same event can potentially occur in different traces, namely attributes are given the same assignment in different traces. This means that potentially the entire same trace can appear multiple times. This motivates why an event log is to be defined as a multiset of traces:[1]

Definition III.2 (Traces & Event Logs)**.** *: Let $\mathcal{E} = \mathcal{A} \times \mathcal{T} \times (\mathcal{V} \nrightarrow \overline{\mathcal{W}})$ be the universe of events. A trace σ is a sequence of events, i.e. $\sigma \in \mathcal{E}^*$. An event-log $\mathcal{L}$ is a multiset of traces, i.e. $\mathcal{L} \subset \mathbb{B}(\mathcal{E}^*)$.*

Given an event $e = (a, t, v)$, the remainder uses the following shortcuts: $activity(e) = a$, $time(e) = t$ and $variables(e) = v$. Also, given a trace $\sigma = \langle e_1, \ldots, e_n \rangle$, $prefix(\sigma)$ denotes the set of all prefixes of σ, including σ: $\{\langle\rangle, \langle e_1 \rangle, \langle e_1, e_2 \rangle, \ldots, \langle e_1, \ldots, e_n \rangle\}$. For building our recommender system, we need to define what we aim to optimize, i.e. the goal of our recommendation: hereafter this is named Key Performance Indicator (KPI), and depends on the specific process domain.

Definition III.3 (KPI Function)**.** *Let $\mathcal{E}$ be the universe of events. A Key Performance Indicator (KPI) is a function $\mathcal{K} : \mathcal{E}^* \times \mathbb{N} \rightarrow \mathbb{R}$ such that, given a (prefix of a) trace $\sigma \in \mathcal{E}^*$ and an integer $1 \leq i \leq |\sigma|$,[2] $\mathcal{K}(\sigma, i)$ returns the KPI value of σ after the occurrence of the first i events.*

As it will be clear later, we need to assume KPI values to be numerical in order for the explanations of the recommendations to be computed. Therefore $img(\mathcal{K})$ is the set of all possible KPI values. With an abuse of notation, we indicate $\mathcal{K}(\sigma) = \mathcal{K}(\sigma, |\sigma|)$, namely the KPI value after the occurrence of events in trace σ. Please note that the KPI values can also belong to the set of timestamps, being them easily associable to numbers. Since our goal is to optimize a KPI, depending on the business requirements and on the KPI's type, we define $\sqsupset_{\mathcal{K}}$ as follows: given two values $a, b \in \mathbb{R}$, we refer to $a \sqsupset_{\mathcal{K}} b$ meaning that a is better than b for $\mathcal{K}$'s definition. Note that our KPI definition is assumed to be computed a posteriori, when the execution is completed and leaves a complete trail as a certain trace σ. In many cases, the KPI value is updated after the occurrence of each event, i.e. after each activity execution. We aim to be generic and account for all relevant domains. Given a trace $\sigma = \langle e_1, \ldots, e_n \rangle$ that records a complete process execution, the followings are example of two potential KPI definitions:

- *Total Time.* Given a σ's prefix of i events, $\mathcal{K}_{total}(\sigma, i)$ measures the difference between the timestamp of the last future event of the trace and the timestamp of the first event (i.e. $time(e_n) - time(e_1)$), In this case, $a \sqsupset_{\mathcal{K}} b$ if and only if $a < b$, meaning it is desiderable to minimize the time.
- *Activity Occurrence.* It measures if a certain activity is going to eventually occur in the future, such as an activity *Open Loan* in a loan-application process. The corresponding KPI definition for the occurrence of an activity a is $\mathcal{K}_{occur\_a}(\sigma, i)$, which is equal to 1 if the activity a occurs in $\langle e_{i+1}, \ldots, e_n \rangle$, 0 otherwise. Here, $\sqsupset_{\mathcal{K}}$ depends on whether or not it is desirable for the activity to happen.

We can now define the prediction problem on logs as follows.

Definition III.4 (Prediction Problem on Event Logs)**.** *Let $\mathcal{L}$ be an event log that records the execution of a given process, for which a KPI $\mathcal{K}$ is defined. Let $\sigma' = \langle e_1, \ldots, e_k \rangle \in \mathcal{L}$ be the trace of a running case, which eventually will complete as $\sigma_T = \langle e_1, \ldots, e_k, e_{k+1}, \ldots, e_n \rangle$. The prediction problem can be formulated as forecasting the value of $\mathcal{K}(\sigma_T, i)$ for all $k < i \leq n$.*

In the process mining literature, this problem has been faced with different machine learning models [4], [14], [15], [16], [17], [18], [5], [6], [9]. We approach the problem by estimating a function $\Phi_{\mathcal{K}} : X_1 \times \ldots \times X_m \rightarrow \mathbb{R}$ which for an incomplete trace σ', forecasts the values of the KPI $\mathcal{K}$. Therefore, a KPI definition $\mathcal{K}$ is necessary as input with the constraint $img(\mathcal{K}) \subset \mathbb{R}$. Afterwards, each prediction technique requires the definition of the domain $X_1 \times \ldots \times X_m$ and a **trace-to-instance encoding function** $\rho_{\mathcal{L}} : \mathcal{E}^* \rightarrow X_1 \times \ldots \times X_m$, which maps each (prefix of a) trace σ in an vector $\rho_{\mathcal{L}}(\sigma) \in X_1 \times \ldots \times X_m$ of m elements that can be of different nature, such as a process attribute, a timestamp, or the number of executions of an activity in σ. The prediction model is trained off-line via a dataset $\mathcal{D}$ that is created from an event log $\mathcal{L} \subset \mathbb{B}(\mathcal{E}^*)$ as follows: each prefix σ^p of each trace $\sigma \in \mathcal{L}$ generates one distinct item in $\mathcal{D}$ consisting of a pair $(x, y) \in (X_1 \times \ldots \times X_m \times img(\mathcal{K}))$ where $x = \rho_{\mathcal{L}}(\sigma^p)$ and $y = \mathcal{K}(\sigma, |\sigma^p| + 1)$. The results is an **oracle function** $\Phi_{\mathcal{K}} : X_1 \times \ldots \times X_m \rightarrow \mathbb{R}$ such as given a trace σ and any

[1] $\mathbb{B}(X)$ indicates the set of all multisets with the elements in set X.

