UNIVERSITÀ DEGLI STUDI DI PADOVA

UNIVERSITÀ DEGLI STUDI DI PADOVA
CORSO DI DOTTORATO IN BRAIN, MIND AND COMPUTER SCIENCE
COMPUTER SCIENCE FOR SOCIETAL CHALLENGES

EXPLAINABLE PREDICTIVE AND PRESCRIPTIVE PROCESS ANALYTICS OF CUSTOMIZABLE BUSINESS KPIs

PhD Candidate
Riccardo Galanti

Supervisor
Prof. Massimiliano de Leoni

Co-Supervisor
Prof. Luciano Gamberini

XXXV SERIES

Abstract

Recent years have witnessed a growing adoption of machine learning techniques for business improvement across various fields. Among other emerging applications, organizations are exploiting opportunities to improve the performance of their business processes by using predictive models for runtime monitoring.

Predictive analytics leverages machine learning and data analytics techniques to predict the future outcome of a process based on historical data. Therefore, the goal of predictive analytics is to identify future trends, and discover potential issues and anomalies in the process before they occur, allowing organizations to take proactive measures to prevent them from happening, optimizing the overall performance of the process.

Prescriptive analytics systems go beyond purely predictive ones, by not only generating predictions but also advising the user if and how to intervene in a running process in order to improve the outcome of a process, which can be defined in various ways depending on the business goals; this can involve measuring process-specific Key Performance Indicators (KPIs), such as costs, execution times, or customer satisfaction, and using this data to make informed decisions about how to optimize the process.

This Ph.D. thesis research work has focused on predictive and prescriptive analytics, with particular emphasis on providing predictions and recommendations that are explainable and comprehensible to process actors. We first propose a prescriptive framework that, given a running process that will be eventually completed, associates an expected KPI value to each of the possible process continuations, based on a predictive model that is trained on historical process executions. Finally, it recommends the continuations that are associated with the best expected KPI values.

However, while the priority remains on giving accurate predictions and recommendations, the process actors need to be provided with an explanation of the reasons why a given process execution is predicted to behave in a certain way and they need to be convinced that the recommended actions are the most suitable ones to maximize the KPI of interest; otherwise, users would not trust and follow the provided predictions and recommendations, and the predictive technology would not be adopted. To address this gap, we proposed an explainable framework based on the Shapley Values game theory approach, which can be adapted to explain any predictive model and any generic KPI, numerical or nominal. On the one hand, we equipped our predictive-monitoring framework with explainable capabilities, in order to highlight every aspect that significantly affects the predicted process outcome. On the other hand, also the prescriptive framework was equipped with explainable capabilities; here, differently from predictions, we focused on highlighting the principal factors that had a negative impact on the KPI, but whose influence could be also largely mitigated by following the proposed recommendations. In order to show the validity of the explanations provided, our explanation strategy was applied to several publicly available datasets; after analyzing the data, the evidence in the explanations demonstrated that the developed predictive framework leveraged attributes that were found to be relevant from a domain viewpoint.

Afterwards, in order to demonstrate the practical application of the research conducted in this Ph.D. thesis, we integrated our explainable predictive framework as a module of a commercial software, the IBM Process Mining Suite. This enabled us to provide process stakeholders with a ready-to-use module that provisions online operational support for their processes. Moreover, we conducted a user evaluation to assess the efficiency and effectiveness of the proposed explainable predictive framework; the evaluation confirmed that the predictions were actually explained in a form that is effective and intelligible for process analysts and that the process stakeholders were satisfied with the explainable predictive process framework.

Finally, a new paradigm, called object-centric, is rapidly gaining popularity in industry; here, the object-centric process is the result of the interactions of many different objects, each with its own life-cycle. These life-cycles are sub-processes that work in concert to carry out process instances, periodically synchronizing and exchanging messages. The existing literature on predictive analytics cannot be directly applied to predict the outcome of object-centric processes, because it relies on a single flow of executions. To address this gap, this Ph.D. thesis proposes an approach to enable predictive analytics in object-centric processes; furthermore, in order to improve the predictive accuracy, we also considered including attributes that synthesize additional information related to the interaction of the different sub-processes, which is a crucial aspect in object-centric processes. By conducting several experiments on real-life datasets, we observed that an increased predictive accuracy was often associated with the adoption of the attributes representing the sub-processes interaction; this aspect was also validated by our proposed explainable framework, which was leveraged to confirm that the designed included attributes were often among the most important ones that were leveraged by our predictive framework.

Contents

Chapter 1

Introduction

Process-aware Recommender systems (hereafter shortened as PAR systems) are a specific class of Information Systems that aim to monitor and predict how process instances are going to evolve, and to recommend the corrective actions to recover the instances with higher risk to not achieve the desired levels of performance (e.g., costs, time deadlines, customer satisfaction). Conceptually, a PAR system is constituted by three main blocks: monitoring, predictive analytics and prescriptive analytics.

Process monitoring refers to the practice of tracking and observing a process to ensure that it is performing as expected and to detect any deviations or issues that may occur. The goal of process monitoring is to maintain control over a process, thus ensuring that the desired process outcome is achieved. The outcome of a process can be defined in various ways depending on the business goals; this can involve measuring process-specific Key Performance Indicators (KPIs), such as costs, execution times, or customer satisfaction, and using this data to make informed decisions about how to optimize the process.

Predictive process monitoring aims to overcome the limitations of traditional monitoring practices by using data produced during process execution to continuously monitor processes performance [84]. In particular, predictive process monitoring leverages machine learning (ML) and data analytics techniques to predict the future outcome of a process based on historical data. Therefore, the goal of predictive process monitoring is to identify future trends, and discover potential issues and anomalies in the process before they occur, allowing organizations to take proactive measures to prevent them from happening, optimizing the overall performance of the process.

Prescriptive analytics goes a step further and recommends the best actions that should be taken in order to optimize the outcome of a process. Overall, PAR systems focus on improving the outcome of processes, hereafter modelled as KPIs. This Ph.D. thesis research work has focused on the predictive and prescriptive blocks, with particular emphasis on providing predictions and recommendations that are explainable and comprehensible to process actors.

In the last years, a lot of research has been on the first two blocks (commonly referred as Predictive Business Process Monitoring techniques) and several approaches have been proposed (see e.g. [63, 106]). Conversely, the prescriptive analytics block has been overlooked, assuming that the users, after being alerted of a potential failure, are able to find the proper corrective actions. However, it has been demonstrated by some on-the-field experts [17] that the assumption of selecting an effective corrective action is not always met in reality. This is due to the fact that, without support, process actors make decisions on the basis of their subjective perception of the process, rather than relying on objective data. In particular, the authors illustrated that, even if the developed predictive-analytics module was able to predict the process outcome rather well, the subsequent interventions selected by the users did not bring significant improvements on the process outcome. This demonstrated that accurate predictions are crucial, but their effect is nullified if it is not matched by effective recommendations, which must be based on objective evidence from historical process data. Therefore, the first goal of this PhD thesis is summarized by the following research question:

Research Question 01 *How can we build a prescriptive business process analytics block that effectively maximizes a given reference KPI?*

Another field that has been overlooked in the last years is explainable artificial intelligence (AI), assuming that a good level of prediction's accuracy is sufficient for the process' stakeholders to trust the recommender system (as well as the prediction system). However, while the priority remains on giving accurate predictions and recommendations, the process actors need to be provided with an explanation of the reasons why a given process execution is predicted to behave in a certain way; moreover, they need to be convinced that the recommended actions are the most suitable ones to maximize the KPI of interest. Previous studies [21, 72] have shown that a necessary condition to build trust is to explain the reason of the provided predictions and recommendations; otherwise, users would not trust the predictive-monitoring technology, they would not follow the suggestions given, and the PAR system would not be adopted. This leads us to the second goal of this PhD thesis:

Research Question 02 *How can users trust the predictions and the recommenda-*

tions provided by a PAR system?

In literature, predictive monitoring has traditionally focused on processes that are composed by a single flow of execution. However, the experience in industry has shown that a new paradigm, called object-centric, is rapidly gaining popularity, since it is observed in many application scenarios. Here, the object-centric process is the result of the interactions of many different objects, each with its own life-cycle. These life-cycles are sub-processes that work in concert to carry out process instances, periodically synchronizing and exchanging messages. The existing literature on predictive monitoring cannot be directly applied to predict the outcome of object-centric processes, because it relies on a single flow of executions. This Ph.D. thesis also focuses on providing a framework to analyze object-centric processes, where the process is carried out through the interaction of different sub-processes. The inclusion of the interaction information in the prediction models is certainly beneficial and increases the accuracy of the predictive model, leading us to the third goal of this Ph.D. thesis:

Research Question 03 *How can one perform predictive analytics of object-centric processes and exploit the information of the complex sub-processes interaction to increase the prediction accuracy?*

1.1 Contribution

In this thesis, we make several contributions to the fields of predictive and prescriptive process monitoring as described below.

First, in order to address *Research Question 01* and build a prescriptive business process analytics framework that effectively maximizes a given reference KPI, the predictive analytics block needs to be build as a first step. Therefore, an assessment of the state of the art of predictive algorithms has been performed, and an empirical evaluation on the accuracy of our predictive framework has been reported. Moreover, the developed predictive monitoring framework has been extended in order to be able to predict a generic KPI of interest. Afterwards, we addressed *Research Question 01* by developing a prescriptive framework; in particular, given a running process that will be eventually completed, we first find all the possible next actions that can be performed, based on the previously observed completed process executions. Then, we leverage the aforementioned predictive framework to associate an expected KPI value to each of the possible process continuations; the recommended actions will be those associated with the best expected KPI values.

In order to address *Research Question 02* and increase the trust in our PAR system, we equipped our previously developed predictive monitoring framework with explanation capabilities, which enabled us to explain any generic KPI, numerical or nominal, by leveraging on current state of the art of Explainable Artificial Intelligence (XAI). In particular, by leveraging post-hoc explainability approaches to explain the outcome of a black box model, we returned for each running process the factors that influenced the prediction the most, with the corresponding magnitude and indication whether the influence was towards increasing or decreasing the predicted KPI's value. The explanations have been validated on several real-life datasets, and the intelligibility of the proposed explanation strategy was assessed through a user study with several real and potential process analysts from academia and industry. To this end, the implementation of the framework is not limited to be an academic proof-of-concept prototype, but has been incorporated into a commercial tool, the IBM Process Mining, in order to provide further strength to the validation of the explanation's understandability by process analysts.

To complete addressing *Research Question 02*, also the prescriptive framework was equipped with explainable capabilities. When explaining the predictions, we focused on every aspect that significantly affects the expected process outcome; here, conversely, we proposed an explanation strategy where we focused on highlighting the principal factors that had a negative impact on the KPI, but whose influence could be also largely mitigated by following the proposed recommendations.

Finally, to address *Research Question 03*, this Ph.D. thesis has proposed a technique to transform object-centric event logs in order to enable predictive analytics in object-centric processes. Afterwards, an empirical evaluation of the accuracy of four different predictive algorithms on several real-life datasets has been reported. In our evaluation, in order to improve the predictive accuracy, we also considered including attributes that synthesize additional information related to the interaction of the different sub-processes, which is a crucial aspect in object-centric processes. By conducting several experiments on real-life datasets, we observed that an increased predictive accuracy was often associated with the adoption of the attributes representing the sub-processes interaction; this aspect was also validated by our proposed explainable framework, which was leveraged to confirm that the designed included attributes were often among the most important ones that were leveraged by our predictive framework to increase the prediction accuracy.

The above contributions have been previously documented in publications that are referenced at the end of the thesis (see Appendix B).

1.2 Thesis outline

Chapter 2 provides definitions, principles and basic concepts from process mining and machine learning areas that will be referenced throughout the thesis. Additionally, we provide the relevant background on explainable AI and we illustrate how machine learning models can be leveraged for predictive process analytics.

In Chapter 3, an assessment of the state of the art of predictive algorithms is performed, and an empirical evaluation of two of the most promising predictive algorithms is reported.

In Chapter 4, the developed predictive monitoring framework is equipped with explanation capabilities, and an empirical evaluation of the provided explanations is performed. In particular, a user evaluation is assessed in order to understand if the explanations provided by the framework are intelligible to process stakeholders. Moreover, the integration of our framework in the IBM Process Mining suite is illustrated, consolidating the research contributions of this thesis.

In Chapter 5, the predictive framework is extended in order to be able to predict the outcome also for object-centric processes and an empirical evaluation of the accuracy of four different predictive algorithms is reported. Moreover, the information about the object interactions is incorporated into the predictive model, illustrating the benefits of their use on the prediction quality. Explainable AI techniques are also leveraged to further confirm the importance of object-interaction.

Chapter 6 reports on the developed prescriptive framework, which not only can recommend the corrective actions to recover the instances with higher risk not to achieve the expected outcome, but can also accompany recommendations with sensible explanations based on the process behavior and the context in which the process is carried on.

Finally, Chapter 7 concludes this thesis by providing a summary of our contributions and discussing possible avenues for future work.

Chapter 2

Preliminaries

In this chapter we introduce the concepts that will be used in this thesis. In particular, Section 2.1 introduces process mining and illustrates the concepts of traditional and object-centric event logs. Section 2.2 gives an overview about relevant concepts from the machine learning field and describes the predictive models that will be used in this thesis. Section 2.3 provides a brief overview of works on explainability of machine learning models. Finally, Section 2.4 concludes the chapter by illustrating how machine learning models can be leveraged for predictive process analytics.

2.1 Process Mining

Business Process Management (BPM) is a systematic approach to improving the performance and efficiency of a company's workflows, processes, and operations [23]. It involves the identification, design, implementation, monitoring, and improvement of business processes to optimize productivity, reduce costs, increase customer satisfaction, and achieve organizational goals.

BPM aims to streamline business operations by mapping out workflows and identifying areas for improvement. This involves analyzing and understanding the current state of business processes, identifying inefficiencies, and developing a plan to optimize them. BPM may also involve automating certain processes using technology to reduce manual labor, errors, and costs.

In this context, a business process is viewed as a collection of inter-connected events, activities and decision points that involve a number of human actors, soft-

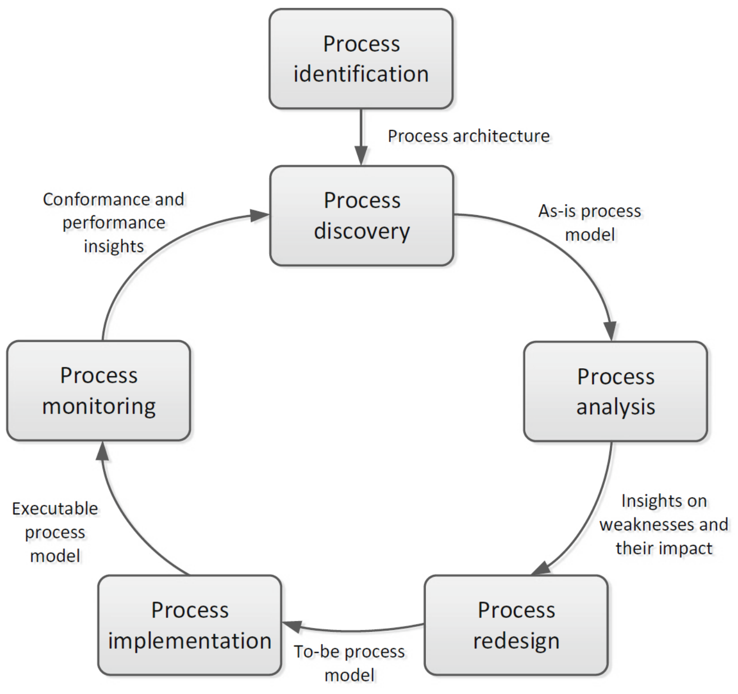


Figure 2.1: BPM lifecycle model [23]

ware systems and physical and digital objects, and that collectively lead to an outcome that adds value to the involved actors. Figure 2.1 illustrates the stages in which BPM activities can be organized:

- *Process identification*: this phase consists in identifying the processes that are relevant to the problem being addressed and critical to the relevant business goals. The outcome of this phase is a process architecture that presents an overview of the identified processes, along with their relations.
- *Process discovery*: it consists in manually or automatically constructing a representation of the current state of an organisation's business process. The current state of a business process is also defined as-is.
- *Process analysis*: it consists in examining a process in order to identify eventual issues related to the as-is process, thus discovering eventual possible improvements.
- *Process redesign*: it consists in identifying different potential changes to the

as-is process in order to tackle the previously identified issues. The result of this operation is the definition of a to-be process model.

- *Process implementation*: this phase consists in the concrete implementation of the aforementioned modifications, in order to achieve the to-be process.
- *Process monitoring*: once the process has been redesigned, this phase consists in collecting and analyzing the data related to the process execution, in order to assess the process performance with respect to its performance criteria.

Modern organizations use process-aware information systems that record information about the execution of business processes that can be extracted and preprocessed to produce event logs [109]. The availability of event logs has lead to a growing interest among organizations to improve their business processes in a data-driven manner. Process mining is a research area within BPM that is concerned with extracting useful insights from the aforementioned event logs. In particular, process mining techniques are able to support various stages of business process management tasks, such as process discovery, analysis, redesign, implementation and monitoring. In this section, we introduce the key process mining concepts. An IEEE standard for representing event logs, called XES (eXtensible Event Stream), has been introduced in [41]. The standard defines the XML format for organizing the structure of traces, events and attributes in event logs.

An event log (see example in Table 2.1) consists of a multiset of traces, i.e. sequences of events that are related to the same case (an instance of a business process). For example, a case can refer to all events related to the same purchase order. An event carries information about the execution of a given activity. The core mandatory elements of every event are the case identifier (e.g. the identifier of the purchase order), the activity name (i.e. the type of the executed event), and a timestamp indicating when the event occurred. In other words, every event represents the occurrence of an activity at a particular point in time and in the context of a given case. Additionally, an event can contain other relevant data; as an example, it can contain information about the payment, such as the order_price, and information about the resource, i.e. the process worker or the software system that performed the specific activity. These additional relevant data are referred to as event attributes. The value of these attributes can change dynamically throughout the trace or it can be static and never change throughout the lifetime of the trace;

Table 2.1: Example of a traditional event log. Each row is an event.

Event ID	Case ID	Activity	Timestamp	Customer	Resource	Product	Order_Price	Order_Quantity
e1	Case1	Order Creation	2017-07-16 15:00	Paul	System	P01	100	1
e2	Case1	Check Availability	2017-07-17 9:00	Paul	Anna	P01	100	1
e3	Case1	Prepare Order	2017-07-17 13:00	Paul	Dennis	P01	100	1
e4	Case1	Order Received	2017-07-20 17:00	Paul	Delivery Company SPA	P01	100	1
e5	Case2	Order Creation	2017-08-05 15:00	Laura	System	P02	50	3
e6	Case2	Check Availability	2017-08-06 9:00	Laura	Jack	P02	50	3
e7	Case2	Cancel Order	2017-08-10 13:00	Laura	System	P02	50	3

an example of the latter can be the type of the ordered product or the age of the customer.

Formally, an event is defined as follows:

Definition 2.1.1 (Event)**.** *Let A be the set of process' activities. Let T the set of possible timestamps and let $\mathcal{AN}$ the set of process attributes. Let $\mathcal{W}_{\mathcal{AN}}$ be a function that assigns a domain $\mathcal{W}_{\mathcal{AN}}(a)$ to each process attribute $a \in \mathcal{AN}$. Let $\overline{\mathcal{W}} = \cup_{a \in \mathcal{AN}} \mathcal{W}_{\mathcal{AN}}(a)$. An event is a tuple $(act, t, val) \in \mathcal{A} \times T \times (\mathcal{AN} \not\rightarrow \overline{\mathcal{W}})$*[1] *where act is the event activity, val is a partial function assigning values to process attributes with $val(a) \in \mathcal{W}_{\mathcal{AN}}(a)$, and t its timestamp.*

In the remainder, given an event $e = (act, t, val)$, we define $\pi_{act}(e) = act$ as the function associating an event to its activity, $\pi_{time}(e) = t$ as the function associating an event to its timestamp, and $\pi_{vmap}(e) = val$ as the function associating an event to a variable-to-value assignment function *val* such that, for each attribute $a \in AN$ in the domain of *val*, *val(a)* indicates the value assigned to a by e.

A trace is a sequence of events. Note that 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:[2]

Definition 2.1.2 (Traces, Event Logs & Prefixes)**.** *Let E be the universe of events, i.e. the set of all possible event identifiers. A trace σ is a sequence of events, i.e. $\sigma \in E^*$. A traditional event-log $\mathcal{L}$ is a multiset of traces, i.e. $\mathcal{L} \subset \mathbb{B}(E^*)$. Moreover, given a trace $\sigma = \langle e_1, \ldots, e_n \rangle$, $prefix(\sigma)$ denotes the set of all prefixes of σ, including σ: $\{\langle\rangle, \langle e_1 \rangle, \langle e_1, e_2 \rangle, \ldots, \langle e_1, \ldots, e_n \rangle\}$.*

In Process Mining, we can distinguish between two types of Event Logs: traditional and *Object-centric Event Logs*. The former type has been described in

[1]The notation $\not\rightarrow$ indicates a partial function.

[2]Given a set X, $\mathbb{B}(X)$ indicates the set of all multisets with the elements in X, and X^* indicates the universe of all sequences over elements in X (Kleene's Star).

this section and illustrated in Table 2.1; in the following, when it is evident from the context, with the term event log we refer to the traditional event log. In a traditional event log, each trace describes the life-cycle of exactly one type of process instance; as we can see in Table 2.1, each event is associated exactly to one process type, i.e. the purchase order. Conversely, in an object-centric event log, each event may refer to one or multiple process types; notice that it is possible to convert object-centric event logs into traditional event logs; more details will be provided in Section 5.3.1.

2.1.1 Object-centric Process Mining

Object-centric processes (also known as Artifact-centric processes) are recently gaining popularity in academia and industry, because their nature is observed in many application scenarios. They are implementations of a paradigm where an instance of one process is not executed in isolation but interacts with other instances of the same or other processes. In fact, the situation is more similar to choreographies where one instance of a process P_1 interacts and synchronizes with several instances of a second process P_2, and the other way around: one instance of P_2 might synchronize with multiple instances of P_1. The situation can be even more complex: instances of P_2 may in turn interact with instances of some P_3, and so on. For instance, consider a retail shop in Padua (Italy): several customers may order products manufactured in a factory in Brisbane (Australia). The factory associates many customer orders to a single manufacturer order to save money. Also, the same customer orders can include products from different manufacturers in different parts of the globe. Customer's and manufacturer's orders are managed via instances of different processes: one instance of customer-order process can be associated to several of manufacturer-order process, and the other way round: each manufacturer-order process instance may be associated to many consumer-order ones. The recent industrial experience is confirming that the assumption of a single execution flow is unfortunately often not met in practice. This led to the introduction of the paradigm of object-centric processes, which has recently been gaining more and more attention because it can model inter-organizational processes more naturally [54, 110, 111]. Any process execution materializes itself as a set of instances of the same/different processes that represent the life cycles of different objects (a.k.a. artifacts) that contribute to the process execution (e.g., the order and the delivery object). These processes for the different objects run independently and synchronize through some bridging events to exchange data needed to progress further.

Object-centric processes are carried on with the support of one or more information systems. It is possible to extract the history of past executions into a transactional data set organized in form of object-centric event logs [28]. In a typical object-centric event log, each event may refer to one or multiple objects (i.e. each event can be associated to one or multiple object identifiers). Therefore, it is not possible to group events to form traces by picking the same identifier for every event.

Example 2.1.1. *Table 2.2 shows an excerpt of an object-centric event log of an Italian utility provider company. It consists of five object types, each with its own object identifier:* `Contract`*,* `Requisition`*,* `Order`*,* `Receipt`*,* `Invoice`*. The first is* `Contract`*, which is the process concerning the stipulation of a contract with a customer, possibly followed by a* `Requisition`*, which is an optional process executed when the order needs a purchase requisition. The* `Order` *process consists of several activities representing mainly quantity, price, or date modifications of the order, eventually approved by the Head of the department. The* `Receipt` *process is then related to the receiving of the goods or the services requested, followed by the* `Invoice` *process, which includes everything related to payments. Some events are associated to a single object identifier, others have multiple (i.e., the so-called bridge events) that enable the synchronization and data exchange between objects.*

Figure 2.2 illustrates how objects are related to each other for synchronization and data exchanges. Note that relationships can be of many-to-many or many-to-one nature. Events are associated with attributes, features based on the properties of requisitions, orders (e.g., `order_price`*), receipts (e.g.,* `receipt_quantity`*), and invoices. In order to better understand how the different objects interact with each other, we represent in Figure 2.3 the excerpt of object-centric event log in Table 2.2 as a sequence diagram. In particular, each column represents a different object of our object-centric event log (e.g., the column with label RQ1 identifies a process of type Requisition with identifier RQ1). The interaction and the exchange of messages between the objects is made possible by the bridge events, which are events that are associated with multiple object identifiers. In this diagram, each bidirectional arrow represents an interaction between the objects, and it is associated with the related bridge event that enabled this communication. The box over each column represents the life-cycle of a particular object; as an example, the life-cycle of object C1 is started at the event e1 and it is concluded at event e5, after interacting with the object O1.*

Table 2.2: Example of an object-centric event log. Each row is an event, and the blank spaces represent attributes' missing values.

ID	Activity	Timestamp	Contract	Requisition	Order	Receipt	Invoice	User	Order_Price	Order_Purch_Group	Rec_Quantity
e1	Contract Line Creation	2017-07-11 9:00	c1					CO01			
e2	Purch Contract Item Material Group Changed	2017-07-14 11:00	c1					CO01			
e3	Purchase Requisition Line Created	2017-07-15 12:00	c1	rq1				A456			
e4	Contract Line Creation	2017-07-15 14:00	c2					CO01			
e5	Purchase Requisition Line Created	2017-07-15 17:00	c2	rq2				A457			
e6	Purchase Order Line Creation	2017-07-16 15:00	c1		o1			A458	100	100_L50	
e7	Contract Line Creation	2017-07-16 16:00	c3					CO01			
e8	Purchase Order Line Creation	2017-07-17 15:00		rq1	o2			A458	200	100_L51	
e9	Purchase Order Line Creation	2017-07-18 15:00		rq2	o3			A458	300	100_L52	
e10	Goods Line Registered	2017-07-22 15:00			o1	r1		A456	100	100_L50	10
e11	Invoice Receipt	2017-07-22 16:00					i1	A125			
e12	Purchase Requisition Group Changed	2017-07-22 19:00		rq1				A456			
e13	Purchase Order Line Creation	2017-07-23 9:00		rq1	o4			A458	600	100_L52	
e14	Purchase Order Line Creation	2017-07-23 12:00	c3		o5			A458	700	100_L50	
e15	Goods Line Registered	2017-07-23 15:00			o2	r2		A456	100	100_L50	10
e16	Invoice Registered	2017-07-29 11:00				r1,r2	i1	A125			10
e17	Invoice Cleared	2017-07-30 12:00					i1	A125			
e18	Goods Line Registered	2017-07-31 15:00			o4	r3		A456	600	100_L52	10
e19	Goods Line Registered	2017-08-09 15:00			o5	r4		A456	700	100_L50	10
e20	Invoice Registered	2017-08-10 11:00				r2,r3,r4	i2	A125			10
e21	Invoice Cleared	2017-08-15 14:00					i2	A125			
e22	Goods Line Registered	2017-08-16 15:00			o3	r5		A456	300	100_L52	5
e23	Purchase Requisition Supplier Changed	2017-08-16 17:00		rq2				A456			
e24	Invoice Registered	2017-08-18 11:00				r5	i3	A125			5
e25	Invoice Cleared	2017-08-20 14:00					i3	A125			

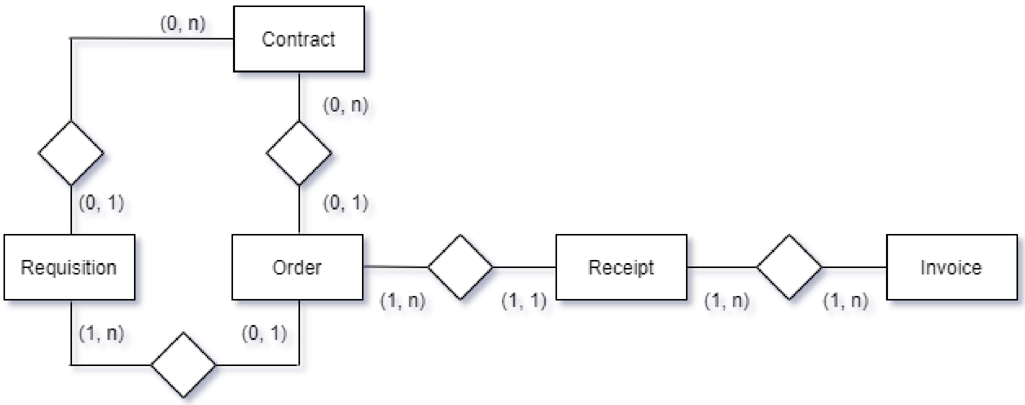


Figure 2.2: Diagram representing cardinality between the different object types in the considered object-centric event log, taken from an Italian utility provider company. For each object type, the cardinality with the subsequent or the previous object type is represented as (*min_cardinality*, *max_cardinality*)

Literature proposes the following definition of object-centric events:

Definition 2.1.3 (Object-Centric Event Log [28]). *Let T be the universe of the timestamps. An object-centric event log is a tuple $\mathcal{L} = (E, T, A, AN, AV, AT, OT, O, \pi_{typ}, \pi_{act}, \pi_{time}, \pi_{vmap}, \pi_{omap}, \pi_{otyp}, <)$ such that:*

- *E is the set of event identifiers,*
- *T is the set of timestamps,*

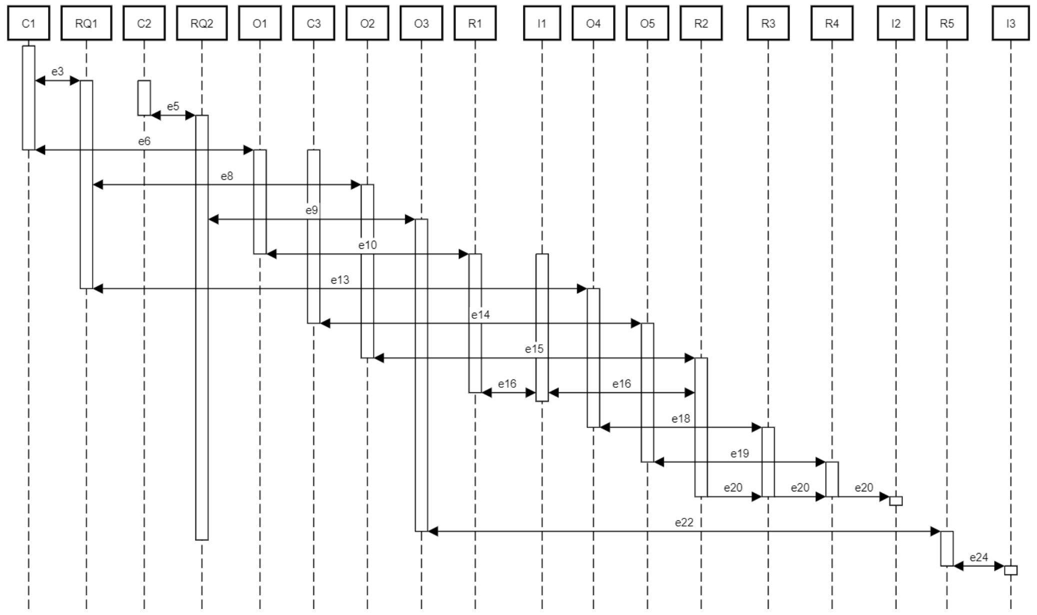


Figure 2.3: Object-centric event log represented in the form of a sequence diagram. Each column represents a different object in the object-centric event log. Each bidirectional arrow represents an interaction between the objects, and it is associated with the related bridge event that enabled the communication. The box over each column represents the life-cycle of a particular object.

- *A is the set of activity names,*
- *AN is the set of attributes names,*
- *AV is the set of attribute values (with the requirement that $AN \cap AV = \emptyset$),*
- *AT is the set of attribute types,*
- *OT is the set of object types,*
- *O is the set of object identifiers,*
- *$\pi_{typ} : AN \cup AV \to AT$ is the function associating an attribute name or value to its corresponding type,*
- *$\pi_{act} : E \to A$ is the function associating an event identifier to its activity,*
- *$\pi_{time} : E \to T$ is the function associating timestamps to event identifiers,*

- $\pi_{vmap} : E \to (AN \not\to AV)$ *is the function associating every event identifier* $e \in E$ *to a variable-to-value assignment function* val *such that, for each attribute* $a \in AN$ *in the domain of* val, $val(a)$ *indicates the value assigned to* a *by* e,
- $\pi_{omap} : E \to 2^O$ *is the function associating an event identifier to a set of related object identifiers,*
- $\pi_{otyp} : O \to OT$ *assigns precisely one object type to each object identifier,*
- $< \subseteq (E \times E)$ *is a partial order of events.*[3]

2.2 Machine Learning

In this section, we explain the relevant concepts from the machine learning field. Section 2.2.1 describes the concept of machine learning and introduces potential different types of predictive tasks. Section 2.2.2 describes the predictive models that will be used later in the thesis.

2.2.1 Overview

Machine learning is a rapidly growing field of computer science that focuses on designing algorithms and models that can learn patterns and make predictions based on data. It is a subset of artificial intelligence and involves teaching machines to recognize patterns in data and use those patterns to make predictions or decisions without explicitly encoding the decision logic into the machines. Nowadays, data are becoming one of the most important crucial assets for companies and, since the amount of data generated is growing consistently, machine learning is becoming increasingly important for businesses and organizations to gain insights from data and thus drive decision-making. In machine learning, large datasets are often used to train the predictive algorithms, which then leverage the knowledge gained from that data to make predictions or decisions about new data. This allows machines to learn from experience, similarly to human learning process.

Machine learning has been applied to a wide range of applications; in this thesis, machine learning algorithms are leveraged in order to predict the outcome

[3]Typically, the partial order is induced by the timestamp, i.e., $e' < e'' \iff \pi_{time}(e') < \pi_{time}(e'')$. However, we do not require to make that assumption.

in the context of business processes. The desired outcome is usually defined by process analysts, process owners and stakeholders, who have a primary interest in monitoring how process instances are going to evolve, in order to promptly being alerted on the instances with higher risk to not achieve the expected outcome; as an example, a delivery company could have the desire to monitor and be alerted whether there is a high probability to not deliver the package on time. Therefore, in this thesis, we will deal with *supervised* learning, which is an approach that involves training a predictive model on label data. More formally, training data is represented as n labeled samples:

$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n) : n \in \mathbb{N}\} \tag{2.1}$$

where $x_i \in \mathcal{X}$ are m-dimensional feature vectors (with $m \in \mathbb{N}$) and $y_i \in \mathcal{Y}$ are the corresponding labels, i.e. the values of the target variable. The feature vectors extracted from the labeled training data are used to train a predictive model; afterwards, the trained predictive model will assign labels to new testing data, and the prediction will be compared with the expected label, assessing the accuracy of the predictive model. In other words, a model generalizes a pattern, providing a mapping $f : \mathcal{X} \to \mathcal{Y}$. The labels can be either continuous, e.g. when indicating the total time of a process, or discrete, e.g. when indicating the risk classification of a given loan. In the former case, the model will be defined as a regression, while in the latter case it will be defined as a classification, which can be further split into two types: multiclass and binary classification. As the name suggests, the latter indicates a classification between only two classes.

From a probabilistic perspective, the machine learning objective is to infer a conditional distribution $P(\mathcal{Y}|\mathcal{X})$. A standard approach to tackle this problem is to represent the conditional distribution with a parametric model, and then to obtain the parameters using a training set containing $\{x_n, y_n\}$ pairs of input feature vectors with corresponding target output vectors. The resulting conditional distribution can be used to make predictions of y for new values of x. Since the conditional distribution discriminates between the different values of y, this is defined as a *discriminative* approach [49], which tries to define a decision boundary that divides the feature space into areas containing feature vectors belonging to the same class (see Figure 2.4).

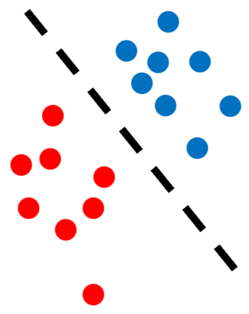


Figure 2.4: Example of discriminative model

2.2.2 Learning algorithms

Predictive process monitoring methods have employed a variety of classification and regression algorithms. In this thesis, we relied on different types of predictive models. The first one, Catboost, performs Gradient Boosting on Decision Trees and belongs to the group of Ensemble Learning algorithms, while the remaining algorithms, namely Long Short-Term Memory (LSTM) network, and Graph Neural Network (GNN), belong to the group of Deep Learning algorithms. In the following, we provide a detailed description of the predictive algorithms that have been used in this thesis.

Decision Trees

One of the main predictive algorithms that will be used in this thesis, Catboost, is based on gradient boosting on decision trees. Therefore, before discussing Gradient Boosting algorithms, Decision Trees are introduced.

Decision Trees are hierarchical structures where each internal node has a condition which is evaluated when an example is being classified [8]. Based on the outcome, the example is sent either to the left or to the right child of the node. The process is repeated until a leaf node is reached, and a prediction has been made about the example's class. Thus, decision making is organized hierarchically. A binary classification tree categorizing a person as healthy or unhealthy is shown in Figure 2.5. The algorithm finds the relationship between the response variable and the predictors and expresses this relation in the form of a tree-structure. As an example, if a person is over 30 years old and does not exercise a lot, it is clas-

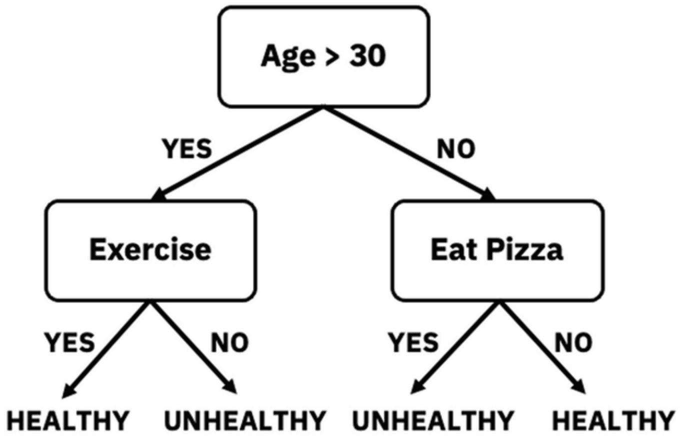


Figure 2.5: Example of Decision Tree model

sified as unhealthy. The model is trained by creating a top-down tree and then this trained decision tree is used to classify and assign a category to new unseen data examples.

Decision tree learning employs a divide and conquer strategy by conducting a greedy search to identify the optimal split points within a tree. This process of splitting is then repeated in a top-down, recursive manner until the data points have been classified under specific class labels. In order to select at each node the best attribute that will be used to split and classify data examples, one of the most common methods is to rely on the *information gain*; however, the concept of entropy needs to be introduced first. Entropy is a concept that stems from information theory, which measures the impurity of the sample values. It is defined as:

$$H(X) = -\sum_{i=1}^{N} p_i * log_2(p_i) \tag{2.2}$$

where i iterates through the possible categories and p_i refers to the probability of the category i; since our example refers to a binary classification, $i = 2$. Entropy values ranges between 0 and 1. If all the samples in the dataset, X, refer to one category, then the entropy will be equal to zero and the uncertainty will be the lowest possible; conversely, if half of the samples belong to one category and the other half to the second category, then the entropy will be 1, i.e. the highest possible.

Table 2.3: Example of dataset. The last column indicates the target variable

Age	Exercise	Eat Pizza	Healthy
40	Yes	Yes	Yes
50	No	Yes	No
20	Yes	No	Yes
25	No	No	Yes
23	No	Yes	No

Information gain can now be represented as the entropy difference before and after a split on a given attribute. The attribute that is associated with the highest information gain (i.e. that reduces the entropy the most) will be selected as the best split in a particular node of the tree, since it indicates the split that will best classify the training data according to the target classification. More formally, information gain is usually defined as:

$$IG(X,a) = H(X) – H(X|a) \tag{2.3}$$

where $H(X)$ is is the entropy for the dataset before splitting (i.e. it is the entropy associated with the parent node), and $H(X|a)$ is the conditional entropy for the dataset after splitting leveraging the attribute a (i.e. it is the entropy associated with the child nodes). In the following, we provide a concrete example showing how information gain is calculated given the sample dataset in Table 2.3. The last column, indicating whether a person is classified as healthy or unhealthy, is the target variable. Here, the entropy is 0.97; it can be calculated by finding the proportion of healthy people, which is 3/5, and the proportion of unhealthy people, which is 2/5. Applying the entropy formula we obtain $H(Healthy) = -((3/5)log_2(3/5) + (2/5)log_2(2/5)) = 0.97$. We can then compute the information gain for each of the attributes individually. As an example, the information gain obtained by splitting on the attribute "Exercise" would be the following: $IG(Healthy, Exercise) = (0.97) - (2/5)*(0) – (3/5)*(0.91) = 0.42$, where 2/5 and 3/5 represent the proportion of values where Exercise=yes and Exercise=no, respectively. 0 represents the entropy when Exercise=yes, while 0.91 is the entropy when Exercise=No. After computing the information gain for all the attributes, the attribute associated with the higher information gain will be selected.

Ideally, the leaf nodes should be homogeneous in the dependant variable, i.e. each leaf node should contain examples related to exactly one category. However, if the size of the tree grows considerably, this usually results in too little data points falling within a given subtree; this problem is known as data fragmentation, and it can often lead to overfitting, which is the impossibility to generalize predictions also for new unseen data points. As a result, it is usually preferrable to have small decision trees; in order to reduce the complexity and prevent overfitting, pruning is usually employed, since it removes branches that split on features with low importance.

Gradient Boosting Trees (Catboost)

Gradient boosting is a machine learning technique that belongs to the family of ensemble learning methods. Typically, in ensemble methods, multiple base (or ”weak”) learners are trained for the same task, and the predictions are then combined together in order to obtain the final prediction. In gradient boosting, predictions from base learners are combined by following a gradient learning strategy [34]. The main difference with other ensemble methods, such as Random Forest [9], is that instead of having parallel weak learners trying to give a prediction for the assigned task, the learners are built in a sequential manner. A base learner is firstly fit to the whole space of data; afterwards, each subsequent learner is trained to correct the errors of the previous one. In particular, in each iteration, the residuals (the differences between the predicted and the target values) are calculated and the next learner is fitted on these residuals, resulting in a boosted version of the previous model. This process is repeated until some stopping criterion is reached. Finally, the output of the gradient boosting model is a kind of weighted mean of the individual predictions of each base learner. Decision trees are typically selected as base learners [108].

To give a better intuition, we illustrate in Figure 2.6 the general idea behind gradient boosting. Let us suppose that we aim to learn a function that is able to divide the ”+” symbols from the ”-” symbols; however, instead of leveraging a unique ”strong” learner, we rely on multiple ”weak” learners, which are able to divide the plane only by drawing vertical or horizontal lines. In box 1, it can be seen that the first learner, D1, divides the plane by drawing a vertical line, correctly classifying the ”+” on the left and the ”-” on the right. The three ”+” on the right side of the line are instead not correctly classified; therefore, more weight will be assigned to them, as it can be seen by the larger size assigned to them in box 2. At this point, the second classifier, D2, draws another vertical line. This time, the two ”-” on the right and the five ”+” on the left are correctly

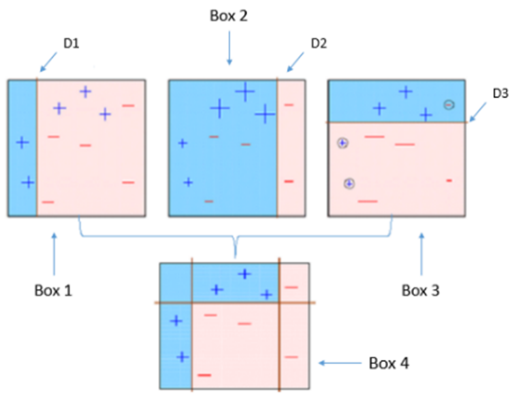


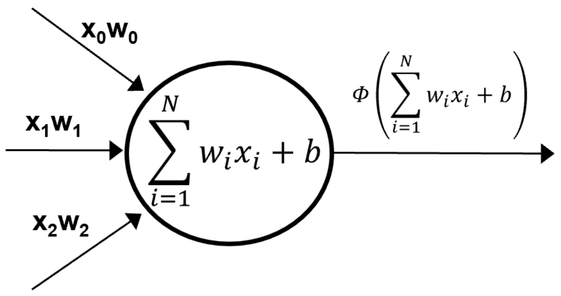
Figure 2.6: Example of a Gradient Boosting model

classified; conversely, the three ”-” on the left side are not correctly classified, and more weight will be given to them in box 3. Finally, the third classifier, D3, draws an horizontal line in box 3, correctly classifying the ”+” above and the ”-” below. As it can be seen, each classifier is predicting by giving more attention to the errors of the previous classifier, but none of them is able to solve the task alone. However, when they are combined together, the task can be correctly solved, as it can be seen in box 4.

In this thesis, as one of the primary prediction algorithms, we rely on Catboost [20], which is a high-performance open source framework for Gradient Boosting on Decision Trees. It is backed by solid theoretical results that explain how strong predictors can be built by iteratively combining weaker models (base predictors) in a greedy manner and it outperforms and solves limitations of current state-of-the-art implementations of gradient boosted decision trees. Catboost performs at each iteration t a random permutation of the features and a tree is constructed on the basis of it. Moreover, for each split of a tree, Catboost combines (concatenates) all categorical features (and their combinations) already used for previous splits in the current tree with all categorical features in the dataset.

Neural Networks

Before discussing deep neural networks models that have been adopted in this thesis, neural networks must be introduced first. A neural network consists of a set of nodes (also called neurons). Figure 2.7 illustrates an example of a neural network node. The equation of a neural network node can be expressed as:



x_0w_0
x_1w_1
x_2w_2
$\sum_{i=1}^{N} w_i x_i + b$
$\Phi\left(\sum_{i=1}^{N} w_i x_i + b\right)$

Figure 2.7: Example of a Neural Network node

$$\sum_{i=1}^{N} w_i x_i + b \tag{2.4}$$

Here, each node receives n inputs x_i, w_i represents the weight assigned to the input x_i and b is the bias term. The result is then passed to a non-linear activation function ϕ, which produces the output of the neural network cell. Several activation functions can be adopted. Common adopted activation functions are the sigmoid, the tanh and the rectified linear unit (Relu) activation functions, which are represented by $sigmoid(x) = \frac{1}{1+e^{-x}}$, $tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ and $relu(x) = max(0, x)$, respectively. In this thesis, we relied on the Relu function, which is proven to better model the non-linear relationships between the input and output variables [29].

