Explainable Predictive Decision Mining for Operational Support

Gyunam Park[1(✉)], Aaron Küsters[2], Mara Tews[2], Cameron Pitsch[2], Jonathan Schneider[2], and Wil M. P. van der Aalst[1]

[1] Process and Data Science Group (PADS), RWTH Aachen University, Aachen, Germany
{gnpark,wvdaalst}@pads.rwth-aachen.de
[2] RWTH Aachen University, Aachen, Germany
{aaron.kuesters,mara.tews,cameron.pitsch, lennart.schneider}@rwth-aachen.de

Abstract. Several decision points exist in business processes (e.g., whether a purchase order needs a manager's approval or not), and different decisions are made for different process instances based on their characteristics (e.g., a purchase order higher than €500 needs a manager approval). Decision mining in process mining aims to describe/predict the routing of a process instance at a decision point of the process. By predicting the decision, one can take proactive actions to improve the process. For instance, when a bottleneck is developing in one of the possible decisions, one can predict the decision and bypass the bottleneck. However, despite its huge potential for such operational support, existing techniques for decision mining have focused largely on describing decisions but not on predicting them, deploying decision trees to produce logical expressions to explain the decision. In this work, we aim to enhance the predictive capability of decision mining to enable proactive operational support by deploying more advanced machine learning algorithms. Our proposed approach provides explanations of the predicted decisions using SHAP values to support the elicitation of proactive actions. We have implemented a Web application to support the proposed approach and evaluated the approach using the implementation.

Keywords: Process mining · Decision mining · Machine learning · Operational support · Proactive action

1 Introduction

A process model represents the control-flow of business processes, explaining the routing of process instances. It often contains decision points, e.g., XOR-split gateway in BPMN. The routing in such decision points depends on the data attribute of the process instance. For instance, in a loan application process, the assessment of a loan application depends on the amount of the loan, e.g., if the amount is higher than €5,000, it requires *advanced assessment* and, otherwise, *simple assessment*.

Decision mining in process mining aims to discover a decision model that represents the routing in a decision point of a business process [8]. The discovered decision model can be used for 1) describing how decisions have been made and 2) predicting how decisions will be made for future process instances. While the focus has been on the former in the literature, the latter is essential to enable proactive actions to actually improve business processes [13]. Imagine we have a bottleneck in *advanced assessment* due to, e.g., the lack of resources. By predicting the decision of a future loan application, we can take proactive action (e.g., suggesting to lower the loan amount to conduct *simple assessment*), thus facilitating the process.

To enable such operational support, a decision model needs to be both 1) predictive (i.e., the model needs to provide reliable predictions on undesired/decisions) and 2) descriptive (i.e., domain experts should be able to interpret how the decision is made to elicit a proactive action). Figure 1 demonstrates these requirements. Figure 1(a) shows a decision point in a loan application process, and there is a bottleneck in *advanced assessment*. Our goal is to accurately predict that a loan application with the amount of €5,500 and interest of 1.5% needs *advanced assessment*, which is undesired due to the bottleneck, and recommend actions to avoid the bottleneck. Figure 1(b) shows four different scenarios. First, if we predict a desired decision (i.e., predicting the simple assessment), no action is required since the simple assessment has no operational issues. Second, if we predict an undesired prediction incorrectly (e.g., incorrectly predicting the advanced assessment), we recommend an inadequate action. Third, if we predict the undesired decision correctly but no explanations are provided, no action can be elicited due to the lack of explanations. Finally, if we predict the undesired decision, and the corresponding explanations are provided (e.g., the amount/interest of the loan has a positive/negative effect on the probability of conducting the advanced assessment, respectively), we can come up with relevant actions (e.g., lowering the amount or increasing the interest rate).

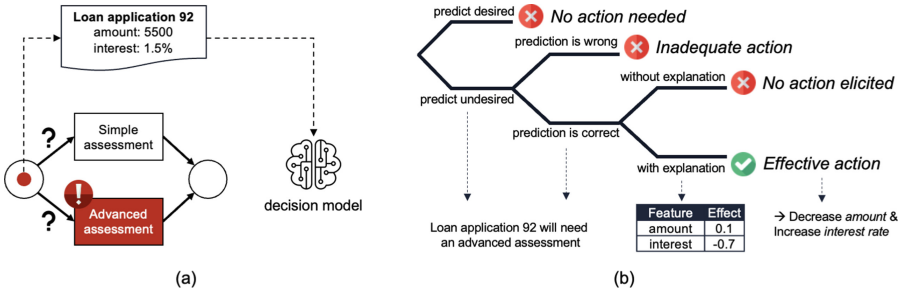


Fig. 1. (a) Decision point in a process model. (b) Different scenarios showing that decision mining needs to be predictive and descriptive to enable operational support.