[2] Given a trace σ, $|\sigma|$ indicates the number of events in σ.

$\sigma^p \in prefix(\sigma)$, $\Phi_{\mathcal{K}}(\rho_{\mathcal{L}}(\sigma^p))$ returns the predicted KPI value $\mathcal{K}(\sigma, |\sigma^p|+1)$.

To keep the notation simple, we are going to refer to $\Phi_{\mathcal{K}}(\rho_{\mathcal{L}}(\sigma))$ as $\Phi_{\mathcal{K}}(\sigma)$. Our framework is independent from the employed predictive model; however, we had to instantiate the framework to prove its effectiveness. In the implementation and experiments, we use the Catboost method [19], [9]. It is a high-performance open source framework for gradient boosting on decision trees. In the domain of Catboost learning, the definition of the trace-to-instance encoding function also considers the history of each prefix of σ. It is done by considering the number of times that each activity has been performed. Consequently, before defining the trace-to-instance encoding function, we define the function $\rho^{aggr}_{\mathcal{A}}(\langle e_1, \ldots, e_n\rangle)$. Here, for each activity $a \in \mathcal{A}$, one dimension exists in $\rho^{aggr}_{\mathcal{A}}(\sigma) : \mathcal{E}^* \to (\mathbb{N})^{|A|}$ that takes on a value equal to the number of events $e \in \sigma$ that refer to a (i.e. such that $activity(e) = a$). The function ρ is then defined as: $\rho_{\mathcal{L}}(\langle e_1, \ldots, e_n\rangle) = \rho^{aggr}_{\mathcal{A}}(\langle e_1, \ldots, e_n\rangle) \bigoplus activity(e_n) \bigoplus_{v \in \mathcal{V}} variables(v)(e_n)$[3]. When the event log $\mathcal{L}$ on which ρ depends is evident from the context, we omit the subscript $_{\mathcal{L}}$.

B. Generating Recommendations

A process aware recommender-system aims to recommend the k-top best next activities to improve the relevant KPI. However, these activities need to be valid from a domain viewpoint. We avoid the strong assumption that a process model exists that prescribes how process instances must be executed. We also assume an activity to be valid in a certain process state if it has been previously observed in other executions for the same state.This requires to provide a *state-representation function.*

Definition III.5 (State-representation function)**.** *Let σ be a trace, and $\mathcal{R}$ a set of the possible representations. The function $l^{state} : \mathcal{E}^* \to \mathcal{R}$ that for each (prefix of a) trace returns the state, is called state-representation function.*

The determination of the activities allowed after the occurrence of a sequence of events requires to build a *Transition System* where nodes are the state observed in the log and arcs are activities observed in those states [20].

Definition III.6 (Transition system)**.** *Let l^{state} be a state-representation function, $\mathcal{L}$ an event log and $\mathcal{E}$ its set of events. A transition system abstracting $\mathcal{L}$ is a tuple $TS_{\mathcal{L}} = (S, T) \subseteq R \times (R \times \mathcal{E} \times R)$ where*

- $S = \cup_{\sigma \in L} \cup_{\sigma' \in \text{ prefix } (\sigma)} l^{\text{state}}\ (\sigma')$
- $T = \{(l^{\text{state}}\ (\sigma'), e, l^{\text{state}}\ (\sigma' \oplus \langle e\rangle))$ *s.t.* $\exists \sigma \in \mathcal{L} : \sigma' \oplus \langle e\rangle \in prefix(\sigma)\}$

Fig 1 shows an example of a transition system in accordance with Def. III.6. It has been built on an event log $\mathcal{L}^{ex} = \{\langle a,b,c,d\rangle, \langle a,b,c,e\rangle, \langle a,b,c,f\rangle, \langle a,b,c,g\rangle, \langle a,c,d,f\rangle\}$[4], using a *sequence-based state-representation function* $l^{state}_{sq}(\langle e_1, \ldots, e_n\rangle) = \langle activity(e_1), \ldots, activity(e_n)\rangle$. Through this function, the state of a (prefix of a) trace is identified with its ordered list of activities. For the example with $\mathcal{L}^{ex}$, the set of possible states is thus $S = \{\langle a,b,c,d\rangle, \langle a,b,c,e\rangle, \langle a,b,c,f\rangle, \langle a,b,c,g\rangle, \langle a,c,d,f\rangle, \langle a,c,d\rangle, \langle a,b,c\rangle, \langle a,b\rangle, \langle a,c\rangle, \langle a\rangle\}$. Transitions systems are built by the recommender system to determine, based on the history, which activities are allowed after observing a sequence of activities. The transition system can naturally be extremely large and not intelligable, but this poses no threat because they are used internally and never shown to process' actors.