The nodes are connected into a network using edges between them. Specifically, a network consists of several layers of nodes, namely the *input layer*, one or several middle *hidden layers* and the *output layer*. A network is activated after receiving input data; afterwards, the output of the input layer constitutes the input of the first hidden layer; this procedure is repeated until the final output layer is reached and the output $\hat{y}$ is produced. The weights w_i and the bias term b are learned iteratively during the training of the neural network using an optimization algorithm to minimize a loss function $L(\hat{y}, y)$, which measures the difference between the predicted output and the actual output. In particular, the weights and the bias are typically learned using the backpropagation algorithm [85]. In a nutshell, during the forward pass, given n inputs x_i, the output is propagated until the final prediction $\hat{y}$ is produced. This value is then compared with the expected target value y (also defined *ground truth*), and the error is calculated based on the defined loss function L. During the backward pass, this error is then used in order

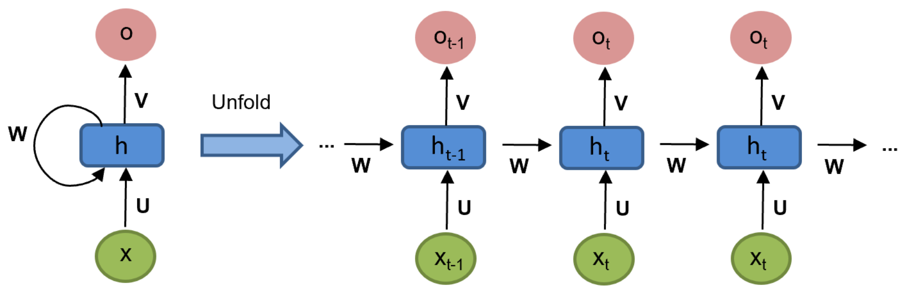


Figure 2.8: Example of a Recurrent Neural Network

to adjust the weights through gradient-based optimization.

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a special type of neural networks where the connections between neurons form a directed cycle. In fact, unlike traditional neural networks, RNNs have a feedback loop that allows them to take into account not only the current input, but also previous inputs and their context. This feedback loop enables RNNs to maintain an internal state, or ”memory”, that allows them to model temporal dependencies and patterns in the input data. RNNs can be unfolded, as shown in Figure 2.8. RNNs can take as input sequences with a variable length; after unfolding, each step is referred to as a time step, where x_t represents one element of the sequence and it is the input at time step t. h_t is the hidden state at time step t and serves as a summary of the information extracted from previous time steps. The hidden state h is updated with information of the new input x_t after each time step: $h_t = \phi(Ux_t + Wh_{t-1})$, where ϕ is the activation function and U and W are vectors of weights that are applied to the new input and to the previous state, respectively. o_t is the output at step t and V is the vector of weights that is applied to the output.

Long Short-Term Memory Networks

One limitation of standard RNNs is the ”vanishing gradient” problem, which causes RNN to perform rather poorly on longer temporal dependencies. As a result, it is difficult for the network to learn long-term dependencies, and the input at the beginning of the sequence may not affect the output at the end. A LSTM network [36] is a special type of recurrent neural network architecture that is designed to address the problem of vanishing gradients. In this thesis, LSTM networks have been considered when evaluating different predictive monitoring frameworks, since the literature has shown that they are among the most suitable methods for predictive business monitoring (cf. Section 3.2).

The main distinction between a regular RNN and a LSTM is that the latter has a more complex memory cell C_t. Here, the amount of information that should be retained in the cell state C_t is controlled via several control gates, which take into consideration both the current input x_t and the previous hidden state h_{t-1}. In particular:

$$\begin{aligned} i_t &= sigmoid(W_i * [h_{t-1}, x_t] + b_i) \\ f_t &= sigmoid(W_f * [h_{t-1}, x_t] + b_f) \\ \widetilde{C}_t &= tanh(W_C * [h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \widetilde{C}_t \end{aligned} \tag{2.5}$$

Here, i_t represents the input gate, which controls the amount of information related to the new input that should be added to the state cell. As already mentioned, it considers the previous hidden state (h_{t-1}) and the current input (x_t) as inputs and applies a linear transformation using the vector of weights W_i and the bias b_i (please note that in all the formulas illustrated above and hereafter described, the vectors of weights W the biases b are both learned during the training phase). The result is then passed through the sigmoid activation function to obtain a value between 0 and 1. In a similar way, f_t represents the forget gate, which determines how much of the previous cell state should be retained. $\widetilde{C}_t$ is the candidate cell state, which represents the new information that should be added to the cell state. Finally, we can illustrate the cell state C_t, which is the memory unit in charge of storing the information from previous time steps. C_t is updated by considering two elements. The first is $f_t * C_{t-1}$, where the forget gate controls the amount of information of the previous state cell C_{t-1} that should be considered. We recall that the forget gate f_t values ranges between 0 and 1, where 0 and 1 indicate that the information from the previous state cell should not or should be considered, respectively. The second element that updates the state cell is $i_t * \widetilde{C}_t$, where i_t, similarly to f_t, controls the amount of information related to the new candidate cell state that should be saved to the state cell. Finally, the information of the state cell C_t will be propagated to the output h_t based on the activation of the output gate o_t:

$$\begin{aligned} o_t = sigmoid(W_o * [h_{t-1}, x_t] + b_o) \\ h_t = o_t * tanh(C_t) \end{aligned} \tag{2.6}$$

Graph Neural Networks

Graph Neural Networks (GNNs) are a type of neural network that is designed to operate on graph data structures, which are used to represent complex relationships between entities. A graph is a data structure that consists of a finite set of nodes and a set of edges connecting them. Formally, graphs, nodes and edges are defined as follows:

Definition 2.2.1 (Graph, Node, Edge)**.** *A graph G is a two-element tuple* (V, EM)*, where V is the set of nodes and* EM *is the set of edges. A node* $v \in V$ *is represented by a single value, and a set of node features can be assigned to it. An edge* $\hat{e}$ *is a tuple* $(v, v') \in V \times V$*, and a set of edge features can be assigned to it.*

Graphs can be used to solve real-life problems that involve representing the problem space as a network. As an example, let us suppose that we want to segment the users of a social network (Figure 2.9); we can model this problem by representing a single user as a node, with the edges between the nodes highlighting their social connection with other users of the social network. Furthermore, each node can contain information about the users, such as the name, the gender, and other sensitive data. In this example, the graph is defined *undirected*, since the nodes are connected by edges that are all bidirectional. Conversely, in *directed* graphs, the nodes are connected only by directed edges. In this thesis, we rely on directed graphs, since we assume a temporal ordering between the events of a business case (more details will be provided in Section 5.3.4).

Figure 2.10 illustrates three different types of prediction tasks on graphs: node-level, edge-level and graph-level prediction tasks [118]. Node-level tasks are concerned with either classifying each node or assigning a value to each node within a graph. With node-level tasks, we may want to divide nodes into distinct classes based on common properties, grouping together similar nodes. With edge-level tasks, the objective is to predict whether or not there is an edge between two or more nodes in a graph. As an example, in a typical recommender system, where the nodes represent the users and the items while the edges represent the user-item interaction, we may want to recommend items that users might like. Finally, in a graph-level task, our goal is to predict the property of an entire graph.

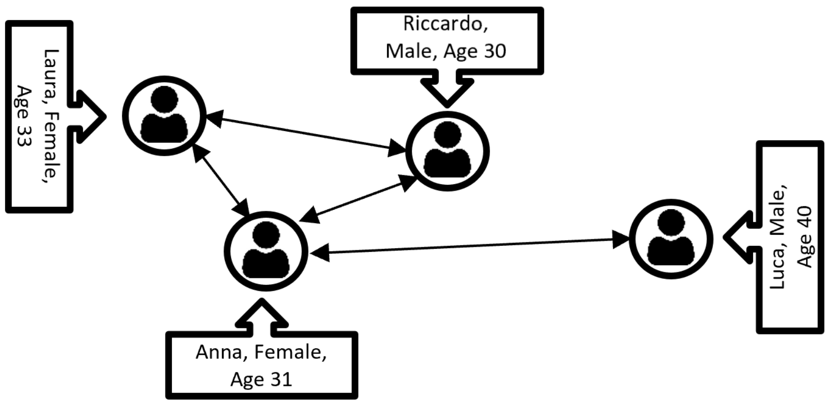


Figure 2.9: Graph illustrating a social network, where the nodes represent the users and the edges the social connections between the users

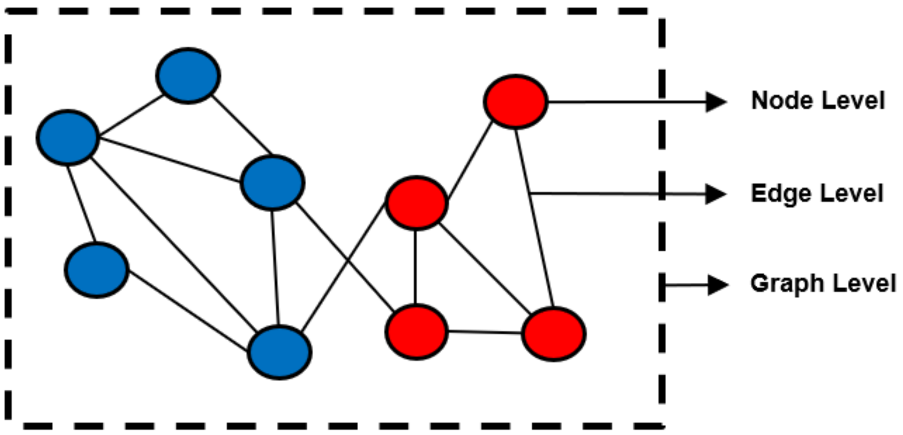


Figure 2.10: Different types of prediction tasks on graphs

As an example, given a graph representing the structure of a molecule (where the nodes represent the atoms and the edges the chemical bonds between the atoms), we may want to classify the smell of the molecule.

In order to train a graph neural network, a message-passing technique is applied. Initially, the embeddings of the nodes are initialized with the input feature vectors x_v, i.e. $h_v^{(0)} = x_v, \forall v \in V$, where $h_v^{(0)}$ is the hidden representation of a node. Afterwards, the message passing operation consists in iteratively updating the previous hidden representation vector h_v of each node v by aggregating messages $M_{N(v)}^{(n)}$ from the neighboring nodes $N(v)$ of node v based on some pooling strategy (such as Max, Mean, etc.); here, n represents the n^{th} step of the message passing. The message passing aggregation and update operations can then be summarized as:

$$\begin{aligned} M_{N(v)}^{(n)} &= Aggregate(h_{v'}^{(n)}, \forall v' \in N(v)) \\ h_v^{(n+1)} &= Update(h_v^{(n)}, M_{N(v)}^{(n)}) \end{aligned} \tag{2.7}$$

Here, the $Aggregate$ function takes the set of representations (embeddings) of the neighboring nodes $N(v)$ of node v and creates a message $M_{N(v)}^{(n)}$ based on the aggregated neighbourhood information at n^{th} iteration. The $Update$ function then combines the message $M_{N(v)}^{(n)}$ with prior embedding $h_v^{(n)}$ of node v to produce the updated embedding $h_v^{(n+1)}$.

More formally, the message passing function illustrated in Equation 2.7 can be also represented as:

$$h_v^{(n+1)} = \phi\left(W_v^{(n+1)} h_v^{(n)} + \sum_{v \in N(v)} W_v^{(n+1)} h_v^{(n)} + b^{(n+1)}\right) \tag{2.8}$$

where ϕ is the non-linear activation function, $W_v^{(n+1)}$ are the trainable parameter matrices of the node v and its neighboring nodes $N(v)$ and $b^{(n+1)}$ is the bias term.

2.3 Explainability in Machine Learning

The training of machine- and deep-learning models for predictive process analytics are typically based on black-box models (e.g. neural networks), which are proven to be more accurate, compared to those based on explicit rules (e.g. classification/regression trees). For instance, in this thesis, Catboost, LSTM, and graph neural networks have been employed.

While the priority remains on giving accurate predictions, users need to be provided with an explanation of the reasons why a given process execution is predicted to behave in a certain way. Black-box models fail to achieve this goal. Previous studies have shown that a necessary condition to build trust is to explain the reason of the provided predictions; otherwise, users would not trust the predictive-monitoring technology, and hence, adopt it [21, 72].

Therefore, it has become more and more evident that conventional performance measures of predictive models, such as accuracy or F-score, are insufficient, and model interpretability/explainability needs to be incorporated into this assessment [98]. Section 2.3.1 summarizes the main strategies for Explainable AI, while Section 2.3.2 details the idea behind Shapley values, which have been leveraged in this thesis for explaining predictive models.

2.3.1 Review of Explainable AI techniques

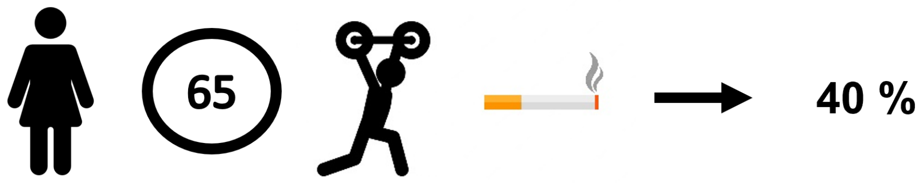
In the last years, two main strategies have been proposed in order to address the problem of explaining the outcome of predictive models. The first group of research proposals focuses on directly designing a transparent classifier that solves the same classification problem. This principle is known as "Transparent Box Design" problem [31] and is often solved by means of interpretable predictors based on extracted rule sets [119, 126]. These models are naturally interpretable; examples are linear regression, logistic regression and decision trees models, from which decision rules can be easily extracted. However, these models often have a significantly lower predictive accuracy. This category also includes attention-based models, which are more accurate and can also provide an underlying mechanism for interpretability [92]. Since these models require the computation of a distribution of weights over inputs, the attention weights can provide some insights to a decision-maker of why a certain prediction was computed by means of not manipulating directly the input features, but rather by manipulating the distribution of weights, which are associated to the input [92].

The second group of research proposals focuses on explaining how the model makes certain predictions. To this end, reverse engineering is typically exploited to understand the black box model. These approaches are also known as post-hoc explainability, and they provide interpretations based on natural language explanations, visualizations or by examples. Thus, we can separate two processes – decision making (prediction) and decision explaining. Post-hoc interpretations are also most suitable to provide intuition for complex non-transparent models, such as deep neural networks. Multiple strategies have been proposed for post-hoc explainability of black box models. Firstly, model explanation methods seek to provide globally explainable models that are able to mimic the behavior of black boxes and that are also understandable by humans. These methods adopt decision trees, sets of rules and linear models as comprehensible global predictors [40]. For example, neural networks can be approximated with decision trees [48] or symbolic rules [127]. Secondly, outcome explanation methods provide a locally explainable model that is able to explain the prediction of the black box in understandable terms for humans for a specific instance or record. In other words, rather than explaining the entire model, the focus is on explaining individual predictions. An example is the Local Interpretable Model-agnostic Explanations (LIME) approach [81], which uses local approximations based on generated random samples near the sample for which the prediction needs to be explained. In the context of image and text analysis, a common approach to explain why neural networks predict a certain outcome is based on designing saliency masks that visually highlight the determining aspects of the analyzed record [96]. Finally, model inspection methods aim to provide a textual or visual representation for understanding some specific property of a black box model or of its predictions. Common properties of interest include sensitivity to attribute changes, and identification of components of the black box (e.g., neurons in a neural network) responsible for specific decisions.

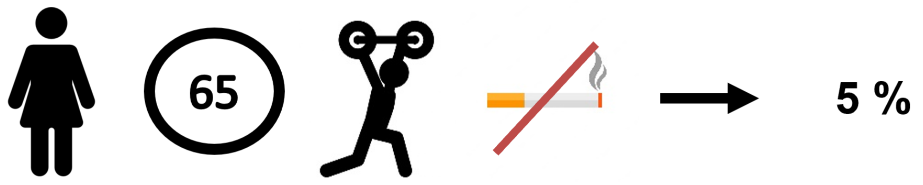
2.3.2 The Theory of Shapley Values

The Shapley Values [93] is a game theory approach to fairly distribute the payout among the players that have collaborated in a cooperative game. This theory can be adapted as an approach to explain a predictive model. In this context, the assumption is that the game is the prediction task, 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*).

To give a better intuition about how the algorithm works, let us suppose as an example that we want to predict if a patient will be affected by lung cancer. Figure 2.11a reports on the predicted probability to develop lung cancer for a 65 years old woman that usually smokes but also practices physical activity on a daily basis. The predicted probability can range between 0 and 1. These values can be interpreted as percentages (from 0% to 100%); in the following example, for readability purposes, we will refer to the predicted probabilities as percentages. Here, the probability is estimated as 40% (compared to an average prediction of 10%), and the objective is to find out how much has each feature value contributed to the prediction compared to the average prediction. The answer could be that the feature *sex=woman* contributed -20%, *age=65* contributed +30%, *sporty=yes* contributed -40% and *smoke=yes* contributed +60%. The contributions add up to +30%, which is the result of the final prediction (40%) minus the average pre-



(a) Predicted probability to develop lung cancer for a 65 years old woman that usually smokes but also practices physical activity on a daily basis.



(b) Predicted probability to develop lung cancer for a 65 years old woman that does not smoke and practices physical activity on a daily basis.

Figure 2.11: Predicted probability to develop lung cancer under different conditions

dicted probability (10%).

The Shapley value, which can be defined as the amount of contribution of the feature value to the model prediction [68], is calculated as the average marginal contribution of a feature value across all possible coalitions. In the following, we explain how a Shapley Value is calculated for one feature. Let us suppose that we want to discover the marginal contribution of the feature *smoke=yes*. This can be done by discovering the contribution when this feature is added to a coalition of *sex=woman*, *age=65* and *sporty=yes*; therefore, the first step is to remove *smoke=yes* from the coalition and replacing it with another value of the smoke feature taken from the dataset (for example *smoke=no*). Now the predicted probability to develop lung cancer for the coalition *sex=woman*, *age=65* and *sporty=yes*, which is reported in Figure 2.11b, is 5%. This means that the contribution of adding the feature *smoke=yes* is 40% - 5% = 35%. In order to get a better estimation of the influence of this feature value, we need to repeat this computation for all possible coalitions. The Shapley value is the average of all the marginal contributions to all possible coalitions. In this example, the possible combinations of feature values that are needed to determine the Shapley value for *smoke=yes* are:

- *No feature values*
- *Sex=Woman*
- *Age=65*
- *Sporty=yes*
- *Sex=Woman* $\wedge$ *Age=65*
- *Sex=Woman* $\wedge$ *Sporty=yes*
- *Age=65* $\wedge$ *Sporty=yes*
- *Sex=Woman* $\wedge$ *Age=65* $\wedge$ *Sporty=yes*

For each of these coalitions we compute the predicted cancer probability with and without the feature value *smoke=yes* and take the difference to get the marginal contribution. The Shapley value is the (weighted) average of marginal contributions.

More formally, given a prediction model, the Shapley Value of a feature can be defined as:

Definition 2.3.1 (Shapley Value). *Let $F = \{f_1, \ldots, f_m\}$ be a set of features defined over the domain $X_1 \times \ldots \times X_n$. Let's consider a prediction model $\Phi : X_1 \times \ldots \times X_n \to \mathcal{Y}$. 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.

Intuitively, the formula in Definition 2.3.1 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.

2.4 Predictive Analytics

This section illustrates how machine learning models can be leveraged for predictive process analytics. In literature, predictive analytics aims to estimate the future KPI values of the running cases. Here, we aim to be generic, meaning that KPIs can be of any nature:[4]

Definition 2.4.1 (KPI). *Let E be the universe of events defined over a set AN of attributes. Let $\mathcal{W}_K$ be the domain of the KPI values. A KPI is a function $\mathcal{K} : E^* \times \mathbb{N} \not\rightarrow \mathcal{W}_K$ such that, given a trace $\sigma \in E^*$ of an event log and an integer index $i \leq |\sigma|$, $\mathcal{K}(\sigma, i)$ returns the KPI value of σ after the occurrence of the first i events.*

Since our goal is to estimate (and ultimately 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 assumes it 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 each activity execution, which is recorded as next event in trace; however, other times, this is only known after the completion. We aim to be generic and account for all relevant cases. Given a trace $\sigma = \langle e_1, \dots, e_n \rangle$ that records a complete process execution, the following are three potential KPI definitions:

Remaining Time. $\mathcal{K}_{remaining}(\sigma, i)$ is equal to the difference between the timestamp of e_n and that of e_i.

Total Time. Given a trace σ of n events, $\mathcal{K}_{total}(\sigma, n)$ measures the difference between the timestamp of the last event of the trace and the timestamp of the first event (i.e., $\pi_{time}(e_n) - \pi_{time}(e_1)$).

Activity Occurrence. It measures whether 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 act is $\mathcal{K}_{occur\_act}(\sigma, i)$, which is equal to true if activity act occurs in $\langle e_{i+1}, \dots, e_n \rangle$ and $i < n$; otherwise false.

Customer Satisfaction. This is the typical KPI to analyze the customer journey [107]. Let us assume, without losing generality, to have a trace $\sigma =$

[4]Given a sequence X, $|X|$ indicates the length of X.

$\langle e_1, \ldots, e_n \rangle$ where the satisfaction is known at the end, e.g. through a questionnaire. Assuming the satisfaction level is recorded with the last event - say $e_n(sat)$ - then, $\mathcal{K}_{cust\_satisf}(\sigma, i) = e_n(sat)$.

The following definition states the prediction problem:

Definition 2.4.2 (The Prediction Problem)**.** *Let $\mathcal{L}$ be an event log that records the execution of a given process, for which a KPI $\mathcal{K}$ is defined. Let $\sigma = \langle e_1, \ldots, e_k \rangle$ 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 $k < i \leq n$.*

Predictive process analytics falls into the problem of supervised learning that aims to learn the model from a training set, for which the values of the dependent variables are known, whereas the value of an independent variable needs to be predicted. Training sets are composed by pairs $(x, y) \in \mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ represents the independent variables (also known as **features**) with their values, and $\mathcal{Y}$ is the value observed for the dependent variable (i.e. the value to predict).

Predictive process analytics requires a KPI definition $\mathcal{K}$ to be specified (cf. Definition 2.4.1). The dependent variable $\mathcal{Y}$ takes on a value from the domain of possible KPI values, which corresponds to the image of $\mathcal{K}$: namely $\mathcal{Y} = img(\mathcal{K})$. Conversely, the characteristics and nature of $\mathcal{X}$ depends on the AI technique that is used for prediction. In abstract terms, each prediction technique requires the definition of the domain $\mathcal{X}$ and, a **trace-to-instance encoding function** $\rho : E^* \to \mathcal{X}$, which maps each (prefix of a) trace σ in an element $\rho(\sigma) \in \mathcal{X}$.

The prediction model is trained off-line via a dataset $\mathcal{D}$ that is created from an event log $\mathcal{L}$ as follows. Each prefix σ' of each each trace $\sigma \in \mathcal{L}$ generates one distinct item in $\mathcal{D}$ consisting of a pair $(x, y) \in (\mathcal{X} \times \mathcal{Y})$ where $x = \rho(\sigma')$ and $y = \mathcal{K}(\sigma, |\sigma'|)$, where y is the value of the KPI $\mathcal{K}$ evaluated over the prefix σ' of the trace σ. Once the dataset item of every trace prefix is created, the model is trained. The resulting prediction model (a.k.a. predictor) can be abstracted as an oracle function $\Phi_{\mathcal{D}} : \mathcal{X} \to \mathcal{Y}$.

The on-line phase aim is to predict the KPI of interest for a set of running cases of the process, identified by a set $\mathcal{L}'$ of partial traces (i.e., a log). It relies on the process predictor $\Phi_{\mathcal{D}}$: for each $\sigma' \in \mathcal{L}'$, the predicted KPI value of the process instance identified by σ' is $\Phi_{\mathcal{D}}(\rho(\sigma'))$.

The definition of the trace-to-instance encoding function requires the intermediate concept of an *event-to-tuple function* $\zeta_{\mathcal{L}} : E \to A \times T \times (AV)^{|AN|}$, which encodes each event of the event log $\mathcal{L}$, where AN is the set of attributes defined in the event log $\mathcal{L}$. Depending on the prediction model, different encodings are possible. In this thesis, the oracle function $\Phi_{\mathcal{D}}$ has been learnt differently depending on the adopted predictive model. Section 2.4.1 illustrates the sequence encoding adopted for Catboost, while Section 2.4.2 describes the sequence encoding adopted for LSTM models.

2.4.1 Sequence encoding for the Catboost model

In the domain of Catboost learning, $\mathcal{X}$ (the independent variables) consists of a tuple with a certain number n of dimensions, i.e. $\mathcal{X} = (\mathbb{R}^n)$. The *event-to-tuple encoding function* $\zeta_{\mathcal{L}} : E \rightarrow \mathbb{R}^n$ is defined as follows: given an event e in $\mathcal{L}$, $\zeta_{\mathcal{L}} = \pi_{act}(e) \bigoplus\limits_{v \in AN} [\pi_{vmap}(e)]$[5], where $\oplus$ denotes the concatenation of two tuples.[6]

Moreover, since the final outcome of the process may be influenced by the previously occurred events, it could be important to also take into consideration the history of the process. Therefore, the function ρ is defined differently depending on the number of past events considered. Given a trace $\langle e_1, \ldots, e_m \rangle$ and a number $0 \leq k < m$ of past events considered, we define $\rho^k(\langle e_1, \ldots, e_m \rangle) = \langle \zeta(e_{m-k}) \bigoplus \ldots \bigoplus \zeta(e_m) \rangle$, where $1 \leq k < m$ represents the number of latest events to be considered. The parameter k can be optimized depending on the dataset using hyper-parameter optimization techniques; an example of application will be discussed in Chapter 3. Note that, if $k = 0$, only the last occurred event is used.

Alternatively, an aggregated history of each trace σ can also be considered, encoding the number of times that each activity has been performed in σ. To this aim, we define a function $\rho^{aggr}_{\mathcal{L}}(\langle e_1, \ldots, e_m \rangle)$; here, for each activity $act \in A$, one dimension exists in $\rho^{aggr}_{\mathcal{L}}(\sigma) : E^* \rightarrow (\mathbb{N})^{|A|}$ that takes on a value equal to the number of events $e \in \sigma$ that refers to act, i.e., such that $\pi_{act}(e) = act$. The function $\rho_{\mathcal{L}}$, supposing that $k = 0$, is then defined as: $\rho_{\mathcal{L}}(\langle e_1, \ldots, e_m \rangle) = \rho^{aggr}_{\mathcal{L}}(\langle e_1, \ldots, e_m \rangle) \bigoplus \zeta_{\mathcal{L}}(e_m)$.

[5]To keep the explanation simple, we assume that the enumerations of all variables v in AN are always returned consistently as if there is a total order among the variables (e.g., the alphabetical order).

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

2.4.2 Sequence encoding for LSTM networks

Let us recall the intermediate concept of an *event-to-tuple function* $\zeta_{\mathcal{L}} : E \to A \times T \times (AV)^{|AN|}$, which encodes each event of the event log $\mathcal{L}$; in the LSTM learning domain, each numerical and boolean attribute a becomes one feature, element of tuple $\zeta_{\mathcal{L}}(e)$. Each literal attribute a is instead represented through the so-called *one-hot encoding*: one different dimension exists for each value $v \in \mathcal{W}_{AN}(a)$, and the dimension referring to value $e(a)$ takes on value 1, with the other dimensions be assigned value 0.

Here, $\mathcal{X}$ consists of sequences of tuples with a certain number n of dimensions, i.e., $\mathcal{X} = (\mathbb{R}^n)^*$.[7] Function ρ is then defined as $\rho(\langle e_1, \ldots, e_m \rangle) = [\zeta(e_1), \ldots, \zeta(e_m)]$.

[7]In literature, LSTMs are often trained based on matrices. However, a sequence of m vectors in $\mathbb{R}^n$ can be seen, in fact, as a matrix in $\mathbb{R}^{n \times m}$. We use here the data set representation as vectors to simplify the formalization.

Chapter 3

Evaluation of Different Models for Predictive Process Analytics

Predictive Process Analytics is becoming an essential aid for organizations, providing online operational support of their processes. It aims to monitor the running instances of a given process and to alert on those that risk to not meet the desired outcome, such as taking too long, costing too much, not sufficiently satisfying customers. Predictive process analytics techniques are typically based on machine- or deep-Learning models that are trained over process' event data.

This chapter reports on the results of the empirical evaluation of our predictive monitoring framework. We acknowledge that a large share of the novelty lays on providing predictions with explanations of the factors that influence them; however, it is also important to report on the quality of the predictions, to illustrate that the explanations are based on high-quality predictions.

3.1 Motivation

Predictive Process Analytics is becoming an essential aid for organizations, providing online operational support of their processes. It aims to overcome the limitations of traditional monitoring practices by using data produced during process execution to continuously monitor processes performance; in particular, it aims to monitor the running instances of a given process and to alert on those that risk to not meet the desired outcome. Therefore, the goal of predictive process analytics is to identify future trends, and discover potential issues and anomalies in the process before they occur, allowing organizations to take proactive measures to prevent them from happening, optimizing the overall performance of the process.

Predictive process analytics techniques are typically based on machine- or deep learning models that are trained over historical process' event data. In this chapter, we perform an analysis of the state of the art of predictive analytics techniques, which revealed that Long Short-Term Memory networks are among the most leveraged models, since they generally outperform other methods. However, since in production environments also the time is an important constraint, we also evaluated alternative models that could be trained in a shorter amount of time, such as Catboost, which was also proposed by other research works in the process management domain. Therefore, this chapter reports on the empirical evaluation of the quality of the predictions produced by LSTM and Catboost models; the experiments conducted on 6 datasets and 17 different KPIs highlighted that Catboost not only can be trained in a shorter amount of time, but it can also generally outperform LSTM models.

We acknowledge that a large share of the novelty lays on providing explanations for the given predictions, highlighting the factors that influenced the predictive model. However, it is also important to report on the quality of the predictions, to illustrate that the explanations are based on high-quality predictions. In fact, while explainability is a necessary condition to build trust, also the predictive accuracy must be taken into account; otherwise, users would not trust and adopt the predictive-monitoring technology.

Section 3.2 summarizes the most relevant works in the field of predictive process analytics. Section 3.3 provides details on the datasets that have been used to assess the predictive accuracy of LSTM and Catboost models. Section 3.4 discusses the experimental setup and reports on the empirical evaluation of the quality of the predictions produced. Finally, Section 3.5 concludes the chapter.

3.2 Related Works

This section reports on the state of the art of Predictive Process Monitoring techniques. Note that predictive process analytics is also related to predictive maintenance [51, 59]. Predictive process analytics can actually be used to predict the eventual defects of machines, and it is especially beneficial when these defects are somewhat influenced by the activities that have previously been carried out through the machine.

The predictive-monitoring survey of Márquez et al. [63] reports on the large repertoire of techniques and tools that were developed to address this problem. In [47], an objective overview of state-of-the-art IoT developments is provided, and it is observed that most researchers' focus is on predicting the system's future state; however, little attention has been given to explaining the prediction values to the users, which is necessary in order to convince the users to trust and adopt the predictive-monitoring technology. Dinis et al. [19] propose a decision support tool for maintenance capacity planning, addressing the problem of forecasting the workload of future maintenance interventions. Conversely, Lughofer et al. [60] propose a self-adaptive predictive algorithm that can effectively suggest modifications in the machine parameters in an online chip production process. However, these works focus only on proposing an improved predictive algorithm, completely overlooking the aspect of explaining why certain predictions were given to the user.

Predictive monitoring has been built on different machine and deep-learning techniques, and also on their ensemble [63]. Different research works have recently illustrated that Long Short-Term Memory networks generally outperform other methods (see, e.g., [70, 73, 104]). However, since in production environments also the time is an important constraint, we wanted to leverage predictive models that could be trained in a shorter amount of time. In particular, we decided to rely on Catboost, which is a high-performance open source framework for gradient boosting on decision trees [20] that showed competitive performances. In fact, several research works in the domain of process management proposed to adopt Catboost for predictive monitoring; in particular, Catboost was shown to have a better accuracy compared to other types of classifiers in [125], where the authors introduced an approach for predicting loan default in peer-to-peer money lending processes, and in [30], where a comparison between different predictive models in order to detect Non-Technical Losses (NTL) in power industries was performed.

3.3 Datasets

For our assessment we used 6 real-life event logs, which are described below, while the event-logs statistics are shown in Table 3.1.

- ***Bank Account Closure*** is a process executed at an Italian banking institu-

Table 3.1: Event logs statistics

Event Log	# traces	# activities	mean events/trace	median events/trace	mean duration	std deviation duration
Bank Account Closure	32429	15	5.5	7	15.5 days	33 days
BPIC 2012	12369	23	14	8	7.9 days	11.7 days
BPIC 2012 - W	9658	6	7.5	6	11.4 days	12.7 days
BPIC 2013	7554	13	8.7	6	12.1 days	28.6 days
HelpDesk 2017	4580	14	4.7	4	40.9 days	8.4 days
Fine Management	125926	11	4	5	384 days	362 days

tion. The process deals with the closure of customer's accounts, which may be requested either by the customer or by the bank, for several reasons.

- ***BPI Challenge (BPIC) 2012***[1] is a real-life log of a Dutch Financial Institute. It represents an application process for a personal loan or overdraft within a global financing organization.

- ***BPIC 2012-W*** is the dataset derived from bpi12 challenge, and it represents the subprocess containing only the states of the work items belonging to the application.

- ***BPIC 2013***[2] is a dataset extracted from Volvo IT Belgium incident management system.

- ***HelpDesk 2017***[3] is a real-life log of SIAV s.p.a. company in Italy, and represents instances of a ticketing process in the company helpdesk area.

- ***Road Traffic Fine Management Process***[4] is a real-life event log of an information system managing road traffic fines.

3.4 Experimental Setup & Assessment of the quality of the Predictions

The framework for predictive monitoring has been implemented in Python, using Pandas to elaborate the data. For the LSTM implementation we relied on the

[1]http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
[2]http://dx.doi.org/10.4121/uuid:500573e6-accc-4b0c-9576-aa5468b10cee
[3]https://doi.org/10.4121/uuid:0c60edf1-6f83-4e75-9367-4c63b3e9d5bb
[4]http://dx.doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5

Keras framework and we built an implementation based on the work done by Navarin et al. [70]. In particular, as learning algorithm, we adopted ADAM with Nesterov Momentum (NAdam) [22]. We also selected 200 training epochs with a patience of 40, a learning rate of 0.001 and a batch size of 32. For the Catboost implementation, instead, we leveraged the open source library available.[5]

We used two/thirds of the traces as training, and one third as test set. Moreover, in order to improve the quality of the trained models, we used hyperparameter optimization, with 20% of the training data employed for this (validation set). Therefore, the training set was used to fit the predictors using combinations of hyperparameters, while the validation set was used to evaluate the performance of each combination. For LSTM, in particular, we tested different network configurations on the validation set, validating the number of LSTM neurons used for each layer (which varied between 100 and 250), and the number of layers (1, 2 and 4), with a 20% dropout for each layer. For Catboost, conversely, 3 different hyperparameters were tuned on the validation set: the number of trees used (which varied between 1500, 3000 and 4000), the depth of each single tree (3, 6 and 10) and the number of past events to be considered in a partial trace when predicting. In fact, as stated in Section 2.4, a particular encoding is needed in tree-based models in order to include information about past events (i.e. the history of the partial trace); here, in the last observed event of the partial trace, for every event-level attribute of each past event that previously occurred, a different dimension (feature) is added. Therefore, since LSTM naturally leverages information about past events, before proceeding to the comparison between Catboost and LSTM, a preliminary study was necessary in order to understand how many past events we should consider when predicting using the Catboost model. In particular, we focused on the scores obtained on the validation set; ideally, we should take the number of events associated to the point with the lowest Mean Absolute Error (MAE) (when predicting a numerical KPI) or associated to the point with the highest F1 score (when predicting a boolean KPI). The number of considered events varies from 0 (no information related to past events is used) to the average number of events per case, which was observed considering all completed process executions. Moreover, we also considered in our evaluations a different encoding of the history of each partial trace σ called *aggregated history* (cf. Section 2.4).

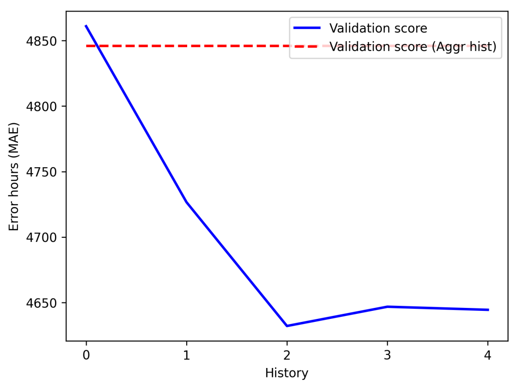
Figure 3.1 shows how the trained model evaluates on the validation set when an increasing number of events is considered for the remaining time prediction in the datasets described in Section 3.3. The y-axis represents the MAE of the

[5]https://catboost.ai/

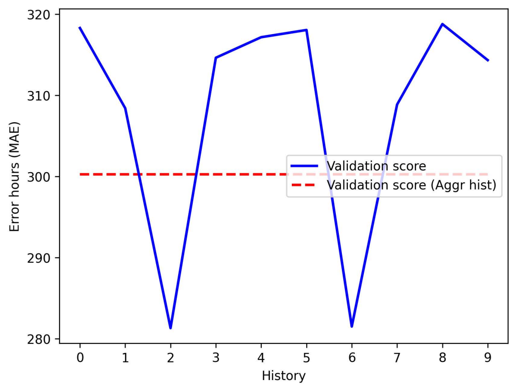
predictive model on the validation set (i.e. the lower the better) and the scale is expressed in hours for remaining time prediction. The blue line indicates the accuracy obtained by the predicting model varying the number of historical events considered, while the red dashed line is related to the quality of the predictions obtained by the model leveraging only aggregated historical information. Including historical information does not always improve the predictive quality of the model. In particular, in Fine Management process (Figure 3.1a) and in BPIC 2013 (Figure 3.1b), it is beneficial to consider historical information when predicting remaining time; here, compared to the situation where no previous events are considered, considering also information about the two previously last occurred events lowers the MAE by 200 and 40 hours, respectively. However, as shown in Figures 3.1c, 3.1d, 3.1e and 3.1f, encoding historical information shows no or very limited improvements in Bank Account Closure, HelpDesk, BPIC 2012 - W and BPIC 2012 processes, respectively.

Figure 3.2 reports on the quality of the predictions when predicting whether or not a given activity is going to eventually occur, when varying the number of past events considered for prediction. The name of the activity to predict clearly varies with the domain, and it is shown in Figure 3.2 in the legend. In particular, each line represents a different boolean KPI that was considered in our analysis. The values on the y-axis here indicate the F1 score (the higher the better). The results highlight that the quality of the prediction of whether or not a certain activity is going to occur is not increased if we consider features related to events that precede the last. This is likely due to the fact that the last event is already a good summary of the history. We also compared the scenario where we optimized the number k of past events (cf. encoding function ρ^k in Section 2.4) and when the aggregated history encoding function was used (cf. encoding function ρ^{aggr} in Section 2.4). This is shown in Figure 3.3: the use of the aggregated history shows no or limited improvement.

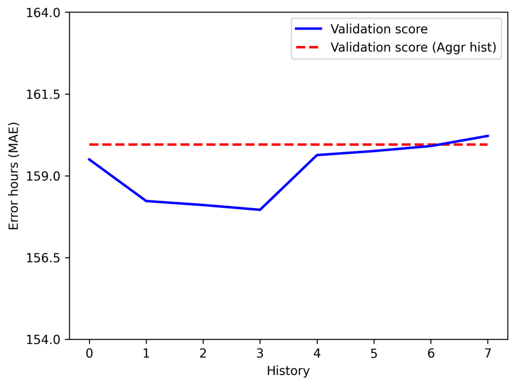
However, training and calculating the scores for each predictive model based on a different configuration of considered past events is a heavy procedure (especially in production environments). Therefore, we elaborated a more lightweight strategy; in particular, if the predictive quality does not improve (i.e. it is worse compared to the previous configuration or it improves by a percentage lower than 1%) for two consecutive steps, we stop our research and we take the best predictive model that was observed until that moment. We then compared the two approaches in order to understand if our heuristic was able to find an optimal accuracy without trying all possible combinations, and we reported the results in



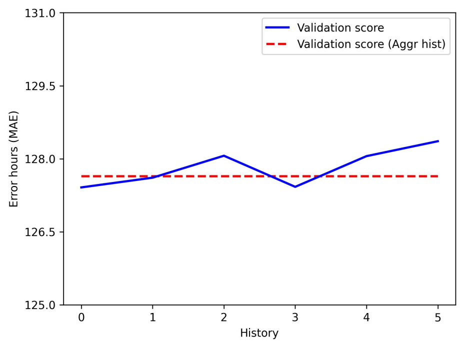
(a) Fine Management process



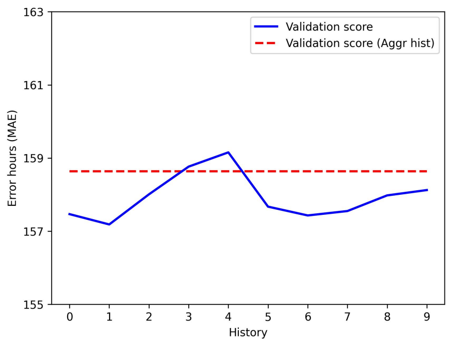
(b) BPIC 2013 process



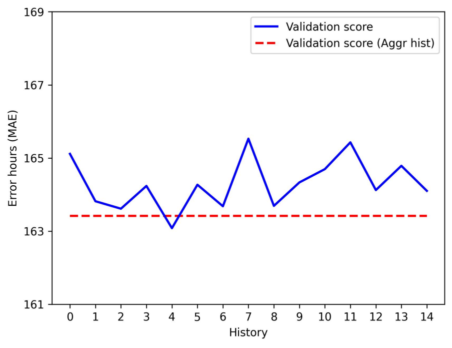
(c) Bank Account Closure process



(d) HelpDesk process



(e) BPIC 2012 - W process



(f) BPIC 2012 process

Figure 3.1: Remaining time prediction accuracy for Catboost model when considering the history of the trace

Table 3.2, where we also highlight the comparison with the model that was not leveraging historical information. The first two columns indicate the event log

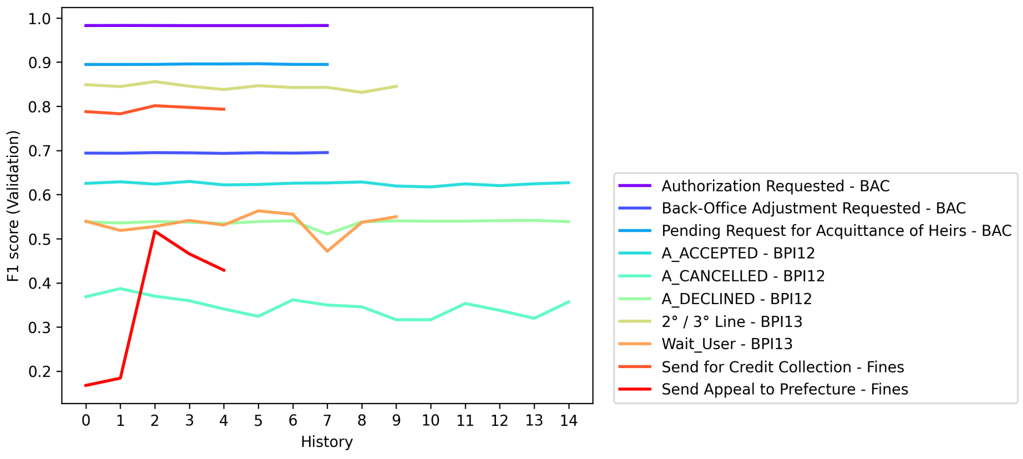


Figure 3.2: Activity occurrence prediction accuracy for Catboost model when considering the history of the trace

and the considered KPI, while the last three columns show the results obtained by Catboost models without considering history, considering the history according to our heuristic and according to the complete procedure respectively. We recall that when the KPI is numerical, then the scores represent the MAE (expressed in days for Remaining Time prediction and in Euros for Total Cost prediction); conversely, when the KPI is Boolean (Activity occurrence prediction), the scores represent the F1. In the last two columns it is also reported the number of previous events considered by the predictive model that were found to have the best predictive quality on the validation set. We can clearly see that in 11 cases out of 17 our heuristic approach found the same number of past events as the complete procedure, with a similar predictive quality in other cases, showing that our approach can be used to find a good configuration but reducing the computational time at the same moment. Moreover, in half of the cases, leveraging historical information can help to achieve a slightly higher accuracy.