Existing work has focused on the descriptive capability of decision models by deploying highly interpretable machine learning algorithms such as decision

trees [8,11,15]. However, it leads to limited predictive capability due to the limitation of decision trees, such as overfitting and instability (i.e., adding a new data point results in regeneration of the overall tree) [16]. In this work, we aim to enhance the predictive capabilities of decision mining, while providing explanations of predicted decisions. To this end, we estimate the decision model by using machine learning algorithms such as support vector machines, random forests, and neural networks. Next, we produce explanations of the prediction of the decision model by using SHAP values.

We have implemented the approach as a standalone web application. Using the implementation, we have evaluated the accuracy of predicted decisions using real-life event logs. Moreover, we have evaluated the reliability of explanations of predicted decisions by conducting controlled experiments using simulation models.

This paper is structured as follows. First, we discuss related work on decision mining and explainability in Sect. 2. Next, we introduce process models and event logs in Sect. 3. In Sect. 4, we provide our proposed approach. In Sect. 5, we explain the implementation of a web application based on the approach. Sect. 6 evaluates the approach based on the implementation using simulated and real-life event logs. We conclude this paper in Sect. 7.

2 Related Work

Several approaches have been proposed to learn decision models from event logs. Rozinat et al. [15] suggest a technique based on Petri nets. It discovers a Petri net from an event log, identifies decision points, and employs classification techniques to determine decision rules. De Leoni et al. [8] extend [15] by dealing with invisible transitions of a Petri net and non-conforming process instances using *alignments*. These methods assume that decision-making is deterministic and all factors affecting decisions exist in event logs. To handle non-determinism and incomplete information, Mannhardt et al. [11] propose a technique to discover overlapping decision rules. In [2], a framework is presented to derive decision models using Decision Model and Notation (DMN) and BPMN. All existing approaches deploy decision trees due to their interpretability. To the best of our knowledge, no advanced machine learning algorithms have been deployed to enhance the predictive capabilities of decision models along with explanations.

Although advanced machine learning approaches provide more accurate predictions compared to conventional white-box approaches, they lack explainability due to their black-box nature. Recently, various approaches have been proposed to explain such black-box models. Gilpin et al. [4] provide a systematic literature survey to provide an overview of explanation approaches. The explanation approaches are categorized into *global* and *local* methods. First, global explanation approaches aim to describe the average behavior of a machine learning model by analyzing the whole data. Such approaches include Partial Dependence Plot (PDP) [6], Accumulated Local Effects (ALE) Plot [1], and global surrogate models [3]. Next, local explanation approaches aim to explain individual predictions

by individually examining the instances. Such approaches include Individual Conditional Expectation (ICE) [5], Local Surrogate (LIME) [14], and Shapley Additive Explanations (SHAP) [10]. In this work, we use SHAP to explain the predictions produced by decision models due to its solid theoretical foundation in game theory and the availability of global interpretations by combining local interpretations [10].

3 Preliminaries

Given a set X, we denote the set of all multi-sets over X with $\mathcal{B}(X)$. $f\restriction_X$ is the function projected on X: $dom(f\restriction_X) = dom(f) \cap X$ and $f\restriction_X(x) = f(x)$ for $x \in dom(f\restriction_X)$.

3.1 Process Models

Decision mining techniques are independent of the formalism representing process models, e.g., BPMN, YWAL, and UML-activity diagrams. In this work, we use Petri nets as the formalism to model the process.

First, a Petri net is a directed bipartite graph of places and transitions. A labeled Petri net is a Petri net with the transitions labeled.

Definition 1 (Labeled Petri Net). *Let $\mathbb{U}_{act}$ be the universe of activity names. A labeled Petri net is a tuple $N{=}(P, T, F, l)$ with P the set of places, T the set of transitions, $P \cap T{=}\emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ the flow relation, and $l \in T \not\to \mathbb{U}_{act}$ a labeling function.*

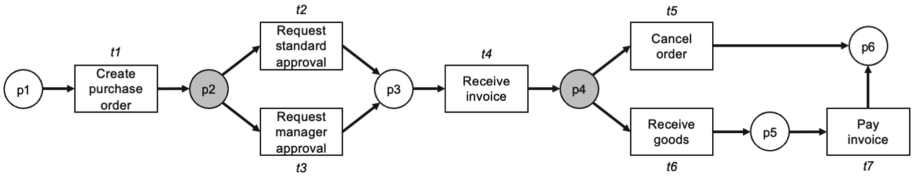


Fig. 2. An example of Petri nets highlighted with decision points.