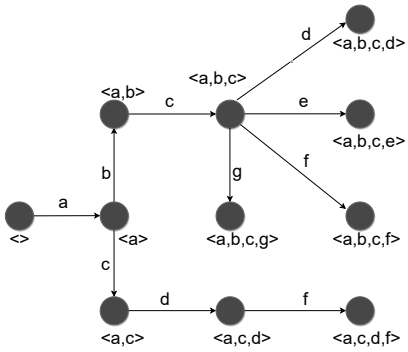

Fig. 1: Transition system for the example log

Let us assume a process instance that leaves a trail of events as per trace $\sigma_R = \langle e_1, \ldots, e_k\rangle$. The trace is running: new activity executions are still expected before completion. We want to recommend the best next activities that optimizes a KPI $\mathcal{K}$. Let us assume a transition system $TS_{\mathcal{L}} = (S_{\mathcal{L}}, T_{\mathcal{L}})$ and an oracle function $\Phi_{\mathcal{K}}$, both constructed from an event log $\mathcal{L}$. We aim to recommend what to do next for σ_R. First, we build the set of **next possible activities** $\mathcal{A}_{\sigma_R}$ that are allowed to occur after observing the events in σ_R, assuming those coincide with what observed in $\mathcal{L}$ and thus modelled by $TS_{\mathcal{L}}$: $\mathcal{A}_{\sigma_R} = \{activity(e)\ : \exists\, (l^{\text{state}}\ (\sigma'), e, l^{\text{state}}\ (\sigma' \oplus \langle e\rangle)) \in T\}$. Then for every activity in $a \in \mathcal{A}_{\sigma_R}$, we evaluate the expected KPI of $\sigma_R \oplus a$ using the oracle function (i.e. $\Phi_{\mathcal{K}}(\sigma' \oplus a)$). Here and later, $\sigma \oplus a$ indicates the trace σ extended with an event $(a, t, \mathbb{B})$ where t is the timestamp of the last event in σ, and $\mathbb{B} : \mathcal{V} \to \{\perp\}$ with $\mathbb{B}(v) = \perp$ for all the v in $\mathcal{V}$ that do not depend directly on a, and therefore cannot be inferred with certainty by just knowing a. The definition of $\mathbb{B}$ reflects the uncertainty on the values that are going to be assigned to attributes through the execution of activity a. As many other predictive methods, Catboost is able to interpret and deal with these missing values, namely attributes whose value is unknown.

This procedure associates each activity with the corresponding expected KPI value, establishing a ranking of possible next activities, if the KPI value is related to total execution time,

[3]Considering $\bigoplus$ as the concatenation of vectors e.g. $[1, 3, 'request\_created'] \bigoplus [2, True] = [1, 3, 'request\_created', 2, True]$

[4]Here, for simplicity, events are just referred to through the activity name.

Possible next activity	Expected total time
d	311h 32min
e	404h 10min
f	261h 56min
g	467h 1min

TABLE I: Ranking for a given trace $\sigma' = \langle a, b, c \rangle$. In this example, relative to the log $\mathcal{L}$ described in the transition system in Fig1, the KPI is the total time, and so we aim to minimize it ($\sqsupset_{\mathcal{K}}$ assumes the value of the $<$ operator). For this, we are going to recommend f, d and e, in this order of preference (k is assumed to be set to 3).

the ranking may be as in Tab I. From it, we can recommend the first k best activities (with k customizable), namely those associated with the best expected KPI values.[5].

We conclude observing that some activities are seldom observed after the execution of certain activities, and considered outliers. We tackle this in the construction of the transition system through the definition of a frequency threshold t: if a certain arc is traversed less than t time while replaying a trace, it is removed. Analysts can set this threshold.

C. Explanation of Prediction Models

Several prediction models, including Catboost, are black boxes, making it difficult to explain the predictions, namely to highlight the features mostly influencing the predictions and, consequently, the recommendations. Explaining predictions and recommendations is essential to build user trust in the models. The remainder briefly summarizes the basic concepts behind the framework for prediction, that has been introduced in [9]. In the remainder, we take this framework as starting point to explain recommendation. The framework leverages on the Shapley Values [21], which is a game theory approach to fairly distribute the payout among the players that have collaborated in a cooperative game. The assumption is that the features from an instance correspond to the players, and the payout is the difference between the prediction made by the predictive model and the average prediction (also called *base value*). Intuitively, given a predicted instance, the Shapley Value of a feature expresses how much the feature value contributes to vary the model prediction from the base value [22]:

Definition III.7 (Shapley Value)**.** *Let $F = \{f_1, \ldots, f_m\}$ be the set of features used by the oracle function $\Phi_{\mathcal{K}} : X_1 \times \ldots \times X_m \to \mathbb{R}$ to predict a KPI $\mathcal{K}$. The Shapley value for feature f_i which assumes value $x_i \in X_i$ is defined as:*

$$\psi_i = \sum_{F' \subseteq F \setminus \{f_i\}} \frac{|F'|!(m-|F'|-1)!}{m!} \left(val\left(F' \cup \{f_i\}\right) - val(F')\right)$$

where $val(F')$ is the so-called payout for only using the set of feature values in $F' \subset F$ in making the prediction.[6]

Note that Shapley values can only be computed when the KPI values are defined over a numerical domain. Intuitively, the formula in Definition III.7 evaluates the effect of incorporating the value $x_i \in X_i$ of the feature f_i into any possible subset of the feature values considered for prediction. In the equation, set F runs over all possible subsets of feature values, the term $val\left(F' \cup \{f_i\}\right) - val(F')$ corresponds to the marginal value of adding the feature f_i which assumes value x_i in the prediction using only the set of feature values in F, and the term $\frac{|F'|!(m-|F'|-1)!}{m!}$ corresponds to all the possible permutations with subset size $|F'|$, to weight different sets in the formula. This way, all possible subsets of attributes are considered, and the corresponding effect is used to compute the Shapley Value of x_i. The starting point for the explainable framework is the trace-to-instance encoding function $\rho : \mathcal{E}^* \to X_1 \times \ldots \times X_m$ (cf. Section III-A), and a event log $\mathcal{L}$. Let us recall that given a trace $\sigma = \langle e_1, \ldots, e_n \rangle \in \mathcal{E}^*$ and its relative encoding $\rho(\sigma) = [x_1, \ldots, x_m]$, each feature f_i has an associated value x_i. As mentioned in Section III-A, a feature f_i can be of different nature, such as a process attribute, a timestamp, or the number of executions of an activity in σ. When applied for prediction explanations, the Shapley values for a trace σ are computed over tuple $\rho(\sigma) = [x_1, \ldots, x_m]$, thus resulting in a tuple of Shapley values $\Psi = [\psi_1, \ldots, \psi_m]$, with ψ_i being the Shapley value of feature f_i that assumes the value x_i. In accordance with the Shapley values theory, the explanation of ψ_i is as follows: since feature $f_i = x_i$, the KPI prediction deviates ψ_i units from the average KPI value of the event-log traces.