After discovering how many past events we should consider when predicting using the Catboost model, we illustrate in Table 3.3 the comparison between the Catboost model (shown in the last column) and the LSTM model (shown in the second last column), which naturally leverages information about past events. As it can be seen, for the KPIs with identifier 1, 2, 6, 10, 11, 14 and 15, which are numerical KPIs, Catboost performs consistently better than LSTM. Moreover, with the exception of the boolean KPIs 3,4 and 17, where LSTM shows a better predictive accuracy, for the other boolean KPIs 5, 7, 8, 9, 12, 13 and 16 Catboost

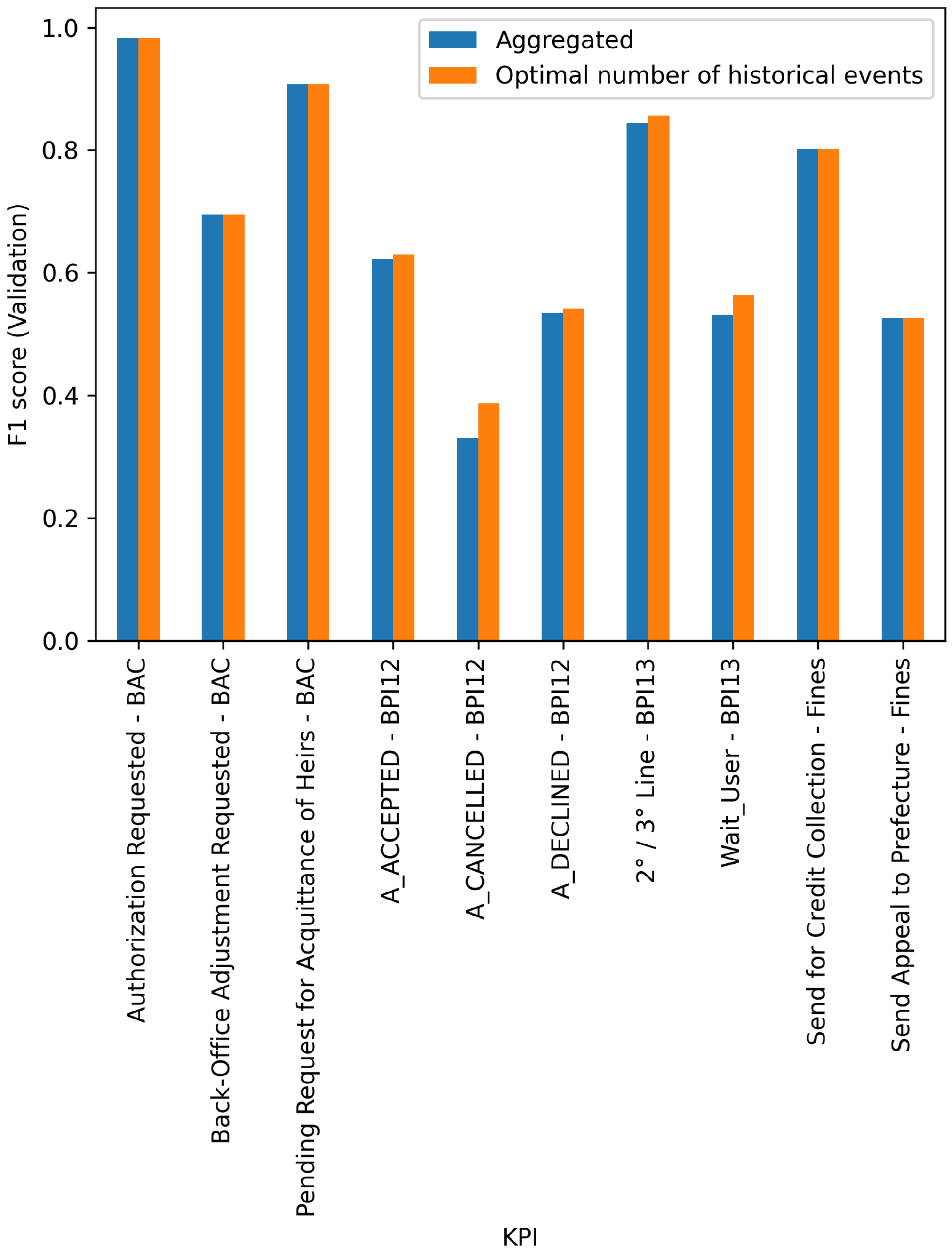


Figure 3.3: Comparison between the result obtained by Catboost with the optimal amount of historical events and the result obtained considering only aggregated historical information when predicting activity occurrence

Table 3.2: Catboost predictive quality when leveraging historical information. The scores represent the MAE when the KPI is numerical; conversely, when the KPI is Boolean (Activity occurrence prediction), the scores represent the F1.

Event Log	**KPI**	**Catboost (no history)**	**Catboost (heuristic)**	**Catboost (complete)**
Bank Account Closure	Remaining Time (MAE)	**4.18**	4.20 (History: 3)	4.20 (History: 3)
Bank Account Closure	Total Cost (MAE)	0.92	0.92 (History: aggr)	0.92 (History: aggr)
Bank Account Closure	Activity *Authorization Requested* (F1)	0.95	**0.96 (History: 1)**	**0.96 (History: 1)**
Bank Account Closure	Activity *Pending Request for acquittance of heirs* (F1)	0.88	0.88 (History: aggr)	0.88 (History: aggr)
Bank Account Closure	Activity *BO Adjustment Requested* (F1)	0.66	0.66 (History: 0)	**0.67 (History: 6)**
BPIC 2012	Remaining Time (MAE)	6.61	6.56 (History: aggr)	**6.53 (History: 4)**
BPIC 2012	Activity *A_ACCEPTED* (F1)	0.69	0.69 (History: 1)	**0.70 (History: 3)**
BPIC 2012	Activity *A_CANCELLED* (F1)	0.55	**0.56 (History: 1)**	**0.56 (History: 1)**
BPIC 2012	Activity *A_DECLINED* (F1)	0.56	0.56 (History: 0)	0.56 (History: 13)
BPIC 2012 - W	Remaining Time (MAE)	7.38	**7.36 (History: 1)**	**7.36 (History: 1)**
BPIC 2013	Remaining Time (MAE)	9.74	**9.58 (History: 2)**	**9.58 (History: 2)**
BPIC 2013	Activity *Push to front (2°/3° line)* (F1)	0.87	0.87 (History: 0)	0.87 (History: 2)
BPIC 2013	Activity *Wait User* (F1)	0.69	0.69 (History: 0)	0.69 (History: 5)
HelpDesk	Remaining Time (MAE)	5.27	5.27 (History: 0)	5.27 (History: 0)
Fine Management	Remaining Time (MAE)	196.93	**185.69 (History: 2)**	**185.69 (History: 2)**
Fine Management	Activity *Send for Credit Collection* (F1)	0.81	0.81 (History: aggr)	0.81 (History: aggr)
Fine Management	Activity *Send Appeal to Prefecture* (F1)	**0.43**	0.37 (History: aggr)	0.37 (History: aggr)

Table 3.3: Predictive quality comparison between Catboost and LSTM. The scores represent the MAE (expressed in days for Remaining Time prediction and in Euros for Total Cost prediction) when the KPI is numerical; conversely, when the KPI is Boolean (Activity occurrence prediction), the scores represent the F1.

ID	**Event Log**	**KPI**	**LSTM**	**Catboost**
1	**Bank Account Closure**	Remaining Time (MAE)	4.37	**4.18**
2	**Bank Account Closure**	Total Cost (MAE)	0.95	**0.92**
3	**Bank Account Closure**	Activity *Authorization Requested* (F1)	**0.99**	0.96
4	**Bank Account Closure**	Activity *Pending Request for acquittance of heirs* (F1)	**0.90**	0.88
5	**Bank Account Closure**	Activity *BO Adjustment Requested* (F1)	0.65	**0.67**
6	**BPIC 2012**	Remaining Time (MAE)	6.66	**6.53**
7	**BPIC 2012**	Activity *A_ACCEPTED* (F1)	0.60	**0.70**
8	**BPIC 2012**	Activity *A_CANCELLED* (F1)	0.37	**0.56**
9	**BPIC 2012**	Activity *A_DECLINED* (F1)	0.51	**0.57**
10	**BPIC 2012 - W**	Remaining Time (MAE)	7.84	**7.36**
11	**BPIC 2013**	Remaining Time (MAE)	11.82	**9.58**
12	**BPIC 2013**	Activity *Push to front (2°/3° line)* (F1)	0.81	**0.87**
13	**BPIC 2013**	Activity *Wait User* (F1)	0.45	**0.69**
14	**HelpDesk**	Remaining Time (MAE)	5.96	**5.27**
15	**Fine Management**	Remaining Time (MAE)	233	**185.69**
16	**Fine Management**	Activity *Send for Credit Collection* (F1)	0.77	**0.81**
17	**Fine Management**	Activity *Send Appeal to Prefecture* (F1)	**0.52**	0.43

can generally outperform LSTM models. In the light of these results, we opted to adopt Catboost, and to put LSTM models away. In the next chapter, explanations will be also reported for our case studies using Catboost, only.

3.5 Summary

In the recent years, organizations are exploiting opportunities to improve the performance of their business processes; to this end, Predictive Process Analytics is becoming an essential aid for organizations, allowing to monitor the running instances of a given process and to alert on those that risk to not meet the desired outcome. This allows discovering potential anomalies before they occur, allowing organizations to take proactive measures to prevent them from happening.

A lot of research has been devoted towards increasingly accurate frameworks for predictive process monitoring; in this chapter, we performed an analysis of the state of the art of predictive analytics techniques, which revealed that Long Short-Term Memory networks are among the most leveraged models, since they generally outperform other methods. However, since in production environments also the time is an important constraint, we also evaluated alternative models that could be trained in a shorter amount of time, such as Catboost, which was also proposed by other research works in the process management domain.

The high popularity of LSTM models in Predictive Business Process Analytics is due to the fact that it has always been considered crucial to take into account the history of the process when predicting the outcome of a running process instance, and LSTM models naturally leverage information about past events. Therefore, before performing the comparison between Catboost and LSTM, a preliminary study was necessary in order to understand how many past events we should consider when predicting using Catboost. However, the experiments conducted on six different datasets revealed that considering historical information shows no or limited improvements in 4 datasets out of 6 when predicting the remaining time, while the quality of the predictions when predicting whether or not a certain activity is going to occur is not increased (or it is very marginally increased) if we consider features related to events that precede the last in all the datasets; therefore, considering the last event or the aggregated historical information is in many cases already a good summary of the history. The comparison performed in this chapter on the quality of the predictions produced by LSTM and Catboost models on 6 processes and 17 different KPIs seems to confirm this finding, since the experiments revealed that Catboost not only can be trained in a shorter amount of time, but it can also generally outperform LSTM models.

We acknowledge that a large share of the novelty lays on providing explanations for the given predictions, highlighting the factors that influenced the predictive model. However, it is also important to report on the quality of the predictions, to illustrate that the explanations are based on high-quality predictions. In fact, while explainability is a necessary condition to build trust, also the predictive accuracy must be taken into account; otherwise, users would not trust and adopt the predictive-monitoring technology.

In the next chapter, our aim is to explain the reasons why a given process execution is predicted to behave in a certain way by leveraging on explainable AI techniques, in order to enhance the trustability of the predictive-monitoring technology.

Chapter 4

Explainable Predictive Process Analytics

We mentioned in Chapter 3 that Predictive Process Analytics is becoming an essential aid for organizations, providing online operational support of their processes. However, process stakeholders need to be provided with an explanation of the reasons why a given process execution is predicted to behave in a certain way. Otherwise, they will be unlikely to trust the predictive monitoring technology and, hence, adopt it. In this chapter, the predictive analytics framework described in Chapter 3 is equipped with explanation capabilities based on the game theory of Shapley Values. The framework has been implemented in the IBM Process Mining suite and commercialized for business users. Furthermore, while in the previous chapter we assessed the quality of the predictions, in this chapter we tested the corresponding evaluations on real-life event data. In particular, a user evaluation has been performed in order to understand if the explanations provided by the system were intelligible to process stakeholders.

4.1 Motivation

Predictive process analytics techniques are typically based on Machine- or Deep-Learning models that are trained over process' event data (cf. Section 3.2 and Chapter 3). However, the majority of these techniques are based on black-box models (e.g. neural networks), which are proven to be more accurate, compared to those based on explicit rules (e.g. classification/regression trees), which tend to be significantly less accurate.

While the priority remains on giving accurate predictions, users need to be provided with an explanation of the reasons why a given process execution is predicted to behave in a certain way. Black-box models fail to achieve this goal. Previous studies have shown that a necessary condition to build trust is to explain the reason of the provided predictions; otherwise, users would not trust the predictive-monitoring technology, and hence, adopt it [21, 72].

Therefore, it has become more and more evident that conventional performance measures of predictive models, such as accuracy or F-score, are insufficient, and model interpretability/explainability needs to be incorporated into this assessment [98]. Stierle et al. [100] report on the repertoire of techniques that have been developed to address the problem of interpreting/explaining the predictive process model. However, they claim that the explanation techniques should be deployed in real-world tools, and should be also assessed with end-users.

In this chapter, we propose a process-monitoring framework that is also able to explain any generic KPI, numerical or nominal, by proposing an explainable framework based on the Shapley Values game theory approach, which can be adapted to explain any predictive model. For the process-monitoring part, in light of the results obtained after comparing Catboost and Long Short Term Memory (LSTM) networks (cf. Chapter 3), we decided to rely on Catboost, which demonstrated comparable results with a drastically reduced model's training time, making it more suitable for production environments. After assessing the intelligibility of the proposed explanation strategy with internal IBM stakeholders, a user study was conducted, where several real and potential process analysts, from academia and industry, were confronted with the implementation of the explainable predictive process analytics framework. In particular, the implementation of the framework is not limited to be an academic proof-of-concept prototype, but has been incorporated into a commercial tool, the IBM Process Mining, and made available to the customers; in this way, it can be easily used also by business users that do not have a technical knowledge, obtaining results in a few hours of automatic computations, instead of running long analyses. Moreover, the user study using a commercial tool provides further strength to validation of the explanation's understandability by process analysts.

Experiments were conducted on different benchmarks and on several real-life datasets, with the aim of explaining the predictions of different outcome indicators (KPIs). The framework returns explanations at a global level, aiming to discover the principal factors driving the predictive model, but also at a local level of each running process instance. The global-level and local-level explanations were provided to 20 users to fulfill 18 tasks related to predictive process analytics: the user study shows that the explanation framework can be understood by users and provides valuable insights into the factors that affect predictions. The chapter is organized as follows. Section 4.2 summarizes the most relevant works related to Explainable AI in the BPM field, illustrating how our explainable framework advances the current state of the art. Section 4.3 describes how Shapley Values can be applied for explainable predictive monitoring, while Section 4.4 reports on our framework for explainable predictive process monitoring. Section 4.5 describes the outcome of our explanation strategy applied on a real-world case study. Section 4.6 provides an overview of the integration of the explainable predictive process monitoring framework with the IBM process mining suite, while Section 4.7 reports on the user evaluation conducted with process analysts. Finally, Section 4.8 concludes the chapter.

4.2 Related Works

This section compares our framework with the literature. In particular, Section 4.2.1 discusses explainability approaches that have been applied to explain the outcome of machine learning models, while Section 4.2.2 focuses on the application of explainability techniques in the Business Process Monitoring (BPM) field.

4.2.1 Explanation of Machine-Learning Models

Some approaches exist in the literature to explain machine learning models, arisen from the need to understand complex black-box algorithms like ensembles of Decision Trees and Deep Learning [4, 61, 62, 65, 81, 94, 95, 102]. Conversely, other approaches exist in the literature that are naturally explainable [6, 11, 117], but these works do not really discuss the relevance and the quality of these explanations.

The survey of Jagatheesaperumal et al. [42] provides an overview of applications and challenges related to Big Data and AI in Industry 4.0, highlighting explainability as one of the main challenges in order to increase machine learning models adoption in industry. The adoption of explanatory methods in industry is at an early stage; one of the most relevant works is by Rehse et al. [79], which also aims at providing a dashboard to process participants with predictions and their explanation. However, the paper does not provide sufficient details on the actual usage of the explainable AI literature, and the very preliminary evaluation is based on one single artificial process that consists of a sequence of five activities. Sachan et al. [86] apply a rule-based method in order to explain the chain of events leading to a decision for a loan application, but the approach requires the support of an expert's domain knowledge and the evaluation is based on a single business case study. In [95] an approach of fake news detection grounded in explainability is introduced.

Some research works have started addressing the issue of guaranteeing that the explanation plots provided to experts convey the information required to establish causality between feature values and process outcomes. In this direction, anchors [80] are a post-hoc explanation technique that yields if-then rules explaining the behavior of the underlying model, together with an indication of the precision and coverage of the rules. As far as we are aware, an approach customizing anchors in the BPM domain is missing in the literature. However, in order to yield meaningful results, anchors perturbation function needs to be explicitly designed for each particular domain/use case and many scenarios require discretization as otherwise results are too specific, have low coverage, and do not contribute to understanding the model.

A significant amount of work in literature is focused on healthcare applications. We highlight the work of Lundberg et al. [62], which proposes an implementation of the Shapley Values in healthcare where the explanatory method is used to prevent hypoxaemia during surgery, and the work of Meacham et al. [65], where explainability is used for analysis of patience re-admittance. In [15], Shapley Values are leveraged in order to validate the correctness of the predictive approach in a utility company. As already mentioned in Section 2.3, in this thesis we relied on the SHapley Additive exPlanations (SHAP) implementation of the Shapley values, which has the strong theoretical foundation of the original game theory approach, with the advantage of providing offline explanations that are consistent with the online explanations. Moreover, SHAP avoids the problems in consistency seen in other explanatory approaches (e.g. the lack of robustness seen in the online surrogate models [3]).

4.2.2 Explanations in the BPM field

As already mentioned in Section 3.2, several works focused on predicting the system's future state or forecasting the workload of future maintenance interventions [19, 47]. However, these works focus only on proposing an improved predictive algorithm, completely overlooking the aspect of explaining why certain predictions were given to the user. These are in fact the problems tackled in this chapter, so as to ensure that the predictive-monitoring system is trusted, and thus used.

The need to incorporate interpretability in addition to conventional performance measures (such as accuracy) when evaluating predictive models has been strongly emerging [98] and several works have been focusing in comparing and evaluating explanations produced by different frameworks [24, 25, 99, 114, 115]. We also highlight [58], whose focus is on stimulating and integrating human feedback into the model exploiting fuzzy rule-based systems; however, no assessment of the intelligibility of the explanations is made.

The explainable survey of Stierle et al. [100] reports on the repertoire of techniques that were developed to address this problem; however, it is observed that the explanation techniques should be deployed in real-world tools, and should be also assessed with end-users. Rizzi et al. [82] are among the first to investigate whether users actually understand the explanation plots returned by XAI techniques in the context of Predictive Process Monitoring. In particular, they investigated the intelligibility of several explanation plots, but only relying on 8 participants, and without employing a consolidated user-interface evaluation methodology.

This chapter does not only propose a framework able to solve some state-of-the-art current limitations, but also evaluates with process analysts (both from the academia and the industry domains) if they actually understand and feel comfortable with the results returned by our explainable predictive monitoring framework; moreover, the framework is also deployed in a real company, providing business stakeholders with online operational support of their processes.

Breuker et al. [10] also try to tackle the problem, but their attempt is not independent of the actual technique employed for predictions; furthermore, their explanations are only based on the activity names, while the explanations can generally involve resources, time, and more (cf. the case studies reported in Section 4.5). Hsieh et al. [38] and Huang et al. [39] apply a counterfactual approach, which provides an understanding on what could have been done differently in order to achieve the desired outcome (e.g. having a loan approved), but their approach focuses only on providing explanations for categorical KPIs. Rizzi et al. [83] propose an approach based on LIME, but they can provide explanations only for nominal KPIs and their main focus is on improving predictive model accuracy rather than highlighting model drivers. Also Coma-Puig and Carmona [14] proposed an approach based on LIME as a rule-based double-checking method to discard high-scored customers with unreliable explanations; however, the authors decided in a subsequent research work [15] to discard LIME in favour of SHAP due to the well-known problems of robustness [3], because of the random component of the algorithm but also the difficulty of having an optimal configuration.

The framework proposed in this chapter specializes the use of Shapley values to the problem of providing explanations for predictive analytics. Furthermore, the use of SHAP enabled us to provide explanations independently from the leveraged predictive model (*model agnostic*), while *model specific* approaches are specifically designed for certain model types; in particular, some approaches [32, 33,74,97,103,122,123] can provide explanations only for neural network models. The study by [66] proposed to generate causal explanations for deep learning-based process outcome predictions by using a global post-hoc explanation approach, called partial dependence plots (PDP); however, because PD plots only show the global average marginal effects of the features, the effect of the features with local heterogeneous effects (for example positively influencing the outcome in half of the cases while negatively influencing the outcome in the other half) might be hidden. Moreover, PDP tend to be hard to read when more than two dimensions are used. Other works [4, 97] use attention mechanisms, which also have the limitation that is linked to the lack of consensus that attention weights are always correlated to feature importance. Using attention weights, especially when complex encoders are used, is found questionable by [87], while [92] claim that attention weights often fail in the task of finding the factors influencing the model's final decisions.

4.3 Explainable Predictions through Shapley Values for Catboost

In this thesis, we leverage the idea of using post-hoc explainability approaches to explain the outcome of a black box model; in particular, for each running case, we aim to return the set of attributes that influence its prediction the most, with the corresponding magnitude and with the indication whether the attributes increase or decrease the predicted KPI's value. Note that increasing the KPI value does not necessarily have a positive connotation because higher values do not necessarily mean better values. For instance, if the KPI is the process-instance cost, it is usually desirable to reduce the value.

In the light of the above, for each trace σ, the problem can be stated as finding a function $\mathcal{T}_{(\sigma,\mathcal{K})}$ such that $\mathcal{T}_{(\sigma,\mathcal{K})}(a, v)$ indicates how much the fact that $a = v$ influences the KPI prediction. The positive or negative sign of $\mathcal{T}(a, v)$ indicates whether the influence is towards increasing or decreasing the KPI value.

Definition 4.3.1 (The Prediction-Explanation Problem)**.** *Let $\mathcal{L}$ be an event log over a set AN of attributes, with domains $\mathcal{W}_{AN}$. Let $\sigma = \langle e_1, \ldots, e_k \rangle$ be a running case with a KPI prediction $\mathcal{K}$. Let be $\overline{\mathcal{W}} = \cup_{a \in AN} \mathcal{W}_{AN}(a)$. Explaining the prediction is the problem of computing a function $\mathcal{T}_{(\sigma,\mathcal{K})} : AN \times \overline{\mathcal{W}} \nrightarrow \mathbb{R}$ defined over a subset of attributes of AN that affects the prediction, where $\mathcal{T}_{(\sigma,\mathcal{K})}(a, v)$ is defined only if $v \in \mathcal{W}_{AN}(a)$.*

In this thesis, in order to explain the outcome of a black box model, we rely on the SHAP implementation of the Shapley values [61], which explains the predictions by assuming that for every instance each feature value of the instance is a player in a game where the prediction is the payout. It has the advantage of providing offline explanations that are consistent with the online explanations, and avoids the problems in consistency seen in other explanatory approaches (e.g. the lack of robustness seen in the online surrogate models [3]); moreover, it can provide explanations independently from the leveraged predictive model (*model agnostic*), while *model specific* approaches are specifically designed for certain model types. Section 2.3.2 introduced the theory of Shapley values, while this chapter illustrates its application and adaptation for predictive process monitoring.

While the proposed explainable predictive framework is independent of the specific technique used for prediction, only explanations of predictions for the Catboost method are reported. This choice is motivated by the fact that, as discussed in Chapter 3, Catboost is preferrable to LSTM for prediction, due to the increased speed to build the model while keeping a better accuracy. Moreover, when comparing the performances of four different predictive models for Object-centric predictive Analytics (cf. Chapter 5), Catboost achieves the highest predictive accuracy in almost all the considered KPIs.

The starting point to build the explainable framework is the trace-to-instance encoding function $\rho : E^* \to \mathcal{X}$, which maps each (prefix of a) trace σ in an element $\rho(\sigma) \in \mathcal{X}$ (cf. Section 2.4); given a trace $\sigma = \langle e_1, \ldots, e_m \rangle$, $\rho(\sigma) = [x^1, \ldots, x^n]$, where each feature f^i has an associated value x^i.

When applied for explainable predictive monitoring, the Shapley values for a trace σ are computed over $\rho(\sigma) = [x^1, \ldots, x^n]$, thus resulting in a tuple of Shapley values $\Psi = [\psi^1, \ldots, \psi^n]$, with ψ^i being the Shapley value of feature f^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 observed in the executions recorded in the event-log. Please note that any Shapley value ψ^i can be either positive or negative. A positive or negative value indicates that the feature contributes to increasing or decreasing the value, respectively.

Definition 4.3.2 (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 2.4. We can define the SHAP function $\Psi_{\Phi_{\mathcal{K}}} : E^* \to \mathbb{R}^m$ 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 $\rho(\sigma) \in X_1 \times \ldots \times X_m$.

The computation is the Shapley value is repeated for each trace of $\mathcal{L}$. However, if f^i is numerical, several different values can be observed for f^i, yielding a large number of explanations $f^i = x^i_1, \ldots f^i = x^i_k$. Some of these explanations are equivalent from a domain viewpoint: e.g., $amount = 10000$, $amount = 10050$ might be referring to the same class of amount in a loan application. Therefore, q representative values $w^i_1, \ldots, w^i_q$ are selected out of values $x^i_1, \ldots x^i_k$ (namely, with $q \ll k$) so as to obtain explanations of type $f^i < w^i_1$, $w^i_1 \leq f^i < w^i_2$, …, $f^i \geq w^i_q$. Values $w^i_1, \ldots, w^i_q$ can be obtained taking the boundaries of the buckets obtained via discretization techniques. In particular, our implementation operationalizes a discretization of each feature f^i on the basis of decision/regression as follows. The training set consists of tuple with only two features: f^i used as the independent variable, and the KPI as target/dependent variable. The values observed at the splits of the tree nodes induce the boundaries and, consequently, the buckets.

While an exact computation of the Shapley values requires to consider all combinations of features (hence, the algorithm is exponential on the number of features), efficient estimations can be obtained through polynomial algorithms that use greedy approaches [68].

4.4 Overall Approach for Explaining Generic KPI Predictions

Explanations can be calculated both offline (i.e. on cases that have been already completed) and online (on cases that have not been completed yet). When used offline they are calculated on the test dataset, a part of the dataset not used for training the model (information about the division between train and test sets have been provided in Chapter 3). Moreover, they can be used with different goals; on one hand, explanations for a set of cases can be aggregated, to provide the stakeholder a global picture of what are the features/factors that the trained model uses to make predictions. This type of explanation is called *Global*, and is very important for gaining trust on predictive process monitoring. On the other hand, *Local* online explanations of a certain prediction can be provided, when there is the need to focus on single running cases with a high risk of failing the desired KPI.

4.4.1 Global Explanations

Our global explanation strategy is to provide a bar chart that overviews the importance of each factor influencing the predictive model. The global explanation strategy can be applied both during the offline and the online phase.

In the offline phase, given an event log $\mathcal{L}$, we consider each prefix σ' of each completed trace in $\mathcal{L}$ related to the test set. Then, we compute the explanations as defined in Section 4.3. At this point, for each explanation we consider the vector of all the Shapley values Ψ associated to it and we compute the average, obtaining the average influence of a particular explanation over all the instances; we then sort the explanations by descending influence, allowing us to find the most relevant explanations influencing the predictive model. In this way, not only do we aim to find the most relevant features, but we also report on their contribution to the output of the predictive model.

Figure 4.1 shows an example of our global explanation strategy for the remaining time prediction, represented as a bar chart reporting the most relevant explanations influencing the prediction during the offline phase. The y axis lists different explanations of types $attr = value$, namely a combination of an at-

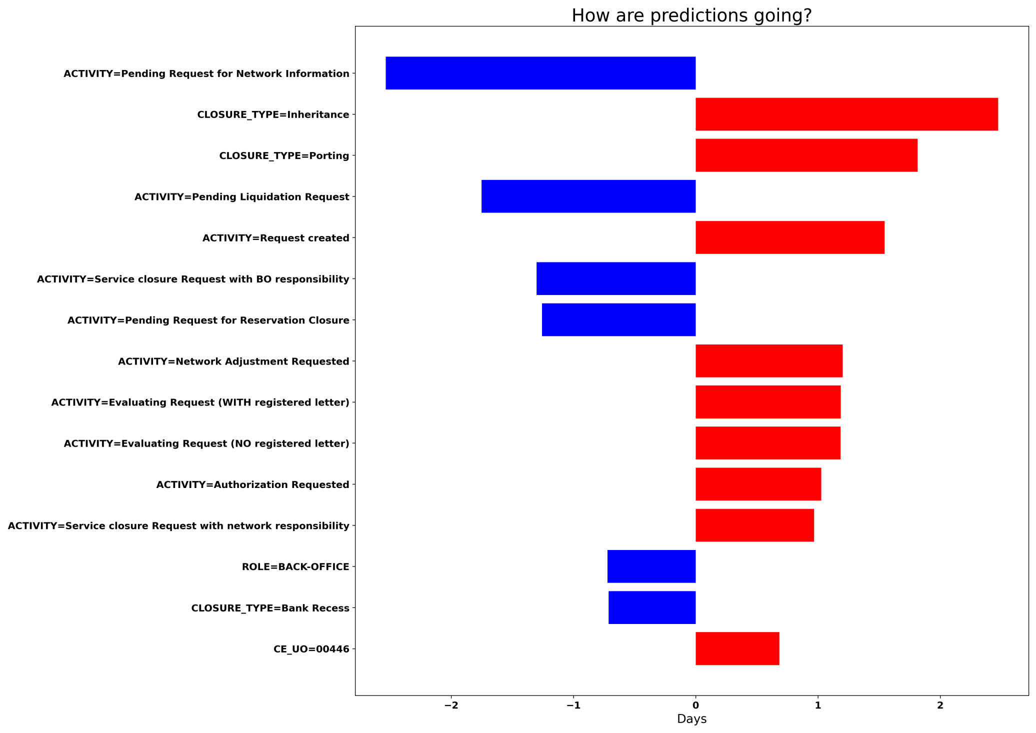


Figure 4.1: Example of global offline explanations for remaining time prediction

tribute with an associated value that was found to be relevant for the predictive model, while the x axis represents the average influence on the predicted time for the considered traces. For instance, let us consider the explanation *Closure_type=Inheritance*, which is associated with a red bar with length greater than 2 days. This indicates that when this association attribute-value (explanation) occurs, then it contributes to increase the KPI value by more than 2 days (i.e. when an instance of a process with Closure type Inheritance occurs, then more time will be needed on average to finish the process).

A similar reasoning can be repeated for global explanations referring to the online phase. For each running case we take the vector of the Shapley values Ψ and associated explanations related to the last observed prefix (which has also the information about previous prefixes); afterwards, for each explanation, we calculate the average value, to obtain the average influence of the explanation over all running cases.

4.4.2 Local Explanations

Our explanation strategy can be also applied in order to generate *Local* explanations of a certain prediction. This is particularly useful when there is the need to focus on single running cases that present a high risk of failing the desired KPI. When we focus on single running cases, we generate a bar chart in the same way we described in the previous section; the only difference is that in order to analyze single running cases, we need to use an interactive tool that enables us to easily perform our analysis. We also considered several alternatives to visualize local explanations, such as representing the explanations in a table or representing them with natural language expressions; however, after an initial interview phase, business process stakeholders indicated to prefer having an uniform view for both local and global explanations.

Further details on the visualization of local explanations will be provided in Section 4.6.

4.5 Explanation evaluation

The previous section showed that the explanations for a learnt prediction model are given as a bar chart, where the y axis lists the different explanations, while the x axis represents the average influence on the considered KPI. However, when predicting a boolean KPI, the values on the x axis represent a probability in the domain $[0 : 1]$; therefore, in order to have a representation of the influence of the explanations that was the same independently from the considered KPI, we shifted the probability in the domain $[-1 : +1]$. As shown in Figure 4.3, this allowed us to give a more clear interpretation of the influence of the explanations.

This section is intended to describe the outcome of our global explanation strategy during the offline phase, in order to show the validity of the explanations provided. In particular, only the explanations of the KPIs related to the bank account closure dataset (cf. Section 3.3) are reported here; additional experiments with different KPIs also on the other publicly-available event logs will be discussed in Appendix A.

4.5.1 Bank Account Closure

For the bank, which deals with the closure of customer's accounts, it is of interest to obtain an estimate of the remaining time until the end for running cases. This allows the bank to decide which cases require special attention, in order to not postpone them too much further. Also, the bank wants to be informed whether there are high chances that one or more of the following activities will occur: *Authorization Requested*, *Pending Request for Acquittance of heirs*, and *Back-Office Adjustment Requested*. They are linked to contingency actions, which should be avoided because they would cause inefficiencies in terms of time, costs, and resource utilization. Finally, the bank is also interested in obtaining an estimate of the total cost of a running case, in order to detect in advance which cases require particular attention. 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.

Global Offline Explanations for Remaining Time Prediction

Figure 4.1 was used in Section 4.4.1 to give an impression on how global explanations are given. Here, we give some more information: it reports on the application of our explainable framework for remaining time prediction. The fact that the closure type is Inheritance (*Closure_Type=Inheritance*) is one of the largest factors that influences the prediction; the color red of the bar and the information that the value is positive (i.e. greater than 2 days) indicates that the influence is towards increasing the remaining time. From a domain viewpoint, when the type of procedure is Inheritance, the bank-account holder is passed away. A further analysis of the data confirms this finding: if the type is *Inheritance*, the process duration is 29 days, versus 14 days when the type is different. Also when the closure type is Porting (*Closure_type=Porting*), the influence is towards increasing the remaining time, and this is also confirmed analyzing the data, since the process duration related to this closure type is 24 days. The evidence in the explanations illustrates that Catboost allowed learning a prediction model that correctly leverages on the closure type to estimate the remaining time.

Other important explanations are related to the activities performed and the resources involved. When activities are performed by a resource director (such as *Authorization Requested*) or when the activity performed is *Network Adjustment Requested* (that only occurs when an error is made in the early stages of the process), the behaviour is considered as exceptional; consequently, the cases usually take longer to complete. This is indicated by the two red bars in the rows *ACTIVITY=Authorization Requested* and *ACTIVITY=Network Adjustment*

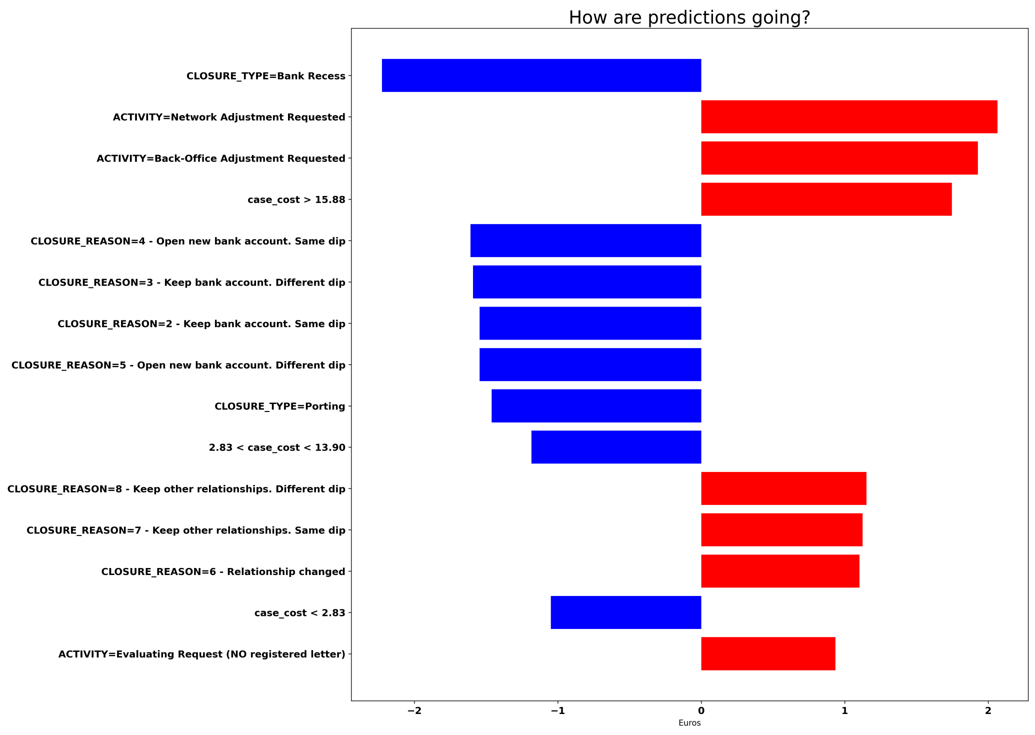


Figure 4.2: Offline explanations for Cost prediction (Bank Account Closure)

Requested, which indicate that the average influence is towards increasing the remaining time by 1 day. Even in this case our framework was able to learn to correctly identify the influencers of the process.

Finally, some explanations are not related to an exceptional behaviour, but rather to the structure of the process; for example, the activity *Request Created* is always performed at the beginning of the process, therefore its influence is towards increasing the remaining time, while explanation *Role=Back-office* is a role associated to resources performing back-office activities, which are generally executed in the final part of the process, thus reducing the remaining time.

Global Offline Explanations for Cost Prediction

Figure 4.2 shows the application of our framework for the case cost prediction. The main factor that contributes to decrease the cost of a case is represented by *Closure_type=Bank Recess*, which is indicated when the closure of the bank account is requested by the bank. The information that the value is negative (i.e. -2 Euros) indicates that the influence is towards decreasing the cost. This is mainly caused by the fact that most of the times here the director does not need to carefully evaluate the request before proceeding and, since the hourly director's wage is certainly higher than that of other bank employees, the predicted case cost will be smaller. The director is similarly not involved when customers, for different reasons, decide to close only one of their bank accounts (labeled respectively as *Closure_Reason=4 - Open new bank account. Same dip*, *Closure_Reason=3 - Keep bank account. Different dip* and *Closure_Reason=2 - Keep bank account. Same dip*), which is a factor that yields lower costs. Another reason is that when only one between different bank accounts of a customer is closed, then the process carried out in the bank is simpler and less Back-office adjustment activities need to be performed compared to when all bank accounts need to be closed, leading to minor costs. The other main factors that contribute to increase the cost of the case are represented by *Activity=Network Adjustment Requested* and *Activity=Back-Office Adjustment Requested*. The information that the values are positive (i.e. 2 Euros) indicates that the influence is towards increasing the cost. This is mainly caused by the fact that these activities should be avoided, since they are only performed when some problem has occurred during the processing of the customer's request and a rework needs to be done, leading to inefficiencies in terms of time, costs, and resource utilization.

Global Offline Explanations for Activity Occurrence Prediction

We mentioned that the financial institute aims to avoid activities related to inefficiencies (e.g. rework), such as *Authorization Requested*, *Pending Request for Acquittance of Heirs* and *Back-Office Adjustment Requested*. The explanations related to the first activity are shown in Figure 4.3. It can be easily seen that the attributes related to the Closure_type and Closure_reason are largely influencing the prediction. In particular, when it is the bank that decides to close the bank account (explanation *Closure_type=Bank Recess*), there is a high probability (0.70) that the director's approval will be necessary; this is also indicated by the red color of the bar, which indicates that the influence is towards predicting that the activity will be performed, and it is confirmed in the data: when the Closure_type is Bank Recess, then the authorization is requested in the 71% of cases (5117 out of 7174). Conversely, when *Closure_type=Porting*, it is highly unlikely that a director's authorization to proceed further is needed, as it is indicated by the blue bar associated with a very high negative probability (-0.80).

Other important attributes are related to the reason triggering the closure's

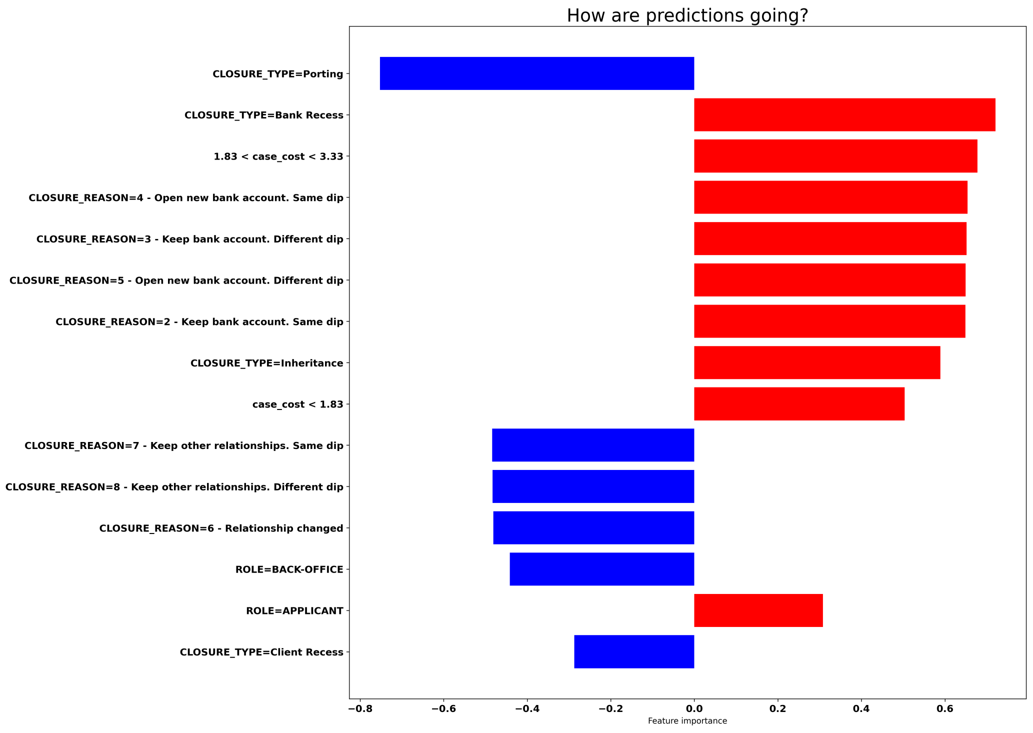


Figure 4.3: Offline explanations for *Authorization Requested* prediction (Bank Account Closure)

request (indicated by the attribute *Closure_reason*). When there is the need to close the old bank account and open a new one in the same department (*Closure_reason=4 - Open new bank account. Same dip*) or when there is the need to keep the same bank account, but transfer the control of it to another department (*Closure_reason=3 - Keep bank account. Different dip*), then there is a high probability that an authorization from the director is needed; these are, in fact, exceptional situations, which need to be managed carefully. A further analysis of the data confirmed that the authorization is needed in the 75% of cases (383 out of 504 cases with the first closure closure) for the first closure reason, and in the 84% of cases (177 out of 209 cases) for the second closure reason.

Finally, when the cost of the case is very low, then the probability to predict that the director's authorization will be needed increases considerably (as it is shown by the red bars associated to the explanations $1.83 < case\_cost < 3.33$ and $case\_cost < 1.83$); this is due to the fact that the authorization to proceed further is usually requested to a director in the early stages of the process, when the cost of the case is still very low.

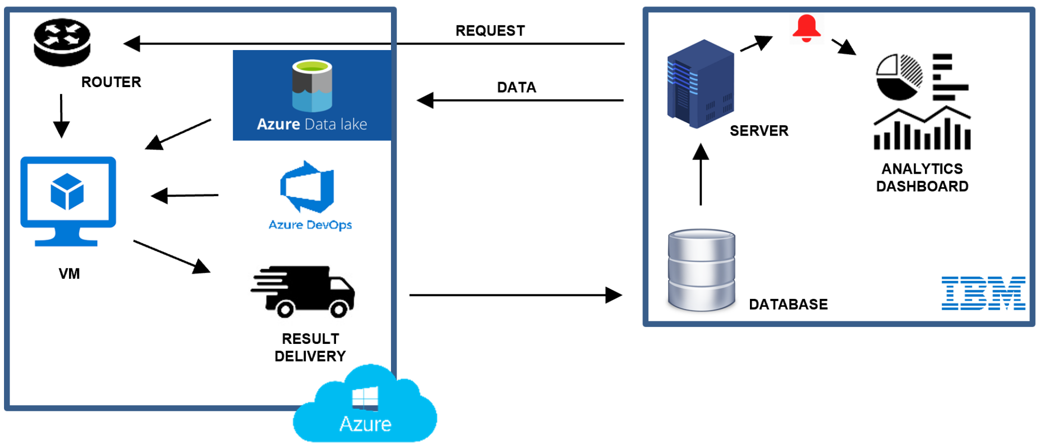


Figure 4.4: High level overview of the main components involved in the explainable decision support system

In order to confirm the findings reported here, we conducted additional experiments with different KPIs also on the other publicly-available event logs, which have been described in Section 3.3. The results are discussed in Appendix A.

4.6 A Software System for Explainable Predictive Process Analytics

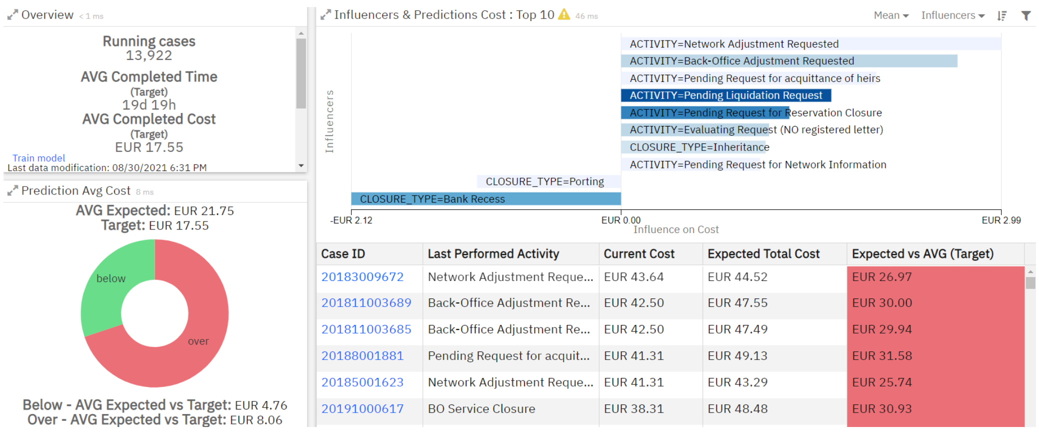
This section describes the integration of the developed explainable predictive framework within the IBM Process Mining suite[1]. This enables us to provide process stakeholders with a ready-to-use module that provisions online operational support for their processes, as well as the influencers driving them, without requiring any specific technical knowledge.

The back-end of the predictive monitoring is based on the Azure infrastructure, which enables us to deploy the technique in the cloud and develop a whole system around it. Figure 4.4 provides a high level overview of the main components involved in the explainable decision support system. IBM Process Mining is represented on the right side, while the Azure infrastructure (also called Machine Learning Platform) hosting our framework is shown on the left. The latter is in charge of processing the requests coming from IBM Process Mining, preparing a compute instance to execute our framework, and delivering the results back. In

[1]https://www.ibm.com/it-it/cloud/cloud-pak-for-business-automation/process-mining

particular, the Machine Learning Platform has been tested to work with datasets up to 10 million events, it can handle multiple requests coming from different users and, in case a customer requests it, multiple compute instances can be easily provided by allocating new clusters, enabling to scale on demand.

The router is the component responsible to process the incoming requests and verify if the domain and the user related to the request are authorized; if the check is positive, the Process Mining suite uploads the event log on the Azure Datalake, which is the component responsible for storing and archiving the data. Once the data has been fully uploaded, the router forwards the request to another component (here shown as Virtual Machine (VM)), which is responsible to create a virtual machine with the necessary computational power and copy the event log data inside the virtual machine; at this point, the code of our explainable predictive framework (which is located in a repository inside Azure DevOps component) is downloaded inside the virtual machine, where it will be executed. The output produced by our explainable predictive framework executed in the VM component will be stored inside the Azure Datalake and then sent to the database of the Process Mining suite via the Result Delivery component. Finally, the results will be shown in the Analytics dashboard via the Application server.