Figure 2 shows a Petri net, $N_1 = (P_1, T_1, F_1, l_1)$, where $P_1 = \{p1, \ldots, p6\}$, $T_1 = \{t1, \ldots, t7\}$, $F_1 = \{(p1, t1), (t1, p2), \ldots\}$, $l_1(t1) =$ *Create purchase order*, $l_1(t2) =$ *Request standard approval*, etc.

The state of a Petri net is defined by its marking. A marking $M_N \in \mathcal{B}(P)$ is a multiset of places. For instance, $M_{N_1} = [p1]$ represents a marking with a token in $p1$. A transition $tr \in T$ is *enabled* in marking M_N if its input places contain at least one token. The enabled transition may *fire* by removing one token from each of the input places and producing one token in each of the output places. For instance, $t1$ is *enabled* in M_{N_1} and *fired* by leading to $M'_{N_1} = [p2]$.

Definition 2 (Decision Points). *Let $N=(P,T,F,l)$ be a labeled Petri net. For $p \in P$, $p\bullet = \{t \in T \mid (p,t) \in F\}$ denotes it outgoing transitions. $p \in P$ is a decision point if $|p\bullet| > 1$.*

For instance, $p2$ is a decision point in N_1 since $p2\bullet = \{t2, t3\}$ and $|p2\bullet| > 1$.

Table 1. An example of event logs.

Case id	Activity	Timestamp	Resource	Total-price	Vendor
PO92	Create purchase order	09:00 05.Oct.2022	Adams	1,000	Apple
PO92	Request standard order	11:00 07.Oct.2022	Pedro	1,000	Apple
PO93	Create purchase order	13:00 07.Oct.2022	Peter	1,500	Samsung
...	...	...	...	...	...

3.2 Event Logs

Definition 3 (Event Logs). *Let $\mathbb{U}_{event}$ be the universe of events, $\mathbb{U}_{attr}$ the universe of attribute names ($\{case, act, time, res\} \subseteq \mathbb{U}_{attr}$), and $\mathbb{U}_{val}$ the universe of attribute values. An event log is a tuple $L = (E, \pi)$ with $E \subseteq \mathbb{U}_{event}$ as the set of events and $\pi \in E \rightarrow (\mathbb{U}_{attr} \nrightarrow \mathbb{U}_{val})$ as the value assignments of the events.*

Table 1 shows a part of an event log $L_1 = (E_1, \pi_1)$. $e_1 \in E_1$ represents the event in the first row, i.e., $\pi_1(e_1)(case) = PO92$, $\pi_1(e_1)(act) = Create\ Purchase\ Order$, $\pi_1(e_1)(time) =$ 09:00 05.Oct.2022, $\pi_1(e_1)(res) = Adams$, $\pi_1(e_1)(total\text{-}price) = 1{,}000$, and $\pi_1(e_1)(vendor) = Apple$.

4 Explainable Predictive Decision Mining

In this section, we introduce an approach to explainable predictive decision mining. As shown in Fig. 3, the proposed approach consists of two phases: *offline* and *online* phases. The former aims to derive decision models of decision points, while the latter aims at predicting decisions for running process instances along with explanations. In the offline phase, we compute *situation tables* based on historical event logs and estimate decision models using the *situation tables*. In the online phase, we predict decisions for ongoing process instances and explain the decision.

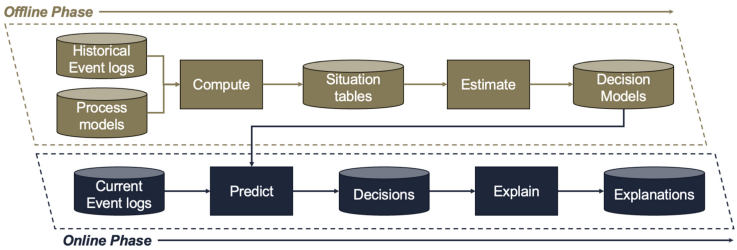


Fig. 3. An overview of the proposed approach.

4.1 Offline Phase

First, we compute situation tables from event logs. Each record in a situation table consists of features (e.g., total price of an order) and a decision in a decision point (e.g., $t2$ at decision point $p2$ in Fig. 2), describing how the decision has been historically made (e.g., at decision point $p2$ in Fig. 2, standard approval (i.e., $t2$) was performed when the total price of an order is €1,000).