Definition III.8 (Shap Function)**.** *Let σ be a (prefix of a) trace of an event log and $\Phi_{\mathcal{K}} : X_1 \times \ldots \times X_m \to \mathbb{R}$ the oracle function trained as defined in III-A. We can define the SHAP function $\Psi_{\Phi_{\mathcal{K}}} : \mathcal{E}^* \to \mathbb{R}^m$ as the vector of Shapley associated to prediction $\Phi_{\mathcal{K}}(\sigma)$.*

In the remainder, we use $\Phi_{\mathcal{K}}(\sigma)[f_i]$ to refer to the Shapley value of feature f_i which assumes a certain value $x_i \in X_i$, which is the i-th entry of the vector $\rho(\sigma) \in X_1 \times \ldots \times X_m$.

IV. A Framework for explaining recommendations

Our recommendations are given choosing the activity that has the best KPI predicted between all the possible next activities reported in the transition system. We now aim to use the Shapley values theory to provide an explanation of the reason why we suggest that activity. The proposed framework leverages on comparing the difference between the Shapley values of the features before and after the recommendation. This allows the user to receive explanations to understand how the contribution of each variable would change following or not the recommendation that we are providing. Given an event log $\mathcal{L}$, a (prefix of a) running trace $\sigma' \in \mathcal{L}$, and one of their next-activity recommended a_{rec} as in III-B, we first evaluate the vector of its associated Shapley values $\Psi_{\mathcal{K}}(\sigma')$, using the SHAP function as described in III.8. We then compute the vector $\Delta(\sigma', a_{rec})$ of the element-wise difference between $\Psi_{\mathcal{K}}(\sigma')$ and $\Psi_{\mathcal{K}}(\sigma' \oplus a_{rec})$, namely between the Shapley values before and after executing a_{rec}:

$$\Delta(\sigma', a_{rec}) = \Psi_{\mathcal{K}}(\sigma') - \Psi_{\mathcal{K}}(\sigma' \oplus a_{rec}) \tag{1}$$

[5]See https://catboost.ai/en/docs/concepts/algorithm-missing-values-processing

[6]Value $val(F)$ is the prediction for feature values in set F that are marginalized over features that are not included in set F. See Sect. 9.5.3.1 in [22] for further details

Reminding that $A_{\sigma'}$ is the set of possible next activities and $\Phi_{\mathcal{K}} : X_1, \ldots, X_m \rightarrow \mathbb{R}$ the oracle function. Let us assume, without loss of generality, that the $\sqsupset_{\mathcal{K}}$ operator is equal to $<$, i.e. we aim to decrease the KPI, a similar discussion could be carried out if $\sqsupset_{\mathcal{K}}$ is equal to $>$. We have two possible scenarios :

1) It is possible to improve the KPI performing one of the activities in $A_{\sigma'}$. So $\exists a \in A_{\sigma'}\ s.t.\ \Phi_{\mathcal{K}}(\sigma') > \Phi_{\mathcal{K}}(\sigma' \oplus a)$, and a_{rec} is therefore the activity that provides the largest KPI's improvement.
2) It is not possible to improve the KPI performing one of the activities in $A_{\sigma'}$, namely $\nexists a \in A_{\sigma'}\ s.t.\ \Phi_{\mathcal{K}}(\sigma') > \Phi_{\mathcal{K}}(\sigma' \oplus a)$. In this case, the provided recommendation a_{rec} is the activity that worsens the KPI the least.

Explaining recommendation is especially relevant for the first scenario, namely when the recommended activity is predicted to improve (decrease, in the example) the KPI values. In this case, we take the dimensions of vector $\Delta(\sigma', a_{rec})$ associated with the top-k larger values, namely the Shapley values that decrease the most after executing a_{rec}. Let us assume feature f_i to be one of the features for which the Shapley values increase the most. The associated explanation can be interpreted as follows: "*The execution of recommendation a_{rec} reduces the influence of feature $f_i = x_i$ of a quantity equal to* $\Delta(\sigma', a_{rec})[f_i]$". As an example, let us consider that the KPI is a total time of a process execution, with lower values to be better. If $\Delta(\sigma', Send\_Letter)[Customer\_Type = Gold] = 200$, the performance of activity $Send\_Letter$ after σ', it is expected to reduce the influence of $Customer\_Type = Gold$ on the total time of 200 time units (e.g., hours).