Overview < 1 ms
Influencers & Predictions Cost : Top 10 46 ms
Mean
Influencers
Train model
Prediction Avg Cost 8 ms
Influencers
ACTIVITY=Network Adjustment Requested
ACTIVITY=Back-Office Adjustment Requested
ACTIVITY=Pending Request for acquittance of heirs
ACTIVITY=Pending Liquidation Request
ACTIVITY=Pending Request for Reservation Closure
ACTIVITY=Evaluating Request (NO registered letter)
CLOSURE_TYPE=Inheritance
ACTIVITY=Pending Request for Network Information
CLOSURE_TYPE=Porting
CLOSURE_TYPE=Bank Recess
-EUR 2.12
EUR 0.00
EUR 2.99
Influence on Cost
below
over

Running cases
13,922
AVG Completed Time
(Target)
19d 19h
AVG Completed Cost
(Target)
EUR 17.55

Last data modification: 08/30/2021 6:31 PM

AVG Expected: EUR 21.75
Target: EUR 17.55

Below - AVG Expected vs Target: EUR 4.76
Over - AVG Expected vs Target: EUR 8.06

Case ID	Last Performed Activity	Current Cost	Expected Total Cost	Expected vs AVG (Target)
20183009672	Network Adjustment Reque...	EUR 43.64	EUR 44.52	EUR 26.97
201811003689	Back-Office Adjustment Re...	EUR 42.50	EUR 47.55	EUR 30.00
201811003685	Back-Office Adjustment Re...	EUR 42.50	EUR 47.49	EUR 29.94
20188001881	Pending Request for acquit...	EUR 41.31	EUR 49.13	EUR 31.58
20185001623	Network Adjustment Reque...	EUR 41.31	EUR 43.29	EUR 25.74
20191000617	BO Service Closure	EUR 38.31	EUR 48.48	EUR 30.93

Figure 4.5: The IBM Analytics dashboard for Explainable Predictive Process Monitoring.

Figure 4.5 shows a screenshot of the *Analytics Dashboard* within the IBM Process Mining suite for prediction of the total cost of cases, namely the cost necessary to complete a case. The use case presented here is related to the Bank Account Closure process process, which deals with the closure of customer's accounts, which may be requested by the customer or by the bank, for several

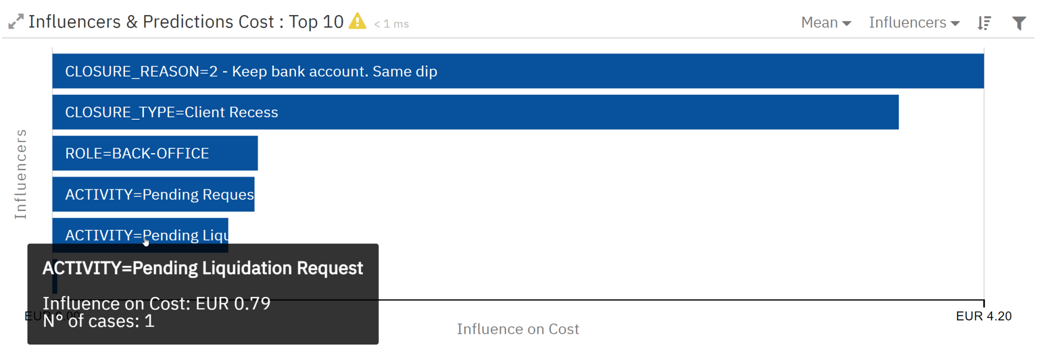

Influencers & Predictions Cost : Top 10
< 1 ms
Mean
Influencers
Influencers
CLOSURE_REASON=2 - Keep bank account. Same dip
CLOSURE_TYPE=Client Recess
ROLE=BACK-OFFICE
ACTIVITY=Pending Reques
ACTIVITY=Pending Liqu
ACTIVITY=Pending Liquidation Request
Influence on Cost: EUR 0.79
N° of cases: 1
EUR 4.20
Influence on Cost

Figure 4.6: Explanations related to one running case. An explanation is visible by passing over with the mouse when a corresponding bar is too short.

reasons.

The upper-left corner reports on general process statistics, such as the number of running cases and the average case total time (here labeled as *Completed Time*) and cost. In the bottom-left corner, the widget shows how many running cases are predicted to cost too much (indicated by the red portion of the pie chart) and what is the average foreseen overcost for those cases. The bottom-right corner lists the running cases, each associated with the case identifier and the last performed activity; since this dashboard refers to the process total cost, each case is also associated with the current cost, the expected total cost as forecasted by the predictive monitor, and its difference wrt. the average completion cost, here also named as target. When a user clicks on a specific running case (e.g. with id 201811010127), it is possible to see the *Local* explanations for that case, named *influencers* in the tool (see Figure 4.6).

Let us consider again the all-cases dashboard in Figure 4.5: the bar chart in the top-right corner provides an helicopter view of the explanations (*Global*). In particular, each row of the bar chart represents an explanation, and extends towards left or right, depending whether the average Shapley value for the explanation is negative or positive. The colour indicates the frequency of an explanation, with darker colours indicating a large number of running cases with that explanation.

As an example, explanation *ACTIVITY=Network Adjustment Requested* has a large bar with a light colour: this means that, for a small number of cases, the fact that the latest activity has been a *Network Adjustment Requested* has contributed to reduce the predicted total case cost by an average value of 2.99 Euros. The explanation *CLOSURE_TYPE=Bank Recess* is conversely associated with a darker colour, namely with a large number of cases. The average shapley value is equal to -2.12 Euros: when the closure of the bank account is requested directly by the bank, the total cost reduces by 2.12 Euros wrt. the average.

4.7 Evaluation of the Human Understandability of the Predictions and Explanations

Chapter 3 has shown an evaluation of the quality of the predictions, while Section 4.5 has shown an evaluation of the quality of the explanations produced by the aforementioned framework. However, it needs to be ultimately used by process analysts, for whom it must be effective and comprehensible. To this aim, we carried out an empirical user evaluation of the explanations produced by our explainable predictive monitoring framework. In particular, for the user evaluation we involved real and potential process analysts, in order to assess whether or not they would understand the explanations and, consequently, trust the whole explainable predictive monitoring framework. The integration between our framework and the IBM Process Mining (which has been described in the previous section) enabled us to perform a complete user evaluation: we did not only analyze the results produced by our framework in isolation, but we also considered the context in which these results were produced.

Section 4.7.1 reports on the methodology carried out for the user evaluation, while Section 4.7.2 discusses the feedback obtained by the process analysts who participated in the user evaluation.

4.7.1 Methodology of the evaluation

A user study was run to test the usability of the Explainable Predictive Analytics module and the users' experience. Below, we report on the characteristics of the users that participated in the experimental session and we describe the experimental settings and the questionnaires that have been adopted to carry out the evaluation session.

Participants

Twenty users were involved in the case study, equally split between males and females, with an average age of 31.20 and Standard Deviation (SD) of 8.43. On average, they had an educational level (years of school) of 18.15 years, with a standard deviation of 1.42. Out of the 20 users, five were Master's or PhD students in Data Science or a related field, 13 were employees in an international IT services company, and two were both students and employees of the same company. Eleven participants were Italian, while nine were not. Experiments were conducted in Italian or English according to the favourite language of each subject.

All participants had some previous knowledge about Process Mining: students had previously taken a course on Process Mining, while employees had already applied Process Mining in their analysis work. Indeed, we asked them about self-evaluating their process mining knowledge on a 5-point Likert Scale (with 1 = superficial knowledge, 2 = basic knowledge, 3 = medium knowledge, 4 = good knowledge, 5 = advanced knowledge), reporting an average score of 3.5 (SD = 1.36). Eighteen out of 20 users reported a previous experience with process mining tools, with an average use frequency of 3.39 and SD of 1.14 (the scale points were 1 = almost never, 2 = rarely, 3 = sometimes, 4 = often, 5 = always). However, none of them had previous experience with the specific module being analyzed.

Materials and Methods

The user study was designed in accordance with the Declaration of Helsinki and approved by the ethics committee for psychological research at the University of Padova (protocol number 4259). All participants were required to read and provide informed consent before starting the experiment. After providing demographic data and some information related to their previous knowledge of process mining, users were presented with an 8-minute video explaining the main structure and functionalities of the Explainable Predictive Analytics module. Then, participants were asked to complete 18 tasks within the module. The tasks were aimed at testing the intelligibility of the information provided by the Explainable Predictive Analytics module regarding the process related to the closure of the customers' bank accounts. Example of tasks are: "According to the prediction of the algorithm, how many cases will cost more than the average?", "What is the most frequent influencer affecting cost prediction?". The complete list of the tasks is reported in Table 4.1.

After each task, participants were asked about the difficulty of the task, which was measured according to the 1-item questionnaire proposed by Tedesco and Tullis ("Overall this task was", 1 = very easy, 2 = easy, 3 = neither easy nor diffi-

Table 4.1: List of the tasks completed by the users within the Explainable Predictive Analytics module. The third and fourth columns report respectively the average (and Standard Deviation) accuracy obtained by participants in completing the task and the evaluation of the task difficulty according to the 1-item questionnaire by Tedesco and Tullis [105]

.

N.	**Task**	**Accuracy (Average, SD)**	**Difficulty (Average, SD)**
1	In the Overview widget, what is the meaning of AVG Completed Cost?	0.83 (0.37)	2.10 (0.79)
2	According to the prediction of the algorithm, how many cases will cost more than the average?	0.95 (0.22)	2.25 (0.85)
3	Focus now on case 20183009672. What is the current case cost?	1.00 (0.00)	1.60 (0.68)
4	Always consider the case 20183009672. According to the algorithm, what will be the total cost of the case?	1.00 (0.00)	1.80 (0.77)
5	In “Influencers & Predictions Cost” widget, what does the column “Expected vs AVG (Target)” represent?	0.70 (0.41)	2.75 (1.02)
6	Click on the practice 20183009672 and write below which is the influencer, relative to the costs, most significant for this case.	0.95 (0.22)	2.70 (0.57)
7	Still in relation to the 20183009672 case, does the most significant influencer raise or lower the prediction of the cost of the case?	0.90 (0.31)	2.25 (0.79)
8	Still in relation to the 20183009672 case, how much does the most significant influencer raise the prediction of the cost of the case?	0.90 (0.31)	2.10 (0.97)
9	How much, on average, is the cost of the process influenced by the Closure type “Bank Recess”?	0.88 (0.28)	2.60 (0.88)
10	How many cases are influenced by the ”Bank Recess” influencer?	1.00 (0.00)	1.95 (1.00)
11	Does the fact that a Network Adjustment Requested (an activity that consists of reworking the file following an error) has been carried out in the process influence the prediction of the total cost?	0.95 (0.22)	2.10 (0.72)
12	How much is the total cost prediction affected by a ”Network Adjustment Requested”?	0.85 (0.37)	1.85 (0.88)
13	What is the most frequent influencer affecting cost prediction?	0.70 (0.47)	2.45 (1.00)
14	Order Influencers bar chart for median influence instead of mean influence. Look at the graph sorted by the median. What is the explanation that most influences cost prediction?	0.90 (0.31)	2.80 (1.15)
15	Order Influencers bar chart again for mean influence. What information provides the ACTIVITY = Back-Office Adjustment Requested histogram?	0.75 (0.30)	2.50 (0.76)
16	In the ”Influencers & Predictions Cost” widget, filter for the influencer relating to the CLOSURE_TYPE Bank Recess. What are the two influencers that best explain cost prediction when the CLOSURE_TYPE is Bank Recess?	0.80 (0.38)	3.10 (0.91)
17	View influencers related to case 20183009672. Focus on the influencers related to the case 20183009672. Explain what information the first histogram provides (Current Cost > 18.82).	0.58 (0.37)	3.55 (1.15)
18	Change the settings, setting the display of ”Completed cases”. In this new view of completed cases, who is the most frequent influencer?	0.80 (0.41)	2.55 (1.10)

cult, 4 = difficult, 5 = very difficult). Previous literature reports this questionnaire to be the most reliable on small subject’s sample in terms of its correlation with performance measures, if compared with other questionnaires on task difficulties [105]. Finally, at the end of the evaluation session, the users filled out two questionnaires for the assessment of usability and user experience [37, 88]:

- Post-Study System Usability Questionnaire (PSSUQ) [52]: it is a 19-items questionnaire that assesses user satisfaction with system usability. In addition to the overall satisfaction score, the PSSUQ provides sub-scores for three specific dimensions: System Usefulness (SysUse), Information Quality (InfoQual), Interface Quality (InterQual). Responses are given on a 7-point Likert Scale from 1 = strongly agree to 7 = strongly disagree.

- User Experience Questionnaire (UEQ) [50]: it includes 26 items testing the users’ experience with the product they interacted with. Items are made up of pairs of opposite adjectives (e.g., “impractical – practical”, “boring

– exciting”), with a response scale on seven points. The UEQ measures both pragmatic and hedonic components of the user experience, providing a score for six different dimensions: Attractiveness (overall impression of the product: do users like or dislike the product?), Perspicuity (is it easy to get familiar with the product? Is it easy to learn how to use the product?), Efficiency (can users solve their tasks without unnecessary effort?), Dependability (does the user feel in control of the interaction), Stimulation (is it exciting and motivating to use the product?), Novelty (is the product innovative and creative? Does the product catch the interest of users?) [90].

The literature proposes normative data for these questionnaires [53]. This has allowed us to derive sound conclusions on the goodness of the numerical results obtained. For PSSUQ, we considered satisfactory the scores that are in or above the normative range reported by Lewis [53]: the mean of the overall satisfaction rating is 2.82 with a 99% Confidence Interval (CI) ranging from 2.62 to 3.02. More specifically going into the three dimensions, the means for the subscales are 2.80 (99% CI 2.57–3.02) for System Usefulness, 3.02 (99% CI 2.79–3.24) for Information Quality, and 2.49 (99% CI 2.28–2.71) for Interface Quality.

For UEQ, an interpretation of the level of satisfaction of the outcome can be achieved by comparing the obtained scores with those of a benchmark data set [89], which contains data from 20190 users from 452 studies concerning different products (business software, web pages, web shops, social networks). In particular, the comparison with the benchmark dataset allows us to qualitatively classify the product for each analyzed dimension as:

- Excellent: the score obtained for the evaluated product is in the range of the 10% best results.
- Good: 10% of the products in the benchmark data set have a better score, while 75% of the products are worse.
- Above average: 25% of the products in the benchmark have a better score than the score obtained for the evaluated product, while 50% of the products are worse.
- Below average: 50% of the products in the benchmark have a better score than the score obtained for the evaluated product, while 25% of the products are worse.

- Bad: the score obtained for the evaluated product is in the range of the 25% worst results.

4.7.2 Evaluation results

This section discusses the results of the usability study, which has focused on four dimensions: accuracy to carry out the tasks, perceived task difficulty, usability of the module and user experience.

Task Accuracy

The task accuracy was calculated as the percentage of tasks correctly fulfilled by the users. For each task, one point was assigned if the task was completed correctly, 0 when the user failed, and 0.5 points if the user's response was partially correct; as an example, 0.5 points were assigned when the user had to find the two main influencers for a particular case, but only one was indicated.

On average, users obtain an accuracy of 0.86 (with SD of 0.11), with the 17 being the most failed task (0.58). The accuracy for each task is reported in the third column of Table 4.1. This level of accuracy can be reasonably considered good; since most of the tasks requires one to leverage on the prediction's explanations, we can conclude that **explanations are generally comprehensible to correctly carry out analysis' tasks**. One should also take into account that users were confronted for the first time with the idea of explainable predictive process analytics and with its operationalization in the IBM Process Mining software suite.

As shown in Table 4.1, tasks 5, 13, 17 have the lowest scores, respectively 0.70, 0.70, 0.58.

Task 5 investigated whether the users could understand the meaning of the column *Expected vs AVG (Target)*, i.e. the last column of the Table in Figure 4.5. In particular, the value for a certain process instance (i.e., case) shows the difference between the predicted value for that process instance (shown in the same Table under the column *Expected Total Cost*) and the actual observed average cost (which is shown in the Overview widget in the top-left corner). The low accuracy for task 5 is likely related to the fact that this value is calculated using the information contained in two different widgets. We plan to show a tooltip explaining the meaning of this column. However, it is worthwhile observing that this task is not related to using the explanations, but is rather related to the visualization of the prediction.

Conversely, task 13 focuses on whether or not the users could correctly indicate the most frequent explanation affecting the cost prediction. An accurate answer to this task should be obtained by inspecting the colours of the bars of the different explanations and focusing on those with darker colours, which mean greater frequencies. Note that the meaning of coloring is consistent with other modules of the IBM Process Mining suite (e.g., in a process model, darker colours are given to the activities that occur more frequently). Indeed, breaking the users down in two groups, composed respectively by users familiar and unfamiliar with other parts of the IBM Process Mining suite, we observed that task 13 was generally answered incorrectly by subjects that never used IBM Process Mining before (among the 6 users that answered incorrectly, 5 users out of 6 never used IBM Process Mining before).

Task 17 had the lowest accuracy, whose perceived difficulty was certainly the highest (cf. last column of Table 4.1). This was reported in some of the user's feedback: "*It was not clear the meaning of the influencer related to the Current Cost, if this is the cost then the influence on the cost is...*". The Current Cost is an attribute that encodes the cost of a case after each event occurred in a process; it is an attribute that is not part of the original event log, but it has been added in order to increase the predictive accuracy of the model.

To address the issue with this type of task, we plan to show in separated views explanations related to process attributes already present in the original event log and explanations related to encoded attributes not present in the original event log (such as the cost). This also came as a comment from one of the users in the feedback: "*For me it was difficult to understand the relations between the influencers related to process attributes and those related to the cost; I would rather prefer to analyze them separately*"). It is also rather important to facilitate users to interpret the semantics of the encoded attributes: a tooltip could then be useful to explain their semantics.

Task Difficulty

Last column of Table 4.1 shows the perceived task difficulty for each of the tasks. The average task difficulty is 2.39, with standard deviation of 0.41. This means that most of the responses given by participants fall in the range between "easy" and "neither easy nor difficult", leading us to conclude that **users have found most of the tasks reasonably easy**.

We also illustrate a graphical distribution of the responses for each task in

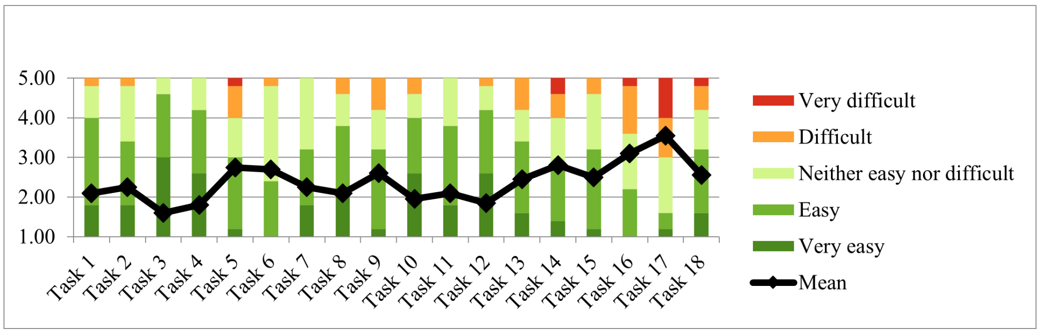


Figure 4.7: Distribution of the responses in the 1-item questionnaire on task difficulty for each task. For each task, the height of the several boxplots varies depending on the percentage of users reporting the corresponding task difficulty.

Figure 4.7. Consistently with what was observed in the accuracy assessment described in Section 4.7.2, the task perceived as the most difficult was the task 17.

Usability

The analysis of the results of the Post-Study System Usability Questionnaire (PSSUQ) is summarized in Figure 4.8: the bars are the values obtained in our evaluation, while the depicted ranges show the normative values with confidence interval (cf. Section 4.7.1). The overall satisfaction score of 2.83 (with SD of 1.24) is within the normative range, thus testifying a good level of user satisfaction of the interface. In the three subscales, participants obtain the following scores: System Usefulness = 2.70 (SD = 1.23), Information Quality = 3.07 (SD = 1.61), Interface Quality = 2.82 (SD = 1.38); these scores fall within the normative range as well.

In other words, **no critical issues emerged regarding the system usefulness, the quality of the information provided by the system, and the quality of the interface (i.e. no major issues regarding the intelligibility of the explanations were found).**

User Experience

As concerns the User eXperience (UX), the six dimensions measured by the UEQ have obtained the following scores: Attractiveness = 1.45 (SD = 1.06), Perspicuity = 1.24 (SD = 1.20), Efficiency = 1.71 (SD = 1.16), Dependability = 1.49 (SD = 0.94), Stimulation = 1.55 (SD = 0.99), Novelty = 1.10 (SD = 1.26). Comparing

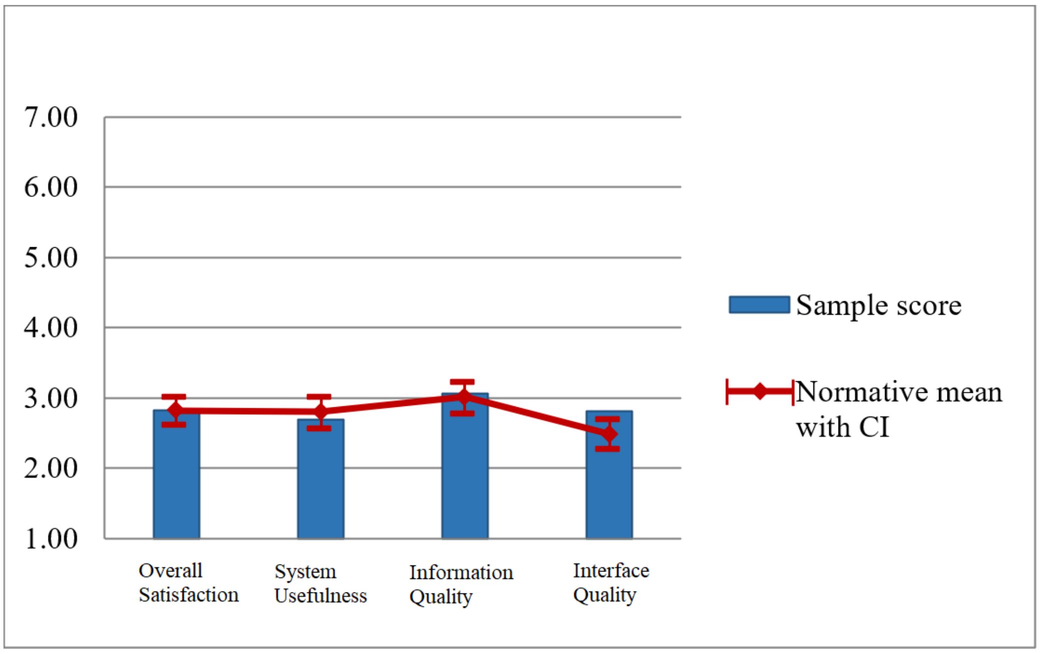


Figure 4.8: Average scores in PSSUQ subscales and their collocation compared to the normative range (mean; CI). Responses are given on a 7-point Likert Scale from 1 = strongly agree to 7 = strongly disagree. Ideally, the scores should be the lowest possible.

these scores with those of the benchmark data set (cf. Section 4.7.1), it should be noted that the Explainable Predictive Analytics module is evaluated as good for Efficiency, Dependability, and Stimulation (see Figure 4.9). It is above average compared to other products for Attractiveness and Novelty. Instead, Perspicuity is just below average; this could be due to the fact that most of the users were not particularly familiar with the adopted process mining tool. Therefore, in order to improve the Perspicuity of the Explainable Predictive Analytics module, it could be important in the future to plan a training phase beforehand, in which users get familiar with the process mining tool. In 4.10, the distribution of the responses for each item of the questionnaire is reported.

Summarizing, we conducted a rigorous user evaluation to understand if process analysts (both from the academia and the industry domains) were comfortable with the results returned by our explainable predictive monitoring framework. Several points emerged from our study; we summarize the main findings here.

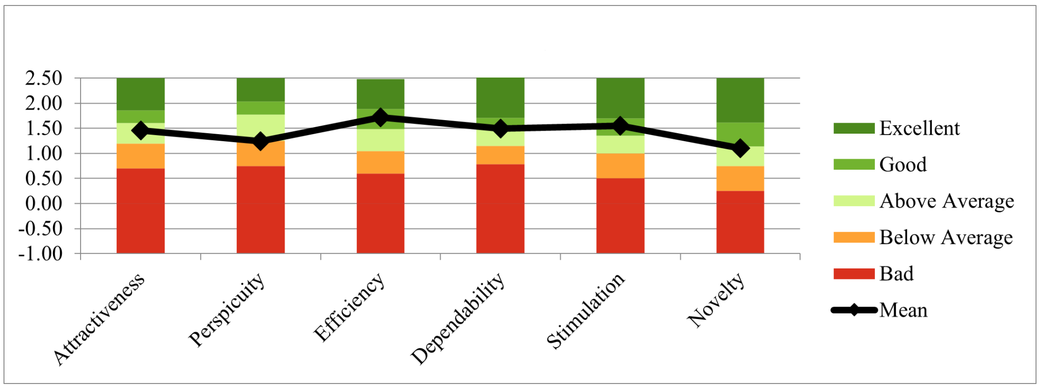


Figure 4.9: Average scores obtained by the users in the UEQ subscales (represented by the black line), and their collocation with respect to other products of the benchmark data set (represented by the coloured blocks). Note that “good” means that 75% of products of the benchmark data set perform worse; “above average” means that 50% of products of the benchmark data set perform worse; “below average” indicates that 25% of the products of the benchmark data set perform worse.

4.7.3 Concluding Remarks

The user study has showed that overall the users achieved a good score and could find the answer for most of the questions without particular problems: cf. an average accuracy of 0.86 out of 1, and an average, perceived task difficulty of 2.39 out of 5. Considering that most of the tasks focus on the explanations, this shows that **predictions are actually explained in a form that is effective and efficient for process analysts**. However, the lower accuracy observed in some tasks has highlighted points for improvement, but these improvements are not related to prediction’s explanations, except for task 17 where users did not well understand explanations that relate to features that are not directly mapped to process’ attributes. However, the problem could likely be overcome by adding a tooltip that indicates the semantics of these additional features.

The Post-Study Usability Questionnaire pointed out that the usability of the explainable predictive process analytics framework and of its operationalization was good, beyond the minimum usability level that literature indicates as suitable. This confirmed that **the proposed explainable framework was considered intelligible by the users**.

Last but not least, the User Experience Questionnaire also showed levels of user satisfaction with the framework that literature consider as appropriate:

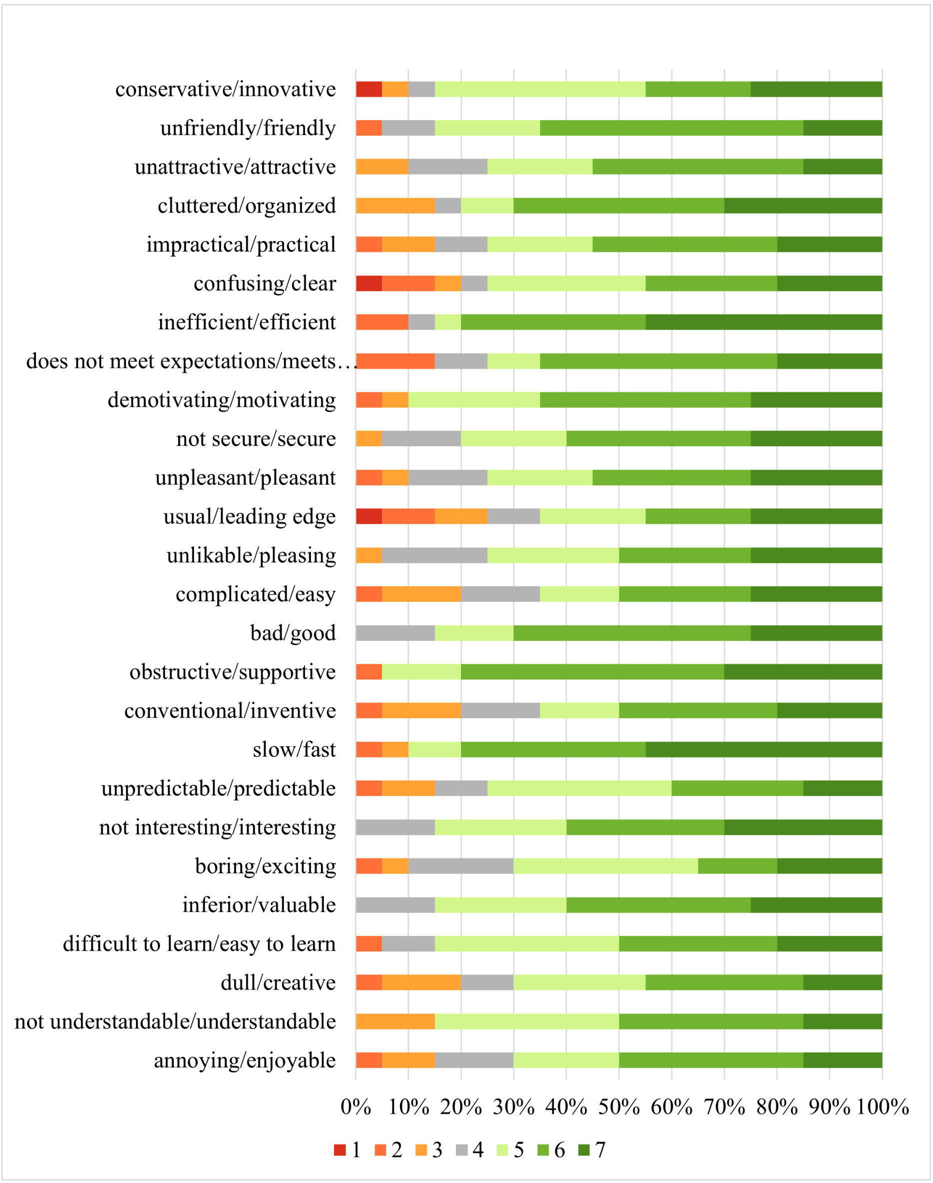


Figure 4.10: Distribution of the responses for each item of the User Experience Questionnaire (UEQ). Each row is an item and reports pairs of opposite adjectives, which are mapped to a response scale on seven points. For each row, the width of the several boxplots varies depending on the percentage of users reporting the corresponding score.

this indicates that **users were satisfied with the explainable predictive process framework**. A slight lower value, still within the boundaries, was observed about perspicuity: this is likely related to the fact that users were confronted for the first time with the framework, and a short training phase beforehand would be beneficial.

4.8 Summary

A lot of research has been devoted towards increasingly accurate frameworks for predictive process monitoring. Moreover, in the last years, growing attention has been paid to ensure that the resulting predictive-monitoring system is workable in practice. With practical workability, here we intend that the process analysts and stakeholders need to trust the system and its predictions. Previous studies have already shown that a necessary condition to build trust is to explain the reason of the provided predictions; proposals that do not put explanation as a core feature are not going to be adopted in practice.

Several works have recently put forward methods to explain the predictions, using a plethora of different techniques (see Section 4.2). However, as also stated by Stierle et al., none of these works verified whether prediction's explanations are provided to process analysts in a form that is intelligible and that can give them actionable insights easily, effectively and efficiently. Furthermore, most of the frameworks have stopped at a prototype phase and have not been deployed in real software suites. Rizzi et al. [82] were the first that previously attempted to conduct some user studies on explainable predictive process analytics, but this attempt was limited to 8 subjects, and without using a commercial, fully-fledged operationalization.

This chapter attempts to address the issues above: assessing the quality of prediction explanations with users and developing the framework into a commercial software. In particular, the chapter's contributions are the following:

1. A framework for explainable predictive process analytics has been introduced. Afterwards, the outcome of our global explanation strategy has been illustrated, in order to show the validity of the explanations provided; in particular, after analyzing the data, the evidence in the explanations demonstrated that Catboost allowed learning a prediction model that leveraged attributes that were found to be relevant from a domain viewpoint.

2. The explainable predictive process framework has been implemented as a module of a commercial software, the IBM Process-Mining software suite (cf. Section 4.6).

3. A user study has been conducted on the IBM Process-Mining suite to assess the efficiency and effectiveness of the proposed explainable predictive process analytics module. Results have shown that, indeed, the module and the form in which explanations are provided are efficient, effective, usable, and satisfactory from a final-user viewpoint (see Section 4.7).

As future work, we aim to perform an assessment against the quality evaluation criteria for Explainable AI introduced in literature (see, e.g. [56]). Moreover, as stated in Section 4.2, many different explainable frameworks have been proposed in these years; it would be interesting to compare the different explanations that would be produced by different (model-agnostic) explainable frameworks for a given trained predictive model, and evaluate the quality of the explanations with users that have a knowledgeable business understanding about the process, possibly empirically identifying strength and weaknesses of the different approaches. Finally, while the explainable framework presented in this chapter focuses on explaining the provided predictions, it can be extended from predictive to prescriptive analytics. In the latter case, the framework needs to suggest which activities to perform as next (and explain why) in order to recover those cases that, otherwise, are predicted to not meet satisfactory KPI values. This extension will be illustrated in Chapter 6. In the next chapter, we aim at developing a framework for object-centric predictive analytics, and enhancing the accuracy of the predictive model by including the information about object interactions. We also leverage on explainable AI techniques to further confirm the importance of object-interaction for improving the prediction quality.

Chapter 5

Object-centric Predictive Process Analytics

Object-centric processes are recently gaining popularity in academia and industry, because their nature is observed in many application scenarios. They are an implementation of a paradigm where an instance of one process is not executed in isolation but interacts with other instances of the same or other processes. Existing research is unable to directly exploit the benefits of these interactions, thus limiting the prediction quality.

Therefore, in this chapter, an approach is first proposed to flatten object-centric event logs (i.e., event logs of object-centric processes) into single-identifier event logs (i.e., event logs of single-flow processes). This enables us to perform predictive analytics in object-centric processes. Afterwards, this chapter proposes an approach to enrich the single-identifier event logs in order to maintain a meaningful abstraction of the object interactions, and the approach is compared with a naïve approach that overlooks the object interactions, illustrating the benefits of their use on the prediction quality. Moreover, this chapter reports on the experience of comparing the quality of the predictions obtained by leveraging four different techniques to tackle predictive analytics in object-centric processes. In particular, one technique is based on Gradient Boosting on Decision Trees, and three are based on Deep Neural Networks, including graph-based models. The four techniques were empirically evaluated on event

logs related to three real object-centric processes and more than 30 different KPI definitions. The experimental results show that the technique based on Gradient Boosting performs consistently better than those based on Deep Neural Networks, both in terms of accuracy and training time, and that considering the object interactions often improves the quality of the predictive model. Finally, Shapley Values and Explainable AI techniques are leveraged to further confirm the importance of object-interaction and aggregated features for improving the prediction quality.

5.1 Motivation

Object-centric processes are recently gaining popularity in academia and industry, because their nature is observed in many application scenarios. They are implementations of a paradigm where an instance of one process is not executed in isolation but interacts with other instances of the same or other processes. These processes for the different objects run independently and synchronize through some bridging events to exchange data needed to progress further.

The IEEE Task Force on Process Mining has recently published a survey with academics, practitioners, consultants and vendors in Process Mining that has shown the importance of object-centric process approaches in the domain of Process Mining [124]. In particular, only 33% of the respondents has indicated to be a minor problem to be forced to analyze processes as if they are composed of a single execution flow (namely with a single case identifier). As mentioned, an object-centric process model removes this limitation, allowing for parallel executions of different sub-processes that periodically synchronize.

This chapter focuses on object-centric predictive process analytics. This is a challenging problem due to the complex intricacy of the process instances that relate to each other via many-to-many associations. Existing research is unable to directly exploit the benefits of these interactions, thus limiting the prediction quality. Moreover, current research in predictive process analytics often relies on the assumption that the instances refer to a single process. Therefore, in this chapter, an approach is proposed to flatten object-centric event logs (i.e., event logs of object-centric processes) into single-identifier event logs (i.e., event logs of single-flow processes), which are enriched to maintain a meaningful abstraction of the object interactions. The complex object interaction is unfolded in traces; in a nutshell, the core idea is that a trace is created for each cluster of events that are connected to each other (i.e. that share an object identifier, either directly or transitively). As an example, let us suppose that an order object with identifier `o1` (representing the life-cycle of the process related to an order purchased by a customer) is associated with multiple receipts `r1, r2` and `r3` (i.e. the items purchased by the customer and associated with the order `o1` have been splitted in multiple deliveries, `r1, r2` and `r3`). The association is established via the so-called bridge events, which contain `o1, r1, r2` and `r3` as object identifiers. Consequently, the order object `o1` is directly associated to the receipt objects with identifier `r1, r2` and `r3`, and the events containing the aforementioned identifiers will be part of the same trace. Afterwards, let us suppose that each of the three items received (identified by the three different receipt identifiers) is payed separately in different invoices, with identifiers `i1, i2` and `i3`. The events containing these invoice identifiers among the different identifiers will be part of the same trace as well, since they are directly related to the receipts with identifiers `r1, r2, r3` and transitively related to order `o1`. Finally, the events will be temporally ordered according to their timestamps. A graphical representation will be illustrated in Section 5.3.1. When the complex interaction is unfolded in a multiset of traces, we can specialize the current state of the art in predictive analytics.

After specializing predictive analytics for object-centric event logs, this chapter reports on the experience of comparing the quality of the predictions obtained by leveraging four different techniques to tackle predictive analytics in object-centric processes. In particular, one technique is based on Gradient Boosting on Decision Trees, and three are based on Deep Neural Networks, including graph-based models. Moreover, when flattening the event log, our first approach for object-centric process analytics does not consider the information about the attributes associated with correlated objects, and their respective value. Neither does it consider the number of objects involved in the execution of a process. This information can improve the quality of the predictions. For instance, the total processing time of a requisition might depend on the number of orders associated with the requisition. Feeding this information into the prediction model can increase the prediction's accuracy, illustrating that the complex interactions of object-centric processes need to be taken into account when predicting, as this allows to consistently improve the predictive performances over simpler techniques. Therefore, when flattening the event log, a second approach is additionally proposed for object-centric predictive process analytics, where the information about object interactions is explicitly encoded as features in the predictive model, with the aim of increasing the predictive model accuracy. Note, however, that the inclusion of this information might not always improve the accuracy, due to overfitting phenomena. The first approach without this additional information is still beneficial on its own when overfitting is detected.

To reasonably generalize the drawn conclusions, the four predictive techniques were empirically evaluated on event logs related to three real object-centric processes, and 30 different KPI definitions. The experimental results illustrate that the technique based on Gradient Boosting performs consistently better than those based on Deep Neural Networks, both in terms of accuracy and training time, and that considering the object interactions often improves the quality of the predictive model. The Gradient Boosting technique guarantees higher prediction quality, while the training time is several orders of magnitude lower than those based on Deep neural networks. In sum, the flattening abstraction retains the important information to guarantee high accuracy, and even prevents some degree of overfitting observed for graph neural networks.

Section 5.2 provides an overview of our contributions with respect to the state of the art, while Section 5.3 presents our proposal to enable predictive analytics in object-centric processes; furthermore, it reports on the results of the empirical evaluation of our predictive monitoring framework for object-centric processes, comparing the results obtained with different predictive techniques. In Section 5.4, we leverage Shapley Values and Explainable AI techniques to assess the importance of object-interaction and aggregated features. Finally Section 5.5 concludes this chapter.

5.2 Related Works

A body of research exists on object-centric processes. Several research works focus on modelling object-centric processes (e.g. [13]) and the verification of the correctness of these models [12, 69]. In the realm of Process Mining, techniques are proposed to discover object-centric process models and behavioral dependencies between objects (i.e. artifacts) [57, 77, 112, 113], and to tackle the problem of the object-centric process conformance checking [2, 27]. However, none of the existing works consider object-centric predictive process analytics.

Predictive analytics (and the approach proposed in this chapter) is generally applicable to any domain where concepts of process executions, namely sequences of executions of activities/steps of any nature that brings from the initial to a final state, can be identified. However, the case studies reported in this chapter refer to business processes, which indeed triggered this line of research. The idea proposed here could be easily extended to processes in other domains. In [35], e.g., predictive process analytics is applied to forecast attacks to IoT systems. However, traditional process analytics assume a single flow of execution with a clear start and end, which means in IoT domains that the IoT system is assumed to be a single execution flow (i.e., an IoT process execution). The reality is in fact different: IoT systems are composed of different autonomous components, each with a different independent life-cycle, which periodically synchronize. Explicitly considering these interactions is certainly relevant for good predictions of, e.g., attacks. In network security, [43, 44] introduce a concept of object. These works compute temporal-association-causal rules in order to discover how human-expert interventions can mitigate security vulnerabilities. The object concept used here is more related to the concept of attributes of events of a log, rather than the concept of object used in this thesis, which is conversely intended as a complex entity with a state and with activities (namely transitions) that takes from one to another state.

A few works consider interactions among different instances of a process [18, 46, 91], but they still rely on the notion of a single process flow (i.e., single case identifier). While some of these works provide valuable insights into inter-case features, their extension to object-centric processes is in fact the goal of the technique proposed in this chapter. Berti and van der Aalst [5] propose an approach to extract object-centric event logs from the data stored in relational databases; however, their work is not aimed at predictive process analytics, which is indeed the primary objective of this chapter.

The approach presented here was built under the assumption that no severe data issues, such as missing events or incorrect timestamps, are present in the event log. In case this happens, correction techniques should be applied in order to resolve these issues [64, 75].

Adams et al. [1] is the only research work on object-centric predictive process analytics. It performs a comparison of graph-based and more traditional machine-learning techniques, and it suggests that the latter does not preserve the whole graph-like structure of object-centric event logs, but only an approximation; however, the comparison between the several techniques is based only on one KPI and one process, and thus the results could not be considered of general applicability. Furthermore, even for one single process and one KPI, the techniques by Adams et al., which rely on graph-based networks, do not show significant improvements. In fact, the improvement was just by 2%, which we have also sometimes observed when comparing techniques based on LSTM and graph neural networks. However, the latter typically requires a large training time (more details will be provided in Section 5.3.5. Furthermore, Adams et al. suggested that the problems caused by flattening the event log, namely convergence and divergence [26, 110]), can possibly strengthen existing directly-follows relationships and make it impossible to apply Process Mining techniques designed for single-flow processes that heavily rely on the directly-follows relationships, such as those for model discovery and conformance checking. However, it has to be noticed that unfolding causes no problem in our process prediction approach because we do not leverage directly-follows relationships.

We also considered several KPIs in our experiments while focusing on different starting and ending points to consider different perspectives of the two object-centric event logs. This idea was inspired by Berti and van der Aalst [5], who introduced a similar concept of viewpoint to extract event logs from the data stored in relational databases. However, the concept of viewpoint did not aim at predictive process analytics.

We already mentioned in Section 3.2 that several research works shown that LSTM and Catboost generally outperform other methods for predictive process monitoring [30, 63, 73, 104, 125]. Therefore, when developing an approach for object-centric predictive process analytics, we decided to include them in our evaluation. About the choice of graph-based neural networks, some works [76, 101, 116] have also proven GNNs to be useful as the control flow of process instances can intuitively be represented as graphs, and the relationships between process activities and between the objects involved can explicitly be modeled.

5.3 Predictive Analytics in Object-centric Processes

This section reports on the results of the empirical evaluation of our predictive monitoring framework for object-centric processes, comparing the results obtained with different predictive techniques. Section 5.3.1 presents our proposal to create single-id event logs from object-centric event logs and enable predictive analytics in object-centric processes. Furthermore, we illustrate the study design for our comparison, which consists of five analysis phases. The first one, which provides details on the three object-centric datasets that have been used to assess the accuracy of the different predictive approaches, is detailed in Section 5.3.2, while Section 5.3.3 describes data preprocessing steps and data selection options that have been used for the prediction models. The third phase, which provides details about the encoding used for each predictive model will be illustrated in Section 5.3.4, while the fourth phase, which illustrates the adopted predictive models, have been already detailed in Section 2.2.2. The fifth phase, containing details on the evaluation in terms of predictive accuracy and run-time performance, will be illustrated and discussed in Section 5.3.5.

5.3.1 Enablement of Object-Centric Predictive Process Analytics: From Object-Centric to Single-Id Event Logs

The starting point is an object-centric log $\mathcal{L} = (E, T, A, AN, AV, AT, OT, O, \pi_{typ}, \pi_{act}, \pi_{time}, \pi_{vmap}, \pi_{omap}, \pi_{otyp}, <)$ (cf. Section 2.1.1). Our object-centric process prediction requires analysts to decide a so-called *viewpoint*, which is an object type $o_t \in OT$ of the process (e.g., ***Requisition***). This defines how the events in an object-centric event log are aggregated to form traces of a single-id event log $\mathcal{L} = (E', T', A', AN', AV', AT', \pi'_{typ}, \pi'_{act}, \pi'_{time}, \pi'_{vmap})$. Notice that the single-id event log definition given in this chapter is equivalent to the traditional event log definition given in Section 2.1.

A single-id event log is created from a chosen viewpoint $o_t \in OT$ as follows. We start by considering each object $o \in O$ such that $\pi_{otyp}(o) = o_t$. At this point, we compute the timestamp of the first event in E that has o as one of the object identifiers with type o_t:

$$t_o = \min_{e \in E.\ o \in \pi_{omap}(e) \wedge \pi_{otyp}(o) = o_t} \pi_{time}(e) \tag{5.1}$$

After finding the object o that has the earliest timestamp among the objects with type o_t, trace σ_o will include every event $e \in E$ such that it at least contains o as identifier (namely, such that $\{o\} \subseteq \pi_{omap}(e)$) or contains an identifier of an object o' in a certain set $R^+_L(o)$ of related objects (namely such that $R^+_L(o) \cap \pi_{omap}(e) \neq \emptyset$).

The set $R^+_L(o)$ is constructed as follows. Let us define $R^1_{L,o_t}(o)$ as the set of objects directly related to o:

$$R^1_{L,o_t}(o) = \{o' \in O : \exists e \in E.\ \{o, o'\} \subseteq \pi_{omap}(e)\}.$$

We can also define the set of objects that are indirectly related o via a "bridge" object o':

$$R^2_{L,o_t}(o) = \bigcup_{o' \in R^1_{L,o_t}(o)} R^1_{L,o_t}(o') \tag{5.2}$$

The definition above can also be extended for any $R^i_{L,o_t}(o)$ with $i > 1$ as follows:

$$R^i_{L,o_t}(o) = \bigcup_{o' \in R^{i-1}_{L,o_t}(o)} R^1_{L,o_t}(o') \tag{5.3}$$

The set $R^+_L(o)$ can thus be defined as the union, for all integer indexes $i \geq 1$, of the sets of the objects that are related to o via $(i-1)$ briding events:

$$R^+_L(o) = \bigcup_{i=1}^{\infty} R^i_{L,o_t}(o) \tag{5.4}$$

After applying the aforementioned procedure, the result is a trace in which all the events contain the identifier of an object that is connected to o, either directly or transitively. Afterwards, we exclude from the set E the events that have been already considered as part of the first identified trace. We then repeat the same procedure illustrated above, finding another object o' with type o_t that has the earliest timestamp among the objects with type o_t; the second trace will include every event $e \in E$ such that it at least contains o' as identifier, and we include all the events associated to objects that are directly or indirectly related to o' via bridge events.

This procedure creates the set T of traces, which in turn induces the other elements of the $\mathcal{L}$ tuple as follows. The set $E' = \cup_{\sigma_o \in T} \cup e' \in \sigma_o e$ of event identifier contains the event identifiers in T. The domain of attribute names, values and types is the same as in the object-centric log: $AN' = AN$, $AV' = AV$, $AT' = AT$, and $\pi'_{typ} = \pi_{typ}$. Functions $\pi'_{act}, \pi'_{time}, \pi'_{vmap}$ are restricted over domain E', namely for each $e' \in E'$ $\pi'_{act}(e) = \pi_{act}(e)$, $\pi'_{time}(e) = \pi_{time}(e)$, $\pi'_{vmap}(e) = \pi_{vmap}(e)$.