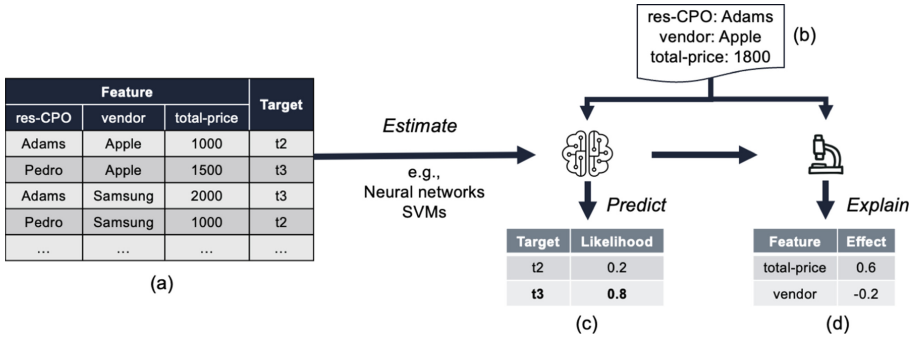


Fig. 4. An example of the proposed approach.

Definition 4 (Situation Table). *Let $\mathbb{U}_{feature}$ be the universe of feature names and $\mathbb{U}_{fmap} = \mathbb{U}_{feature} \nrightarrow \mathbb{U}_{val}$ the universe of feature mappings. Let $N{=}(P, T, F, l)$ be a labeled Petri net and $p \in P$ a decision point. $sit_p \in \mathbb{U}_L \rightarrow \mathcal{B}(\mathbb{U}_{fmap} \times p\bullet)$ maps event logs to situation tables (i.e., multi-sets of feature mappings and decisions). $S_p{=}\{sit_p(L) \mid L \in \mathbb{U}_L\}$ denotes the set of all possible situation tables of p.*

The table in Fig. 4(a) represents a situation table of $p2$ in Fig. 2 derived from the event log depicted in Table 1. For instance, the first row in Fig. 4(a) describes that *request standard approval* ($t2$) was executed when human resource *Adams* performed *create purchase order* (i.e., *res-CPO*) for the order of €1,000 (i.e., *total-price*) with *Apple* (i.e., *vendor*). Formally, $s_1 = (fmap_1, t2) \in sit_{p2}(L1)$

where $fmap_1 \in \mathbb{U}_{fmap}$ such that $fmap_1 = \{(\textit{res-CPO}, \textit{Adams}), (\textit{vendor}, \textit{Apple}), (\textit{total-price}, 1{,}000)\}$. Note that, s_1 corresponds to event e_2 in Table 1 and $fmap_1$ is derived from all historical events of $PO92$.

A decision model provides the likelihood of each transition in a decision point based on a given feature, e.g., when the total price of an order (i.e., feature) is €1,800, standard approval will be performed with the likelihood of 0.2 and manager approval with the likelihood of 0.8.

Definition 5 (Decision Model). *Let $N{=}(P, T, F, l)$ be a labeled Petri net and $p \in P$ a decision point. Let $dmap_p \in p\bullet \rightarrow [0, 1]$ be a decision mapping that maps decisions to likelihoods such that the sum of all likelihoods adds up to 1, i.e., $\Sigma_{p' \in p\bullet} dmap_p(p') = 1$. D_p denotes the set of all possible decision mappings. $DM_p \in \mathbb{U}_{fmap} \rightarrow D_p$ is the set of all possible decision models of p that map feature mappings to decision mappings.*

We estimate decision models based on situation tables.

Definition 6 (Estimating Decision Models). *Let $N{=}(P, T, F, l)$ be a labeled Petri net and $p \in P$ a decision point. $estimate_p \in S_p \rightarrow DM_p$ is a function estimating a decision model from a situation table.*

The estimation function can be built using many machine learning algorithms such as neural networks, support vector machines, random forests, etc.

4.2 Online Phase

Using the decision model derived from the offline phase, we predict the decision of a running process instance and explain the prediction. Using the feature of a running process instance depicted in Fig. 4(b), a decision model may produce the prediction shown in Fig. 4(c), leading to the final decision of *request manager approval* that has the highest likelihood. Next, we compute an explanation for the decision (i.e., the effect of each feature on the prediction) as shown in Fig. 4(d), e.g., *total-price* has a positive effect of 0.6 while *vendor* has a negative effect of 0.2. In other words, *total-price* increases the likelihood of predicting the decision of *request manager approval* by the magnitude of 0.6 and *vendor* decreases it by the magnitude of 0.2, respectively.