Each recommendation should be associated with at least one explanation, namely at least one feature f_i for which $\Delta(\sigma', a_{rec})[f_i] > 0$. Theorem IV.2 below guarantees that it is always the case. The proof requires one intermediate lemma:

Lemma IV.1 (Disequation's properties)**.** *Given $a, b \in \mathbb{R}^m$, if $\sum_{i=1}^{m} a_i > \sum_{i=1}^{m} b_i$ there exists at least a $j \in \{1, \ldots, m\}$, such that $a_j > b_j$*

Proof. Suppose by contradiction that

$$\forall i \in \{1, \ldots, m\}\, a_i \leq b_i$$

Applying then the summation for all $i \in \{1, \ldots, m\}\, a_i \leq b_i$, we get the hypothesis falsified, and then the thesis. □

The theorem of the presence of at least one explanation can now be formulated and proven:

Theorem IV.2. *Let $\sigma_1, \sigma_2 \in \mathcal{E}^*$ two different traces, $\Phi_{\mathcal{K}} : X_1 \times \ldots \times X_m \rightarrow \mathbb{R}$ be the oracle function and $\Psi_{\Phi_{\mathcal{K}}} : \mathcal{E}^* \rightarrow \mathbb{R}^m$ be the associated SHAP function. If $\Phi_{\mathcal{K}}(\sigma_1) > \Phi_{\mathcal{K}}(\sigma_2)$, then exists at least an $i \in \{1, \ldots, m\}$ such that $\Psi_{\Phi_{\mathcal{K}}}(\sigma_1)[f_i] > \Psi_{\Phi_{\mathcal{K}}}(\sigma_2)[f_i]$*

Proof. Let $\overline{\Phi_{\mathcal{K}}(\mathcal{X}))}$ the average value (a.k.a. *base value*) for the prediction of elements belonging to the domain set $\mathcal{X}$. From [22] we know that the sum of the elements of the Shapley Values vector is equal to the difference between its relative predicted value and the base value, analitically

$$\sum_{i=1}^{m} \Psi_{\Phi_{\mathcal{K}}}(\sigma)[f_i] + \overline{\Phi_{\mathcal{K}}(\mathcal{X}))} = \Phi_{\mathcal{K}}(\sigma)\ \forall \sigma \in \mathcal{E}^* \tag{2}$$

Applying this formula to both σ_1 and σ_2 we obtain the system

$$\begin{cases} \sum_{i=1}^{m} \Psi_{\Phi_{\mathcal{K}}}(\sigma_1)[f_i] + \overline{\Phi_{\mathcal{K}}(\mathcal{X}))} = \Phi_{\mathcal{K}}(\sigma_1) \\ \sum_{i=1}^{m} \Psi_{\Phi_{\mathcal{K}}}(\sigma_2)[f_i] + \overline{\Phi_{\mathcal{K}}(\mathcal{X}))} = \Phi_{\mathcal{K}}(\sigma_2) \end{cases}. \tag{3}$$

By subtracting the second from the first equation we obtain

$$\Phi_{\mathcal{K}}(\sigma_1) - \Phi_{\mathcal{K}}(\sigma_2) = \sum_{i=1}^{m} \Psi_{\mathcal{K}}(\sigma_1)[f_i] - \sum_{i=1}^{m} \Psi_{\mathcal{K}}(\sigma_2)[f_i]$$

Since $\Phi_{\mathcal{K}}(\sigma_1) > \Phi_{\mathcal{K}}(\sigma_2)$ by hypothesis, we get that $\Phi_{\mathcal{K}}(\sigma_1) - \Phi_{\mathcal{K}}(\sigma_2) > 0$, and so

$$\sum_{i=1}^{m} \Psi_{\mathcal{K}}(\sigma_1)[f_i] > \sum_{i=1}^{m} \Psi_{\mathcal{K}}(\sigma_2)[f_i]$$

Then, applying the lemma IV.1, we obtain that exists at least a feature f_j for which $\Psi_{\Phi_{\mathcal{K}}}(\sigma_1)[f_j] > \Psi_{\Phi_{\mathcal{K}}}(\sigma_2)[f_j]$ □

This ensures that at least a value exists for which we can provide the change in Shapley value as an explanation, and it is the one at the $j-th$ entry. It is also important to note that since the theorem also holds when inverting both the inequalities, we can apply the same procedure if $\sqsupset_{\mathcal{K}}$ operator is equal to $>$ (i.e. we aim to maximize the KPI) and providing as explanations the features with lower $\Delta(\sigma', a_{rec})$ associated.

The discussion above corresponds to the scenario where there is a recommendation that predicts to improve on the KPI. In the alternative scenario where there is no recommendation for KPI improvement, we do not have formal guarantee that an explanation exists. The framework's implementation builds on the work by Galanti et al. [9] to train the oracle function $\Phi_{\mathcal{K}}$, leveraging also the libraries Pandas and NumPy. The explanations are provided using the library provided by Shap, in its framework dedicated to CatBoost[7]. The code is available on GitHub at `https://github.com/Pado123/explainable-prescriptive-analytics`.

V. Introduction to Use Cases and Recommendation Results

Although the paper's focus is on the explanations of the recommendations, it is clear that explanations only make sense for sensible recommendations that can positively affect the process' KPIs. Section V-A discusses how the event log has been split in a training log, which is used to build the Catboost model and the transition system, and in test log, used for evaluation. This section concludes by illustrating the test-log usage to assess the quality of recommendations. Section V-B presents the considered case studies, while Section V-C reports on the evaluation of the recommendations' quality.

[7]https://catboost.ai/en/docs/concepts/shap-values

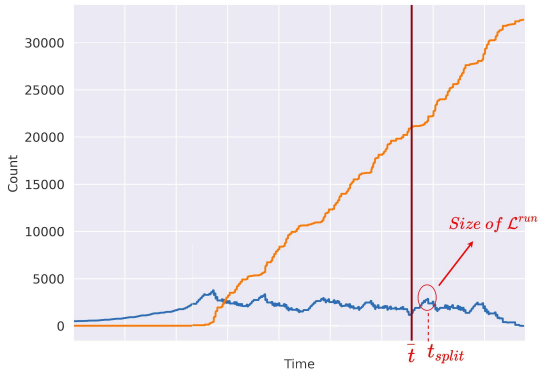

Fig. 2: The orange and blue lines represent the number of completed and active traces, respectively. The vertical red bar is the timestamp $\bar{t}$ when the 65% of traces have completed.

