



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.




변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#) 

Master's Thesis

A rule-based framework for interpretable predictions
of business process outcomes using event logs

Suhwan Lee

Department of Industrial Engineering

Ulsan National Institute of Science and Technology

2021

A rule-based framework for interpretable predictions
of business process outcomes using event logs

Suhwan Lee

Department of Industrial Engineering

Ulsan National Institute of Science and Technology

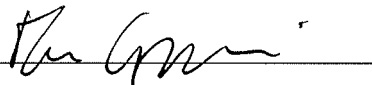
A rule-based framework for interpretable predictions
of business process outcomes using event logs

A thesis/dissertation submitted to
Ulsan National Institute of Science and Technology
in partial fulfillment of the
requirements for the degree of
Master of Science

Suhwan Lee

12/16/2012 of submission

Approved by



Advisor

Marco Comuzzi

A rule-based framework for interpretable predictions of business process outcomes using event logs

Suhwan Lee

This certifies that the thesis/dissertation of Suhwan Lee is approved.


12/16/2020 of submission

Signature



Advisor: Marco Comuzzi

Signature



Chiehyeon Lim: Thesis Committee Member #1

Signature



Sunghoon Lim: Thesis Committee Member #2

Abstract

The development of models for process outcome prediction using event logs has evolved with a clear focus on performance improvement. In this thesis we take a different perspective, focusing on obtaining interpretable predictive models for outcome prediction. In particular, we propose a method based on association rule mining, which results in inherently interpretable classification models. While association rule mining has been used with event logs for process model approximation and anomaly detection in the past, its application to outcome-based predictive model is novel. The proposed method defines how to pre-process logs, obtain the rules, prune the rules to a limited number that can be handled by human decision makers, and use the rules to predict process outcomes. The experimental results on real world event logs show that in most cases the performance of the proposed method is aligned with the one of traditional approaches, with only a slight decrease in some cases. We argue that such a decrease of performance is an acceptable trade-off in return for a predictive model that is interpretable by design.

Contents

I	Introduction	1
II	Background and related work	4
2.1	Business process management and process mining	4
2.1.1	Process discovery	7
2.1.2	Conformance checking	8
2.1.3	Improvements/Enhancement	8
2.2	Predictive process monitoring	8
2.2.1	General framework	8
2.2.2	Encoding features and predictive model	9
2.2.3	Next activity prediction and timestamp monitoring	9
2.3	Outcome predictive process monitoring	9
2.4	Explainable predictive model in process mining	10
2.5	Association rule mining	11
III	Problem definition and Framework	13
3.1	Problem definition	13
3.2	Framework	15
3.2.1	Pre-processing	15
3.2.2	Outcome prediction	16
IV	Evaluation	21
4.1	Datasets, model parameters and hyperparameters	21
4.2	Results and discussion	22

4.3	Visualising association rules	28
4.3.1	Predictive model	28
4.3.2	Case prediction	31
V	Conclusion	37
	References	39

List of Figures

1	BPMN 2.0 example	4
2	Event log in dependency graph	6
3	Business process life cycle	7
4	Predictive process monitoring workflow [1]	10
5	Framework	16
6	Example of pre-processing to obtain extended item matrix	17
7	3-step rule pruning process	19
8	Performance with/without intra-label pruning - BPIC 2015 event logs	23
9	Performance with/without intra-label pruning - BPIC 2011 and Sepsis event logs	24
10	ARM-based and ML-based outcome classification performance - BPIC 2015 event logs	26
11	ARM-based and ML-based outcome classification performance - BPIC 2011 and Sepsis event logs	27
12	BPIC 2015_1 event log network graph for different labels	29
13	Sepsis event log network graph (label 0) for different prefix lengths	30
14	BPIC 2015 event log process instance and rule visualization: Case outcome label 1	32
15	BPIC 2015 event log process instance and rule visualization: Case outcome label 0	33
16	Process instance and rule visualization with different prefix length	36

I Introduction

A business process comprises the series of operations conducted by an organization to achieve a given business goal. Business goals are the backbone of the organization to define the major objective of establishment and describe the essence of business. Typical examples of business process are patient diagnosis and treatment in a hospital or raw materials procurement in a traditional manufacturing companies. While performing these actions, organizations monitor their productive efficiency, modifying the business process to increase the probability of business goal accomplishment whenever necessary. *Business process management* (BPM) comprises a set of management tasks intended to guarantee that the process outcome maximizes the organization's benefit and to reduce the risk of missing the goal, e.g., analysis of bottlenecks of diagnosis waiting time in a hospital to enhance patient satisfaction or supply chain optimization in a procurement process to achieve cost reduction [2].