Example 5.3.1. *As an example, let us consider the* `Requisition` *as a viewpoint. The outcome of applying our technique to the object-centric event log represented in Table 2.2 can be seen in Table 5.1, where the white and blue coloured rows refer to two separate identified traces. In particular, the first (white) trace contains all the events that are associated to objects directly or indirectly related to the first requisition,* `rq1`*; therefore, we first considered the events containing* `rq1` *as identifier, namely* `e3`, `e8`, `e12`, `e13`*); afterwards, we considered*

Table 5.1: Single-id event log obtained from unfolding the object-centric event log in Table 2.2 when applying our object-centric approach and considering the *Requisition* viewpoint. Each row is an event, which is labelled with a different colour based on the trace identified. The identifier of the trace is also reported in the first column. The blank spaces represent attributes missing values.

Case ID	ID	Activity	Timestamp	Contract	Requisition	Order	Receipt	Invoice	User	Order_Price	Order_Purch_Group	Rec_Quantity
1	e3	Purchase Requisition Line Created	2017-07-15 12:00	c1	rq1				A456			
2	e5	Purchase Requisition Line Created	2017-07-15 17:00	c2	rq2				A457			
1	e6	Purchase Order Line Creation	2017-07-16 15:00	c1		o1			A458	100	100_L50	
1	e7	Contract Line Creation	2017-07-16 16:00	c3					CO01			
1	e8	Purchase Order Line Creation	2017-07-17 15:00		rq1	o2			A458	200	100_L51	
2	e9	Purchase Order Line Creation	2017-07-18 15:00		rq2	o3			A458	300	100_L52	
1	e10	Goods Line Registered	2017-07-22 15:00			o1	r1		A456	100	100_L50	10
1	e11	Invoice Receipt	2017-07-22 16:00					i1	A125			
1	e12	Purchase Requisition Group Changed	2017-07-22 19:00		rq1				A456			
1	e13	Purchase Order Line Creation	2017-07-23 9:00		rq1	o4			A458	600	100_L52	
1	e14	Purchase Order Line Creation	2017-07-23 12:00	c3		o5			A458	700	100_L50	
1	e15	Goods Line Registered	2017-07-23 15:00			o2	r2		A456	100	100_L50	10
1	e16	Invoice Registered	2017-07-29 11:00				r1,r2	i1	A125			10
1	e17	Invoice Cleared	2017-07-30 12:00					i1	A125			
1	e18	Goods Line Registered	2017-07-31 15:00			o4	r3		A456	600	100_L52	10
1	e19	Goods Line Registered	2017-08-09 15:00			o5	r4		A456	700	100_L50	10
1	e20	Invoice Registered	2017-08-10 11:00				r2,r3,r4	i2	A125			10
1	e21	Invoice Cleared	2017-08-15 14:00					i2	A125			
2	e22	Goods Line Registered	2017-08-16 15:00			o3	r5		A456	300	100_L52	5
2	e23	Purchase Requisition Supplier Changed	2017-08-16 17:00		rq2				A456			
2	e24	Invoice Registered	2017-08-18 11:00				r5	i3	A125			5
2	e25	Invoice Cleared	2017-08-20 14:00					i3	A125			

Table 5.2: Single-id event log obtained from unfolding the object-centric event log in Table 2.2 when considering the *Requisition* viewpoint and with the existing approach, assuming each instance belongs to a single process. Each row is an event, which is labelled with a different colour based on the trace identified. The identifier of the trace is also reported in the first column. The blank spaces represent attributes missing values.

Case ID	ID	Activity	Timestamp	Contract	Requisition	Order	Receipt	Invoice	User	Order_Price	Order_Purch_Group	Rec_Quantity
1	e3	Purchase Requisition Line Created	2017-07-15 12:00	c1	rq1				A456			
2	e5	Purchase Requisition Line Created	2017-07-15 17:00	c2	rq2				A457			
1	e8	Purchase Order Line Creation	2017-07-17 15:00		rq1	o2			A458	200	100_L51	
2	e9	Purchase Order Line Creation	2017-07-18 15:00		rq2	o3			A458	300	100_L52	
1	e12	Purchase Requisition Group Changed	2017-07-22 19:00		rq1				A456			
1	e13	Purchase Order Line Creation	2017-07-23 9:00		rq1	o4			A458	600	100_L52	
2	e23	Purchase Requisition Supplier Changed	2017-08-16 17:00		rq2				A456			

the events related transitively to `rq1`, i.e. with at least one identifier of an object in $R_L^+(rq1)$. For instance, we considered event `e6` because it contains the contract identifier `c1` that is related to `rq1` via event `e3`, which contains both `rq1` and `c1` among the identifiers. The second (blue) trace contains instead all the events that are associated to objects directly or indirectly related to the second requisition, `rq2`. It is worth noting that, since we are considering the `Requisition` viewpoint, the first events of the two traces will be the ones associated with the earliest timestamp among the objects that contain an object identifier of type `Requisition`, namely `e3` for the first trace and `e5` for the *second trace.*

Conversely, existing techniques would be unable to deal with multiple object types, namely with multiple interleaving processes; they would only restrict to one single process, which would likely be the process related to objects of type `Requisition`*. Therefore, there would again be two traces, one per requisition. Differently from our approach, the event log would only contain the events that present the first or the second requisition among the identifiers. The resulting log is in Table 5.2.*

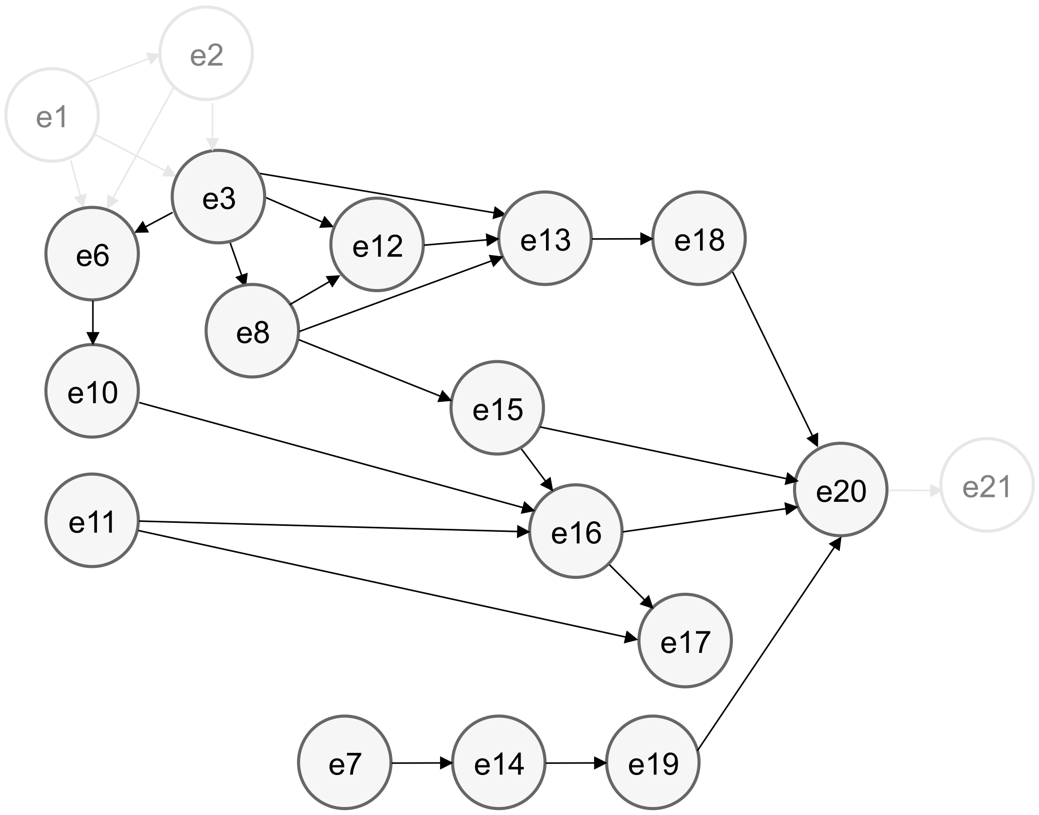


Figure 5.1: Directed graph representing the connections between the events of the first (white) trace illustrated in Table 5.1. The nodes highlighted in gray represent the events considered when the viewpoint is *Requisition* and the selected target activity is the last *Invoice Registered.*

To give a better intuition about how we extract traces from an object-centric event log, our technique can be also visually described by relying on graphs. Given the event log $\mathcal{L}$ represented in Table 2.2, where no indication about traces exists, each event becomes a node of the graph, and the object identifiers associated with each event are added as properties of the node. Finally, an edge exists between two nodes e_i and e_j if there is at least one object identifier that is both in e_i and e_j. This edge is directed since the events are temporally ordered according to their timestamps. The procedure is then repeated for all the events on the event log.

A visual representation of the connections between the events of the first identified trace can be found in Fig. 5.1. Let us recall that, for each object o of the viewpoint, we find the earliest event e that has o among its objects; since we are considering the *Requisition* as viewpoint, *e3* will be the first event of the trace and a node of the aforementioned graph. We then create a trace for object o by navigating the graph from node e, following the edge directions, and by selecting every node that is encountered in the navigation. As an example, the node represented by the event identifier $e3$ is connected to node $e6$ because $c1$ is a property that is in common between these two nodes, and it is connected to nodes $e8$, $e12$, $e13$ as the object identifier $rq1$ is a property that is in common between these four nodes.

The concept of viewpoint has a direct influence on the number of events that are included. An example is provided in Fig. 5.1. Here, the gray colour indicates the nodes that are included when the selected viewpoint is the *Requisition*, and the selected target activity is the last *Invoice Registered.* Statistics describing the impact of the several viewpoints considered in our evaluation on the number of events included will be provided in Section 5.3.5.

Summarising, the approach starts from an object-centric event log $\mathcal{L} = (E, T, A, AN, AV, AT, OT, O, \pi_{typ}, \pi_{act}, \pi_{time}, \pi_{vmap}, \pi_{omap}, \pi_{otyp}, <)$, which is unfolded to a single-id event log $\mathcal{L} = (E', T', A', AN', AV', AT', \pi'_{typ}, \pi'_{act}, \pi'_{time}, \pi'_{vmap})$ around a viewpoint, which defines how the events in an object-centric event log are aggregated to form traces of a single-id event log.

These traces can be used to train and test a proper prediction model using the off-the-shelf techniques discussed in Section 2.2.2. However, this unfolding to a single-id event log does not preserve information about the number of objects correlated to the viewpoint object. In fact, it excludes the information about the attributes associated with correlated objects, and their respective value. Let us suppose again that we aim to use the total processing time of requisitions as KPI, and this time is somehow correlated to the number of orders associated to the requisitions (e.g., a requisition for more orders takes longer to be processed). The prediction of this KPI could be more accurate if the number of orders objects associated with the requisition were used to train and use the prediction model.

To mitigate this information loss, we extend our first approach to a richer one, where we introduce aggregation attributes that synthesize additional interaction information. Given a prefix σ'_o, of the trace σ_o for a viewpoint object o, the following aggregation attributes are included.

1. A given attribute a that can take different values $v_1, \dots, v_n$ in a set $\{o'_1, \dots, o'_m\} \subseteq R^+_L(o)$ of objects related to o. The standard encoding would only retain one value for a, namely the value observed in the latest event in σ_o that assigns a value to a. We define:
   - if a is numerical (i.e., $\pi_{typ}(a) \subseteq \mathbb{R}$), one feature f is added to the feature set that contains an aggregated value $\psi(v_1, \dots, v_n)$ summarizing observed values. The aggregation function $\psi(\vec{v})$ may be customized depending on the domain, such as the average value in $\vec{v}$;
   - if a is categorical, a feature is added for each pair (a, v_i) with value defined as the ratio between the number of attribute assignments to the value v_i over the total assignments of a.
2. For each object type $o'_t \in OT$, one feature encodes the number of objects of type o'_t associated with events of σ'_o, namely $|\{o' \in O : \pi_{otyp}(o) = o'_t \wedge \exists e \in \sigma'_o.\ o' \in \pi_{omap}(e)\}|$.
3. For each object type $o'_t \in OT$ and for each activity $a \in A$ one feature is added with value equal to the percentage of objects of type o'_t for which activity a has occurred at least once in trace σ'_o.

Example 5.3.2. *The result of introducing the aggregation attributes to the feature set is presented in Table 5.3:*

- *The first type of aggregation attribute that was introduced is represented by the column* `Avg order_price`*, which represents the average order price (indicated by the column* `order_price`*) considering all the orders in place in the specific trace. It can be seen that event* `e6` *is associated to an* `Avg order_price` *of 100, since there is only one order with* `order_price` *100; conversely, event* `e8` *is associated to an* `Avg order_price` *of 150, since there is one order (o1) with* `order_price` *100 and one order (o2) with* `order_price` *200.*

Table 5.3: Single-id event log obtained from unfolding the object-centric event log in Table 2.2 with the proposed object-centric approach when considering the Requisition viewpoint and the aggregation attributes. Each row is an event, which is labelled with a different colour based on the trace identified. The identifier of the trace is also reported in the first column. The blank spaces represent attributes missing values.

Case ID	ID	activity	timestamp	Contract	Requisition	Order	Receipt	Invoice	user	order_price	order_purch_group	rec_quantity	# Contracts	# Reqs	# Orders	# Receipts	# Invoices	Avg order_price	% order_purch_group=100_L50	% order_purch_group=100_L51	% order_purch_group=100_L52	Avg rec_quantity	(Order, % Purchase Order Line Creation)	(Order, % Goods Line Registered)	(Receipt, % Goods Line Registered)	(Invoice, % Invoice Receipt)	(Receipt, % Invoice Registered)	(Invoice, % Invoice Registered)	(Invoice, % Invoice Cleared)
1	e3	Purchase Requisition Line Created	2017-07-15 12:00	c1	rq1				A456				1	1	0	0	0	0	0	0	0		0	0	0	0	0	0	0
1	e6	Purchase Order Line Creation	2017-07-16 15:00	c1		o1			A458	100	100_L50		1	1	1	0	0	100	1	0	0		1	0	0	0	0	0	0
1	e7	Contract Line Creation	2017-07-16 16:00	c3					CO01				2	1	1	0	0	100	1	0	0		1	0	0	0	0	0	0
1	e8	Purchase Order Line Creation	2017-07-17 15:00		rq1	o2			A458	200	100_L51		2	1	2	0	0	150	0.5	0.5	0		1	0	0	0	0	0	0
1	e10	Goods Line Registered	2017-07-22 15:00			o1	r1		A456	100	100_L50	10	2	1	2	1	0	150	0.5	0.5	0	10	1	0.5	1	0	0	0	0
1	e11	Invoice Receipt	2017-07-22 16:00					i1	A125				2	1	2	1	1	150	0.5	0.5	0	10	1	0.5	1	1	0	0	0
1	e12	Purchase Requisition Group Changed	2017-07-22 19:00		rq1				A456				2	1	2	1	1	150	0.5	0.5	0	10	1	0.5	1	1	0	0	0
1	e13	Purchase Order Line Creation	2017-07-23 9:00		rq1	o4			A458	600	100_L52		2	1	3	1	1	300	0.33	0.33	0.33	10	1	0.33	1	1	0	0	0
1	e14	Purchase Order Line Creation	2017-07-23 12:00	c3		o5			A458	700	100_L50		2	1	4	1	1	400	0.5	0.25	0.25	10	1	0.25	1	1	0	0	0
1	e15	Goods Line Registered	2017-07-23 15:00			o2	r2		A456	200	100_L50	10	2	1	4	2	1	400	0.5	0.25	0.25	10	1	0.5	1	1	0	0	0
1	e16	Invoice Registered	2017-07-29 11:00				r1,r2	i1	A125			10	2	1	4	2	1	400	0.5	0.25	0.25	10	1	0.5	1	1	1	1	0
1	e17	Invoice Cleared	2017-07-30 12:00					i1	A125				2	1	4	2	1	400	0.5	0.25	0.25	10	1	0.5	1	1	1	1	1
1	e18	Goods Line Registered	2017-07-31 15:00			o4	r3		A456	600	100_L52	10	2	1	4	3	1	400	0.5	0.25	0.25	10	1	0.75	1	1	0.66	1	1
1	e19	Goods Line Registered	2017-08-09 15:00			o5	r4		A456	700	100_L50	10	2	1	4	4	1	400	0.5	0.25	0.25	10	1	1	1	1	0.5	1	1
1	e20	Invoice Registered	2017-08-10 11:00				r2,r3,r4	i2	A125			10	2	1	4	4	2	400	0.5	0.25	0.25	10	1	1	1	0.5	1	1	0.5
1	e21	Invoice Cleared	2017-08-15 14:00					i2	A125				2	1	4	4	2	400	0.5	0.25	0.25	10	1	1	1	0.5	1	1	1
2	e5	Purchase Requisition Line Created	2017-07-15 17:00	c2	rq2				A457				1	1	0	0	0	0	0	0	0		0	0	0	0	0	0	0
2	e9	Purchase Order Line Creation	2017-07-18 15:00		rq2	o3			A458	300	100_L52		1	1	1	0	0	300	0	0	1		1	0	0	0	0	0	0
2	e22	Goods Line Registered	2017-08-16 15:00			o3	r5		A456	300	100_L52	5	1	1	1	1	0	300	0	0	1	5	1	1	1	0	0	0	0
2	e23	Purchase Requisition Supplier Changed	2017-08-16 17:00		rq2				A456				1	1	1	1	0	300	0	0	1	5	1	1	1	0	0	0	0
2	e24	Invoice Registered	2017-08-18 11:00				r5	i3	A125			5	1	1	1	1	1	300	0	0	1	5	1	1	1	0	1	1	0
2	e25	Invoice Cleared	2017-08-20 14:00					i3	A125				1	1	1	1	1	300	0	0	1	5	1	1	1	0	1	1	1

- *The second type of aggregation attribute that was introduced is represented by the column* `%order_purchase_group=100_L50`*, which indicates the fraction of objects correlated with objects of type order where attribute* `order_purchase_group` *takes on value 100_L50: in particular, it indicates the fraction of the orders that are related to the purchase group 100_L50, compared to the total number of orders. As an example, the event* `e6` *is associated with a* `%order_purchase_group=100_L50` *of 1, since there is only one order with purchase group 100_L50; conversely, event* `e8` *is associated with a* `%order_purchase_group=100_L50` *of 0.5, since there is one order (o1) with purchase group 100_L50 and one order (o2) with purchase group 100_L51.*

- *The third type of aggregation attribute introduced is represented by the column* `#Orders`*, which represents the number of concurrent processes (objects) of type order in the specific trace. In the event* `e6` *there is only one order (o1), while in the event* `e8` *there are two orders (o1 and o2) that are being executed in parallel in the specific trace.*

- *Finally, the fourth type of aggregation attribute introduced is represented by the column* `Order, %Goods Line Registered`*, which indicates the percentage of orders that have performed the activity* `Goods Line Registered` *at least once; as an example, it has a value of 0.5 for event* `e10`*, since among the two orders (o1 and 02) there is only one order (o1) that has performed* `Goods Line Registered`*.*

5.3.2 Datasets

We previously mentioned that our study design for comparing the performances of different predictive models in the object-centric case consists of five analysis phases, which are detailed in Figure 5.2. This section describes the first phase, which provides details on the object-centric datasets that have been used to assess the accuracy of the different predictive approaches.

The first object-centric process, which has been already illustrated in Section 2.1.1, was executed by a well-known Italian utility-provider company, one of the major energy companies in Europe. The company focuses on the production/extraction of electricity and gas and on their distribution in different parts of the world. As mentioned in Section 2.1.1 and shown in Figure 2.2, this object-

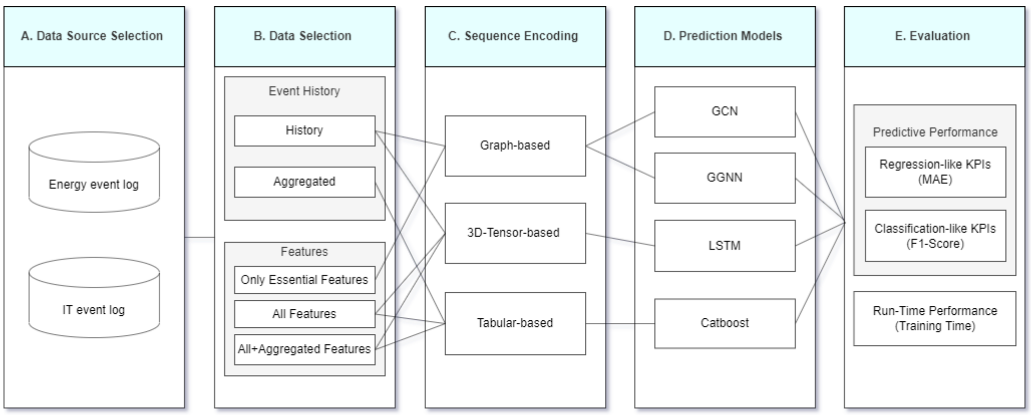


Figure 5.2: Study design for comparing Machine-Learning-based techniques for predictive analytics in object-centric processes.

centric process runs through the intertwining of five different processes (i.e., object types): the *Requisition*, the *Order*, the *Invoice*, the *Contract*, and the *Receipt*.

The second object-centric process was executed by a well-known American technology company, one of the major companies worldwide. This object-centric process runs through the intertwining of three different processes (i.e., object types), the *Requisition*, the *Order*, and the *Invoice*; Figure 5.3 illustrates how objects are related to each other for synchronization and data exchanges.

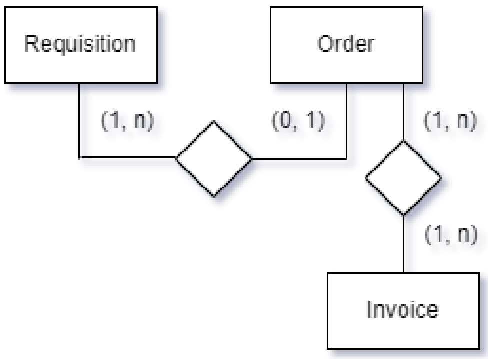


Figure 5.3: Diagram representing cardinality between the different object types in the IT object-centric event log. For each object type, the cardinality with the subsequent or the previous object type is represented as (*min_cardinality*, *max_cardinality*)

Finally, the third object-centric process was executed by a well-known American manufacturing company, whose aim is to sell processed consumer foods through retail stores. This object-centric process runs through the intertwining of three different processes (i.e., object types), the *Order*, the *Receipt*, and the *Invoice*; Figure 5.4 illustrates how objects are related to each other for synchronization and data exchanges.

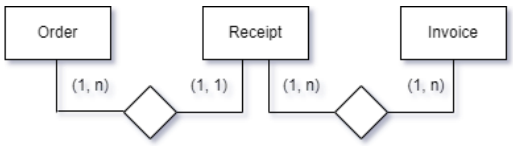


Figure 5.4: Diagram representing cardinality between the different object types in the manufacturing object-centric event log. For each object type, the cardinality with the subsequent or the previous object type is represented as (*min_cardinality*, *max_cardinality*)

5.3.3 Data Preprocessing and Selection

The second phase begins with a data preprocessing procedure consisting of three steps. First, we removed attributes with missing values in more than 80% of the cases or attributes with the same values in all cases. Second, we used domain knowledge to remove the somehow duplicated attributes. For example, the event log contains an attribute that refers to the order plant name, which is unique, and a second related to the plant identifier. Thus, one of the two can be removed. Third, the large dimension of both companies is also reflected in the cardinality of some categorical attributes. For instance, for the utility-provider company, the codes of the materials shipped worldwide (*order_material_code*) are stored in an attribute that counts up to $4,179$ different values. We applied the 80-20 rule to reduce the cardinality of the attributes with thousands of different values [71]. Specifically, we kept the most frequent attribute values that covered 80% of the cases and labeled the remaining values as ”other”.

After data preprocessing, we considered three different feature settings, varying in the number of features used. First, *All Features*, indicates that all features are leveraged for prediction. This setting has been used for LSTM and Catboost models. Second, *All+Aggregated Features*, including also aggregated features, which synthesize additional information about object interaction and which were encoded according to the procedure that has been already described in Section 5.3.1. This setting has been selected as the previous one for both Catboost and

LSTM models. Third, *Only Essential Features*, including activity features, the four temporal features calculated based on the events' timestamps (as proposed by Tax et al. [104]), and five features specifying the object type of the events. This setting has been used for graph-based models. Note that the selection of the feature setting and the selection of the models' architecture (that will be illustrated in the Section 5.3.5) were selected based on the best predictive accuracy that was observed on the validation set. For Catboost and LSTM models, we decided to include 2 different feature settings (*All Features* and *All+Aggregated Features*), since in most of the times Catboost and LSTM showed a better accuracy when leveraging also the aggregated features; however, there were cases in which leveraging them did not improve the predictive accuracy, due to overfitting phenomena. The approach without the additional information about aggregated features is still beneficial on its own when overfitting is detected.

5.3.4 Sequence encoding for Graph Neural Networks

The sequence encoding adopted for Catboost models and LSTM networks has been already described in Section 2.4.1 and Section 2.4.2, respectively. Here, we limit ourselves to illustrate the sequence encoding that has been adopted for graph neural networks.

Graph neural networks, which have been introduced in Section 2.2.2), require a graph-oriented representation as input.

Graph neural networks cannot be directly trained on raw graphs; therefore, an adaptation needs to be done in order to leverage graph neural networks for predictive process analytics. In particular, we transformed raw graphs into instance graphs. When graph neural networks are employed for predictive process analytics, each prefix of every event-log trace is encoded as an instance graph. Since nodes and edges of an instance graph can be interpreted in different ways, there are different forms of how an instance graph can encode a trace prefix; in this thesis, an instance graph assumes (time-ordered) events of a prefix as graph nodes and considers edges between the prefix's events. Regarding edge types, we differ between (1) *Repeat* (activity of a target event is equal to an activity of a source event), (2) *Backward* (activity of a target event was observed in a previous event of the current prefix), and (3) *Forward* (activity of a target event was not observed in previous events of the current prefix). In the following, we provide a concrete example illustrating how a trace prefix can be encoded as an instance graph. Given the prefix σ' (which can be also represented with a graph structure, as it can be seen in Figure 5.5), where each row represents an event of the prefix, with the first attribute representing the case identifier, the second attribute representing the performed activity, the third the timestamp, and the last one the resource that has performed the activity:

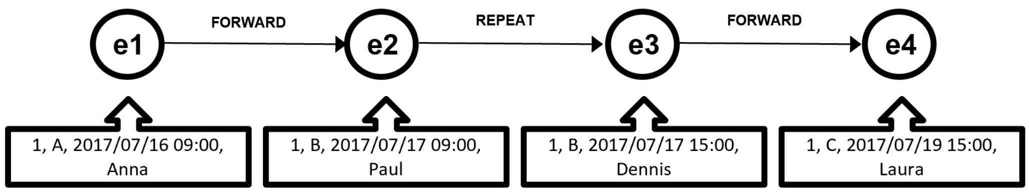

FORWARD
REPEAT
FORWARD
e1
e2
e3
e4
1, A, 2017/07/16 09:00, Anna
1, B, 2017/07/17 09:00, Paul
1, B, 2017/07/17 15:00, Dennis
1, C, 2017/07/19 15:00, Laura

Figure 5.5: Graph structure representing the prefix example 5.5. Each box illustrates the attributes of the related event.

$$\begin{aligned}\sigma' = \langle &(1, A, 2017/07/16\ \ 09{:}00, Anna),\\ &(1, B, 2017/07/17\ \ 09{:}00, Paul),\\ &(1, B, 2017/07/17\ \ 15{:}00, Dennis),\\ &(1, C, 2017/07/19\ \ 15{:}00, Laura)\rangle\end{aligned} \tag{5.5}$$

the process instance graph $\mathcal{I}_{\sigma'}$ for the prefix σ' can be represented as follows:

$$\mathcal{I}_{\sigma'} = \left(\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} A & 0.0 & 7 & Anna \\ B & 1.0 & 1 & Paul \\ B & 0.25 & 1 & Dennis \\ C & 2.0 & 3 & Laura \end{bmatrix}, \begin{bmatrix} 1 & 2 & \textit{Forward} \\ 2 & 3 & \textit{Repeat} \\ 3 & 4 & \textit{Forward} \end{bmatrix} \right) \tag{5.6}$$

Here, the first matrix of the instance graph $\mathcal{I}_{\sigma'}$ is the adjacency matrix, which stores the connection between nodes, which are events in our setting. For example, the first row [0, 1, 0, 0] indicates that an edge goes from the first event to the second event of the prefix. The second is the node matrix, which stores all information (features) related to the nodes. In this small example, we considered as features the activity, the time since the previous event in the process instance calculated based on the events' timestamps (in days format), the day of the week, and the resource that performed the activity. The third matrix is the edge matrix, which stores the event identifier of the source node, the identifier of the target node, and the type of edge defined by the source and target node.

More formally, instance graph, adjacency matrix, node matrix, and edge matrix can be defined as:

Definition 5.3.1 (Instance Graph, Adjacency Matrix, Node Matrix, Edge Matrix)**.** *An instance graph $\mathcal{I}$ is a three-element tuple (AM, V, EM). AM is an adjacency matrix storing which nodes of the graph are connected by an edge and lies in $\mathbb{R}^{|V|\times|V|}$, where $|V|$ is the number of nodes of the graph. V is a node matrix storing features that describe the graph's nodes and lies in $\mathbb{R}^{|V|\times q}$, where q is the number of node features. EM is an edge matrix that is added in order to store features that describe the edges of the graph and lies in $\mathbb{R}^{|V|\times p}$, where p refers to the number of features describing the edge, i.e., the source node, the target node, and features describing the edge.*

After describing the required preliminary concepts, we can finally illustrate the sequence encoding adopted for graph neural networks. In particular, in this thesis, we relied on two different graph neural networks: GCN (Graph Convolutional Network) and GGNN (Gated Graph Neural Networks) [55]. The main difference between the two models lies in the fact that the edge matrix of the input graph for the GCN is omitted as the original GCN from [45] does not allow it to compute it. When leveraging the former, $\mathcal{X}$ is represented by a two-element tuple (AM, V), where AM is the adjacency matrix and V is the node matrix; conversely, for the GGNN, $\mathcal{X}$ represents a three-element tuple (AM, V, EM), where EM is the edge matrix that is added in order to store features that describe the edges of the graph. In the example reported above, which was referring to a traditional process, the edge types described in the edge matrix were encoded with respect to the activity; however, when dealing with object-centric processes, these edge types are defined taking into consideration also the object identifiers. For example, an edge in an object-centric process is of type *Backward* iff the activity has already been observed in previous events of the current prefix for the same object identifier. Finally, please notice that, for each node (event) of the GCN and GGNN, the node features have been encoded in the same way as LSTM. In particular, let us recall the intermediate concept of an *event-to-tuple function* $\zeta_{\mathcal{L}} : E \to A \times T \times (AV)^{|AN|}$, which encodes each event of the event log $\mathcal{L}$; here, each numerical and boolean attribute a becomes one feature, element of tuple $\zeta_{\mathcal{L}}(e)$. Each literal attribute a is instead represented through the so-called *one-hot encoding*: one different dimension exists for each value $v \in \mathcal{W}_{AN}(a)$, and the dimension referring to value $e(a)$ takes on value 1, with the other dimensions be assigned value 0.

5.3.5 Evaluation & Results

Our comparison includes prediction models constructed with four different machine learning algorithms. The first three algorithms, namely GCN, GGNN, and LSTM, belong to the group of Deep Learning algorithms, whereas the remaining one, namely Catboost, belongs to the group of Ensemble Learning algorithms. We already described the architectures and parameters that have been evaluated for both LSTM and Catboost models in Section 3.4; in the following, we provide a quick overview about the architectures and parameters that have been evaluated for the GCN and the GGNN models.

In particular, we used the Tensorflow framework for the GCN and the GGNN implementation. For both algorithms, we leveraged ADAM as learning algorithm, we selected 200 training epochs with a patience of 20, a learning rate of 0.001 and a batch size of 32. In each experiment, we used 2/3 of the traces as a training set and 1/3 of the traces as a test set. During training, a hyperparameter optimization was performed, in which we used the last 20% of the training set as a validation set.

Regarding the GCN implementation, the architecture was inspired by the work of [116]; the original architecture included a GCN layer with one channel, followed by a Global Average Pooling layer, a Dropout layer with a dropout rate of 50%, two Dense layers with 256 neurons and $tanh$ activation, and a second Dropout layer with the same dropout rate. We tested different network configurations on the validation set, validating the number of channels in the GCN layer (which varied between 1, 2, and 4), the number of final Dense layers (we considered keeping the two layers or removing them) and the number of neurons for each Dense layer (which varied between 100 and 250).

Conversely, the architecture of the second algorithm (GGNN), which integrates a Gated Recurrent Unit (GRU) cell that explicitly considers the temporal aspect of sequences, was inspired by the work of [120]; the original architecture included a Gated Graph layer with four GRU cell iterations and $tanh$ activation, followed by a Global Attention layer with 100 output channels and three Dense layers with a dropout rate of 50% each and with 300, 200 and 100 neurons, respectively. We tested different network configurations on the validation set, validating in particular the number of GRU cell iterations in the Gated Graph layer (which varied between 1, 2, and 4), the number of final Dense layers (which varied between 1, 2, 3 or no layers at all) and the number of neurons for each Dense layer (which varied between 100 and 250).

All the KPIs were defined wrt. a viewpoint and considering different target activities. We considered several KPIs in our evaluation, which can be grouped into three categories:

- *Elapsed Time between the first occurrence of the considered object and the last occurrence of a selected target activity.* From the customers' perspective, it is of interest to predict the time required to complete the last occurrence of some selected activity, considering starting from different viewpoints. Regarding the utility-provider company, the first target activity of interest is *SES Line Registered*. It indicates that the service requested by the customer is provided. However, as the customer can require several services, it is of interest to know when all the services requested are provided. The second target activity, *SES Line Released*, indicates that a further step is performed, which is the confirmation from the manager that everything is received correctly. Another interesting activity to be monitored is *Invoice Receipt*, which indicates that the invoice is correctly charged to the customer; conversely, *Invoice Cleared* indicates that the invoice is paid. Similar target activities were of interest for the manufacturing company, in particular *Invoice Receipt*, *Invoice Cleared*, *SES Line Registered* and *Goods Line Registered*; the last one indicates that the materials requested by the customer are provided and, since the customer can require several materials, it is of interest to know when all the materials requested are provided. Also for the technology company, two interesting activities to be monitored were *Invoice Receipt* and *Invoice Reconciled*; the last one, similarly to *Invoice Cleared*, indicates that the invoice is paid. The third target activity of interest is *Invoice Submit*, which indicates that the Invoice is registered into the system. The last interesting activity to be monitored is *Invoice Approved*, which indicates that the invoice that is submitted for registration is approved by a manager.

- *Pay Delay estimation.* It refers to the number of days exceeding the planned payment date, starting from the contract's creation to the last occurrence of *Invoice Cleared*.

  These two categories are defined over a numerical domain and descriptive statistics relevant to understanding predictive accuracy are summarized in Table 5.4a.[1]

[1]Please notice that the selection of the cases and the observed average standard deviation of the cases greatly vary depending on the selected KPI.

Table 5.4: Descriptive statistics of selected KPIs.

(a) Numerical KPIs. AVG stands for Average Value, SD stands for Standard Deviation. Log 1 refers to the energy company, while Log 2 refers to the IT company.

Log	ID	Viewpoint	KPI	# Cases	# Events	Avg (Days)	SD (Days)
1	1	Contract	Elapsed Time from Contract to the last SES Line Registered	2,960	148,685	278.22	230.96
1	2	Contract	Elapsed Time from Contract to the last SES Line Released	2,960	153,572	279.42	230.94
1	3	Contract	Elapsed Time from Contract to the last Invoice Receipt	6,821	372,918	237.11	218.61
1	4	Contract	Elapsed Time from Contract to the last Invoice Cleared	6,880	427,606	287.14	229.69
1	5	Contract	Pay Delay estimation from Contract to the last Invoice Cleared	6,880	427,606	11.41	53.57
1	6	Order	Elapsed Time from Order to the last Invoice Receipt	28,798	465,488	28.34	38.20
1	7	Order	Elapsed Time from Order to the last Invoice Cleared	35,896	622,868	45.89	51.89
1	8	Requisition	Elapsed Time from Requisition to the last Invoice Receipt	2,366	92,781	61.58	49.44
1	9	Requisition	Elapsed Time from Requisition to the last Invoice Cleared	2,436	111,077	115.68	60.18
1	10	Requisition	Elapsed Time from Requisition to the last SES Line Released	1,310	51,526	50.76	52.44
1	11	Requisition	Elapsed Time from Requisition to the last SES Line Registered	1,310	49,510	49.79	52.25
2	12	Requisition	Elapsed Time from Requisition to the last Invoice Reconciled	87,722	671,296	28.95	54.94
2	13	Requisition	Elapsed Time from Requisition to the last Invoice Receipt	86,702	567,387	26.57	53.06
2	14	Requisition	Elapsed Time from Requisition to the last Invoice Submit	87,525	589,172	27.65	54.31
2	15	Requisition	Elapsed Time from Requisition to the last Invoice Approved	87,566	594,500	27.99	54.74
2	16	Order	Elapsed Time from Order to the last Invoice Reconciled	92,674	755,384	37.65	71.45
2	17	Order	Elapsed Time from Order to the last Invoice Submit	92,378	667,973	36.29	70.93
2	18	Order	Elapsed Time from Order to the last Invoice Approved	92,392	674,461	36.74	71.40
3	19	Order	Elapsed Time from Order to the last Invoice Receipt	50,178	1,187,587	105.49	240.92
3	20	Order	Elapsed Time from Order to the last Invoice Cleared	85,936	1,884,186	86.03	193.08
3	21	Receipt	Elapsed Time from Receipt to the last Invoice Receipt	37,626	107,249	30.11	84.50
3	22	Receipt	Elapsed Time from Receipt to the last Invoice Cleared	73,463	385,689	39.54	68.22
3	23	Invoice	Elapsed Time from Invoice to the last Goods Line Registered	36,703	149,089	10.87	8.75
3	24	Invoice	Elapsed Time from Invoice to the last SES Line Registered	1558	56,517	28.28	98.21

(b) Categorical KPIs. Column *% Cases* is the percentage of cases in which the activity or the attribute is present with that value.

Log	ID	Viewpoint	KPI	# Cases	# Events	% Cases
1	25	Contract	Occurrence of Activity Purchase Order Blocked (from Contract to the last Invoice Cleared)	6,880	427,606	27%
1	26	Contract	Occurrence of Activity Pay Method Changed (from Contract to the last Invoice Cleared)	6,880	427,606	26%
1	27	Contract	Occurrence of Attribute Pay Type Assuming Value Late (from Contract to the last Invoice Cleared)	6,880	427,606	61%
3	28	Order	Occurrence of Attribute Pay Type Assuming Value Late (from Order to the last Invoice Cleared)	85,936	1,884,186	60%
3	29	Order	Occurrence of Activity Pay Method Changed (from Order to the last Invoice Cleared)	85,936	1,884,186	20%
3	30	Order	Occurrence of Activity Invoice Released (from Order to the last Invoice Cleared)	85,936	1,884,186	3%

- *Occurrence of activity / occurrence of attribute with a particular value.* It refers to whether a certain activity or condition (e.g., a late payment) will occur in the future. It is calculated after selecting the path from the creation of the contract to the last occurrence of *Invoice Cleared* for the utility-provider company and from the creation of the order to the last occurrence of *Invoice Cleared* for the manufacturing company. This category is boolean, with true indicating the occurrence, and false the absence. In particular, this category includes six boolean KPIs related to activities that the utility-provider and the manufacturing companies want to prevent, as

Table 5.5: Predictive accuracy for KPIs and prediction models. Regression-like and classification-like KPIs are measured with MAE and F1-Score, respectively. Training times are reported in brackets.

Log	ID	Viewpoint	KPI	GCN	GGNN	LSTM w.out aggr. features	LSTM w. aggr. features	Catboost naive	Catboost w.out aggr. features	Catboost w. aggr. features
1	1	Contract	Elapsed Time from Contract to the last SES Line Registered	42.85 (1h 5m)	41.17 (1h 45m)	45.22 (3h 34m)	51.70 (7h 50m)	50.62 (5m)	33.94 (20m)	**31.9 (11m)**
1	2	Contract	Elapsed Time from Contract to the last SES Line Released	42.31 (1h 13m)	47.99 (3h 12m)	50.77 (4h 9m)	59.64 (3h 54m)	48.94 (1m)	34.14 (19m)	**33.09 (21m)**
1	3	Contract	Elapsed Time from Contract to the last Invoice Receipt	42.64 (2h 59m)	41.19 (8h 7m)	37.72 (5h 23m)	40.53 (12h)	39.43 (2m)	**29.44 (30m)**	30.78 (42m)
1	4	Contract	Elapsed Time from Contract to the last Invoice Cleared	47.86 (6h 12m)	44.37 (9h 17m)	39.05 (12h 6m)	43.05 (9h)	44.81 (2m)	**34.06 (35m)**	34.4 (49m)
1	5	Contract	Pay Delay estimation from Contract to the last Invoice Cleared	17.09 (2h 12m)	18.47 (10h 4m)	14.05 (8h 53m)	14.69 (18h 11m)	16.16 (1m)	12.88 (8m)	**12.36 (11m)**
1	6	Order	Elapsed Time from Order to the last Invoice Receipt	27.51 (2h 29m)	36.44 (5h 45m)	26.71 (2h 47m)	28.31 (7h 25m)	23.77 (5m)	20.6 (12m)	**19.15 (21m)**
1	7	Order	Elapsed Time from Order to the last Invoice Cleared	29.95 (7h 56m)	37.04 (11h 50m)	23.22 (4h 19m)	25.48 (8h 22m)	24.19 (12m)	22 (56m)	**20.08 (1h 21m)**
1	8	Requisition	Elapsed Time from Requisition to the last Invoice Receipt	34.39 (41m)	45.62 (33m)	41.36 (56m)	41.98 (1h 43m)	35.57 (1m)	31.72 (7m 25s)	**31.08 (5m)**
1	9	Requisition	Elapsed Time from Requisition to the last Invoice Cleared	35.33 (16m)	67.04 (27m)	40.96 (2h 15m)	37.61 (2h 47m)	42.97 (1m)	**32.10 (22m)**	36.71 (17m)
1	10	Requisition	Elapsed Time from Requisition to the last SES Line Released	32.23 (5m)	75.21 (20m)	55.4 (49m)	41.87 (1h 48m)	37.74 (1m)	29.5 (3m 45s)	**26.96 (3m)**
1	11	Requisition	Elapsed Time from Requisition to the last SES Line Registered	31.52 (5m)	81.62 (14m)	48.97 (48m)	39.48 (1h 15m)	38.25 (1m)	**28.05 (1m 57s)**	31.2 (6m)
2	12	Requisition	Elapsed Time from Requisition to the last Invoice Reconciled	45.86 (2h 56m)	42.12 (4h 47m)	40.5 (1h 16m)	39.29 (3h 40m)	**20.36 (2m)**	29.7 (11m)	27.5 (13m)
2	13	Requisition	Elapsed Time from Requisition to the last Invoice Receipt	48.16 (3h 10m)	44.31 (4h 17m)	49.06 (1h 9m)	41.81 (1h 52m)	**18.34 (2m)**	31.53 (9m)	29.44 (12m)
2	14	Requisition	Elapsed Time from Requisition to the last Invoice Submit	47.88 (1h 45m)	43.02 (5h 12m)	43 (1h 3m)	41.79 (1h 58m)	**18.93 (2m)**	30.21 (10m)	27.82 (13m)
2	15	Requisition	Elapsed Time from Requisition to the last Invoice Approved	47.81 (3h 54m)	43.87 (3h 50m)	42.53 (1h 2m)	41.63 (2h)	**19.42 (2m)**	30.83 (7m)	28.75 (8m)
2	16	Order	Elapsed Time from Order to the last Invoice Reconciled	51.3 (2h 50m)	46.6 (8h)	50.60 (1h 28m)	51.64 (2h 44m)	36.43 (3m)	26.99 (7m)	**25.48 (9m)**
2	17	Order	Elapsed Time from Order to the last Invoice Submit	53.13 (4h 51m)	46.89 (10h 31m)	45.23 (42m)	44.84 (2h 22m)	36.45 (3m)	25.97 (11m)	**24.02 (14m)**
2	18	Order	Elapsed Time from Order to the last Invoice Approved	51.61 (5h 14m)	47.14 (5h 3m)	47.49 (42m)	46.01 (2h 15m)	37.66 (3m)	27.13 (12m)	**25.26 (14m)**
3	19	Order	Elapsed Time from Order to the last Invoice Receipt	48.10 (17h 37m)	41.28 (1d 5h)	53.30 (10h 48m)	56.62 (11h 47m)	59.41 (10m)	**33.58 (25m)**	37 (31m)
3	20	Order	Elapsed Time from Order to the last Invoice Cleared	31.49 (2d 15h)	24.66 (1d 15h)	37.86 (23h 16m)	37.49 (18h 11m)	48.81 (21m)	**20.26 (47m)**	21.77 (52m)
3	21	Receipt	Elapsed Time from Receipt to the last Invoice Receipt	37.68 (1h 4m)	32.83 (37m)	30.76 (19m)	29.61 (40m)	38.98 (2m)	22.83 (2m)	**22.73 (3m)**
3	22	Receipt	Elapsed Time from Receipt to the last Invoice Cleared	28.94 (4h 25m)	18.09 (6h 46m)	47.72 (2h 53m)	16.98 (1h 41m)	19.98 (6m)	11.47 (9m)	**11.42 (10m)**
3	23	Invoice	Elapsed Time from Invoice to the last Goods Line Registered	4.31 (50m)	1.93 (2h 39m)	2.59 (1h 15m)	2.49 (1h 41m)	4.18 (1m)	**1.05 (3m)**	1.06 (4m)
3	24	Invoice	Elapsed Time from Invoice to the last SES Line Registered	18.37 (20m)	19.14 (1h 28m)	26.06 (1h 9m)	**16.59 (1h 8m)**	22.09 (2m)	17.14 (2m)	17.32 (3m)
1	25	Contract	Occurrence of Activity Purchase Order Blocked	0.33 (4h 45m)	0.37 (9h 12m)	0.51 (6h 4m)	0.52 (9h)	0.53 (2m)	0.48 (15m)	**0.60 (20m)**
1	26	Contract	Occurrence of Activity Invoice Pay Method Changed	0.38 (3h 35m)	0.50 (13h)	0.64 (6h 22m)	0.66 (12h 39m)	0.67 (2m)	0.70 (14m)	**0.74 (20m)**
1	27	Contract	Occurrence of Attribute Pay Type Late	0.73 (1h 6m)	0.75 (5h 38m)	**0.82 (7h 12m)**	**0.82 (14h)**	0.80 (2m)	**0.82 (14m)**	**0.82 (19m)**
3	28	Order	Occurrence of Attribute Pay Type Late	0.76 (1d)	0.78 (1d 2h)	0.75 (15h 45m)	0.76 (18h 50m)	0.80 (16m)	**0.83 (39m)**	**0.83 (42m)**
3	29	Order	Occurrence of Activity Invoice Pay Method Changed	0.34 (1d 9h)	0.47 (1d 4h)	0.47 (16h 12m)	0.50 (17h 29m)	0.46 (15m)	0.58 (39m)	**0.60 (41m)**
3	30	Order	Occurrence of Activity Invoice Released	0.06 (4d)	0.12 (1d 12h)	0.50 (16h)	0.28 (18h 27m)	0.56 (30m)	0.62 (1h 15m)	**0.63 (1h 17m)**

summarized in Table 5.4b.[2] First, they were both interested to know in advance whether there would be changes to the payment method (represented by the activity *Invoice Pay Method Changed*). When this activity happens, there are usually delays in payments. Moreover, they were interested to know whether there will be delays with the payments (represented by the attribute *Pay Type* assuming value Late). Lastly, the former company was also interested in predicting whether there will be problems with the order (represented by the activity *Purchase Order Blocked*) since this situation can bring additional delays caused by the reworks needed to fix the problem. Similarly, for the latter company, it was interesting to know whether there will be problems with the invoice since this situation can bring additional delays caused by the reworks (represented by the activity *Invoice Released*) needed to fix the problem.