A. Definition of the Training and Test logs and Evaluation of Recommendation

The starting point for an evaluation is an event log $\mathcal{L}$. From this, we first extract the training log $\mathcal{L}^{comp}$, used to train the recommender system as a whole, namely the oracle function and the transition system. Then, we create the test log $\mathcal{L}^{run}$ of the running cases on which the system is evaluated.

To extract the training log $\mathcal{L}^{comp} \subset \mathcal{L}$ and test log $\mathcal{L}^{run} \subseteq \mathcal{L} \setminus \mathcal{L}^{comp}$, we compute the earliest time $\bar{t}$ in which 65% of the traces of $\mathcal{L}$ are completed, see, e.g. the example in Fig 2. Then, we compute the time $t_{split} \geq \bar{t}$ with the largest number of running cases. This allows us to define $\mathcal{L}^{comp}$ as the set of traces of $\mathcal{L}$ completed at time t_{split}, and $\mathcal{L}^{run}$ as the set of traces of $\mathcal{L}$ running at time t_{split}.

The traces of test log $\mathcal{L}^{run}$ are truncated to a set $\mathcal{L}^{trunc}$ that is obtained from $\mathcal{L}^{run}$ by removing every event with a timestamp larger than t_{split}: $\mathcal{L}^{trunc}$ only contains the events occurred before time t_{split}. This procedure tries to mimic the reality at time t_{split}.

The accuracy of recommending the activity a for the running trace $\sigma' \in \mathcal{L}^{trunc}$ is evaluated as the average KPI of traces similar to $\sigma' \oplus a$, belonging to $\mathcal{L}^{run}$:

$$acc(a, \sigma') = avg_{\sigma \in \mathcal{L}^{sim}_{a,\sigma'}} \mathcal{K}(\sigma) \tag{4}$$

where

$$\mathcal{L}^{sim}_{a,\sigma'} = \{\sigma \in \mathcal{L}^{run} : \exists \sigma^p \in prefix(\sigma) \wedge l^{state}_{sq}(\sigma^p) = l^{state}_{sq}(\sigma' \oplus a)\}$$

And l^{state}_{sq} is the *sequence state-representation function* (cf. Section III-B).

B. Use cases

The validity of our approach was assess using two different event logs with their associated use case. The first is so called **Bank Account Closure** (BAC), a log referring to a process of an Italian Bank Institution that deals with the closures of bank accounts. From the bank's information system, we extracted an event log with 212,721 events containing 15 different activities and 32,429 completed traces, divided in 22,013 for train and 10,286 for test. Each trace is associated with an attribute, *Closure_Type*, which encodes the type of procedure that is carried out for the specific account holder, and the *Closure_Reason*, namely the reason triggering the closure's request. For this case, we aim to avoid the occurrence of the activity *Pending Liquidation Request*. The KPI value can be 1 or 0 if the activity occurs or not, while the oracle function $\Phi_{\mathcal{K}}$ is represented by the probability of the activity occurring (i.e. $\Phi_{\mathcal{K}}(\sigma) \in [0, 1]$). Note that one wants to reduce the activity-occurrence probability: the activity *Pending Liquidation Request* is considered as rework, thus being an inefficiency in time and costs.

Use Case	Accuracy when best recommendations are		Δ
	not followed	followed	
Occurrence of activity *PLR*	0.96	0.34	64.4%
Total Time for the VINST process	484h 46min	441h 14min	9.4%

TABLE II: The accuracy when executions are left to be carried without influencing the outcome with recommendation, versus the scenario when the best recommendation is always followed. The KPI value of the first is given as probability of occurrence, while the second is the actual time value. Activity *PLR* is a shortcut name for *Pending Liquidation Request*

The second log was used by the BPI challenge in 2013[8]. It is provided by Volvo Belgium and contains events from an incident and problem management system called **VINST**. We extracted 7,456 completed traces and 64,975 events. It contains 13 different activities. In selecting traces from the log to obtain training and test, we generate a training log of 5,103 traces and a test log of 2,236 traces. For this case we aim to decreasing the total execution time, the KPI value and the oracle function are respectively the total time and its expected value.

C. Evaluation results

Table II reports on the results that we obtained using the recommender system. In the first column, the use cases are reported. The second and third column reports on the accuracy (cf. Equation 4) when recommendations are or are not followed, respectively. Given a trace σ', the recommendation accuracy $acc(a, \sigma')$ is computed for the activity a that is the top recommended activity. Conversely, the non-recommendation accuracy $acc(a', \sigma')$ is computed for the activity a' that follows in the trace σ of which σ' is prefix. Last columns highlights the percentage improvement between and the second and the third column, computed as $\Delta = (1 - followed/not\_followed) * 100\%$ with $followed$ and $not\_followed$ being the accuracy when recommendations are followed or not followed. Results show a sensitive improvement for both of cases, especially for the minimization of the occurrence of the rework activity *Pending Liquidation Request*. Further investigation is out of scope of this paper: we only aimed to illustrate the validity of the recommendations on which explanations are computed. The latter is the novel contribution of the paper.