As business processes are executed, events capturing the activities executed in them can be generated and stored in a database. Process mining is a technique that aims at understanding these events and utilizing them to analyze the outcome and conformance of business processes [2]. Process mining includes process discovery, to discover visual models of the business processes, conformance checking, to check the extent to which the events generated by the execution of a process fit a process model, and process enhancement, i.e., analyzing event data to decide how to improve a process. Process mining is an academic discipline that has recently gained increasing traction in industry. The global process analytics market size was USD 190.14 Million in 2018 and the market capitalization is projected to grow at USD 4,759.24 Million by 2026 [3].

One of the emerging process mining techniques [4] is predictive process monitoring, which aims at building predictive models of various aspects of interests in a process. Regarding "aspects of interest in a process", predictive models are trained using event logs, i.e., collections of events logged during the execution of business processes, to predict the next activity that will be executed in running process cases [5], the expected time at which process cases will terminate [6], or a given outcome of process cases, for instance whether they will satisfy or not given execution constraints or service level objectives. More in general, any recorded information about events or activities could be the objective of a predictive model. For instance, a synthetic process outcome label may be defined by a data analyst based on domain-specific attributes available in an event log.

Beyond creating more accurate and higher performing predictive models, the data mining research is increasingly shifting its focus towards explainability and interpretability of predictive models. Predictive models that lack transparency provide users with answers, but leave them without a real understanding of the input and output causal relation behind the model [7]. As decision makers, users would like to perceive the logic of predictions from the moment a predictive model is created. This allows users to feel more confident when using the predictions to take decisions and when defending, if necessary, the decisions that they have taken. Explainability

and interpretability are obviously relevant also for predictive process mining where users, for example, may require to understand why a particular model predicts a particular activity to be the next one that will be executed based on the previously occurred events.

There are two ways to approach interpretability of machine learning models [7], i.e., interpretable model extraction and post-hoc explanation. The former implies to generate models that are inherently interpretable, such as rule-based models and, to a lesser extent, decision trees. The latter aims at explaining what type of knowledge has been acquired by models during the training, e.g., explaining the values obtained for weights and biases in a neural network in a human-interpretable way [8]. Post-hoc explanation exploits knowledge from the parameters of trained models and presents the acquired information in human-interpretable manners to the users. One way to interpret traditional machine learning method is understanding the input feature weight in terms of frequency, coverage, and accuracy to trained model by adding or excluding the features. Post-hoc explanation is also typical in deep learning models, which have been for long considered *black box*, where providing global explanations is more challenging than with more traditional machine learning techniques because of the complex structure, i.e., layers and hyperparameters, of the learning model.

In the predictive process monitoring field, several approaches to understand the trained model following the post-hoc explanation paradigm have been proposed. Previous research has mainly utilized deep learning models and calculated the feature importance weight from parameters from model layers, e.g., in the case of next activity prediction with Gated Graph Neural Networks (GGNN) [9] or estimating the running case outcome using LSTM models [10]. In case of the next activity prediction, parameters in neural networks are modified to classify historical executed activities weight into hierarchical groups to measure their impact on next activity prediction. Other examples of post-hoc explanation in predictive monitoring will be discussed more in depth in Chapter 2.4.

In this thesis, we propose to adopt a different paradigm to achieve explainability of models, i.e., interpretable model extraction. In particular, we consider the use case of predicting process outcomes using an association rule-based prediction model. Association rules provide a monotonic representation of relations in a dataset that delivers intuitive comprehension to users [11]. In the proposed approach, in a pre-processing stage, case and event level attributes, including process outcomes, are encoded into input vectors; then association rules are extracted and, specifically, we consider only the rules that contain the outcome labels in their consequent. Next, significant rules sorted by support level are pruned in order to retain a limited number of rules to determine (predict) the outcome of running cases. Depending on the number of rules that they satisfy, incomplete/running cases are classified either to have a positive or negative outcome.