We calculated the MAE for the 24 numerical KPIs as values were reasonably well balanced. By contrast, we calculated the F1-Score for the last six boolean KPIs. Finally, we report between parenthesis the training time required to train every prediction model for each KPI of interest.

[2]The last column reports the distribution of the classes. For example, the activity *Invoice Pay Method Changed* in the utility event log was performed in 26% of the cases.

Table 5.5 summarizes the results of our comparison. As mentioned before, our comparison takes into account four different predictive models, namely GCN, GGNN, LSTM, and Catboost.

As described in Section 5.3.3 and illustrated in Figure 5.2, the selection of the feature setting that have been used for the several prediction models have been chosen based on the best predictive accuracy that was observed on the validation set. For Catboost and LSTM, however, we included two different feature settings (*All Features* and *All+Aggregated Features*), since both of them were beneficial depending on the selected KPI. Moreover, we also included a baseline (*Catboost naive*) representing an example of existing technique that focuses only on one single process, thus being unable to deal with multiple interleaving processes. The baseline relied on Catboost, since the comparison illustrated in Chapter 3 highlighted a better accuracy compared to LSTM models when predicting the outcome in traditional processes.

First of all, examining the last three columns in Table 5.5, which refer to the results obtained with three different settings adopted for Catboost predictive model, we can immediately observe that *overlooking the object interactions has a great impact on the predictive accuracy: the approach that considers all the events directly or indirectly connected to objects of the selected viewpoint and the approach that also considers the aggregated features have the best predictive accuracy in almost all the considered KPIs.* The only exception is related to the results obtained for the second event log when considering the Requisition viewpoint (KPIs 12 to 15).

Moreover, examining the accuracy obtained when leveraging or not the aggregated attributes, it can be observed that leveraging the aggregated attributes improved LSTM model accuracy in 19 cases out of 30 (especially in the KPIs related to the second and the third case study), while it improved Catboost accuracy in 22 cases out of 30; this further confirms that taking into account the object interactions can often improve the accuracy, independently from the type of the adopted predictive model (LSTM attributes' encoding substantially differs from the encoding adopted by Catboost).

Furthermore, we observe that Catboost achieves the highest predictive accuracy among the four prediction models in almost all the considered KPIs; there is only one case (KPI 9) where the GCN obtains a slightly better predictive accuracy compared to Catboost with the aggregated features, while there are two cases (27 and 24) where, compared to Catboost, the LSTM obtains an equivalent and slightly better predictive accuracy, respectively. However, Catboost models *are generally learned significantly faster compared to other predictive models.*

We further compare the obtained results with the statistics of the selected KPIs in Table 5.4; in particular, for the first case study, we noticed that the event logs obtained for the numerical KPIs 3 to 7 and for the categorical KPIs 25 to 27 are those that contain more events. In these settings, the predictive accuracy of LSTM is closer to that of Catboost, and considerably outperforms those of GGNN and GCN. However, for the numerical KPIs 12 to 18 related to the second case study, while GGNN and LSTM consistently perform better than GCN and, except for one case (KPI 16), LSTM performs consistently better than GGNN, Catboost systematically outperforms other methods. Linked to the point above, LSTM can naturally learn from sequences of events, thus learning from the interaction among process objects. By contrast, GGNN and GCN tend to focus on adjacency matrices of nodes (i.e., events) in proximity, being less capable of reason on events that are indirectly connected. However, while the LSTM does not always outperform GGNN, Catboost systematically performs better than the other models because of the aggregated features, designed to capture the object-interaction. Conversely, for the numerical KPIs 8 to 11, which are characterized by fewer events, the GCN outperforms LSTM and shows a predictive accuracy relatively comparable to Catboost. Also for numerical KPIs 1 and 2, which are characterized by fewer events, the GGNN and GCN consistently outperform LSTM models. *From this, we can conclude that the GCN can occasionally have slightly better performances in the presence of limited amount of data (which instead poses an issue with LSTM) but, if enough data is provided, Catboost systematically outperforms graph-based approaches, which conversely struggle to learn more complicated interaction patterns.*

When the KPI is related to the (non) occurrence of a process' activity (e.g., Occurrence of Activity Purchase Order Blocked) that is seldom observed (see KPIs 25 and 26), we observed that Catboost models and LSTM networks significantly surpass graph-based neural networks. Particularly relevant is KPI 30, where the target activity to be monitored, *Invoice Released*, is very rare (it was observed only in the 3% of cases); here, the predictive accuracy of graph-based neural networks drops dramatically, and both Catboost and LSTM networks outperform graph neural networks. When the activity is more common (see KPIs 27 and 28), graph-based neural networks show better predictive accuracy (in KPI 28 graph-based models even perform slightly better than LSTM); however, their predictive accuracy remain poorer than that of Catboost models.

Finally, when the KPI is related to the (non) occurrence of a process' activity

(KPIs 25 to 30), we can see that the designed aggregated attributes can consistently increase the accuracy of the predictive model; in fact, it improves in 4 cases out of 6 for LSTM (in another case it achieves the same accuracy), while it improves in 4 cases out of 6 for Catboost (in the remaining two cases it achieves the same accuracy).

5.4 Explanations of Object-centric Process Predictions

As a further confirmation of the importance of object-interaction and aggregated features, we computed the Shapley Values in accordance to the framework discussed in Chapter 4; in particular, explanations were calculated on the test dataset. While the adopted explainability framework is independent of the actual technique employed to make the predictions, we reported here only on the explanation of predictions for the Catboost method. This choice is motivated by the fact that, as discussed in the previous section, Catboost is preferrable both to LSTM and graph neural networks for prediction, due to the increased speed to build the model and increased accuracy.

Figure 5.6 illustrates the boxplot representing the distribution of the Shapley values over the different trace prefixes for some of the most important features influencing the prediction of the KPI *Elapsed Time from Contract to the last Invoice Receipt* in the first event log for the approach with the aggregated features. In particular, each boxplot is ordered by the average Shapley value on the selected KPI considering the absolute value. Each row of the boxplot is linked to an explanation, which extends towards left or right, depending whether the observed Shapley values for the explanation were negative or positive. It can be clearly seen in Figure 5.6 that many aggregated features are indeed considered relevant by the predictive model; as an example, the most important explanation is *% ORDER_PURCH_GROUP=100_L50* $>$ *0.33* and the average associated Shapley value is -15 days: this means that, when more than the 33% of the orders are related to the purchase group *100_L50*, the estimated Elapsed Time from the signature of the Contract to the last Invoice Receipt reduces on average by 15 days.

A similar reasoning can be applied to Figure 5.7, which reports on some of the most important features influencing the prediction of the KPI *Elapsed Time from Contract to the last Invoice Cleared* for the approach without the aggregated features. As it can be seen, several features were added after considering

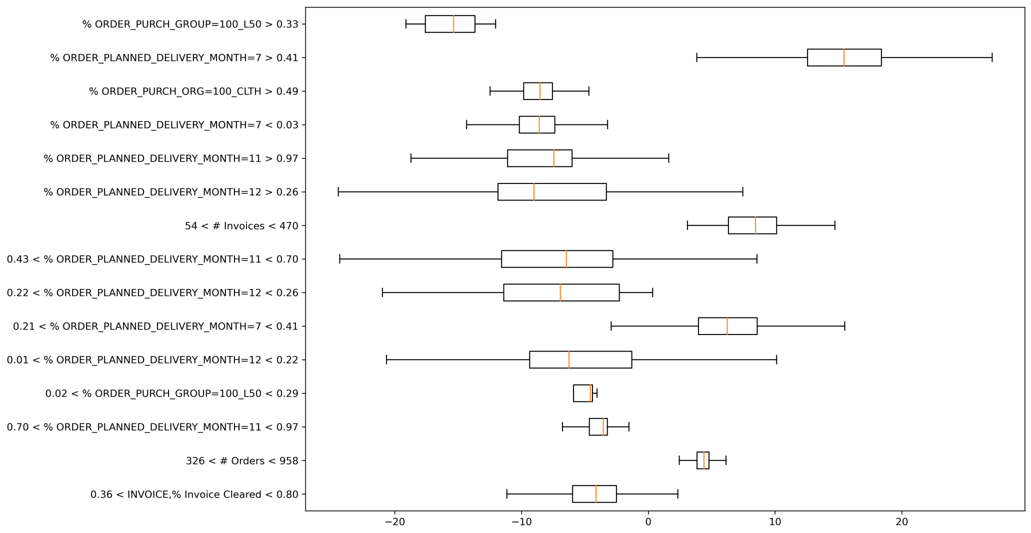


Figure 5.6: Boxplot representing the impact of some of the most important aggregated features that were added to the predictive model in order to improve the accuracy for the KPI *Elapsed Time from Contract to the last Invoice Receipt* related to the first event log in the approach with aggregated features.

the object interaction and were considered useful by the predictive model, such as *PAY_DUE_MONTH_ADJ* and the occurrence of the activity *Invoice Due Date Changed*; these are attributes related to the *Invoice* object type, whose values are related to the last opened invoice (or to the last performed activity).

However, since we do not use aggregated features, we miss several factors significantly influencing the prediction. Indeed, when we consider the aggregated features for the same KPI (*Elapsed Time from Contract to the last Invoice Cleared*) and compute the significance of the influence (i.e. the average Shapley value), we obtain the boxplots in Figure 5.8. It can be clearly seen that the aggregation attributes are now among the most important factors significantly influencing the prediction. As an example, the two most important factors that are contributing to increase the estimated time are represented by *%Order_planned_delivery_month=7 > 0.18* and *318 < #Orders < 966*. The former is associated with an average Shapley value of 20, which means that when more than 18% of the orders are delivered in July (month 7), the estimated time to clear the last invoice is 20 days larger that the average. The latter, associated with an average Shapley value of 8, indicates that when there are a lot of orders that needs to be managed at the same time (between 318 and 966), the estimated

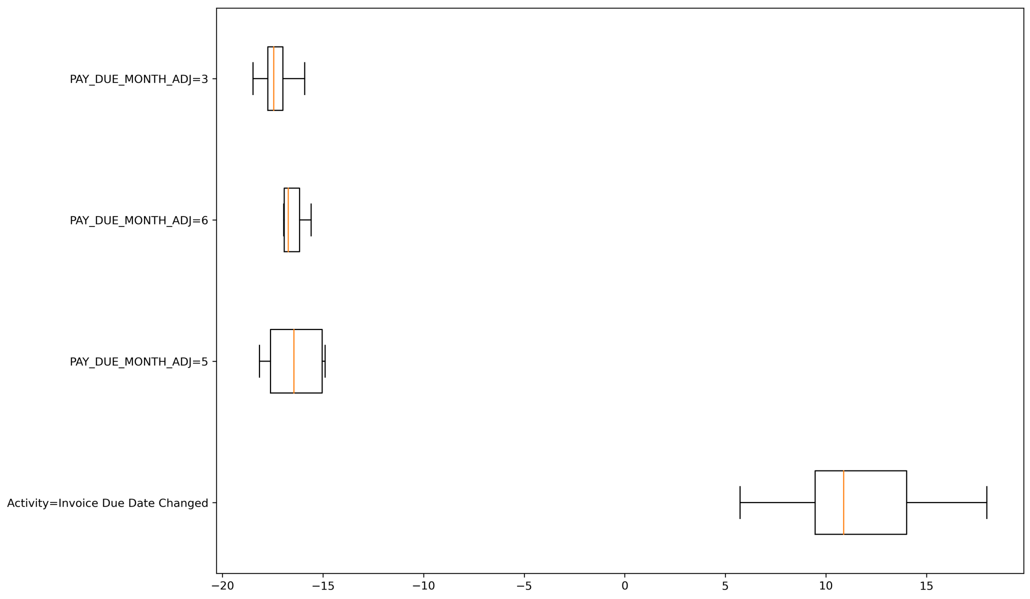


Figure 5.7: Boxplot representing the impact of some of the most important features that were added to the predictive model (such as the attributes related to the *Invoice* object type) in order to improve the accuracy for the KPI *Elapsed Time from Contract to the last Invoice Cleared* related to the first event log in the approach without aggregated features.

time to clear the last invoice is 8 days larger that the average. Conversely, one of the most important factors contributing to decrease the estimated time is *ORDER, %Goods Line Registered* $>$ *0.97*, which is associated with an average Shapley value of -10; this means that, when the activity Goods Line Registered (which represents the fact that the goods have been received) has been performed at least once in more than 97% of the orders, the estimated time reduces on average by 10 days.

Figure 5.9 illustrates the boxplot representing the distribution of the Shapley values over the different trace prefixes for some of the most important features influencing the prediction of the KPI *Elapsed Time from Order to the last Invoice Submit* in the second event log for the approach with the aggregated features. It can be clearly seen that many aggregated features are indeed considered relevant by the predictive model; as an example, the most important explanation is *% INVOICE_ID_VENDOR=100_1000197058* $>$ *0.17* and the average associated Shapley value is +70 days: this means that, when more than the 17% of the invoices are related to the vendor *100_1000197058*, the estimated Elapsed

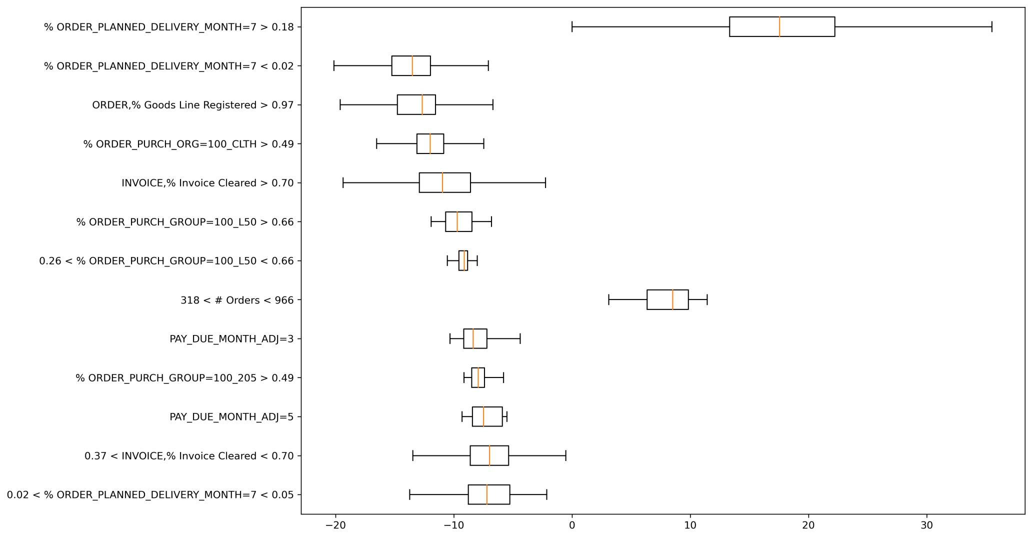


Figure 5.8: Boxplot representing the impact of some of the most important aggregated features that were added to the predictive model in order to improve the accuracy for the KPI *Elapsed Time from Contract to the last Invoice Cleared* in the approach with aggregated features.

Time from the request of the Order to the last Invoice Submit increases on average by 70 days. A further analysis of the data confirms this finding: when this Invoice Vendor is involved, the average duration of the process is 157d 9h, compared to an average process duration of 36d. Other important explanations are associated to the average number of items purchased by the customers, calculated considering all the orders that are being managed (represented by the attribute *Avg ORDER_LINE_AMOUNT*). As an example, the second most important explanation is *Avg ORDER_LINE_AMOUNT* $<$ *659* and it is associated with an average Shapley value of -34 days, while the third most important explanation is *Avg ORDER_LINE_AMOUNT* $>$ *25666* and it is associated with an average Shapley value of +26 days. This means that when the average number of purchased items (considering all the orders) is high ($>$ *25666*), the time needed to fulfill all the orders is increased by 26 days; conversely, when the average number of purchased items is lower ($<$ *659*) the time needed to fulfill all the orders is decreased on average by 34 days.

Regarding the third case study, we mentioned that the manufacturing company aims to avoid activities related to inefficiencies, such as *Invoice Pay Method Changed* (when this activity happens, there are usually delays in payments), and

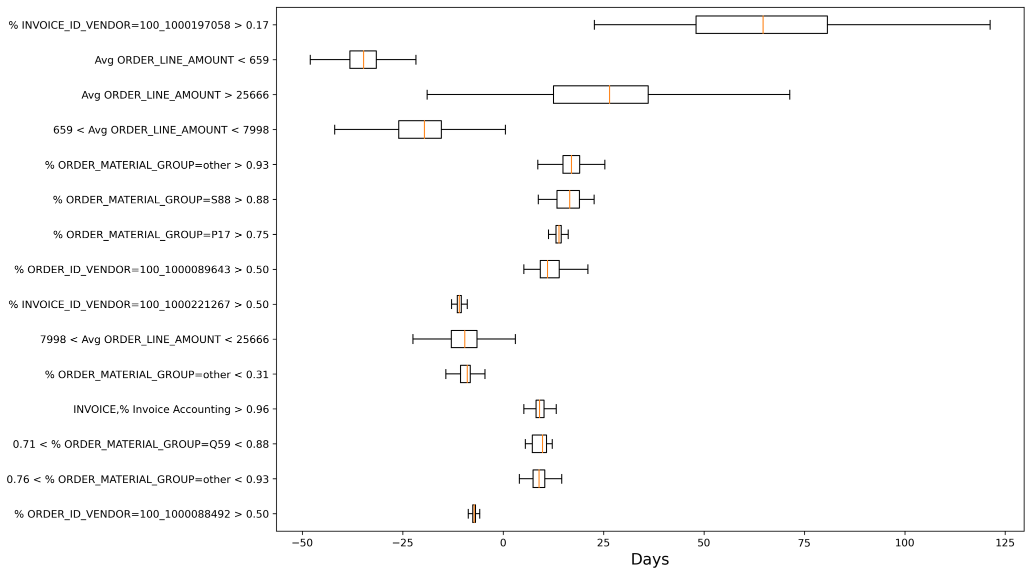


Figure 5.9: Boxplot representing the impact of some of the most important aggregated features that were added to the predictive model in order to improve the accuracy for the KPI *Elapsed Time from Order to the last Invoice Submit* related to the second event log in the approach with aggregated features.

Invoice Released (i.e. the activity that needs to be performed when problems with the invoice occur). The explanations related to the first activity are shown in Figure 5.10. As it can be seen, the two most important aggregated features that are contributing to decrease the probability that the Payment Method will be changed are *Avg ORDER_LINE_AMOUNT* < 8 and *INVOICE, % INVOICE Send Payment* > 0.72. The former indicates that when the average number of purchased items (calculated considering all the orders) is low (< 8), then the probability that the payment method will be changed is decreased by 37%. The latter means that when the payment has been already sent for more than 72% of the invoices, then the probability that the payment method will be changed is decreased by 30%. Analyzing the process, we noticed that when the Payment Method has to be changed, it is usually changed immediately after registering the invoice but before sending the payment; therefore, if the payment has been already sent for a lot of invoices, the probability that the payment method changes decreases. Conversely, the aggregated attribute *# Order* > 996 associated with a Shapley value of 0.22 indicates that when there are a lot of orders that need to be managed at the same time (more than 966), the probability that the payment method will be changed is

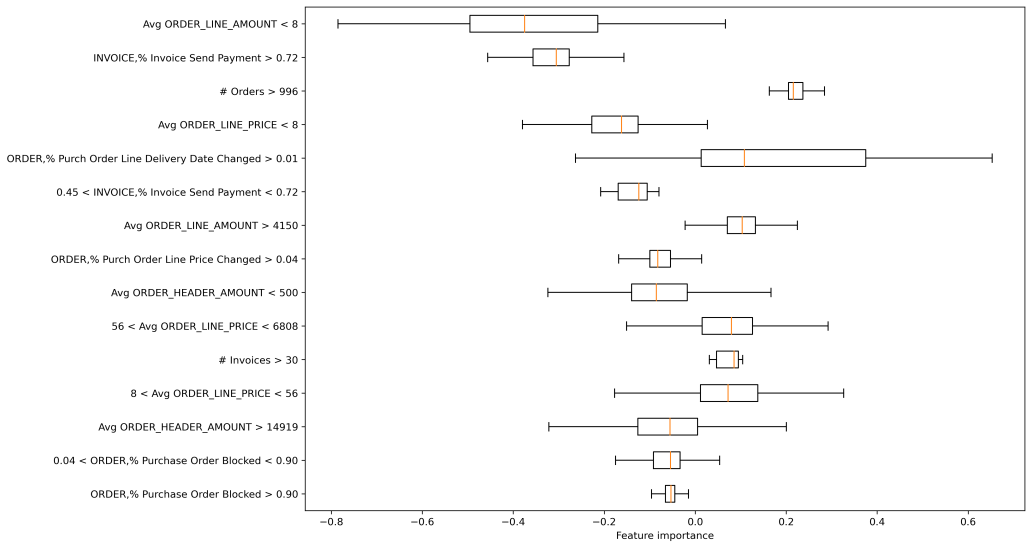


Figure 5.10: Boxplot representing the impact of some of the most important aggregated features that were added to the predictive model in order to improve the accuracy for the KPI *Occurrence of activity Invoice Pay Method Changed* related to the third event log in the approach with aggregated features.

increased by 22% compared to the average.

Finally, the explanations related to the occurrence of the second activity (*Invoice Released*), which is related to the third case study, are shown in Figure 5.11. Here, the two most important explanations are *0.08 < INVOICE, % Invoice Blocked: Locked for payment < 0.75* and *INVOICE, % Invoice Blocked: Locked for payment > 0.75*; they indicate that, when the percentage of blocked invoices among all the considered invoices is between 8% and 75% or is greater than 75%, then the probability of the occurrence of *Invoice Released* is increased by 65% and 63%, respectively. While these two explanations could seem trivial at a first glance (if one of the invoices is blocked, there will be the need to unblock it, i.e. to perform the activity *Invoice Released*), the evidence in the explanations demonstrates that Catboost allowed learning a prediction model that leverages on the aggregated attributes to correctly estimate the occurrence of *Invoice Released*. Notice also that, when one or more invoices have performed an *Invoice Released* at least once, the probability that also other invoices will perform *Invoice Released* is increased; this is represented by the two explanations *0.26 < INVOICE, % Invoice Released < 0.62* and *0.62 < INVOICE, % Invoice Released < 0.78*, which are associated with a probability of 58% and 48%, respectively.

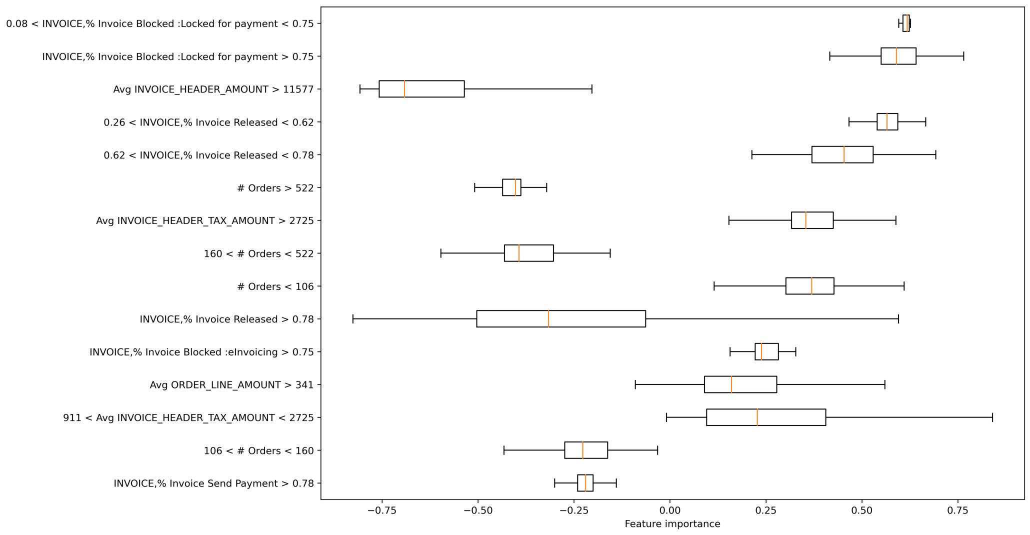


Figure 5.11: Boxplot representing the impact of some of the most important aggregated features that were added to the predictive model in order to improve the accuracy for the KPI *Occurrence of activity Invoice Released* related to the third event log in the approach with aggregated features.

5.5 Summary

The lion's share of attention in (Business) Process Management has traditionally been on designing and analyzing processes that are based on a unique notion of case identifier, with a single flow of execution from an initial state to one of the potential final states. In practice, organizations execute more complex processes that are often interacting with each other: one instance of a given process synchronizes with instances of other processes, possibly exchanging data. In light of the above, the object-centric process paradigm is nowadays attracting more and more attention in academia and industry. In this paradigm, the process is seen as the interplay of numerous sub-processes that constitute the life cycles of different objects, which periodically synchronize with each other.

This chapter tackles the problem of predictive analytics over object-centric processes. The large share of research in predictive analytics cannot be directly applied here, because it traditionally focuses on the problem of predicting the outcome of cases (i.e., process instances) that run in isolation. Also recent techniques that capture the inter-case dynamics assume instances to be of the same process, namely referring to the same object type. If the valuable information of the interaction between objects of the same or different type is not fed into the construction of prediction models, the resulting model might be of low accuracy.

A first viable solution proposed in this chapter, is to flatten object-centric event logs into single-identifier event logs and to enrich the latter via features that encode the object interactions. This solution approximates the graph-like structure of the object interactions, with some degree of information loss. The accuracy of the proposed approach that also encodes the object interactions has been firstly compared with that of a naïve approach, which only considers the events related to the process of the viewpoint objects. Experiments have shown that the naïve approach performs rather poorly, confirming the importance of our approach to consider the object interactions when predicting.

Typically, the operation of flattening the event log typically leads to a duplication of events, possibly strengthening existing directly-follows relationships or, even, adding new relationships that do not exist in reality. These problems, known as convergence and divergence [26, 110], make it impossible to apply process-mining techniques designed for single-flow processes that heavily rely on the directly-follow relationships, such as those for model discovery and conformance checking. Conversely, unfolding causes no problem in our process-prediction approach, because we do not use the direct-follow relationships.

After flattening the object-centric event logs into single-identifier event logs, this chapter reports on the experience of comparing the quality of the predictions obtained by leveraging four different techniques to tackle predictive analytics in object-centric processes. In particular, one technique is based on Gradient Boosting on Decision Trees, and three are based on Deep Neural Networks, including graph-based models. The four techniques were empirically evaluated on event logs related to three real object-centric processes, and 30 different KPI definitions. The experimental results show that the technique based on Gradient Boosting performs consistently better than those based on Deep Neural Networks, both in terms of accuracy and training time, and that considering the object interactions often improves the quality of the predictive model. Finally, Shapley Values and Explainable AI techniques have been leveraged to further confirm the importance of object-interaction and aggregated features for improving the prediction quality. In the next chapter, we illustrate how our framework for explainable predictive process monitoring can be extended from predictive to prescriptive analytics.

Chapter 6

Explainable Prescriptive Process Analytics

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 to not trust the PAR system and overlook these recommendations is high.

This chapter 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 potential relevance of these explanations for process owners is illustrated in two use cases.

6.1 Motivation

Recent literature has put forward a number of proposals of prescriptive analytics, whose focus is mostly on recommending the corrective actions, without providing the reasons that led the systems to propose the suggested recommendations [16, 121]. 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 the suggested corrective actions, which are based on data, but rather to enact contingency actions that are subjective and, thus, can even potentially worsen the KPI outcome [17].

This chapter 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 explaining the selected recommendations leverages on current state of the art of Explainable AI, specifically on the Shapley Values game-theory approach (cf. Section 2.3.2). 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 [16] 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 the 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.

6.2 Related works

Literature has largely focused on predicting the future outcome of process instances (cf. Section 3.2). A recent body of research has been focusing on recommending which activities to work on as next, to improve process' KPIs of interest, or to suggest the most common continuations [7, 16, 67, 121].

In parallel, several research works focused on explaining the reasons of these predictions and the affecting factors, using several approaches (cf. Section 4.2 and our approach based on leveraging the Shapley Values, which has been described in Chapter 4).

However, none of the existing works 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.

In this chapter, we always decided to rely on the Shapley Values in order to build our framework. As already mentioned in Section 4.2, we also considered using attention-based mechanisms. However, this type of approach can provide explanations only for neural network models; moreover, attention mechanisms are disputed whether or not they are always correlated to feature importance [87], while Serrano and Smith [92] claim that attention weights often fail in the task of finding the factors influencing the model's final decisions.

6.3 A Framework for Generating Recommendations

A process aware recommender-system aims to recommend the k-top best next activities in order to improve the relevant KPI. However, these activities need to be valid from a domain viewpoint. We avoid the strong assumption of having a process model in order to prescribe 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 6.3.1 (State-representation function)**.** *Let σ be a trace, and $\mathcal{R}$ a set of the possible representations. The function $l^{state} : E^* \to \mathcal{R}$ 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 [109].

Definition 6.3.2 (Transition system)**.** *Let l^{state} be a state-representation function, $\mathcal{L}$ an event log and E its set of events. A transition system abstracting $\mathcal{L}$ is a tuple $TS_{\mathcal{L}} = (S, T) \subseteq R \times (R \times E \times R)$ where*

- $\mathrm{S} = \cup_{\sigma \in \mathcal{L}} \cup_{\sigma' \in \mathit{prefix}\,(\sigma)} l^{state}\ (\sigma')$

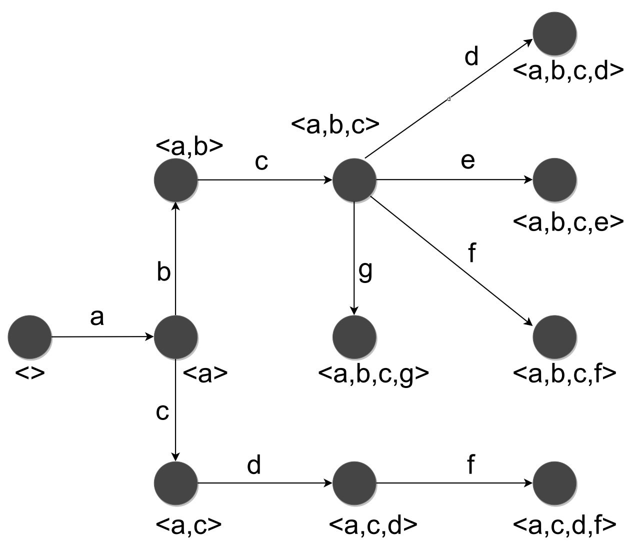


Figure 6.1: Transition system based on the example log $\mathcal{L}^{ex} = \{\langle a,b,c,d\rangle, \langle a,b,c,e\rangle, \langle a,b,c,f\rangle, \langle a,b,c,g\rangle, \langle a,c,d,f\rangle\}$

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

Fig 6.1 shows an example of a transition system in accordance with Def. 6.3.2. It has been built on an event log $\mathcal{L}^{ex} = \{\langle a,b,c,d\rangle, \langle a,b,c,e\rangle, \langle a,b,c,f\rangle, \langle a,b,c,g\rangle, \langle a,c,d,f\rangle\}$[1], using a *sequence-based state-representation function* $l^{state}_{sq}(\langle e_1,\ldots,e_n\rangle) = \langle \pi_{act}(e_1),\ldots,\pi_{act}(e_n)\rangle$. 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 in order 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 intelligible, but this poses no threat because it is only used internally and it is never shown to process' actors.

Let us assume that a process instance 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 activi-**

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

ties $\mathcal{A}_{\sigma_R}$ 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 $TS_{\mathcal{L}}$: $\mathcal{A}_{\sigma_R} = \{\pi_{act}(e) : \exists \left(l^{\text{state}}\left(\sigma'\right), e, l^{\text{state}}\left(\sigma' \oplus \langle e \rangle\right)\right) \in T\}$. Then, for every activity $act \in \mathcal{A}_{\sigma_R}$, we evaluate the expected KPI of $\sigma_R \oplus act$ using the oracle function (i.e. $\Phi_{\mathcal{K}}(\sigma' \oplus act)$). Here and later, $\sigma \oplus act$ indicates the trace σ extended with an event $(act, t, \mathbb{V})$ where t is the timestamp of the last event in σ, and $\mathbb{V} : AN \rightarrow \{\perp\}$ with $\mathbb{V}(a) = \perp$ for all $a \in \mathcal{AN}$. Here, the attributes are given a value $\perp$ to indicate that a value was not assigned for the specific event. The definition of $\mathbb{V}$ reflects the uncertainty on the values that are going to be assigned to attributes through the execution of activity act. As many other predictive methods, Catboost is able to interpret and deal with these missing values, namely attributes whose value is unknown. [2]

This procedure associates each activity with the corresponding expected KPI value, establishing a ranking of possible next activities; afterwards, we recommend the first k best activities act_{rec} (with k customizable), namely those associated with the best expected KPI values.

To give a better intuition of how our recommender system works, let us suppose that we have a running trace $\sigma' = \langle a, b, c \rangle$ and that we aim to estimate what will be the total time of the trace upon completion. According to our transition system illustrated in Fig 6.1, the possible next activities allowed for this trace could be only *d*, *e*, *f* or *g*. Afterwards, we leverage our Catboost model, which has been previously trained on the completed traces, in order to give a total time estimation for each of the possible continuations $\langle a, b, c, d \rangle$, $\langle a, b, c, e \rangle$, $\langle a, b, c, f \rangle$, $\langle a, b, c, g \rangle$. Finally, we can establish a ranking of the possible continuations, suggesting the ones (depending on the threshold k) associated with the lowest predicted total time. Let us suppose that the threshold k is configured to 1; in the example represented in Tab 6.1, it would be recommended to perform the activity f as next activity, since it is associated with an expected total time of 261h 56min, which is the lowest one among all the possible continuations.

6.4 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.

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

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

Table 6.1: 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 Fig6.1, the KPI is the total time, and so we aim to minimize it ($\sqsupset_{\mathcal{K}}$ assumes the value of the $<$ operator). In this example, we would recommend f, d and e, in this order of preference (assuming k to be set to 3).

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 and 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 the next recommended activities act_{rec} as defined in the previous section, we first evaluate the vector of its associated Shapley values $\Psi_{\mathcal{K}}(\sigma')$, using the SHAP function as described in 4.3.2. We then compute the vector $\Delta(\sigma', act_{rec})$ of the element-wise difference between $\Psi_{\mathcal{K}}(\sigma')$ and $\Psi_{\mathcal{K}}(\sigma' \oplus act_{rec})$, namely between the Shapley values before and after executing act_{rec}:

$$\Delta(\sigma', act_{rec}) = \Psi_{\mathcal{K}}(\sigma') - \Psi_{\mathcal{K}}(\sigma' \oplus act_{rec}) \tag{6.1}$$

Reminding that $\mathcal{A}_{\sigma'}$ is the set of possible next activities and $\Phi_{\mathcal{K}} : X_1, \ldots, X_m \to \mathbb{R}$ 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 $\mathcal{A}_{\sigma'}$. So $\exists act \in \mathcal{A}_{\sigma'}\ s.t.\ \Phi_{\mathcal{K}}(\sigma') > \Phi_{\mathcal{K}}(\sigma' \oplus act)$, and act_{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 $\mathcal{A}_{\sigma'}$, namely $\nexists act \in \mathcal{A}_{\sigma'}\ s.t.\ \Phi_{\mathcal{K}}(\sigma') > \Phi_{\mathcal{K}}(\sigma' \oplus act)$. In this case, the provided recommendation act_{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 $\Delta(\sigma', act_{rec})$ associated with the top-k larger values, namely the Shapley values that decrease the most after executing act_{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 act_{rec} reduces the influence of feature $f_i = x_i$ of a quantity equal to* $\Delta(\sigma', act_{rec})[f_i]$". As an example, let us consider that the KPI is the total time of a process execution (the lower the value the better it is). 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 $\Delta(\sigma', act_{rec})[f_i] > 0$. Theorem 6.4.2 below guarantees that it is always the case. The proof requires one intermediate lemma:

Lemma 6.4.1 (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 6.4.2. *Let $\sigma_1, \sigma_2 \in E^*$ two different traces, $\Phi_{\mathcal{K}} : X_1 \times \ldots \times X_m \rightarrow \mathbb{R}$ be the oracle function and $\Psi_{\Phi_{\mathcal{K}}} : E^* \rightarrow \mathbb{R}^m$ 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 [68] 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

$$\sum_{i=1}^{m} \Psi_{\Phi_{\mathcal{K}}}(\sigma)[f_i] + \overline{\Phi_{\mathcal{K}}(\mathcal{X}))} = \Phi_{\mathcal{K}}(\sigma)\ \forall \sigma \in E^* \tag{6.2}$$

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

$$\begin{cases} \sum_{i=1}^{m} \Psi_{\Phi_{\mathcal{K}}}(\sigma_1)[f_i] + \overline{\Phi_{\mathcal{K}}(\mathcal{X}))} = \Phi_{\mathcal{K}}(\sigma_1) \\ \sum_{i=1}^{m} \Psi_{\Phi_{\mathcal{K}}}(\sigma_2)[f_i] + \overline{\Phi_{\mathcal{K}}(\mathcal{X}))} = \Phi_{\mathcal{K}}(\sigma_2) \end{cases} . \tag{6.3}$$

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 6.4.1, we obtain that exists at least a feature f_j for which $\Psi_{\Phi_{\mathcal{K}}}(\sigma_1)[f_j] > \Psi_{\Phi_{\mathcal{K}}}(\sigma_2)[f_j]$ $\square$

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 $\Delta(\sigma', act_{rec})$ associated.

The discussion above corresponds to the scenario where there is a recommendation that predicts to improve 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 in order to train the oracle function has been already described in Chapter 3; in particular, we leveraged the libraries Pandas and NumPy. The explanations are provided using the library provided by SHAP, in its framework dedicated to CatBoost[3].

6.5 Recommendations results

Although this chapter's focus is on the explanations of the recommendations, it is clear that explanations only make sense for recommendations that can positively

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

affect the process' KPIs. Section 6.5.1 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 a test log, used for the evaluation. This section concludes by illustrating the test-log usage to assess the quality of recommendations. Section 6.5.2 provides details on the considered datasets, while Section 6.5.3 reports on the evaluation of the recommendations' quality.

6.5.1 Experimental Settings and Recommender Evaluation

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 6.2. 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 act for the running trace $\sigma' \in \mathcal{L}^{trunc}$ is evaluated as the average KPI of traces similar to $\sigma' \oplus act$, belonging to $\mathcal{L}^{run}$:

$$acc(act, \sigma') = avg_{\sigma \in \mathcal{L}^{sim}_{act,\sigma'}} \mathcal{K}(\sigma) \tag{6.4}$$

where

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

And l^{state}_{sq} is the *sequence state-representation function* (cf. Section 6.3).

6.5.2 Datasets

The validity of our approach was assessed using two different event logs. 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. We already provided statistics about this event log in Chapter 3. here, we obtained 22,013

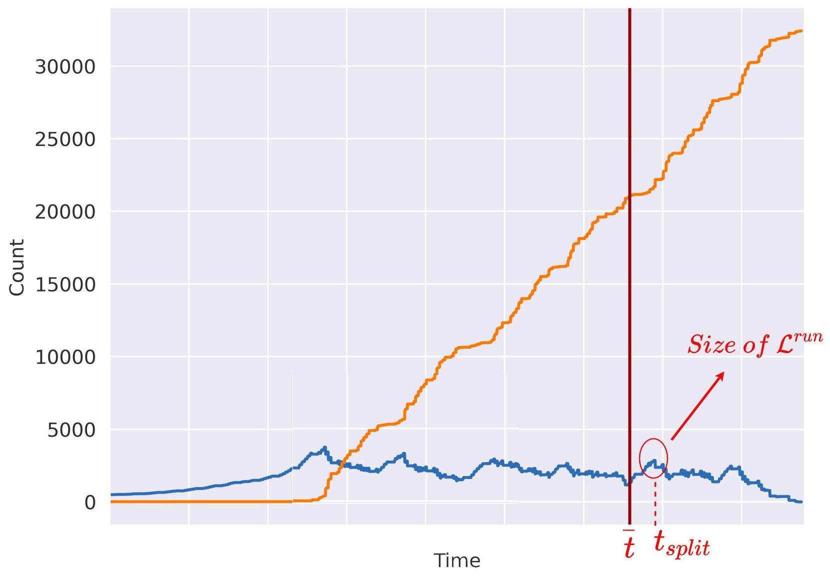


Figure 6.2: The orange and blue lines represent the number of completed and active traces, respectively. The vertical red bar is the timestamp $\bar{t}$ when the 65% of traces have completed.

for training and 10,286 for testing. Let us recall that 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. In this considered use case, we aimed 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.

The second log was used by the BPI challenge in 2013[4]. 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. When selecting the traces from the log in order to obtain a train and test set, we generated a training log of 5,103 traces and a test log of 2,236 traces. In this considered use case, we aimed to decrease the total execution time.

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

Use Case	**Average KPI value 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 6.2: Comparison of the average KPI value observed when the process execution is carried out without recommendation, versus the scenario when the best recommendation is always followed. The KPI value of the first use case is given as probability of occurrence, while the second is the actual time value. Activity *PLR* is a shortcut name for *Pending Liquidation Request*

6.5.3 Evaluation results

Table 6.2 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 average KPI value observed (cf. Equation 6.4) when recommendations are or are not followed, respectively. Given a trace σ', the average KPI value $acc(act, \sigma')$ is computed for the activity act that is the top recommended activity. Conversely, the average KPI value $acc(act', \sigma')$ is computed for the non-recommended activity act' that follows in the trace σ of which σ' is prefix. Last columns highlights the percentage improvement between the second and the third column, computed as $\Delta = (1 - followed/not\_followed) * 100\%$ with $followed$ and $not\_followed$ being the average KPI value when recommendations are followed or not followed. Results show a sensitive improvement for both cases, especially for the minimization of the occurrence of the rework activity *Pending Liquidation Request*. Further investigation is out of scope for this chapter: we only aimed to illustrate the validity of the recommendations on which explanations are computed. The latter is the novel contribution that is illustrated in this Chapter.

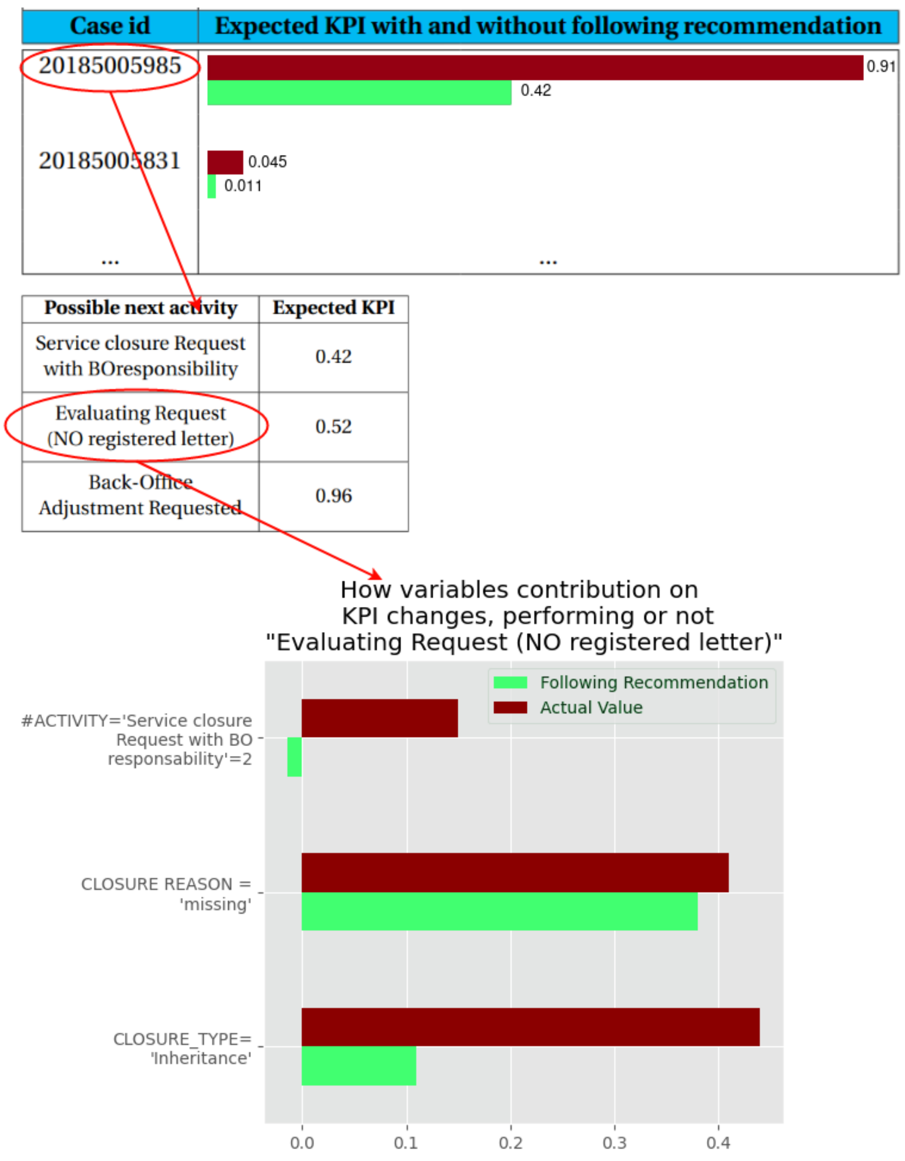
6.6 Provision and Display of Recommendations Explanations

Figure 6.3 shows how explanations are expected for our PAR system, for the use case of the process of Bank Account Closure, where we aim to minimize the occurrence of the activity *Pending Liquidation Request*. The outcome is a list that, for each running case, illustrates the expected KPI value when the recommendation is or is not followed, respectively. Figure 6.3 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 is 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 6.3). Let us assume that the process actor opts to perform the second best recommended activity, circled in red in figure; in this way, by performing the activity *Evaluating Request (No Registered Letter)*, the probability of the occurrence of *Pending Liquidation Request* would be reduced to 52%. In this example, we decided to consider the second best action to highlight the fact that the process actor might have reasons based on aspects not modelled in the process to not choose the recommendation that minimizes the probability (e.g., performing the activity could require accesses to systems that are currently under maintenance). When the recommendation is selected, the explanations are provided in the form of a bar chart (see bottom of Figure 6.3):

- Currently, the fact that the activity *Service closure Request with BO responsibility* has been executed twice contributes to increasing the probability of the occurrence of the undesired activity by ca. 15% (value 0.15); conversely, by following the recommendation and executing the activity *Evaluating Request (No Registered Letter)*, the aforementioned contribution would be nullified. In fact, we can clearly see that the double occurrence of the activity *Service closure Request with BO responsibility* would become a positive contribution, reducing the probability of the occurrence of the undesired activity.