[8]https://www.win.tue.nl/bpi/doku.php?id=2013:challenge

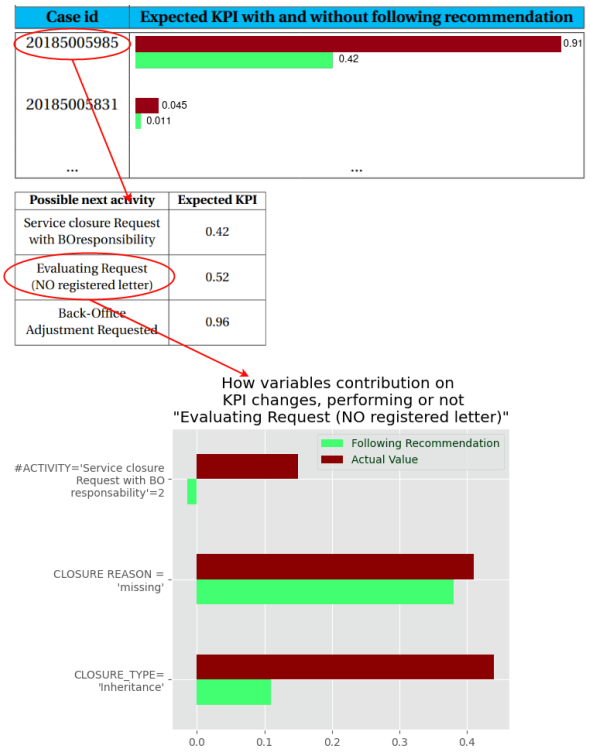

Fig. 3: Example of output for the PAR system with recommendation for the KPI value related to the (undesired) occurrence of activity *Pending Liquidation Request*. Explanation label `#ACTIVITY=actname=actnumber` means that the activity named "act name" happened "act number" times

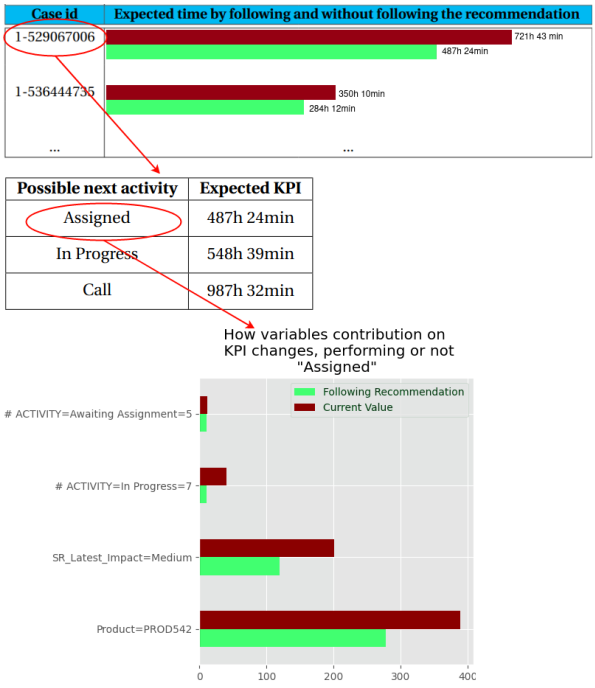

Fig. 4: Example of the procedure followed for providing recommendations and explanations for the KPI value "Total time" for VINST process. The x-axis scale of the bar chart is in hours.

VI. Provision and Display of Explanations

Figure 3 shows how explanations are expected for our PAR system, for the use case of the process of Bank Account Closure, namely aiming to minimize the occurrence activity *Pending Liquidation Request*. The outcome is a list that, for each running case, illustrates the expected KPI value when both no or the best recommendation is followed. Figure 3 illustrates an excerpt of this list with two running cases: for the first case, 20185005985, the probability of the occurrence of activity *Pending Liquidation Request* is 91% (namely, the KPI value is 0.91), while the probability drops to 42% if the best recommended activity would be performed. For case 20185005985, it is possible to significantly reduce the probability. When the process actor decides to intervene and focuses on the specific case, a list of potential recommendations is offered, each with the expected KPI value (see table in the middle of Figure 3). Let us assume that the process actor opts to perform the second best recommended activity, circled in red in figure, namely that reducing the probability of the occurrence of *Pending Liquidation Request* to 52%. In fact, the actor might have reasons based on aspects not modelled in the process to not choose the recommendation that minimizes the probability (e.g., the activity requires accesses to systems that are currently in maintenance). When the recommendation is selected, the explanations are provided in form of bars (see bottom of Figure 3):

- Currently the fact that activity *Service closure Request with BO responsibility* has been executed twice contributes to an increase of probabilities of the undesired activity to occur of ca. 15% (value 0.15), the execution of *Evaluating Request* (the recommendation) will nullify this contribution. In fact, the double occurrence of activity becomes a positive contribution to a reduction of the activity occurrence.
- The reason for bank account closure is unknown contributes to an increase of ca. 41% of the probability of the undesired activity to occur. The recommendation activity is able to partly mitigate this contribution, which lowers to ca. 38%.
- Similarly to the point above, the recommendation activity allows mitigating the negative probability contribution to the occurrence of the undesired activity from an original value of ca. 45% to ca. 11%.

This form of explanation is in line with the frameworks for Explainable AI of black-box models (cf. Section II). In accordance with these frameworks, *we also aim to pinpoint the individual contribution of each of the different process' features to the predicted KPI value, but with the notable difference that we want to just focus on those features whose influence on the KPI values can be largely impacted by performing a recommendation activity*. In fact, we are not interested in returning explanations for those features that have a large impact on the KPI values but that the recommendations cannot influence. Note that, for the specific example in question, the pairwise sum of the difference of the Shapley's values

of these three explanations is around accounting for ca. 48% of the positive contribution to reducing the KPI value (recall that the overall expected KPI value improvement with the recommendation is of ca. 52%). These are indeed the first three more relevant explanations. Others possibly exist, but with a smaller impact on reducing the probability for the activity to occur.

Figure 4 shows a similar output for the VINST process (cf. Section V-B) when one aims to minimize the KPI of the total execution time. The recommendation of activity *Assigned* for case *1-529067006* allows the total execution time to be reduced from 712 hours and 43 minutes to 487 hours and 24 minutes. Four explanations are present, and the most significant (i.e. enabling a larger decrease of the total time) is `Product=PROD542`: performing the activity *Assigned* allows one to mitigate the negative impact on KPI related to the product being *PROD542* from almost 400 hours to ca. 280 hours.