We also propose a visual approach to support user interpretability of the predictive models. This approach comprises a way to visualize the rules composing a prediction model and a different way to visualize effectively how the rules of the model are activated (or not) to predict the

outcome of each individual running case. The significant association rules construct visualization method of predictive model in network graph form which has features and its frequency as node and edge respectively. For predicted case explanation, event and case attributes are illustrated in a dependency graph from start to final event and label-rule list with colored attributes and rules which fulfill the condition.

The thesis is organised as follows. Chapter II reviews background and the related work. Chapter III formalises the problem of outcome prediction, discussing the proposed approach and introducing the proposed framework of in detail. Experimental results on real world event logs and visualization methods are presented in Section IV and conclusions are finally drawn in Section V.

II Background and related work

In this chapter research background and related work are discussed to reveal where this research is positioned. We review here the literature about business process management and process mining and, more specifically, research contributions in the area of predictive monitoring, which is the focus of this thesis. This chapter is organized as follows. Section 2.1 reviews the business process and process mining in the general scope. Outcome predictive monitoring, one of the research topics on process mining, is discussed in Section 2.2. Section 2.3 introduces the application of association rule mining in the predictive data mining field as well as in process mining and the papers regarding interpretable predictive models in process mining are discussed in Section 2.4.

2.1 Business process management and process mining

In this section, first, we will discover what is process mining and business processes management. Then, the main use cases of process mining will be analysed, i.e., process discovery, conformance checking, and predictive monitoring.

Considering the relation between process mining and business process, it would be advised to figure the definition of business process before discussing process mining. Business process is a standardized set of activities from an organization, for example, a company or hospital, which operates through flows of events to achieve business goals [12]. One example scenario of business process is patient diagnosis process in hospital. To manage operation efficiency, a hospital records activities of patients from the beginning of diagnosis and treatment to the end of process (patient dismissal). For example, the process for a single patient may begin with reserving a doctor appointment and include seeing a doctor, collecting medicine from pharmacy, and finishing with payment.

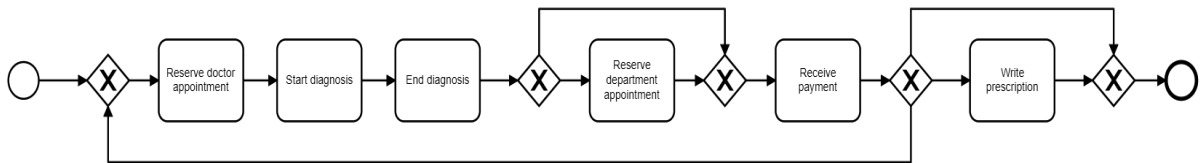


Figure 1: BPMN 2.0 example

A hospital may design the patient diagnosis process as shown in Figure 1 and observe how patients act regarding to the business process model, which is the blueprint of the process. Each patient’s behavior according to the business process model is an *instance* of this process. [12]

Table 1 is an event log of patients in the patient diagnosis process. The event log may have been collected, for instance, from the logs of the information systems of the hospital supporting the execution of the process. The event log contains a patient identifier, activity, and timestamp

Case ID	Activity	Timestamp
0001	Reserve doctor appointment	10/15/20 09:30 AM
0001	Start diagnosis	10/15/20 10:00 AM
0001	End diagnosis	10/15/20 10:20 AM
0001	Receive payment	10/15/20 10:35 AM
0001	Write prescription	10/15/20 10:55 AM
0002	Receive payment	10/15/20 10:10 AM
0002	Reserve doctor appointment	10/15/20 10:25 AM
0002	Start diagnosis	10/15/20 10:57 AM
0002	End diagnosis	10/15/20 11:06 AM
0002	Reserve department appointment	10/15/20 11:12 AM
0002	Receive payment	10/15/20 11:24 AM

Table 1: Hospital patient care system event log

information. The patient identifier indicates the process case (or instance) to which an event belongs and it is also called case id.