In this work, we use SHAP values [10] to provide explanations of decisions. SHAP values are based on *Shapley* values. The concept of *Shapley* values comes from *game theory*. It defines two elements: a game and some players. In the context of predictions, the game is to reproduce the outcome of the model, and the players are the features used to learn the model. Intuitively, Shapley values quantify the amount that each player contributes to the game, and SHAP values quantify the contribution that each feature brings to the prediction made by the model.

Definition 7 (Explaining Decisions). *Let $fmap \in \mathbb{U}_{fmap}$ be a feature mapping and $F = \{f_1, f_2, \ldots, f_i, \ldots\} = dom(fmap)$ denote the domain of $fmap$.*

Let $N{=}(P,T,F,l)$ be a labeled Petri net, $p \in P$ a decision point, and dm_p a decision model. Let $t \in p\bullet$ be a target transition. The SHAP value of feature f_i for predicting t is defined as:

$$\psi^t_{f_i} = \sum_{F' \subseteq F \setminus \{f_i\}} \frac{|F'|!(|F| - |F'| - 1)!}{|F|!} (dm_p(fmap{\restriction}_{F' \cup \{f_i\}})(t) - dm_p(fmap{\restriction}_{F'})(t))$$

For $fmap$, $exp_{dm_p,t}(fmap) = \{(f_1, \psi^t_{f_1}), (f_2, \psi^t_{f_2}), \dots\}$ is the explanation of predicting t using dm_p.

As shown in Fig. 4(d), for feature mapping $fmap'$ described in Fig. 4(b), the explanation of predicting $t3$ (i.e., request manager approval) using decision model dm'_{p2} is $exp_{dm'_{p2},t3}(fmap') = \{(\textit{total-price}, 0.6), (\textit{vendor}, -0.2)\}$. In other words, *total-price* has a positive effect with the magnitude of 0.6 on the decision of $t3$ and *vendor* has a negative effect with the magnitude of 0.2.

Moreover, we can provide a global explanation of a decision model by aggregating SHAP values of multiple running instances. For instance, by aggregating all SHAP values of *total-price* for predicting $t3$, e.g., with the mean absolute value, we can compute the global effect of *total price* to the prediction.

5 Implementation

We have implemented a Web application to support the explainable decision mining with a dedicated user interface. Source code and user manuals are available at https://github.com/aarkue/eXdpn. The application comprises three functional components as follows.

Discovering Process Models. This component supports the discovery of process models based on inductive miner [7]. The input is event data of the standard XES. The discovered accepting Petri net is visualized along with its decision points.

Decision Mining. This component supports the computation of situation tables from event logs and the estimation of decision models from the computed situation table. First, it computes situation tables with the following three types of features:

- *Case features*: Case features are on a case-level and used for predicting all decisions related to that case.
- *Event features*: Event features are specific to an event and used for predicting decisions after the occurrence of the event.
- *Performance features*: Performance features are derived from the log. It includes *elapsed time of a case* (i.e., time duration since the case started) and *time since last event* (i.e., time duration since the previous event occurred).

Next, the estimation of decision models uses the following machine learning algorithms: *Random Forests*, *XGBoost*, *Support Vector Machines (SVMs)*, and *Neural Networks*.

Visualizing Decisions and Explanations. This component visualizes the F1 score of different machine learning algorithms and suggests the best technique based on the score. Moreover, it visualizes the explanation of the decision both at local and global levels. Local explanations are visualized with *force plot* (cf. Fig. 5(a)), *decision plot* (cf. Fig. 5(b)), and *beeswarm plot* (cf. Fig. 5(c)), whereas global explanations are visualized with *bar plot* (cf. Fig. 5(d)), *force plot* (cf. Fig. 5(e)), and *beeswarm plot* (cf. Fig. 5(f)). We refer readers to [9] for the details of different plots.

6 Evaluation

In this section, we evaluate the approach by conducting experiments using the implementation. Specifically, we are interested in answering the following research questions.

- RQ1: Does the advanced machine learning algorithm efficiently predict the decisions?
- RQ2: Does the approach provides reliable explanations for the predictions?

6.1 RQ1: Prediction Accuracy

In order to answer RQ1, we conduct experiments using real-life event logs: Business Process Intelligence Challenge (BPIC) 2012[1] and BPIC 2019[2]. For each event log, we first discover a process model and determine decision points. Then we estimate different decision models for each decision point and compare the performance of the decision models using 5-fold cross-validation. To measure the performance of the decision model, we use F1 scores. Each model is instantiated with suitable, event-log-specific parameters, which have largely been obtained from a parameter grid search on each decision point as well as manual test runs. For decision tree algorithms, we apply pruning steps to avoid too many splits that result in decision trees harder to interpret in practice due to their complexity.