- In the second explanation, we can see that the fact that the reason for the bank account closure is unknown contributes to an increase of ca. 41% of the probability of the undesired activity to occur. By executing the recommended activity, this contribution lowers to ca. 38% and can be partly mitigated.

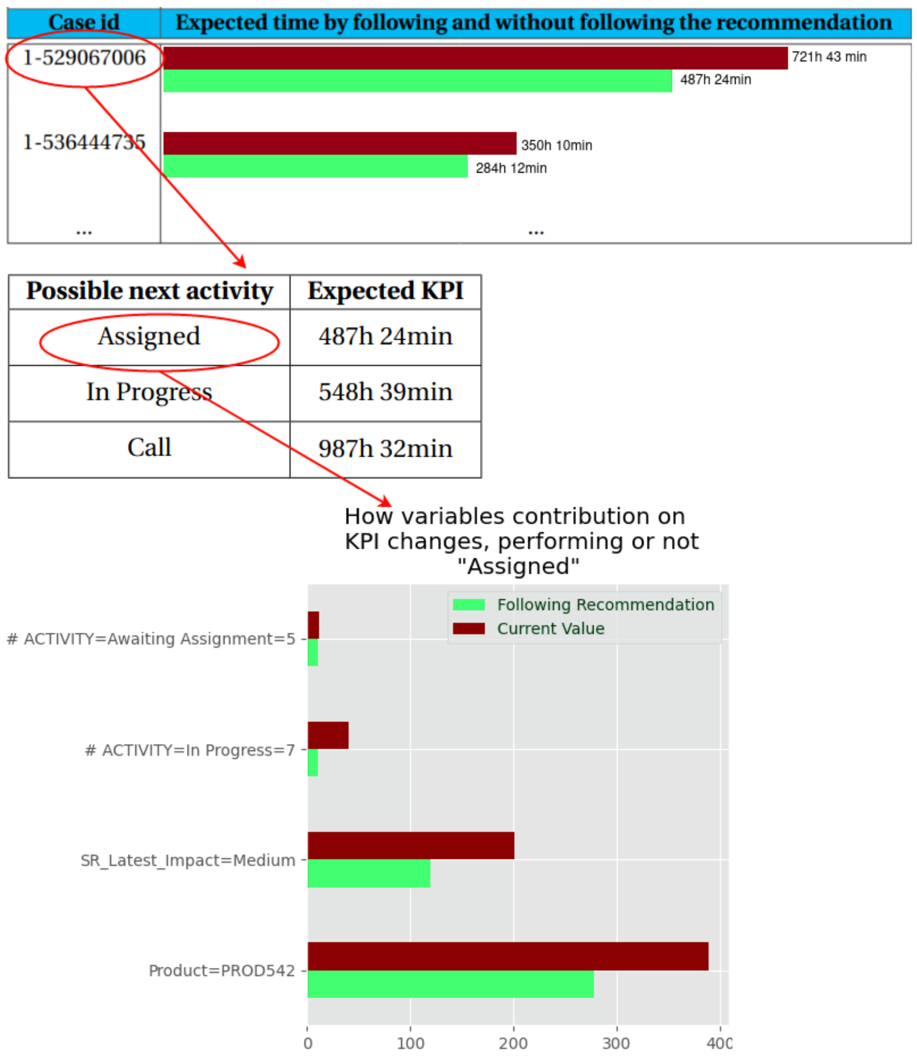
- Similarly to the point above, following the recommended activity allows mitigating the influence of the type of the bank account closure (*Inheritance*); in fact the probability contribution to the occurrence of the undesired activity drops from an original value of ca. 45% to ca. 11%.



Case id	Expected KPI with and without following recommendation
20185005985	🟥 0.91 🟩 0.42
20185005831	🟥 0.045 🟩 0.011
...	...

Possible next activity	Expected KPI
Service closure Request with BOresponsibility	0.42
Evaluating Request (NO registered letter)	0.52
Back-Office Adjustment Requested	0.96

Figure 6.3: Example of output produced by the PAR system when providing recommendations in order to minimize the KPI value related to the (undesired) occurrence of the target activity *Pending Liquidation Request*. Explanation label `#ACTIVITY=actname=actnumber` means that the activity named ”act name” happened ”act number” times



Case id	Expected time by following and without following the recommendation
1-529067006	721h 43 min 487h 24min
1-536444735	350h 10min 284h 12min
...	...

Possible next activity	Expected KPI
Assigned	487h 24min
In Progress	548h 39min
Call	987h 32min

Figure 6.4: 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.

The explanations were visualized in accordance with the frameworks for Explainable AI of black-box models (cf. Section 4.2). Here, *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 are not influenced by the recommendations. Note that, for the specific example in question, the pairwise sum of the difference of the Shapley's values of these three explanations is 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. Other explanations could have been selected, but with a smaller impact on reducing the probability for the activity to occur.

Figure 6.4 shows a similar output for the VINST process (cf. Section 6.5.2), where the aim was 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 of the product being *PROD542* from almost 400 hours to ca. 280 hours.

6.7 Summary

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 chapter presents the first attempt to report on a framework that explains process' recommendations. 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 these process executions.

This chapter 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. Moreover, we aim to test our explainable prescriptive framework on other publicly available datasets and KPIs, in order to further validate the findings reported in this chapter. 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. [56]).

Chapter 7

Conclusions

Process monitoring consists in analyzing historical process execution data in order to gauge process performance with respect to a set of performance objectives, providing a realtime overview of process performance and identifying performance issues as they arise. Recently, the rapid adoption of enterprise systems with logging capabilities has increased the development of data-driven predictive process monitoring, which exploits historical process execution data to predict the future outcome of ongoing business process instances. Thus, potential deviations from the expected process behaviour can be anticipated and proactively addressed. To this end, various approaches have been proposed to tackle typical predictive monitoring problems, such as whether a package will be correctly delivered in a delivery company or if the customer will be satisfied.

The analysis of several research proposals in predictive business process monitoring revealed that Long Short-Term Memory networks generally outperform other methods. However, since in production environments also the time is an important constraint, we wanted to evaluate alternative models that could be trained in a shorter amount of time, such as Catboost, which was also proposed by other research works in the process management domain. Afterwards, we empirically evaluated the quality of the predictions produced by LSTM and Catboost models; the experiments conducted on 6 datasets and 17 different KPIs highlighted that Catboost not only can be trained in a shorter amount of time, but it can also generally outperform LSTM models. Therefore, we decided to leverage Catboost for our predictive monitoring framework.

The purpose of a predictive process monitoring system is to provide operational decision support for process actors. Namely, given a prediction, the user can decide to mitigate the effects of a likely undesired outcome. However, the analysis of the literature has shown that the prescriptive part has often been overlooked, assuming that the users, after being alerted of a potential failure, are able to autonomously find the proper corrective actions. Therefore, the first contribution of this thesis consisted in developing a process-aware recommender system that not only aims to monitor and predict how process instances are going to evolve, but also recommends the corrective actions to recover the instances with higher risk to not achieve the desired outcome. In particular, we developed a PAR system able to recommend actions not only for improving a specific KPI (e.g. reducing the total time of a process), but also for improving a generic, user-customizable KPI. Note that we aimed at recommending the best actions that could be performed in order to optimize a particular KPI, instead of recommending just the next likely ones. Experiments were run on two real-life datasets, which confirmed the quality of the recommendations to improve the KPIs of interest.

The literature review also revealed that predictive process monitoring techniques are traditionally evaluated in terms of conventional performance measures, such as accuracy or F-score; however, in order to gain the trust of users, a necessary condition is to explain the reason of the provided predictions and recommendations; otherwise, the PAR system would not be trusted and adopted. Therefore, it has become more and more evident that conventional performance measures of predictive models are insufficient, and model interpretability/explainability needs to be incorporated into this assessment. Explainable AI techniques have been overlooked, assuming that a good level of prediction's accuracy is sufficient for the process' stakeholders to trust the recommender system (as well as the prediction system). As a second contribution, in this thesis we equipped our predictive-monitoring framework with explainable capabilities, by proposing an explainable framework based on the Shapley Values game theory approach, which can be adapted to explain any predictive model. In order to show the validity of the explanations provided, our explanation strategy was applied to several publicly available datasets; after analyzing the data, the evidence in the explanations demonstrated that the developed predictive framework leveraged attributes that were found to be relevant from a domain viewpoint.

Afterwards, in order to demonstrate the practical application of the research conducted in this thesis, we integrated our explainable predictive framework as a module of a commercial software, the IBM Process Mining Suite. This enabled us to provide process stakeholders with a ready-to-use module that provisions online operational support for their processes, as well as the influencers driving them, without requiring any specific technical knowledge.

Moreover, we conducted a user evaluation to assess the efficiency and effectiveness of the proposed explainable predictive process analytics module. The evaluation confirmed that predictions are actually explained in a form that is effective and efficient for process analysts and that the proposed explainable framework was considered intelligible by the users; furthermore, process stakeholders were satisfied with the explainable predictive process framework.

The analysis of the literature also highlighted that none of the existing works could provide an explanation of the recommendations. As a further contribution, we extended our proposed PAR system with explainable capabilities, to provide process actors with the rationale behind the choice of the recommended activities; in fact, recommendations without sensible explanations prevent process actors from feeling engaged in the decision process of the recommendations, thus not following the suggested corrective actions. Let us recall that explaining recommendations is certainly different than explaining predictions: while explaining the predictions focuses on every aspect that significantly affects the expected process' outcome, when explaining the recommendations we principally focused on highlighting the principal factors that had a negative impact on the KPI, but whose influence could be also largely mitigated by following the proposed recommendations.

The last contribution of the thesis is the proposal and application of a predictive analytics framework to predict the outcome of object-centric processes. In this paradigm, the process is seen as the interplay of numerous sub-processes that constitute the life cycles of different objects, which periodically synchronize with each other. The large share of research in predictive analytics could not be directly applied here, because it traditionally focuses on the problem of predicting the outcome of cases (i.e., process instances) that run in isolation. Moreover, if the valuable information of the interaction between objects of the same or different type is not fed into the construction of prediction models, the resulting model might be of low accuracy. Therefore, after proposing an approach to enable predictive analytics in object-centric processes, we enriched our predictive framework by exploiting the information of the complex sub-processes interaction; the experiments conducted on three real object-centric processes and 30 different KPI definitions indeed showed an increased prediction accuracy when considering the object interactions. Finally, we also leveraged our explainable framework to further confirm that the principal factors influencing the predictive model were indeed related to attributes that were designed to describe the complex object interactions.

7.1 Future Work

This thesis has laid the foundations of an explainable predictive and prescriptive process monitoring framework for custom business KPIs. At the same time, our research contributions open up a number of directions for future research.

In particular, we have designed and evaluated a technique for explainable predictive and prescriptive process monitoring based on the Shapley Values game theory approach, which leverages the idea of fairly distributing the payout among the players (features) that have collaborated in a cooperative game (i.e. the prediction task). The first future direction could be to evaluate alternative explainable approaches that have been emerging such as counterfactual approaches; here, the focus is on explaining the outcome of a black-box model through a hypothetical 'what-if' scenario, understanding what features should be changed in order to achieve a desired outcome [78]. To give a better intuition, an example of a counterfactual explanation would be: if the person's income were higher than $15,000, then he/she would have been granted a loan? Therefore, it could be worth comparing several explanations strategies in order to understand potential limitations and advantages of different explainable approaches.

To this end, another interesting future direction would be to define a clear quality criteria for Explainable AI techniques and objectively verify the validity of the explanations against it (see, e.g. [56]). In fact, there are numerous criteria to choose from that have been emerging in the last years; as an example, in literature several popular criteria are mentioned to assess the quality of an explanation method, such as performance, appropriate trust, explanation satisfaction and fidelity. However, there is no general consensus on which criteria should be used.

Afterwards, the approach proposed in this Ph.D. thesis to enable predictive analytics on object-centric event logs relies on flattening object-centric event logs into event logs of single-flow processes, which are enriched to maintain a meaningful abstraction of the object interactions. However, different techniques could have been used to extract and calculate process executions from object-centric event logs, discovering potentially different and novel insights.

Finally, a natural extension of the proposed recommender system would be to integrate it inside the IBM Process Mining suite, and assess what would be an optimal user experience that would help process workers and operational managers adopt it. Therefore, a future direction would be to validate if the provided recommendations would be understandable and easy to use for the end users, and how explainable the resulting recommendations are. In fact, the reasoning behind the recommendations must be clear in order to increase the trust, otherwise users would not follow and adopt the recommended actions. Similarly, we plan on leveraging the feedbacks received from the conducted user evaluation on the explainable predictive framework, in order to improve the general user experience in IBM Process Mining and improve our explanation strategy.

Appendix A

Additional explanations evaluations

This appendix is intended to report on the outcome of our global explanation strategy during the offline phase, in order to show the validity of the explanations provided. In particular, we extend the description provided in Section 4.5 by including additional experiments with different KPIs also on the other publicly-available event logs.

A.1 Bank Account Closure

The bar chart related to *Back-Office Adjustment Requested* prediction (Figure A.1) shows that the attributes related to the type and the reason of bank account closure are influencing the most.

When the closure type is Porting, it indicates that the customer has decided to move one or more services and the current-account balance from one bank account to another. Here, it is the most influential factor and its influence is towards strongly increasing the probability that a Back-Office Adjustment will be requested (as it can be seen by the red bar associated with a value of 0.85). Also the influence of *Closure_type=Bank Recess* is towards increasing the probability; a further analysis of the data confirmed that more than 2/3 of the total back-office adjustments is performed when the process is characterized by one of these two closure types. When a customer decides to open a new bank account in the same department of the previous bank account and decides to close the old one, it is less likely that a Back-office Adjustment will be performed; this can be seen by looking at the explanation *Closure_reason=4 - Open new bank account. Same dip*, which is associated with a negative probability of -0.20. Finally, an

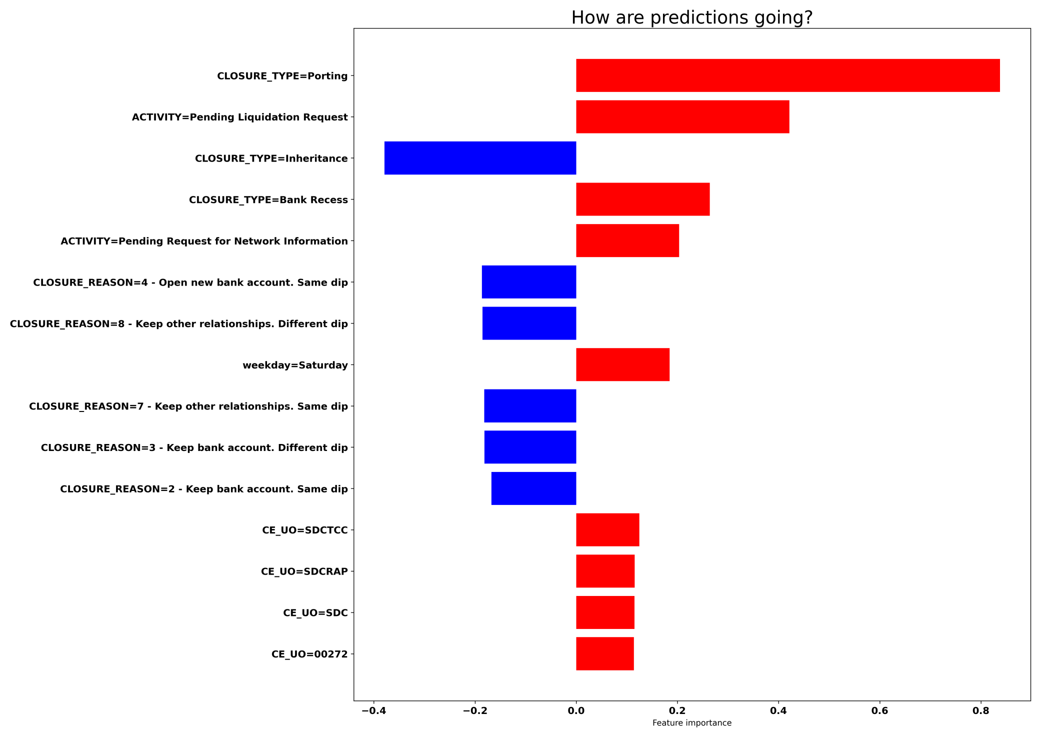


Figure A.1: Offline explanations for *Back-Office Adjustment Requested* prediction (Bank Account Closure)

interesting thing to be noticed is that when one or more of the activities previously performed in the process were performed on Saturday, then it's more likely that a Back-Office Adjustment will be needed in the future, as highlighted by the explanation *weekday=Saturday* associated with the positive value 0.20.

The results for the third activity on which the bank decided to focus (*Pending Request for Acquittance of Heirs*) are reported in Figure A.2. Here, the closure type is one of the most influential factors. In particular, the fact that the closure type is Inheritance (*Closure_Type=Inheritance*) strongly influences towards predicting that a *Pending Request for Acquittance of Heirs* will occur; this can be seen by the large red bar associated with a value that is near 1. Analyzing the data, it has been seen that a *Pending Request for Acquittance of Heirs* is performed most of the times when the closure type is Inheritance, while it is very unlikely to be performed when other closure types are performed. This also reflects in the blue bars related to the other three types of closure type (*Closure_Type=Client Recess*, *Closure_Type=Bank Recess* and *Closure_Type=Porting*), which are associated with a value -0.60. Moreover, in the lower part of the plot, it can be seen

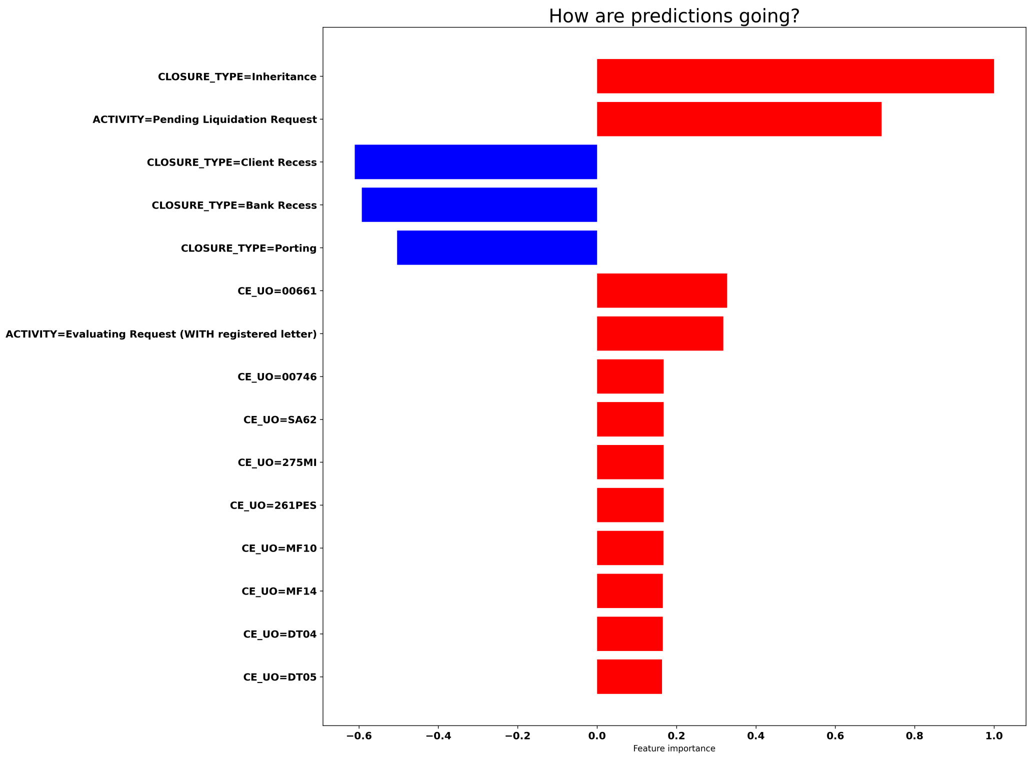


Figure A.2: Offline explanations for *Pending Request for Acquittance of Heirs* prediction (Bank Account Closure)

that other attributes that are influencing the prediction by slightly increasing the probability are related to the resource (shown in the plot as *Ce_Uo*) that has performed one or more of the activities preceding *Pending Request for Acquittance of Heirs*.

A.2 BPIC 2012

This is the dataset from BPI 2012 challenge, and it represents an application process for a personal loan or overdraft within a global financing organization. The event log is a composition of three merged intertwined sub processes. The first letter of each task name identifies from which sub process (source) it originated from. Figure A.3 shows the application for the remaining time prediction in the bpi12 dataset.

It can be clearly seen by the color blue of the bars and the negative values that the influence of the principal factors is towards decreasing the remaining

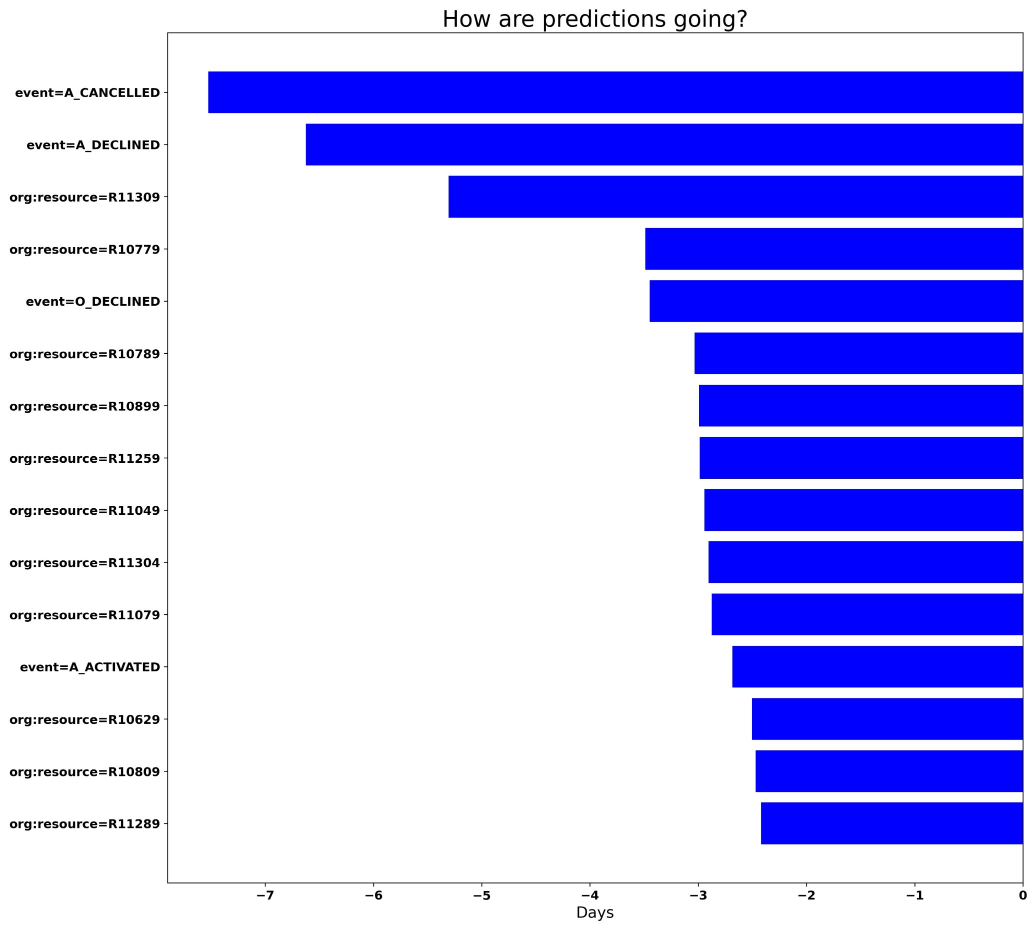


Figure A.3: Offline explanations for Remaining time prediction (BPIC 2012)

time prediction. In particular, the fact that a particular activity has been previously performed in the process, is one of the largest factors that influences the prediction. This can be seen in the explanations *event=A_CANCELLED*, *event=A_DECLINED*, and *event=O_DECLINED*, which are associated to the values -7.5, -6.5, and -3.5 respectively. From a domain viewpoint, the application for a personal loan can be rejected by the financial organization (action represented by the task A_DECLINED), or it can be directly closed because the user decides to cancel the request (represented by the task A_CANCELLED); therefore, when these actions are performed, the loan application process will be suddenly interrupted and it will finish earlier compared to the average duration of the application process, motivating the very high negative impact on the remaining time prediction. This also happens when the offer that has been sent by the organization is

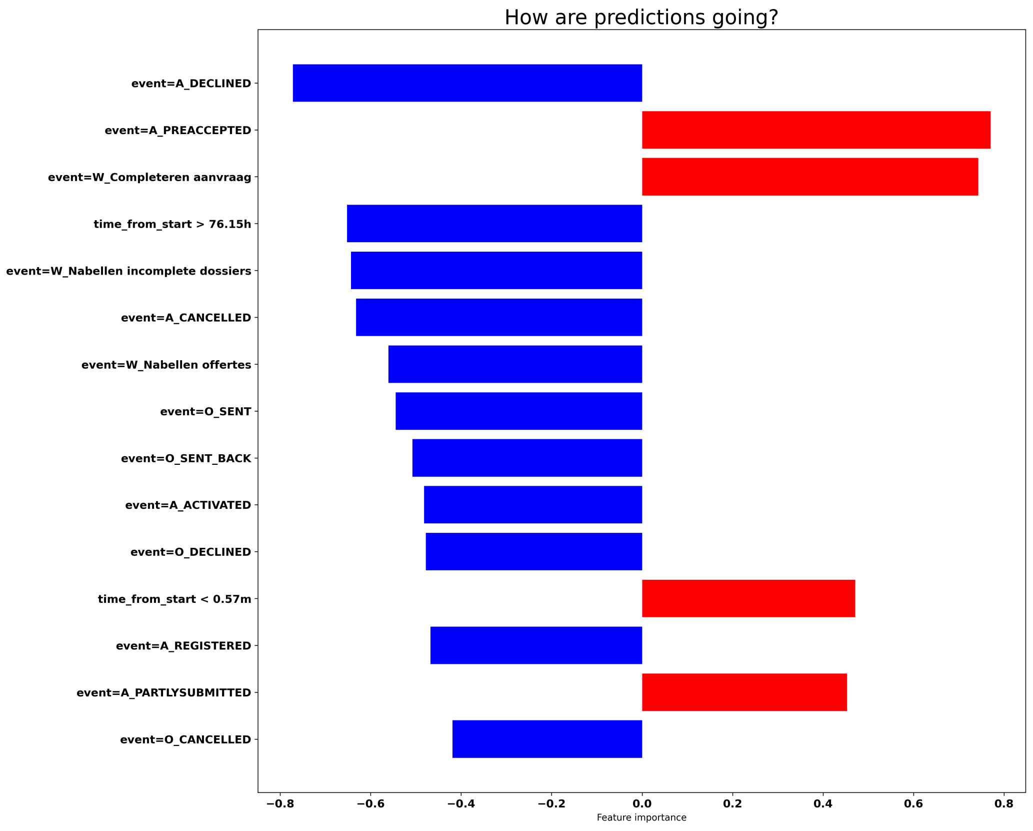


Figure A.4: Offline explanations for *A_ACCEPTED* prediction (BPIC 2012)

rejected by the user (action represented by the task O_DECLINED); the negative impact on the remaining time is motivated by the fact that the organization can still make a counter-proposal to the customer, but it can also lead to the closure of the application process. However, since this action is generally performed later in the process compared to the two previous actions (which occur in the early stages of the process), it has a lower impact on the remaining time (-3.5 compared to -6.5 and -7.5). The other attributes that are influencing the prediction are related to the resources that have performed one or more activities in the process.

In this process we also focused on three activities, namely *A_ACCEPTED*, *A_DECLINED*, and *A_CANCELLED*, whose occurrence would be interesting to be predicted beforehand since they indicate if the application has been accepted, declined or cancelled respectively.

Figure A.4 reports on the outcome of the application of our explainable framework for activity *A_ACCEPTED* occurrence prediction. It can be easily seen that the attributes related to the activity performed are largely influencing the prediction. The most influential factor here is *event=A_DECLINED*. When this activity is performed, there is a very high probability (-0.80) that the loan application is not going to be accepted. Conversely, as it can be seen by the red bars, the influence of *event=A_PREACCEPTED* and *event=W_Completeren aanvraag* is towards highly increasing the probability to have the application accepted; these are in fact activities which are usually performed immediately before proceeding with the acceptance of the application. Moreover, other activities similarly influence the prediction but, since in the process they are usually performed after the acceptance of the application, their influence is towards lowering the probability that the application will be accepted afterwards, as it can be seen by the blue bars related to the explanations *event=W_Nabellen incomplete dossiers*,*event=W_Nabellen offertes*, *event=O_SENT*, *event=O_SENT_BACK*, *event=A_ACTIVATED* and *event=O_DECLINED* and associated with values -0.50 and -0.60. Finally, since the application is usually accepted in the initial part of the process, the time elapsed since the beginning of the process is a crucial feature. As it can be seen, when the time elapsed is above a certain threshold ($time\_from\_start > 76.15h$), then the influence is towards lowering the probability that the application will be accepted by -0.65; instead, when the time elapsed is below a certain threshold ($time\_from\_start < 0.57m$), then the influence is towards increasing the probability by 0.50.

The explanations related to the activity *A_DECLINED* are shown in Figure A.5. As before, attributes related to the activity performed are the most important influencers for the prediction. When the application is declined (*event= A_DECLINED*), there is still a chance that the organization makes a counter-proposal to the customer; however, since it is rare that an application is declined twice, if *A_DECLINED* is performed once, than the influence is strongly towards not predicting that this activity will occur again in the future (as it can be seen by the strong negative value -0.85). The same reasoning applies also for *event=A_CANCELLED*, which is associated with value -0.80. Moreover, also when the acceptance process for the loan application has started, the probability to predict that the loan will be declined decreases; this is highlighted by the blue bars associated with the explanations *event=A_ACTIVATED*, *event=A_REGISTERED*, *event=A_APPROVED*, *event=O_ACCEPTED*, with values -0.65, -0.60, -0.50, -0.45, respectively. Conversely, activity *O_DECLINED* can be directly followed by *A_DECLINED*; therefore when activity *O_DECLINED* is performed (*event= O_DECLINED*), the influence is towards predicting that the application will be

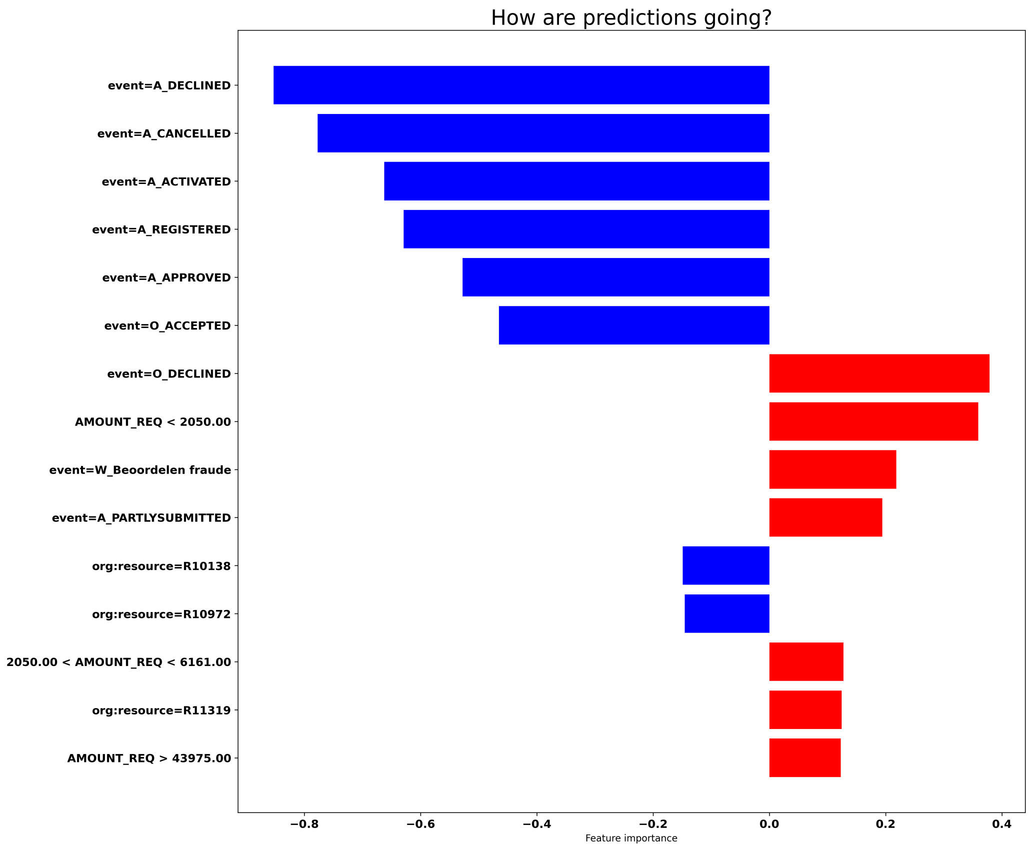


Figure A.5: Offline explanations for *A_DECLINED* prediction (BPIC 2012)

declined (as underlined by the red bar with value 0.40).

Finally, Figure A.6 reports on the explanations for *A_CANCELLED* occurrence prediction, which are very similar to the explanations shown in Figure A.5. Since it is rare that an application is cancelled twice, if *A_CANCELLED* is performed once, than the influence is strongly towards not predicting that this activity will occur again (as it can be seen by the strong negative value -0.85). The same reasoning applies also for *event=A_DECLINED* and *event=O_DECLINED*, which is associated with value -0.80. Additionally, as it was previously mentioned, when the acceptance process for the loan application has started, the probability to predict that the loan will be cancelled decreases; this is underlined by the blue bars associated with the explanations *event=A_ACTIVATED*, *event=A_REGISTERED*, *event=A_APPROVED*, and *event=O_ACCEPTED*, with values -0.60, -0.60, -0.55, -0.55, respectively. Conversely, activity *O_DECLINED* can be directly followed by *A_DECLINED*; therefore when activity *O_DECLINED* is performed (*event=*

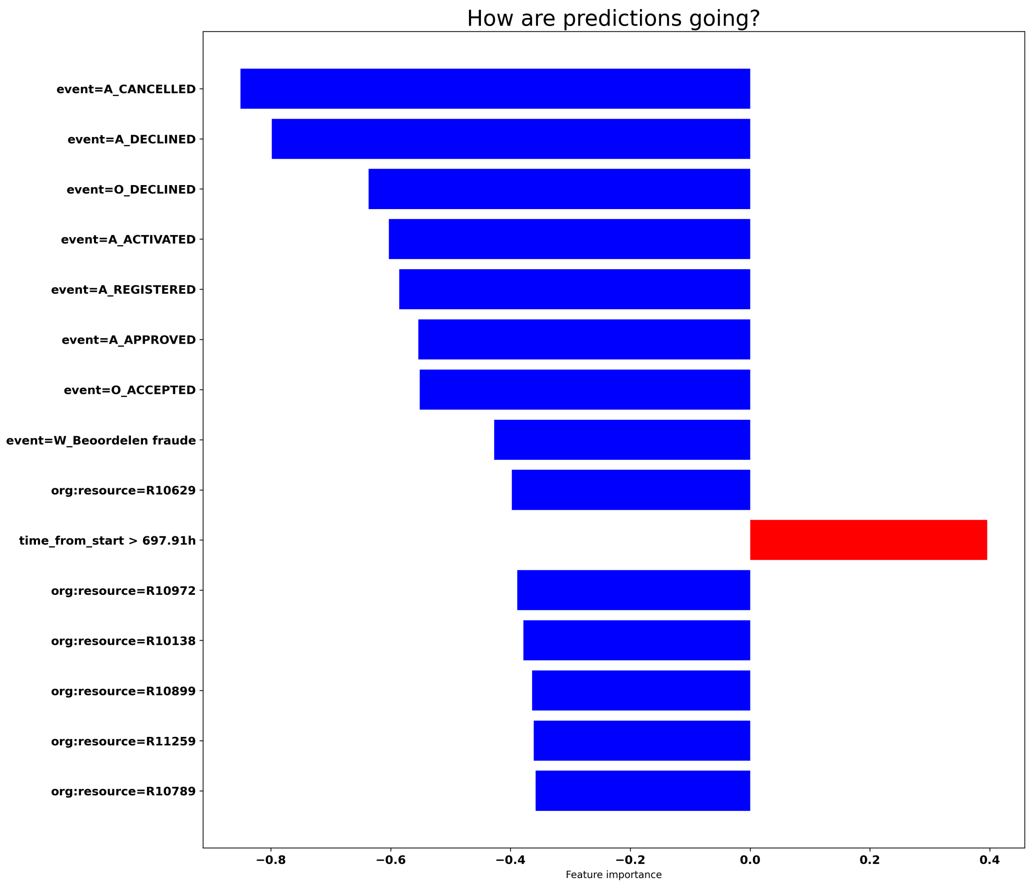


Figure A.6: Offline explanations for *A_CANCELLED* prediction (BPIC 2012)

O_DECLINED), the influence is towards predicting that the application will be declined (as underlined by the red bar with value 0.40).

A.3 BPIC 2012 - W

Figure A.7 shows the application for the remaining time prediction in the Bpi12 - W dataset, which is the dataset derived from bpi12 challenge, representing the subprocess containing only the states of the work items belonging to the application. Here, the most important influencers are the attributes related to the activity and the resource performing the activity. The main factor that contributes to decrease the remaining time of a case is represented by *concept:name=W_Nabellen incomplete dossiers*. The information that the value is negative (i.e. -4.5 days) indicates that the influence of performing the activity *W_Nabellen incomplete*

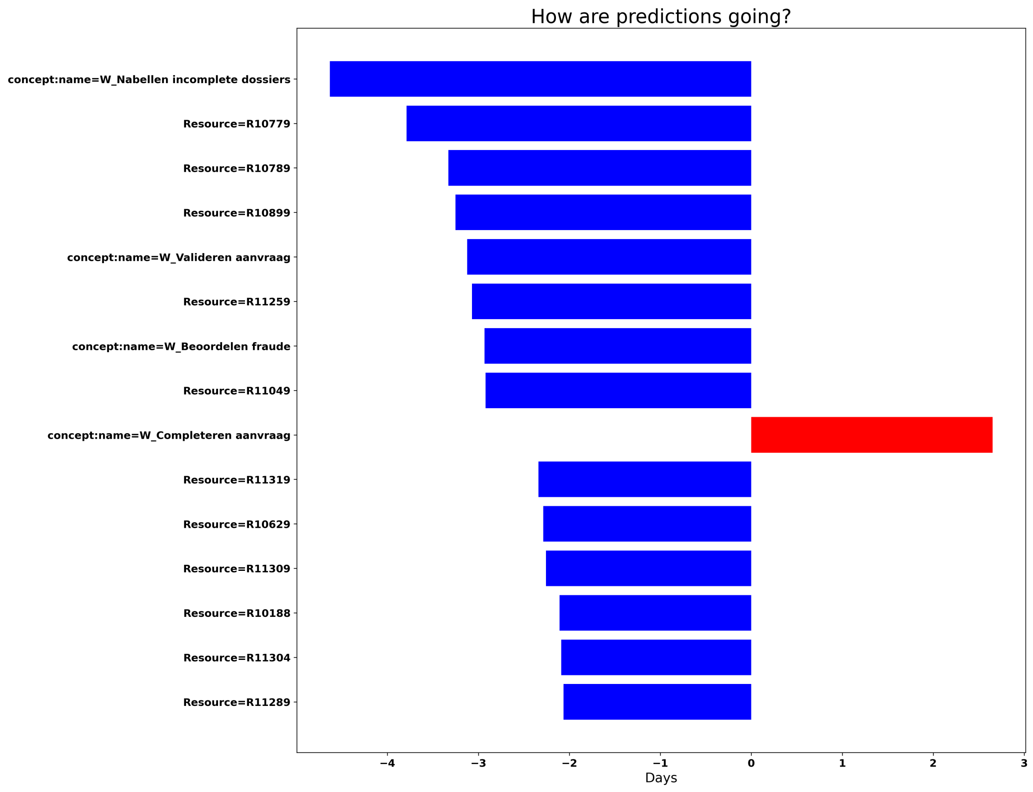


Figure A.7: Offline explanations for Remaining Time prediction (BPIC 2012 - W)

dossiers is towards decreasing the remaining time. This is mainly caused by the fact that this activity is usually performed in the final part of cases, therefore the remaining time to finish the case will be smaller. This also regards other activities performed in the final part of cases (*concept:name=W_Valideren aanvraag* and *concept:name=W_Beoordelen fraude*) and the related resources performing these activities, which influence is towards decreasing the remaining time. Conversely, *concept:name=W_Completeren aanvraag* is one of the first activities that are performed in this process, therefore the contribution is towards increasing the remaining time prediction, as it can be seen by the red bar associated with a value of 3 days.

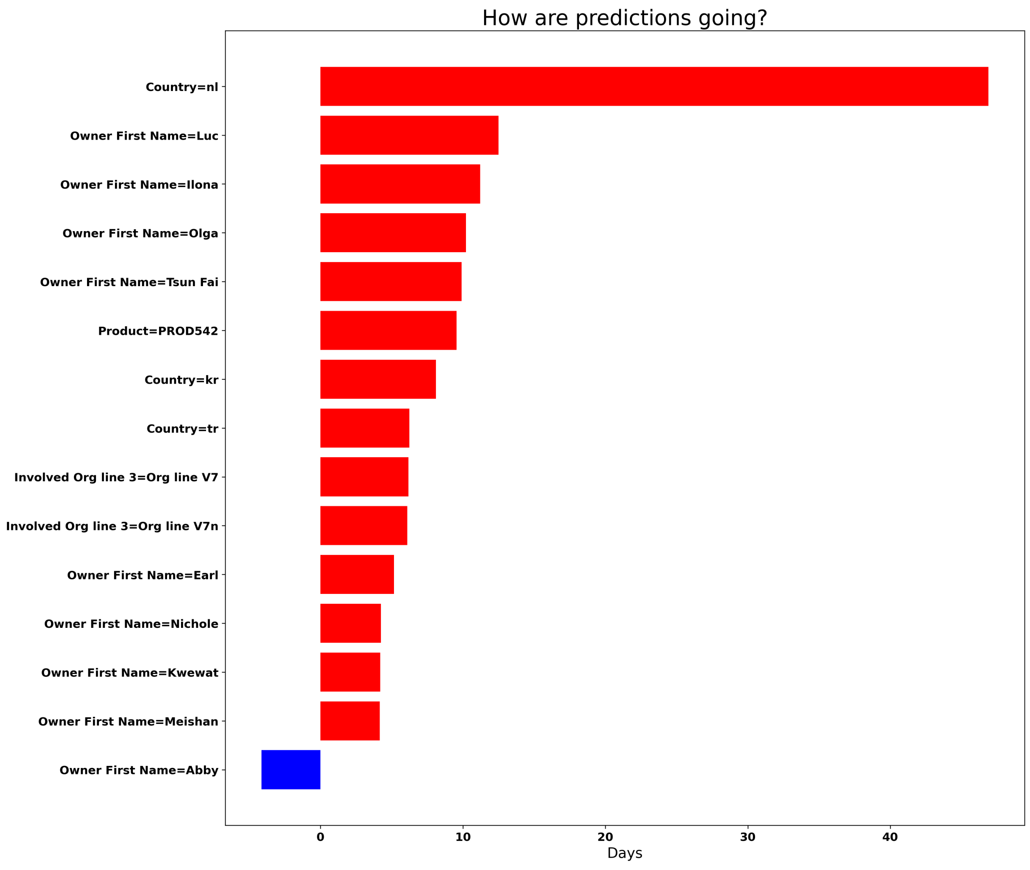


Figure A.8: Offline explanations for Remaining Time prediction (BPIC 2013)

A.4 BPIC 2013

This is a dataset from BPI 2013 challenge, extracted from Volvo incident management system. Here, we focused on obtaining an estimate of the remaining time until the end for running cases, to detect the cases requiring special attention.

Figure A.8 reports on the outcome of the application of our explainable framework for remaining time prediction in Bpi13 dataset. The fact that the country in which the incident is managed is Netherlands (*Country=nl*) is one of the largest factors that influences the prediction; the color red of the bar and the information that the value is positive (50 days) indicates that the influence is towards increasing the remaining time. A further analysis of the data confirms this finding: if the country is Netherlands, the process duration is 197 days, versus 12.1 days when the type is different. This also applies to the situation where the incident is managed in South Korea (*Country=kr*) or in Turkey (*Country=tr*), where the influence is towards increasing the remaining time by 8 and 6 days respectively; again, this finding was supported by the data, since it was observed a longer duration of the process (16d 4h and 19d 9h respectively). Other important attributes are related to the resource of the support team (indicated by *Owner First Name*) in charge of working on the incident. Here, for example, the fact that one or multiple activities were performed by Luc, Ilona or Olga, contributed to increase the remaining time prediction by 10 days. Also the support team that is in charge of solving the problem can influence the remaining time. Here, the fact that a 3rd level support team has been involved (indicated by *Involved Org line 3=Org line V7*, *Involved Org line 3=Org line V7n*) increased the remaining time by 7 days; this is probably due to the fact that only the problems of a certain severity level are passed to the 3rd level support team, hence requiring more time to be solved.