To depict all events and connections not only single case trace, event log could be also described in graphical form, i.e., using a dependency graph. Figure 2 is an example of an event log in graphical form using process analysis program 'DISCO'. The first graph in Figure 2 represents the frequency of activities and subsequence, while the second one concentrates on mean event duration between two activities. The green node at the left top and the red node at the right bottom of the graph are the start and end node of the process. Regarding the first graph, the darker node color and thicker edge mean the activity and the activity sub sequence more frequently occurred in an event log, respectively. In case of the second graph, mean event duration of sub sequence is presented along each edge. Also in this case the value determines the thickness of edges. By glancing these graphs, we could inspect basic knowledge from the event log and notice interesting facts such as activity importance in frequency level, multiple traces observed, or possible bottleneck points.

Naturally after perceiving and obtaining the business process data, the attempt to analyze and utilize the data are followed, i.e., Business Process Management. Figure3 is the business process management life cycle, which represents broad scope to prospect interest for organization improvements. From design phase to optimize phase, each phase along life cycle are correlated and dependant. Design and model phase are about how to conduct business process in the organization and depict observed process instance into graphical notation, such as the ones shown in Figure1. Based on the designed process and figured model, real process execution is followed. The monitoring phase is critical to detect and check whether recorded process instances follow the model as intended which contains estimating time to finish the process or prediction

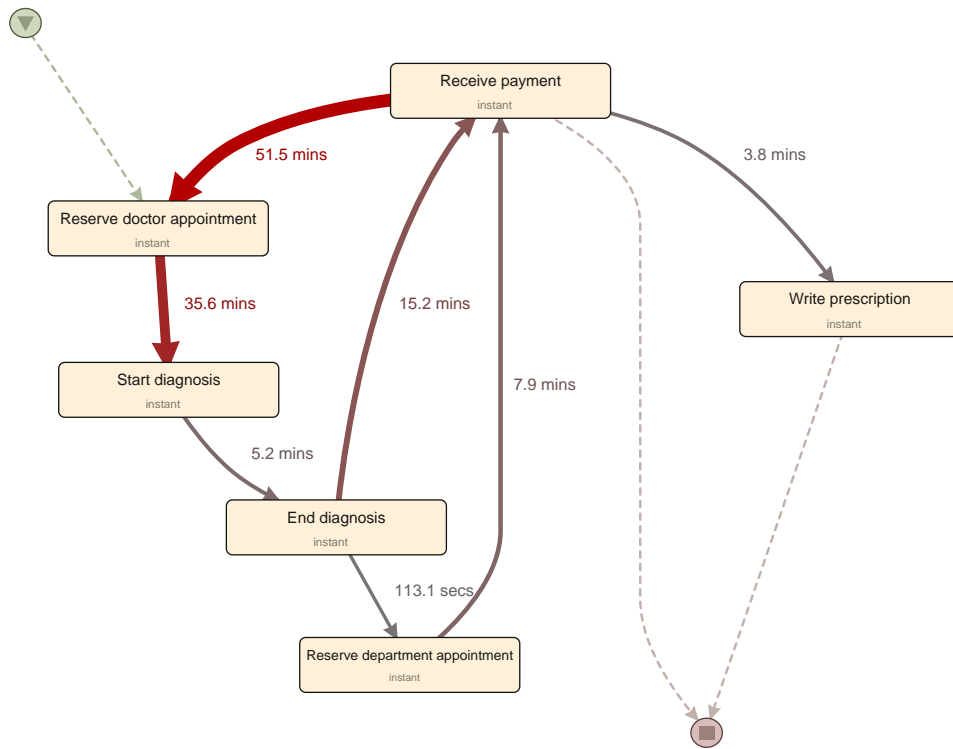
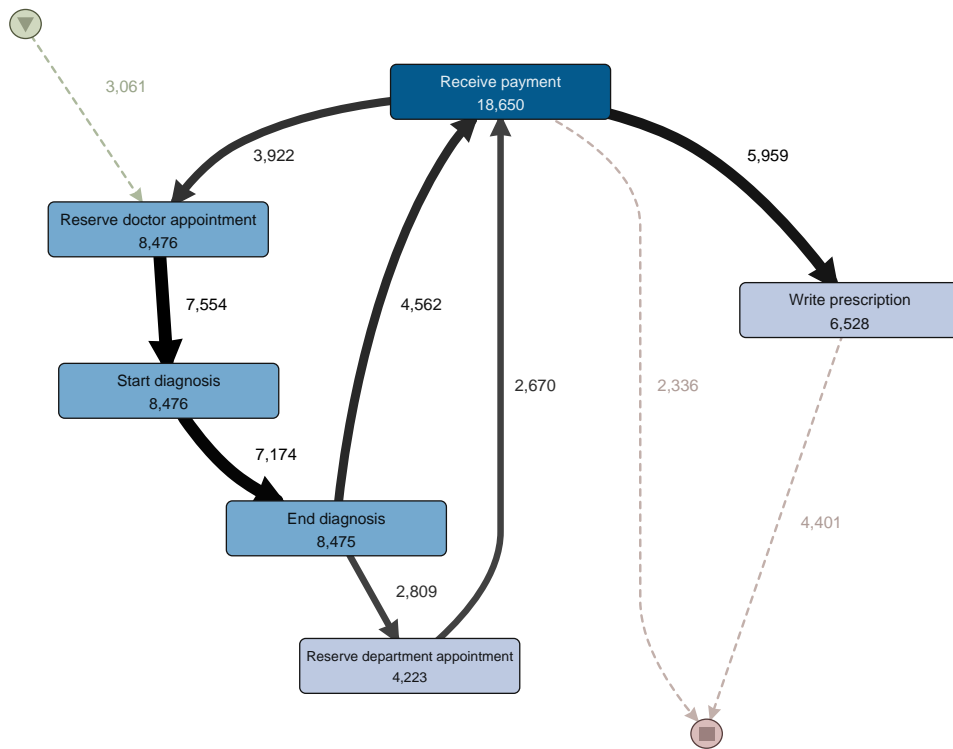