Table 2 shows the F1 score of different machine learning algorithms in different real-life event logs[3]. The top two scores for each decision point are highlighted with bold fonts. *XGBoost* shows good scores for all decision points except $p14$ in BPIC 2012 and $p4$ in BPIC 2019. The scores for *Support Vector Machine* belong to the top two scores for most of the decisions except $p4$ and $p6$ in BPIC 2012

[1] https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f.
[2] https://doi.org/10.4121/uuid:d06aff4b-79f0-45e6-8ec8-e19730c248f1.
[3] The experimental results are reproducible in https://github.com/aarkue/eXdpn/tree/main/quantitative_analysis along with the corresponding process model.

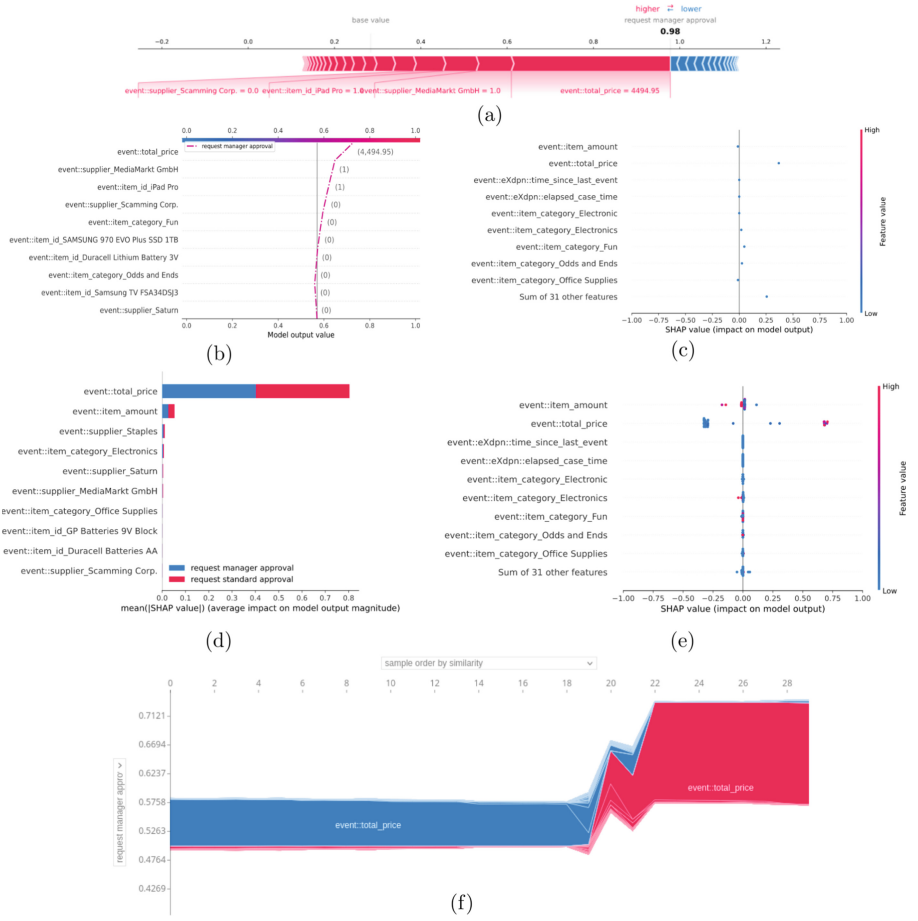


Fig. 5. Local explanations: (a) Force plot, (b) Decision plot, and (c) Beeswarm plot explain how the model arrived at the decision of a running instance (i.e., *request manager approval* with the likelihood of 0.98). For instance, (a) visualizes the positive (red-colored) and negative (blue-colored) features with increasing magnitudes. **Global explanations**: (d) Bar plot, (e) Beeswarm plot, and (f) Force plot explain how the model arrived at the decision of all running instances (both on *request standard approval* and *request manager approval*). For instance, (d) visualizes the mean absolute SHAP value for each feature on predicting *request standard approval* (blue-colored bar) and *request manager approval* (red-colored bar), showing that total-price has the highest impact on both predictions. (Color figure online)

and $p8$ and $p11$ in BPIC 2019, whereas the ones for *Neural Network* belong to the top two scores in $p4$, $p6$ and $p14$ in BPIC 2012 and $p4$ and $p8$ in BPIC 2019. *Decision Tree* shows the top two scores only for $p16$ in BPIC 2012 and $p11$ in BPIC 2019.