VII. Conclusions and Further directions

Existing research on PAR systems has focused on providing recommendations that can bring executions back on track. However, recent literature has overlooked the problem of ensuring that process actors feel engaged, trust these recommendations, and consequently follow them. Engagement and trust pass through combining recommendations with understandable explanations for process actors.

This paper is the first attempt to report on a framework to explain process' recommendation. As discussed, the explanations are given in terms of values of process characteristics that process actors would understand. In particular, the explanations focus on those characteristics that affect the process' outcome, and whose negative influence can be mitigated by the recommendations.

Explaining the given recommendations is beneficial to gain a better insight into how the performance of certain activities in a given process' state can influence the relevant KPI. This provides further insights for process actors into how to improve the process executions.

The paper showcases how explanations would look in two use cases related to real-life processes. As future work, the utmost priority will be given to assess the framework with real process actors, thus determining whether the shape in which explanations are given is insightful and increases the trust of the suggested recommendation. The intuition seems to support this, but the definitive answer can only be given through an extensive user study. In parallel, we aim to more objectively verify the validity of the explanations against the quality criteria for Explainable AI introduced in literature (see, e.g. [23]).

Acknowledgement. The PhD. scholarship of Mr. Padella is partly funded by IBM Italy, and by the BMCS Doctoral Program, University of Padua. This research is also supported by the Department of Mathematics, University of Padua, through the BIRD project "Data-driven Business Process Improvement" (code `BIRD215924/21`).

References

[1] M. Dees, M. de Leoni, W. M. P. van der Aalst, and H. A. Reijers, "What if process predictions are not followed by good recommendations?" in *Proceedings of the Industry Forum at BPM 2019*, ser. CEUR Workshop Proceedings, vol. 2428. CEUR-WS.org, 2019.

[2] M. de Leoni, M. Dees, and L. Reulink, "Design and evaluation of a process-aware recommender system based on prescriptive analytics," in *2020 2nd International Conference on Process Mining (ICPM)*, 2020.

[3] S. Weinzierl, S. Dunzer, S. Zilker, and M. Matzner, "Prescriptive business process monitoring for recommending next best actions," in *Business Process Management Forum*, 2020.

[4] A. E. Márquez-Chamorro, M. Resinas, and A. Ruiz-Cortés, "Predictive monitoring of business processes: A survey," *IEEE Transaction on Services Computing*, vol. 11, no. 6, 2018.

[5] L. Lin, L. Wen, and J. Wang, *MM-Pred: A Deep Predictive Model for Multi-attribute Event Sequence*. SIAM, 05 2019.

[6] M. Camargo, M. Dumas, and O. González-Rojas, *Learning Accurate LSTM Models of Business Processes*. Springer, 07 2019.

[7] A. Metzger, T. Kley, and A. Palm, "Triggering proactive business process adaptations via online reinforcement learning," in *Business Process Management*. Cham: Springer International Publishing, 2020.

[8] Z. D. Bozorgi, I. Teinemaa, M. Dumas, M. L. Rosa, and A. Polyvyanyy, "Prescriptive process monitoring for cost-aware cycle time reduction," in *2021 3rd International Conference on Process Mining (ICPM)*, 2021.

[9] R. Galanti, B. Coma-Puig, M. de Leoni, J. Carmona, and N. Navarin, "Explainable predictive process monitoring," in *Proceedings of the 2nd International Conference on Process Mining (ICPM 2020)*. IEEE, 2020.

[10] R. Sindhgatta, C. Moreira, C. Ouyang, and A. Barros, "Exploring interpretable predictive models for business processes," in *Proceedings of BPM 2020*. Springer, 2020.

[11] T.-H. Huang, A. Metzger, and K. Pohl, "Counterfactual explanations for predictive business process monitoring," in *Proceedings of EMCIS 2021*. Springer, 2022.

[12] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *The 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[13] J. Sarthak and B. C. Wallace, "Attention is not explanation," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2019.

[14] G. Park and M. Song, "Prediction-based resource allocation using LSTM and minimum cost and maximum flow algorithm," in *Proceedings of the International Conference on Process Mining, ICPM 2019, Aachen, Germany, 2019*. IEEE, 2019.

[15] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, "Predictive business process monitoring with LSTM neural networks," in *Proceedings of the 29th International Conference on Advanced Information Systems Engineering (CAiSE 2017)*, vol. 10253. Springer, 2017.

[16] N. Navarin, B. Vincenzi, M. Polato, and A. Sperduti, "LSTM networks for data-aware remaining time prediction of business process instances," in *Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI 2017)*, 2017.

[17] M. Polato, A. Sperduti, A. Burattin, and M. de Leoni, "Time and activity sequence prediction of business process instances," *Computing*, vol. 100, no. 9, Sep 2018.

[18] I. Verenich, M. Dumas, M. La Rosa, F. Maggi, and I. Teinemaa, "Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, 07 2019.

[19] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: gradient boosting with categorical features support," in *Proceedings of the Workshop on ML Systems at NIPS 2017*, 2017.

[20] W. M. P. van der Aalst, *Process Mining: Data Science in Action*. Berlin: Springer-Verlag, 2011.

[21] L. S. Shapley, *A value for n-person games*. RAND Corporation, 1953, vol. 2, no. 28.

[22] C. Molnar, *Interpretable Machine Learning*, 2022, Available online at https://christophm.github.io/interpretable-ml-book/.

[23] H. Löfström, K. Hammar, and U. Johansson, "A meta survey of quality evaluation criteria in explanation methods," in *Proceedings of CAiSE Forum 2022*, ser. LNBIP, vol. 452. Springer, 2022.