Figure 2: Event log in dependency graph

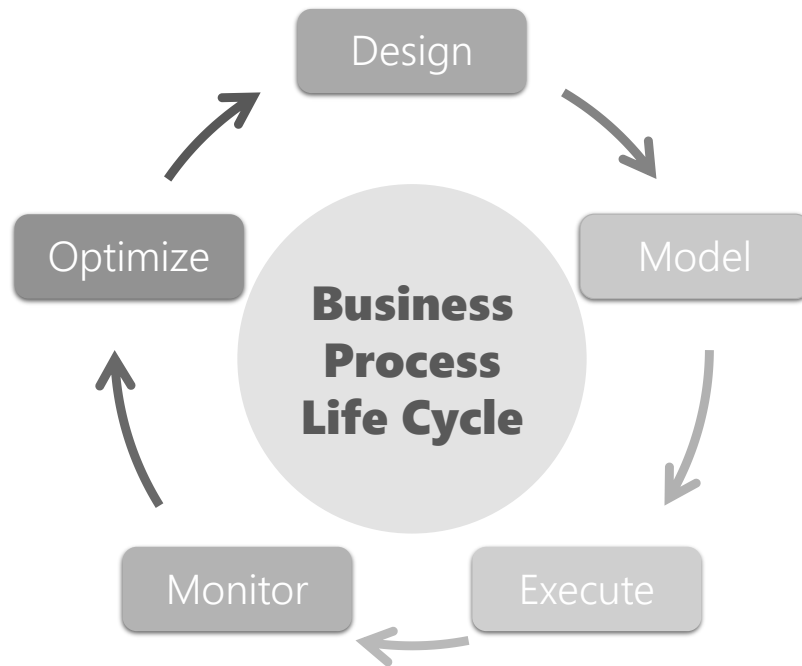


Figure 3: Business process life cycle

on potential amount of resource to execute the process. Along with the analysis and monitoring result, business process are continuously optimized and enhanced [13]. In terms of life cycle at business process, this thesis is positioned at the monitoring phase which predicts outcome of process instance aiming improvements with explainable reasoning.

From the extracted knowledge from event logs, there are three main point of research area in the process mining. Brief explanation of research area is followed.

- Process discovery
- Conformance checking
- Improvements/Enhancement

2.1.1 Process discovery

Process discovery aims at discovering models from the event log. One of the representative method is α -algorithm which figures out relation between activities in event log by time sequence order [14]. Between a pair of activities 4 types of relations could be discovered such as direct succession, causality, parallel, and choice depends on observed sequence patterns. Then visualization model in network graph depicts obtained activity relations and aggregates possible options of process in event log. Figure 1 shows BPMN 2.0 example with nodes, edges, and gates to describe traces along activity relations.