Table 2. F1 scores of applying different machine learning algorithms in different decision points. The bold font shows the top two results in each decision point.

Event logs		BPI challenge 2012 (only offers)						BPI challenge 2019 (filtered)			
Decision point		p4	p6	p12	p14	p16	p19	p3	p4	p8	p11
Algorithms	Decision tree	0.6888	0.7545	0.7955	0.9633	**0.9612**	0.9263	0.9555	0.9948	0.8135	**1.0000**
	XGBoost	**0.7189**	**0.7897**	**0.8004**	0.9697	**0.9612**	**0.9407**	**0.9632**	0.9948	**0.8293**	**1.0000**
	Support vector machine	0.7151	0.7799	**0.8023**	**0.9701**	**0.9612**	**0.9414**	**0.9649**	**0.9950**	0.8096	0.9997
	Neural network	**0.725**	**0.8048**	0.7955	**0.9698**	0.9607	0.9317	0.9583	**0.9981**	**0.8191**	0.9949

6.2 RQ2: Reliability of Explanations

To answer RQ2, we design a simulation model to simulate a Purchase-To-Pay (P2P) process using CPN tools [12]. The simulation model allows us to fully define the decision logic of decision points. Based on the decision logic, we qualitatively evaluate if the generated explanation is reliable. Figure 6 shows the Petri net discovered using inductive miner [7] from an event log generated by the simulation model, with highlighted decision points. Decision point (c) describes the decision of whether the purchase order is held at customs or not. The decision logic in the simulation model is as follows: *If 1) a purchase order originates from outside the EU and 2) the base price per item is higher than €50, the order is held at customs.*

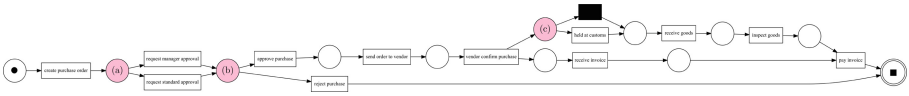
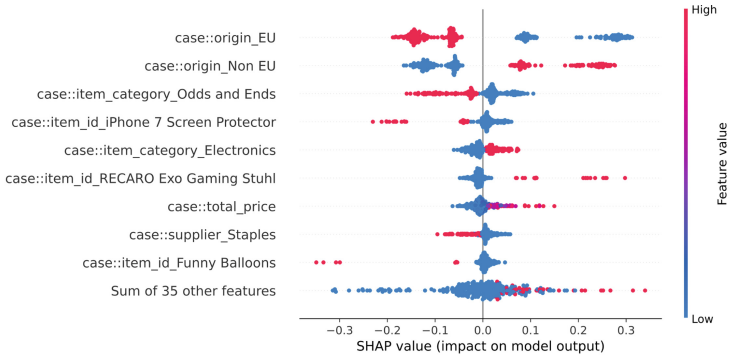
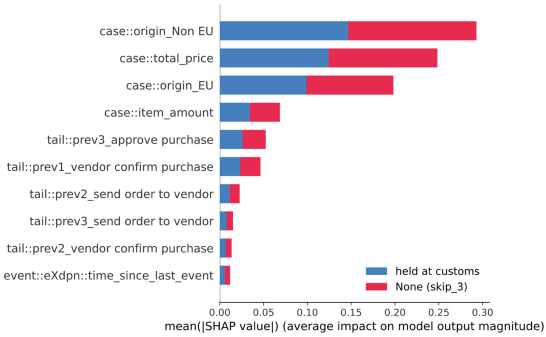


Fig. 6. Petri net discovered from the simulated P2P event logs.

The beeswarm plot in Fig. 7(a) explains the decision at decision point (c). The Non-EU origin (high value of `origin_Non EU`) has a strong positive impact on the probability of being held at customs according to the decision model. Moreover, the existence of items in category `Odds and Ends`, which have low base prices, has a negative impact on the probability, whereas the existence of items in category `Electronics`, which have high base prices, has a positive impact on the probability. When the individual product names, categories, and vendors are excluded (see Fig. 7b), the four most impactful features that remain are exactly the ones used in the logic of the underlying simulation model: The EU or Non-EU origin, the total price and the number of items in the order. Overall the decision logic as interpretable through the plots corresponds to the underlying logic applied in the simulation model, showing that the explanation obtained is reliable.