This leads us to analyze one of the KPIs in which the company was interested in, the push to front strategy. In particular, the company wanted to resolve most of the incidents with the first line support teams without involving 2nd or 3rd support line teams (strategy called push to front), in order to have a more efficient process and avoid having too much work concentrated on 2nd and 3rd line support teams. Therefore, we also focused on predicting if at least one activity is going to be performed by a resource not belonging to the first line support team.

The bar chart related to the push to front prediction is shown in Figure A.9; It can be clearly seen that the attributes related to the product that presented a problem or that was involved in an incident are the most important ones. The influence for all of them is towards strongly predicting that a resource from the 2° or 3° line support team will be involved in the management of the incident, as it can be seen by the color red of the bars and the very high positive values (0.80). The data confirmed that the products reported in these explanations (such as (*Product=PROD691*)) were always associated to the involvement of a resource from the 2° or 3° level support team, thus confirming the findings reported here.

Finally, the company had also an interest in understanding if people working in the company were abusing the *Wait - User* substatus to hide inefficiencies in the process, which would otherwise being detected by KPIs measuring the total resolution time of an incident. Therefore, the main objective here is to try to detect as soon as possible if this status will be set.

Figure A.10 shows the results for the *Wait - User* substatus prediction. Here, the fact that the *Resolved* substatus has been set (labeled as *Sub Status=Resolved*) is the most important factor, and its contribute is towards strongly decreasing the probability that a *Wait - User* substatus will be set (as it can be seen by the blue

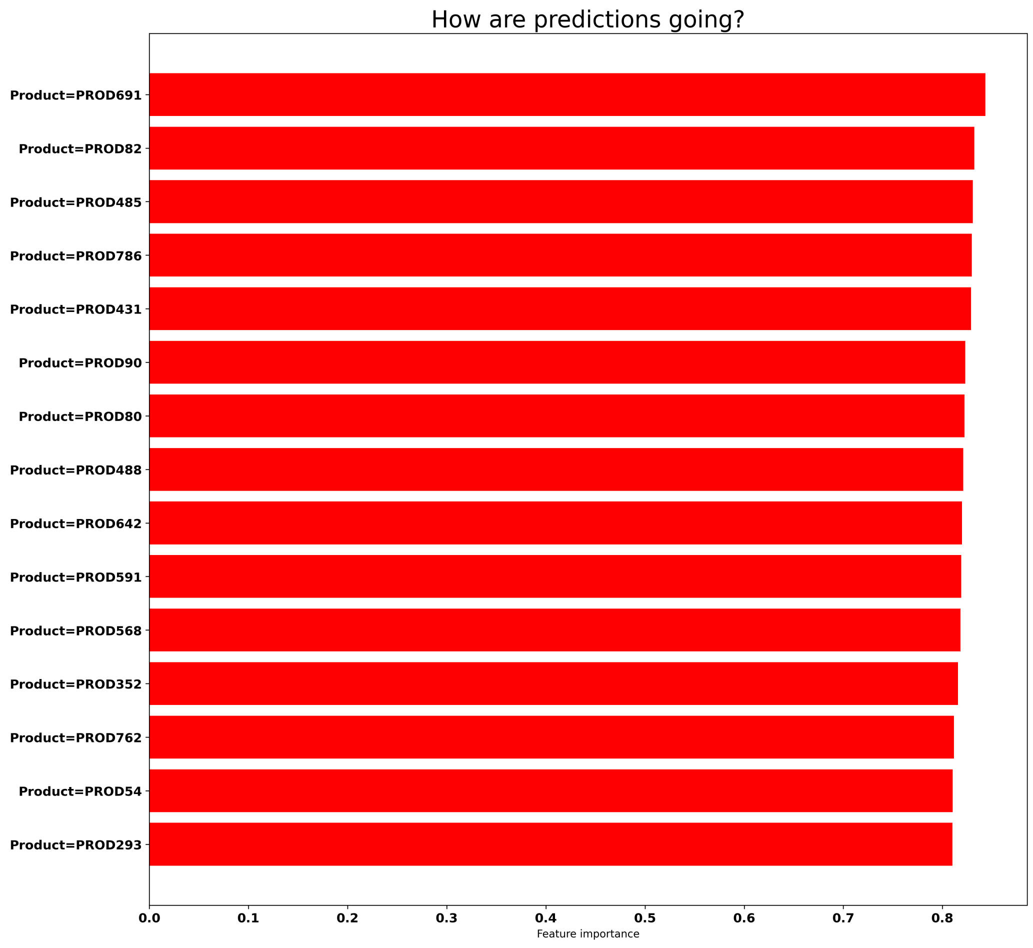


Figure A.9: Offline explanations for *Push to Front* prediction (BPIC 2013)

bar and the associated value -0.80). The analysis of the process confirmed that it is unlikely that a *Wait - User* status is set after the *Resolved* status, which is indeed the status indicating that the incident has been already solved. Other important attributes are related to the product that presented a problem or that was involved in an incident; the influence for all of them is towards strongly predicting that the *Wait - User* substatus will not be set during the resolution of the problem. The data also confirmed that the products reported in these explanations (such as *Product=PROD691*) were never associated to the setting of the status *Wait - User*, thus confirming the findings reported here.

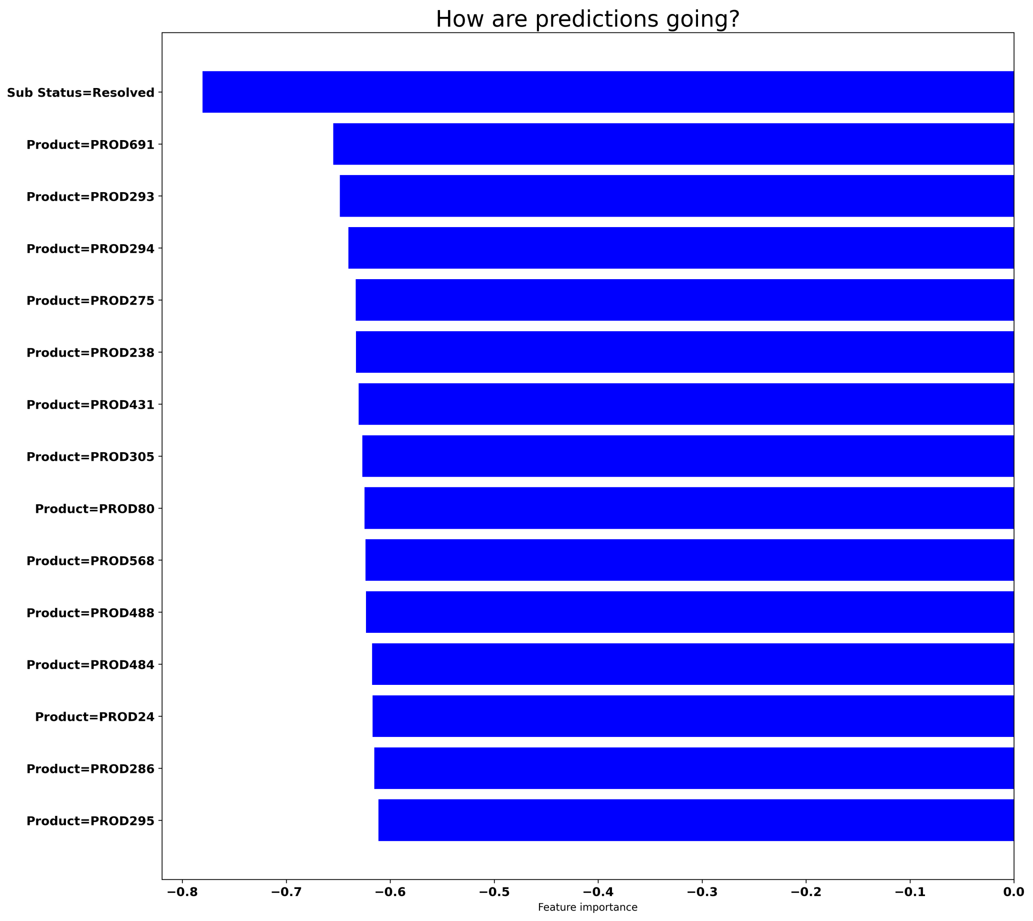


Figure A.10: Offline explanations for *Wait - User* prediction (BPIC 2013)

A.5 HelpDesk 2017

This dataset is a real-life log of SIAV s.p.a. company in Italy, and represents instances of a ticketing process in the company helpdesk area.

Figure A.11 refers to the application for the remaining time prediction. It can be clearly seen by the color blue of the bars and the negative values that the influence of the principal factors is towards decreasing the remaining time prediction. In particular, the time elapsed since the beginning of the process is a crucial feature. As it can be seen, when the time elapsed is between certain thresholds ($199.01h < time\_from\_start < 960.51h$), then the influence is towards lowering the predicted remaining time by -12 days. Moreover, when the time elapsed is larger than the average process duration (which was 40 days, the same duration highlighted in $time\_from\_start > 960.51h$), the influence towards lowering the

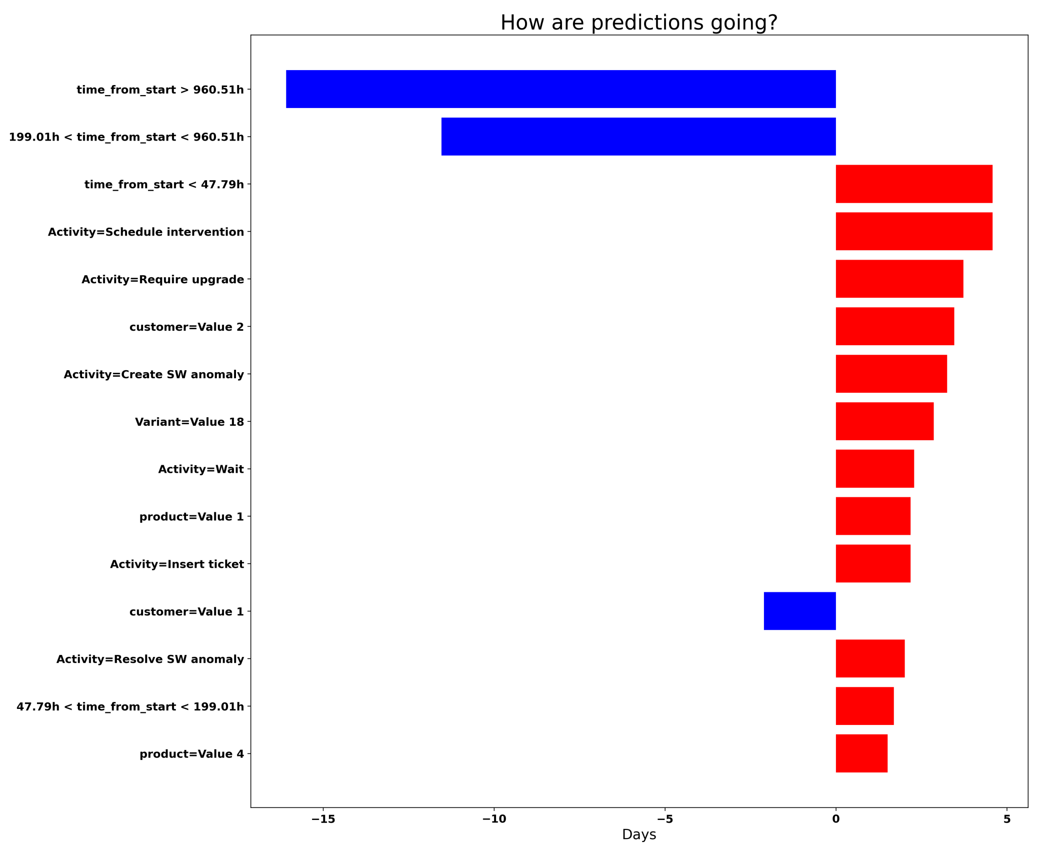


Figure A.11: Offline explanations for Remaining Time prediction (HelpDesk 2017)

predicted remaining time becomes even stronger (-16 days); this is mainly caused by the fact that when the elapsed time is above the average process duration, it is highly probable that the case is going to be closed soon, thus lowering the remaining time prediction. Conversely, when the process is still in the initial phases ($time\_from\_start < 47.79h$), the influence is towards predicting a higher remaining time (+5 days). Other important attributes are related to the performed activity. In this process a ticket, after having been opened and after assigning a seriousness level, is usually taken in charge by a resource and resolved; however, after having been taken in charge, some activities can occur, such as *Schedule intervention*, *Require upgrade*, *Create SW anomaly* and *Wait*. These are all exceptional activities that are rarely performed, thus increasing the remaining time prediction, as it can be seen by the red bars associated to *Activity=Schedule intervention*, *Activity=Require upgrade*, *Activity=Create SW anomaly* and *Activ-*

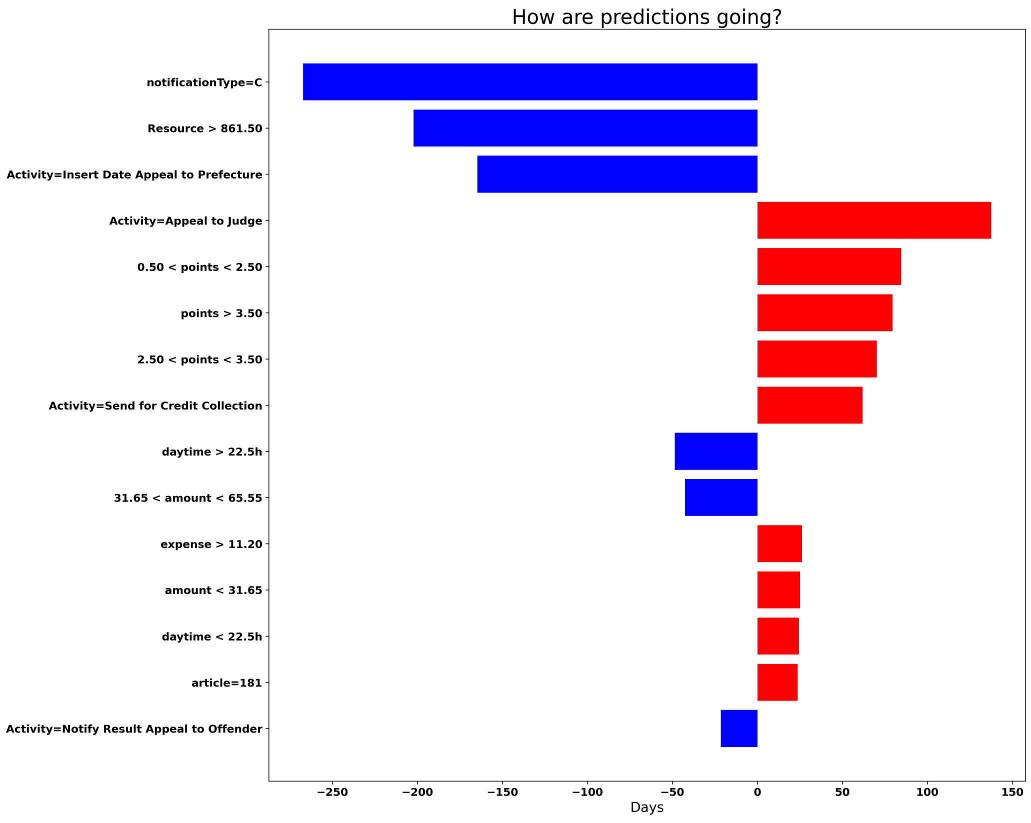


Figure A.12: Offline explanations for Remaining Time prediction (Fine Management)

ity=Wait.

A.6 Road Traffic Fine Management

This dataset is a real-life event log of an information system managing road traffic fines.

Figure A.12 reports on the outcome of the application of our explainable framework for remaining time prediction in Fine Management dataset. The fact that the fine is related to a parking ticket (*notificationType=C*) is one of the largest factors that influences the prediction; the color blue of the bar and the information that the value is negative (-270 days) indicates that the influence is towards strongly decreasing the remaining time. This is principally due to the fact that parking tickets can be paid immediately after the ticket creation, in which

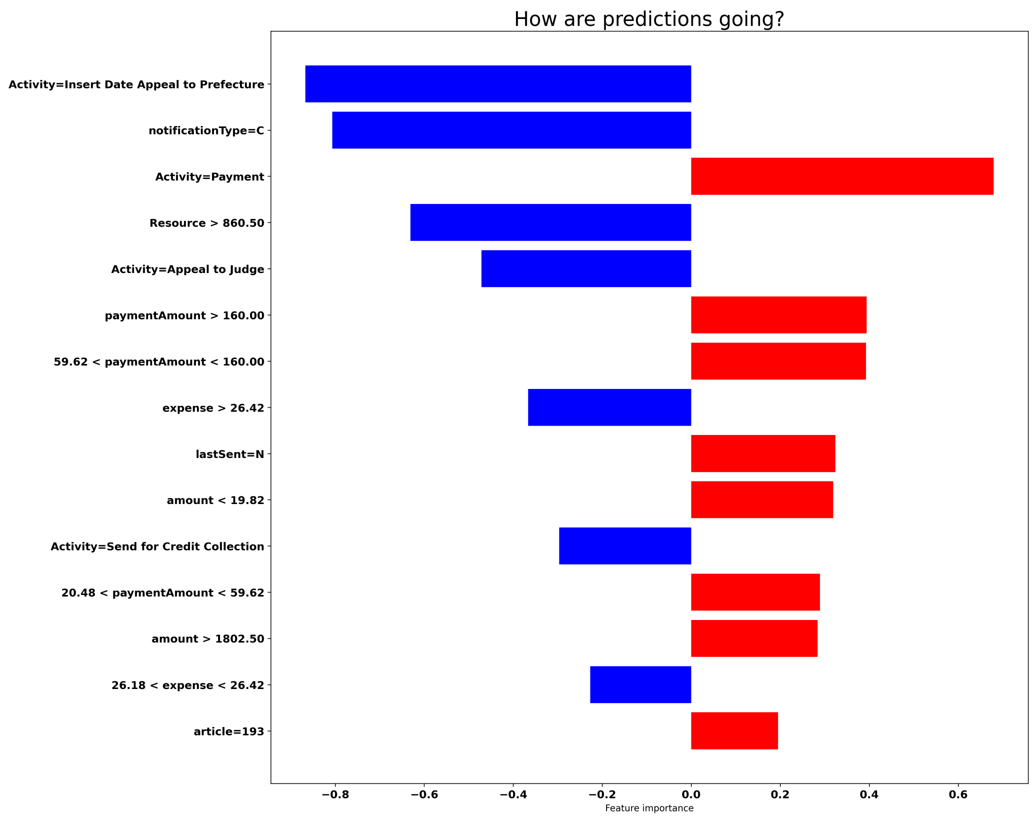


Figure A.13: Offline explanations for *Send for Credit Collection* prediction (Fine Management)

case the paperwork is bypassed and unnecessary administration time and costs are avoided. A similar explanation can be given for the second largest factor ($Resource > 861.50$), which is decreasing the remaining time prediction by -200 days. The resource here is the person responsible to notify the fine; a high identification code indicates that the resource belongs to a local police department, which is usually responsible to notify the fines related to parking tickets. Other important attributes that are influencing the prediction are related to the performed activity. In particular, on the other side of the spectrum, explanation *Activity=Appeal to Judge* has the largest positive shapley value (145 days). This can also be justified: when the offender appeals against the payment of the fine, the execution takes longer due to the involvement of the judge.

Finally, Figure A.13 reports on the explanations for *Send for Credit Collection* occurrence prediction, which is the activity performed when the case is sent to an external credit collection agency that will contact the offender to collect the payment. Here, when an appeal against the fine is initiated within 60 days and it is correctly registered by the corresponding prefecture (*Activity=Insert Date Appeal to Prefecture*) is the most influential factor and its influence is towards strongly decreasing the probability that a *Send for Credit Collection* will be performed (as it can be seen by the blue bar and the associated probability of -0.85). A further analysis of the data confirmed this finding: when an appeal is sent to the prefecture, the external credit collection agency is alerted only in the 7% of cases (282 over 4043 cases). A similar situation happens when, after receiving the results of the appeal, the offender appeals against the result (*Activity=Appeal to Judge*): the influence is again towards decreasing the probability that the external credit collection agency will be alerted (-0.50). The data confirmed that when the judge is appealed, activity *Send for Credit Collection* will be performed only in the 27% of cases (148 over 541 cases). Other attributes contributing to decrease the probability that a *Send for Credit Collection* will be performed are *notificationType=C* and $Resource > 860.50$, which are decreasing the probability respectively by -0.80 and -0.65. As it was previously said, these factors indicate that the fine is related to a parking ticket and that the resource belongs to a local police department, which represents a simpler case in which tickets can be paid immediately after the ticket creation, thus not usually requiring the involvement of an external agency responsible for the credit collection. Conversely, when the payment has been already performed (*Activity=Payment*) the influence is towards increasing the probability that a *Send for Credit Collection* will be performed (0.65); this was actually surprising, but a further analysis of the data confirmed that there is a small chance that after performing the payment, a *Send for Credit Collection* action is initiated in the 3% of cases (2126 out of 65800 cases). This means that there could be cases where an action is initiated by the external credit collector agency even if the payment has been already made, indicating a possible lack of synchronization between the accounting department and the external credit collection agency.

Appendix B

Publications

This research activity has led to several publications in international journals and conferences. These are summarized below.[1]

International Journals

1. **Galanti, R.**, de Leoni, M., Monaro, M., Navarin, N., Marazzi, A., Di Stasi, B., Maldera, S., "An Explainable Decision Support System for Predictive Process Analytics", *Engineering Applications of Artificial Intelligence*, vol. 120, p. 105904, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095219762300088X

2. **Galanti, R.**, de Leoni, M., Navarin, N., Marazzi, A., "Object-centric Process Predictive Analytics", *Expert Systems with Applications*, vol. 213, p. 119173, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422021911

International Conferences and Workshops

1. Padella, A., de Leoni, M., Dogan, O., **Galanti, R.**, "Explainable Process Prescriptive Analytics", *the 4th International Conference on Process Mining (ICPM)*, Bolzano, Italy, IEEE Computational Intelligence Society, 2022, pp. 16–23.

2. **Galanti, R.**, de Leoni, M., Marazzi, A., Bottazzi, G., Delsante, M., Folli, A., "Integration of an Explainable Predictive Process Monitoring System into IBM Process

[1]The author's bibliometric indices are the following: H-index = 2, total number of citations = 80 (source: Google Scholar on May 27, 2023).

Mining Suite". *Proceedings of the ICPM Doctoral Consortium and Tool Demonstration Track 2021 collocated with the 3rd International Conference on Process Mining (ICPM)*, Eindhoven, Netherlands, 2021. CEUR Workshop Proceedings, vol. 3098, pp. 53-54.

3. **Galanti, R.**, Coma-Puig, B., de Leoni, M., Carmona, J., Navarin, N., "Explainable Predictive Process Monitoring", *the 2nd International Conference on Process Mining (ICPM)*, IEEE Computational Intelligence Society, 2020, pp. 1-8. DOI:10.1109/ICPM49681.2020.00012

4. **Galanti, R.**, "Explainable Prescriptive Process Analytics (Extended Abstract)". *Proceedings of the ICPM Doctoral Consortium and Tool Demonstration Track 2020 co-located with the 2nd International Conference on Process Mining (ICPM)*, Padova, Italy, 2020. CEUR Workshop Proceedings, vol. 2703, pp. 1-2. **(Best PhD Proposal)**

Bibliography

[1] J. N. Adams, G. Park, S. Levich, D. Schuster, and W. M. P. van der Aalst, "A framework for extracting and encoding features from object-centric event data," in *Service-Oriented Computing*. Springer Nature Switzerland, 2022, pp. 36–53.

[2] J. N. Adams and W. M. P. van der Aalst, "Precision and fitness in object-centric process mining," in *Proceedings of the 3rd International Conference on Process Mining (ICPM 2021)*. IEEE, 2021, pp. 128–135.

[3] D. Alvarez-Melis and T. S. Jaakkola, "On the robustness of interpretability methods," *arXiv preprint arXiv:1806.08049*, 2018.

[4] 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.

[5] A. Berti and W. M. P. van der Aalst, "Extracting multiple viewpoint models from relational databases," in *Proceedings of the 9th Internation Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2019)*, ser. Lecture Notes in Business Information Processing, vol. 379. Springer, 2019, pp. 24–51.

[6] K. Böhmer and S. Rinderle-Ma, "Logo: Combining local and global techniques for predictive business process monitoring," in *Advanced Information Systems Engineering*, S. Dustdar, E. Yu, C. Salinesi, D. Rieu, and V. Pant, Eds. Cham: Springer International Publishing, 2020, pp. 283–298.

[7] 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.

[8] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees," 1984.

[9] L. Breiman, "Random forests," *Machine Learning*, 2001.

[10] D. Breuker, P. Delfmann, M. Matzner, and J. Becker, “Designing and evaluating an interpretable predictive modeling technique for business processes,” in *Business Process Management Workshops*. Springer, 2015, pp. 541–553.

[11] J. Brunk, M. Stierle, L. Papke, K. Revoredo, M. Matzner, and J. Becker, “Cause vs. effect in context-sensitive prediction of business process instances,” *Information Systems*, vol. 95, 09 2020.

[12] D. Calvanese, M. Montali, M. Estañol, and E. Teniente, “Verifiable uml artifact-centric business process models,” in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, ser. CIKM ’14. New York, NY, USA: Association for Computing Machinery, 2014, p. 1289–1298.

[13] D. Cohn and R. Hull, “Business artifacts: A data-centric approach to modeling business operations and processes,” *IEEE Data Engineering Bulletin*, vol. 32, no. 2, pp. 3–9, 2009.

[14] B. Coma-Puig and J. Carmona, “A quality control method for fraud detection on utility customers without an active contract,” 04 2018, pp. 495–498.

[15] B. Coma-Puig and J. Carmona, “Non-technical losses detection in energy consumption focusing on energy recovery and explainability,” *Machine Learning*, vol. 111, no. 2, pp. 487–517, 2022.

[16] 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.

[17] 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.

[18] V. Denisov, D. Fahland, and W. M. P. van der Aalst, “Predictive performance monitoring of material handling systems using the performance spectrum,” in *Proceedings of the International Conference on Process Mining, ICPM 2019*, 2019, pp. 137–144.

[19] D. Dinis, Ângelo Palos Teixeira, and A. Barbosa-Póvoa, “Foresim-bi: A predictive analytics decision support tool for capacity planning,” *Decision Support Systems*, vol. 131, p. 113266, 2020.

[20] 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.

[21] F. Doshi-Velez and B. Kim, “Towards a rigorous science of interpretable machine learning,” 2017.

[22] T. Dozat, “Incorporating nesterov momentum into adam,” 2016.

[23] M. Dumas, M. La Rosa, J. Mendling, and H. Reijers, *Fundamentals of business process management*, 2nd ed. Springer, 2018.

[24] G. Elkhawaga, M. Abuelkheir, and M. Reichert, “Explainability of predictive process monitoring results: Can you see my data issues?” 2022.

[25] G. Elkhawaga, M. Abuelkheir, and M. Reichert, “Xai in the context of predictive process monitoring: Too much to reveal,” 2022.

[26] S. Esser and D. Fahland, “Multi-dimensional event data in graph databases,” *Journal on Data Semantics*, vol. 10, no. 1, pp. 109–141, 2021.

[27] D. Fahland, M. de Leoni, B. F. van Dongen, and W. M. P. van der Aalst, “Conformance checking of interacting processes with overlapping instances,” in *Proceedings of the 9th International Conference on Business Process Management (BPM 2011)*, vol. 6896. Springer Berlin Heidelberg, 2011, pp. 345–361.

[28] A. Farhang Ghahfarokhi, G. Park, A. Berti, and W. M. P. van der Aalst, “OCEL Standard,” January 2020, Available at http://ocel-standard.org/1.0/specification.pdf.

[29] K. Fukushima, “Cognitron: A self-organizing multilayered neural network,” *Biol. Cybern.*, vol. 20, no. 3–4, p. 121–136, sep 1975. [Online]. Available: https://doi.org/10.1007/BF00342633

[30] K. M. Ghori, R. A. Abbasi, M. Awais, M. Imran, A. Ullah, and L. Szathmary, “Performance analysis of different types of machine learning classifiers for non-technical loss detection,” *IEEE Access*, vol. 8, pp. 16 033–16 048, 2020.

[31] R. Guidotti, A. Monreale, F. Turini, D. Pedreschi, and F. Giannotti, “A survey of methods for explaining black box models,” *CoRR*, vol. abs/1802.01933, 2018. [Online]. Available: http://arxiv.org/abs/1802.01933

[32] K. M. Hanga, Y. Kovalchuk, and M. M. Gaber, “A graph-based approach to interpreting recurrent neural networks in process mining,” *IEEE Access*, vol. 8, pp. 172 923–172 938, 2020.

[33] M. Harl, S. Weinzierl, M. Stierle, and M. Matzner, “Explainable predictive business process monitoring using gated graph neural networks,” *Journal of Decision Systems*, vol. 0, no. 0, pp. 1–16, 2020.

[34] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)*, 02 2009.

[35] A. Hemmer, R. Badonnel, and I. Chrisment, “A process mining approach for supporting iot predictive security,” in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–9.

[36] S. Hochreiter and J. Urgen Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[37] A. Hodrien and T. Fernando, “A review of post-study and post-task subjective questionnaires to guide assessment of system usability,” *Journal of Usability Studies*, vol. 16, no. 3, pp. 203–232, 2021. [Online]. Available: http://usir.salford.ac.uk/id/eprint/60928/

[38] C. Hsieh, C. Moreira, and C. Ouyang, “Dice4el: Interpreting process predictions using a milestone-aware counterfactual approach,” in *3rd International Conference on Process Mining, ICPM 2021, Eindhoven, Netherlands, October 31 - Nov. 4, 2021*, C. D. Ciccio, C. D. Francescomarino, and P. Soffer, Eds. IEEE, 2021, pp. 88–95.

[39] T.-H. Huang, A. Metzger, and K. Pohl, “Counterfactual explanations for predictive business process monitoring,” in *European, Mediterranean, and Middle Eastern Conference on Information Systems*. Springer, 2021, pp. 399–413.

[40] J. Huysmans, K. Dejaeger, C. Mues, J. Vanthienen, and B. Baesens, “An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models,” *Decision Support Systems*, vol. 51, no. 1, pp. 141–154, 2011.

[41] IEEE, “Ieee standard for extensible event stream (xes) for achieving interoperability in event logs and event streams,” *IEEE Std 1849-2016*, pp. 1–50, 2016.

[42] S. K. Jagatheesaperumal, M. Rahouti, K. Ahmad, A. Al-Fuqaha, and M. Guizani, “The duo of artificial intelligence and big data for industry 4.0: Review of applications, techniques, challenges, and future research directions,” 2021.

[43] S. Khan and S. Parkinson, “Eliciting and utilising knowledge for security event log analysis: An association rule mining and automated planning approach,” *Expert Systems with Applications*, vol. 113, pp. 116–127, 2018.

[44] S. Khan and S. Parkinson, “Discovering and utilising expert knowledge from security event logs,” *Journal of Information Security and Applications*, vol. 48, p. 102375, 2019.

[45] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2017.

[46] E. L. Klijn and D. Fahland, “Identifying and reducing errors in remaining time prediction due to inter-case dynamics,” in *Proceedings of the 2nd International Conference on Process Mining, ICPM 2020*, 2020, pp. 25–32.

[47] M. Koot, M. R. Mes, and M. E. Iacob, “A systematic literature review of supply chain decision making supported by the internet of things and big data analytics,” *Computers & Industrial Engineering*, vol. 154, p. 107076, 2021.

[48] R. Krishnan, G. Sivakumar, and P. Bhattacharya, “Extracting decision trees from trained neural networks,” *Pattern Recognition*, vol. 32, no. 12, pp. 1999–2009, 1999.

[49] J. Lasserre, C. Bishop, J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, “Generative or discriminative? getting the best of both worlds,” *BAYESIAN STATISTICS*, vol. 8, pp. 3–24, 01 2007.

[50] B. Laugwitz, T. Held, and M. Schrepp, “Construction and evaluation of a user experience questionnaire,” in *HCI and Usability for Education and Work*, A. Holzinger, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 63–76.

[51] J. Levitt, *Complete Guide to Preventive and Predictive Maintenance*. Industrial Press Inc., 2011.

[52] J. Lewis, “Psychometric evaluation of the post-study system usability questionnaire: The PSSUQ,” *Proceedings of the Human Factors Society Annual Meeting*, vol. 36, pp. 1259–1260, 10 1992.

[53] J. R. Lewis, “Psychometric evaluation of the PSSUQ using data from five years of usability studies,” *International Journal of Human–Computer Interaction*, vol. 14, no. 3-4, pp. 463–488, 2002.

[54] G. Li, R. Medeiros de Carvalho, and W. M. P. van der Aalst, “Object-centric behavioral constraint models: A hybrid model for behavioral and data perspectives,” in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC 2019)*. ACM, 2019, p. 48–56.

[55] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, “Gated Graph Sequence Neural Networks,” in *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, 2016.

[56] 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.

[57] X. Lu, M. Nagelkerke, D. van de Wiel, and D. Fahland, “Discovering Interacting Artifacts from ERP Systems,” *IEEE Transactions on Services Computing*, vol. 8, no. 6, pp. 861–873, 2015.

[58] E. Lughofer, *Model Explanation and Interpretation Concepts for Stimulating Advanced Human-Machine Interaction with “Expert-in-the-Loop”*, 06 2018, pp. 177–221.

[59] E. Lughofer and M. Sayed-Mouchaweh, *Predictive Maintenance in Dynamic Systems*. Springer, 2019.

[60] E. Lughofer, A.-C. Zavoianu, R. Pollak, M. Pratama, P. Meyer-Heye, H. Zörrer, C. Eitzinger, and T. Radauer, “Autonomous supervision and optimization of product quality in a multi-stage manufacturing process based on self-adaptive prediction models,” *Journal of Process Control*, vol. 76, pp. 27–45, 2019.

[61] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in *Advances in neural information processing systems*, 2017, pp. 4765–4774.

[62] S. M. Lundberg, B. Nair, M. S. Vavilala, M. Horibe, M. J. Eisses, T. Adams, D. E. Liston, D. K.-W. Low, S.-F. Newman, J. Kim *et al.*, “Explainable machine-learning predictions for the prevention of hypoxaemia during surgery,” *Nature biomedical engineering*, vol. 2, no. 10, p. 749, 2018.

[63] 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, pp. 962–977, 2018.

[64] N. Martin, *Data Quality in Process Mining*. Cham: Springer International Publishing, 2021, pp. 53–79.

[65] S. Meacham, G. Isaac, D. Nauck, and B. Virginas, *Towards Explainable AI: Design and Development for Explanation of Machine Learning Predictions for a Patient Readmittance Medical Application*, 06 2019, pp. 939–955.

[66] N. Mehdiyev and P. Fettke, “Prescriptive process analytics with deep learning and explainable artificial intelligence,” in *European Conference on Information Systems*, 2020.

[67] 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.

[68] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

[69] M. Montali and D. Calvanese, “Soundness of data-aware, case-centric processes,” *International Journal on Software Tools for Technology Transfer*, vol. 18, no. 5, pp. 535–558, 2016.

[70] 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, pp. 1–7.

[71] M. E. J. Newman, *Power laws, Pareto distributions and Zipf’s law*. Taylor & Francis, 2005, vol. 46, no. 5, pp. 323–351.

[72] I. Nunes and D. Jannach, “A systematic review and taxonomy of explanations in decision support and recommender systems,” *User Modeling and User-Adapted Interaction*, vol. 27, no. 3–5, p. 393–444, Dec. 2017.

[73] 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, pp. 121–128.

[74] V. Pasquadibisceglie, G. Castellano, A. Appice, and D. Malerba, “FOX: a neuro-fuzzy model for process outcome prediction and explanation,” in *3rd International Conference on Process Mining, ICPM 2021, Eindhoven, Netherlands, October 31 - Nov. 4, 2021*, C. D. Ciccio, C. D. Francescomarino, and P. Soffer, Eds. IEEE, 2021, pp. 112–119.

[75] M. Pegoraro, “Probabilistic and non-deterministic event data in process mining: Embedding uncertainty in process analysis techniques,” *arXiv.org*, vol. arXiv.2205.04827, 2022.

[76] P. Philipp, R. X. M. Georgi, J. Beyerer, S. Robert, and J. Beyerer, “Analysis of control flow graphs using graph convolutional neural networks,” *2019 6th International Conference on Soft Computing & Machine Intelligence (ISCMI)*, pp. 73–77, 2019.

[77] V. Popova and M. Dumas, "Discovering unbounded synchronization conditions in artifact-centric process models," in *Proceedings of Business Process Management Workshops*, ser. LNBIP, vol. 171. Springer, 2014, pp. 28–40.

[78] R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. De Bie, and P. Flach, "Face: Feasible and actionable counterfactual explanations," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, ser. AIES '20, 2020, p. 344–350.

[79] J.-R. Rehse, N. Mehdiyev, and P. Fettke, "Towards explainable process predictions for industry 4.0 in the dfki-smart-lego-factory," *KI - Künstliche Intelligenz*, vol. 33, no. 2, pp. 181–187, Jun 2019.

[80] M. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," vol. 32, 2018.

[81] M. T. Ribeiro, S. Singh, and C. Guestrin, "“why should I trust you”: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco*, 2016, pp. 1135–1144.

[82] W. Rizzi, M. Comuzzi, C. Di Francescomarino, C. Ghidini, S. Lee, F. M. Maggi, and A. Nolte, "Explainable predictive process monitoring: A user evaluation," 2022.

[83] W. Rizzi, C. Di Francescomarino, and F. Maggi, *Explainability in Predictive Process Monitoring: When Understanding Helps Improving*, 09 2020, pp. 141–158.

[84] M. v. Rosing and A.-W. Scheer, *The Complete Business Process Handbook, Volume 1 - Body of Knowledge from Process Modeling to BPM*, 03 2015, vol. 1.

[85] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[86] S. Sachan, J.-B. Yang, D.-L. Xu, D. Benavides, and Y. Li, "An explainable ai decision-support-system to automate loan underwriting," *Expert Systems with Applications*, vol. 144, Apr. 2020.

[87] 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, pp. 3543–3556.

[88] J. Sauro and J. Lewis, "Standardized usability questionnaires," in *Quantifying the User Experience*. Elsevier, 2016, pp. 185–248.

[89] M. Schrepp, “User experience questionnaire handbook. all you need to know to apply the ueq successfully in your project.” 09 2015.

[90] M. Schrepp, A. Hinderks, and J. Thomaschewski, “Applying the user experience questionnaire (UEQ) in different evaluation scenarios,” in *Design, User Experience, and Usability. Theories, Methods, and Tools for Designing the User Experience*, A. Marcus, Ed. Cham: Springer International Publishing, 2014, pp. 383–392.

[91] A. Senderovich, C. Di Francescomarino, and F. Maggi, “From knowledge-driven to data-driven inter-case feature encoding in predictive process monitoring,” *Inf. Syst.*, vol. 84, pp. 255–264, 2019.

[92] S. Serrano and N. A. Smith, “Is attention interpretable?” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 2931–2951.

[93] L. S. Shapley, *A value for n-person games*. RAND Corporation, 1953, vol. 2, no. 28, pp. 307–317.

[94] A. Shrikumar, P. Greenside, and A. Kundaje, “Learning important features through propagating activation differences,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3145–3153.

[95] K. Shu, L. Cui, S. Wang, D. Lee, and H. Liu, “Defend: Explainable fake news detection,” in *International Conference on Knowledge Discovery & Data Mining, SIGKDD*. ACM, 2019, pp. 395–405.

[96] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *CoRR*, vol. abs/1312.6034, 2013.

[97] R. Sindhgatta, C. Moreira, C. Ouyang, and A. Barros, “Exploring interpretable predictive models for business processes,” in *Business Process Management - 18th International Conference, BPM 2020, Seville, Spain, September 13-18, 2020, Proceedings*, ser. Lecture Notes in Computer Science, D. Fahland, C. Ghidini, J. Becker, and M. Dumas, Eds., vol. 12168. Springer, 2020, pp. 257–272.

[98] R. Sindhgatta, C. Ouyang, and C. Moreira, “Exploring interpretability for predictive process analytics,” in *Service-Oriented Computing - 18th International Conference, ICSOC 2020, Dubai, United Arab Emirates, December 14-17, 2020, Proceedings*, ser. Lecture Notes in Computer Science, E. Kafeza, B. Benatallah, F. Martinelli, H. Hacid, A. Bouguettaya, and H. Motahari,

Eds., vol. 12571. Springer, 2020, pp. 439–447. [Online]. Available: https://doi.org/10.1007/978-3-030-65310-1_31

[99] A. Stevens and J. De Smedt, "Explainable artificial intelligence in process mining: Assessing the explainability-performance trade-off in outcome-oriented predictive process monitoring," 2022.

[100] M. Stierle, J. Brunk, S. Weinzierl, S. Zilker, M. Matzner, and J. Becker, "Bringing light into the darkness - A systematic literature review on explainable predictive business process monitoring techniques," in *28th European Conference on Information Systems - Liberty, Equality, and Fraternity in a Digitizing World , ECIS 2020, Marrakech, Morocco, June 15-17, 2020*, F. Rowe, R. E. Amrani, M. Limayem, S. Matook, C. Rosenkranz, E. A. Whitley, and A. E. Quammah, Eds., 2021.

[101] M. Stierle, S. Weinzierl, M. Harl, and M. Matzner, "A technique for determining relevance scores of process activities using graph-based neural networks," *Decision Support Systems*, vol. 144, p. 113511, 2021.

[102] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3319–3328.

[103] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," 2017.

[104] 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, pp. 477–492.

[105] D. Tedesco and T. Tullis, "A comparison of methods for eliciting post-task subjective ratings in usability testing," in *Usability Professionals Association Conference (UPA)*, 2006, pp. 1–9.

[106] I. Teinemaa, M. Dumas, M. La Rosa, and F. Maggi, "Outcome-oriented predictive process monitoring: Review and benchmark," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, 07 2017.

[107] A. Terragni and M. Hassani, "Optimizing customer journey using process mining and sequence-aware recommendation," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019,*. ACM, 2019, pp. 57–65.

[108] R. Urraca, J. Antonanzas, F. Antonanzas-Torres, and F. J. Martinez-de Pison, "Estimation of daily global horizontal irradiation using extreme gradient boosting machines," in *International Joint Conference SOCO'16-CISIS'16-ICEUTE'16*, M. Graña, J. M. López-Guede, O. Etxaniz, Á. Herrero, H. Quintián, and E. Corchado, Eds. Cham: Springer International Publishing, 2017, pp. 105–113.

[109] W. M. P. van der Aalst, *Process Mining: Data Science in Action*. Springer, 2016.

[110] W. M. P. van der Aalst, "Object-centric process mining: Dealing with divergence and convergence in event data," in *Proceedings of 17th International Conference on Software Engineering and Formal Methods (SEFM 2019)*, ser. LNCS, vol. 11724. Springer, 2019, pp. 3–25.

[111] W. M. P. van der Aalst, A. Artale, M. Montali, and S. Tritini, "Object-centric behavioral constraints: Integrating data and declarative process modelling," in *Description Logics*, ser. CEUR Workshop Proceedings, vol. 1879. CEUR-WS.org, 2017.

[112] W. M. P. van der Aalst and A. Berti, "Discovering object-centric petri nets," *Fundamenta Informaticae*, vol. 175, no. 1-4, pp. 1–40, 2020.

[113] M. L. van Eck, N. Sidorova, and W. M. P. van der Aalst, "Guided interaction exploration in artifact-centric process models," in *Proceedings of the 19th Conference on Business Informatics (CBI)*, vol. 01. IEEE, 2017, pp. 109–118.

[114] M. Velmurugan, C. Ouyang, C. Moreira, and R. Sindhgatta, "Evaluating fidelity of explainable methods for predictive process analytics," in *Intelligent Information Systems*, S. Nurcan and A. Korthaus, Eds. Cham: Springer International Publishing, 2021, pp. 64–72.

[115] M. Velmurugan, C. Ouyang, C. Moreira, and R. Sindhgatta, "Evaluating stability of post-hoc explanations for business process predictions," in *Service-Oriented Computing*, H. Hacid, O. Kao, M. Mecella, N. Moha, and H.-y. Paik, Eds. Cham: Springer International Publishing, 2021, pp. 49–64.

[116] I. Venugopal, J. Töllich, M. Fairbank, and A. Scherp, "A comparison of deep-learning methods for analysing and predicting business processes," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[117] I. Verenich, M. Dumas, M. La Rosa, and H. Nguyen, "Predicting process performance: A white-box approach based on process models," *Journal of Software: Evolution and Process*, vol. 31, p. e2170, 03 2019.

[118] L. Waikhom and R. Patgiri, “A survey of graph neural networks in various learning paradigms: methods, applications, and challenges,” *Artificial Intelligence Review*, 11 2022.

[119] T. Wang, C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl, and P. MacNeille, “A bayesian framework for learning rule sets for interpretable classification,” *Journal of Machine Learning Research*, vol. 18, no. 70, pp. 1–37, 2017. [Online]. Available: http://jmlr.org/papers/v18/16-003.html

[120] S. Weinzierl, “Exploring gated graph sequence neural networks for predicting next process activities,” in *Business Process Management Workshops*. Springer International Publishing, 2022, pp. 30–42.

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

[122] S. Weinzierl, S. Zilker, J. Brunk, K. Revoredo, M. Matzner, and J. Becker, “XNAP: making lstm-based next activity predictions explainable by using LRP,” in *Business Process Management Workshops - BPM 2020 International Workshops, Seville, Spain, September 13-18, 2020, Revised Selected Papers*, ser. Lecture Notes in Business Information Processing, A. del-Río-Ortega, H. Leopold, and F. M. Santoro, Eds., vol. 397. Springer, 2020, pp. 129–141.

[123] B. Wickramanayake, Z. He, C. Ouyang, C. Moreira, Y. Xu, and R. Sindhgatta, “Building interpretable models for business process prediction using shared and specialised attention mechanisms,” *Knowledge-Based Systems*, vol. 248, p. 108773, 2022.

[124] M. T. Wynn, J. Lebherz, W. M. P. van der Aalst, R. Accorsi, C. Di Ciccio, L. Jayarathna, and H. Verbeek, “Rethinking the input for process mining: Insights from the xes survey and workshop,” in *Proceedings of the ICPM 2021 Process Mining Workshops*, ser. LNBIP, vol. 433. Springer, 2022, pp. 3–16.

[125] Y. Xia, L. He, Y. Li, N. Liu, and Y. Ding, “Predicting loan default in peer-to-peer lending using narrative data,” *Journal of Forecasting*, vol. 39, no. 2, pp. 260–280, 2020.

[126] X. Yin and J. Han, “Cpar: Classification based on predictive association rules,” *Lecture Notes of the Institute for Computer Sciences Social Informatics & Telecommunications Engineering*, vol. 24, pp. 236–255, 2003.

[127] Z.-H. Zhou, Y. Jiang, and S. Chen, “Extracting symbolic rules from trained neural network ensembles,” *AI Commun.*, vol. 16, pp. 3–15, 2003.