(a) Beeswarm plot visualizing the impact of high or low feature values on the model probability of being held at customs. The Non-EU origin (high value of `origin_Non EU`) has a strong positive impact on the probability of being held at customs.



(b) Bar plot visualizing the mean absolute SHAP value of each selected feature, per output class

Fig. 7. Qualitative analysis showing the explanation plots of decision point (c) using a neural network model.

7 Conclusions

In this paper, we proposed an approach to explainable predictive decision mining. In the offline phase of the approach, we derive decision models for different decision points. In the online phase, we predict decisions for running process instances with explanations. We have implemented the approach as a web application and evaluated the prediction accuracy using real-life event logs and the reliability of explanations with a simulated business process.

This paper has several limitations. First, the explanation generated by the proposed approach is less expressive than the logical expression generated by traditional decision mining techniques. Also, we abstract from the definition of features that can be used to construct the situation tables, focusing on explaining several possible features in the implementation. In future work, we plan to extend

the approach with a taxonomy of features to be used for the comprehensive construction of situation tables. Moreover, we plan to connect the explainable predictive insights to actual actions to improve the process.

Acknowledgment. The authors would like to thank the Alexander von Humboldt (AvH) Stiftung for funding this research.

References

1. Apley, D.W., Zhu, J.: Visualizing the effects of predictor variables in black box supervised learning models. CoRR arXiv:abs/1612.08468 (2016)
2. Bazhenova, E., Weske, M.: Deriving decision models from process models by enhanced decision mining. In: Reichert, M., Reijers, H.A. (eds.) BPM 2015. LNBIP, vol. 256, pp. 444–457. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_36
3. Frosst, N., Hinton, G.E.: Distilling a neural network into a soft decision tree. CoRR arXiv:1711.09784 (2017)
4. Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M.A., Kagal, L.: Explaining explanations: An approach to evaluating interpretability of machine learning. CoRR arXiv:1806.00069 (2018)
5. Goldstein, A., Kapelner, A., Bleich, J., Pitkin, E.: Peeking inside the black box: visualizing statistical learning with plots of individual conditional expectation. J. Comput. Graph. Stat. **24**(1), 44–65 (2015)
6. Greenwell, B.M., Boehmke, B.C., McCarthy, A.J.: A simple and effective model-based variable importance measure. CoRR arXiv:1805.04755 (2018)
7. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs - a constructive approach. In: Colom, J.-M., Desel, J. (eds.) PETRI NETS 2013. LNCS, vol. 7927, pp. 311–329. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38697-8_17
8. de Leoni, M., van der Aalst, W.M.P.: Data-aware process mining: discovering decisions in processes using alignments. In: Shin, S.Y., Maldonado, J.C. (eds.) 28th Annual ACM Symposium on Applied Computing, pp. 1454–1461. ACM (2013)
9. Lundberg, S.: Shap library documentation. https://shap.readthedocs.io/en/latest/index.html#. Accessed 05 Aug 2022
10. Lundberg, S.M., Lee, S.: A unified approach to interpreting model predictions. In: Guyon, I., et al. (eds.) NeurIPS 2017, pp. 4765–4774 (2017)
11. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Decision mining revisited - discovering overlapping rules. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) CAiSE 2016. LNCS, vol. 9694, pp. 377–392. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39696-5_23
12. Park, G., van der Aalst, W.M.P.: Towards reliable business process simulation: a framework to integrate ERP systems. In: Augusto, A., Gill, A., Nurcan, S., Reinhartz-Berger, I., Schmidt, R., Zdravkovic, J. (eds.) BPMDS/EMMSAD -2021. LNBIP, vol. 421, pp. 112–127. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-79186-5_8
13. Park, G., van der Aalst, W.M.P.: Action-oriented process mining: bridging the gap between insights and actions. Prog. Artif. Intell. (2022). https://doi.org/10.1007/s13748-022-00281-7

14. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should I trust you?": explaining the predictions of any classifier. In: Krishnapuram, B., Shah, M., Smola, A.J., Aggarwal, C.C., Shen, D., Rastogi, R. (eds.) 22nd SIGKDD, pp. 1135–1144. ACM (2016)
15. Rozinat, A., van der Aalst, W.M.P.: Decision mining in ProM. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 420–425. Springer, Heidelberg (2006). https://doi.org/10.1007/11841760_33
16. Safavian, S.R., Landgrebe, D.A.: A survey of decision tree classifier methodology. IEEE Trans. Syst. Man Cybern. **21**(3), 660–674 (1991)