2.1.2 Conformance checking

The next research topic of process mining is conformance checking, which points at comparing discovered process model and event log to check conformance and discrepancy. The method which detects behavior of traces between process model and measures the difference to event log is called alignment-based conformance checking [15]. Each trace found in event log has a fitness score which is a cost to modify trace to match the process model by counting mismatch or omitted activities. Through calculating holistic event log fitness score, the performance of discovered process model in terms of reenacting recorded traces and depict event log can be measured.

2.1.3 Improvements/Enhancement

The last type of process mining research area is enhancement and improvement in real business process with extracted knowledge from process model and event log. The main theme of this thesis, predictive process monitoring, is allocated at this area and will be introduced at the next section.

2.2 Predictive process monitoring

One perspective to enhance the business process is predictive process monitoring. Predictive process monitoring is predictions and recommendations on future executed activity or required input data to minimize the likelihood of violation on business constraints and business goal [4]. There are different aspects of a process that can be predicted: 1) *the next occurred activity*, 2) *execution time of process*, or 3) *outcome of running cases*.

2.2.1 General framework

The general predictive monitoring framework is described in Figure 4. The framework consists of two main phases to get prediction results, which are training (offline) phase and testing (online) phase respectively. Building predictive models from event log to implement test case is the purpose of training phase. Therefore the testing phase is about preparation steps of ongoing test case for applying the predictive model.

There are 4 detailed steps in training phase from event log to predictive models. The first step is case filtering by certain prefix length and prefix log generation. Among the cases which has longer case length than given prefix length, prefixes of each case are extracted to construct prefix log. Bucketing homogeneous case within prefix log is the next step. The homogeneity criteria is defined by similarities of process state. The third step is encoding feature vectors of cases in buckets as proper input format for predictive models. Building predictive model using machine-learning model is the last step of training phase [16].

The testing (online) phase is about prediction on live and running case. In this phase to pre-process test case for predictive model bucketing based on filtered prefix and feature vector

encoding in training phase are reused. Encoding feature vectors is followed after clustering test cases considering similarities to buckets. The final step of framework is applying predictive model to pre-processed ongoing test case and getting prediction result.

2.2.2 Encoding features and predictive model

There are possible combinations of feature vector encoding techniques and machine-learning model selection for predictive model. By standardized form, there are 4 types of sequence encoding, last-state, aggregation, index-base, and tensor-based encoding. Sequence encoding techniques are vary by snap shot point of prefixes and information amount to consider. Last-state encoding takes the last event data of the case, on the other hand, aggregation and index-base encoding retains all event data of given prefixes. Depends on expression shape, aggregation encoding represents entire event sequence data into single entity. Index-base encoding enumerates series of data by distinguish able entities. And tensor-based encoding is extension of index-based encoding in 3-dimension matrix for deep learning algorithm, for example recurrent neural network (RNN) or long short-term memory (LSTM). In encoded feature vector, attributes in each case and event are recorded by data types. Attributes with continuous values are written down in as-is, on the other hand, categorical values are one-hot encoded. The most literature select different training model which is already developed in machine learning or deep learning area such as random forest, extreme gradient boosting, RNN, or etc. This is the predictive process monitoring along general framework and detailed prediction step explanation. Related literature will be introduced.

2.2.3 Next activity prediction and timestamp monitoring

Regarding activity and timestamp monitoring, Evermann et al. [5] implements LSTM and RNN deep learning model mainly used for natural language processing to predict next activity. This thesis is the early research to combine deep learning algorithm to predictive monitoring field. Tax et al. [17] suggest to implement deep neural network, LSTM model to estimate next activity and timestamp of next event. Since this thesis forecast sequence of upcoming events along timestamp until the end event, remaining cycle time could also be calculated by difference between predicted last event timestamp to predicted next event timestamp.

2.3 Outcome predictive process monitoring

Regarding outcome-oriented predictive monitoring, Teinmaa et al. [16] provide a general framework for outcome-based process monitoring and an insightful benchmark comparing the performance of different approaches from the perspective of sequence encoding, trace bucketing, and classification algorithms.

Notable approaches in this field are the ones presented by Di Francescomarino et al. [18], which adopts trace clustering before predictive classification on running cases, and Senderovich

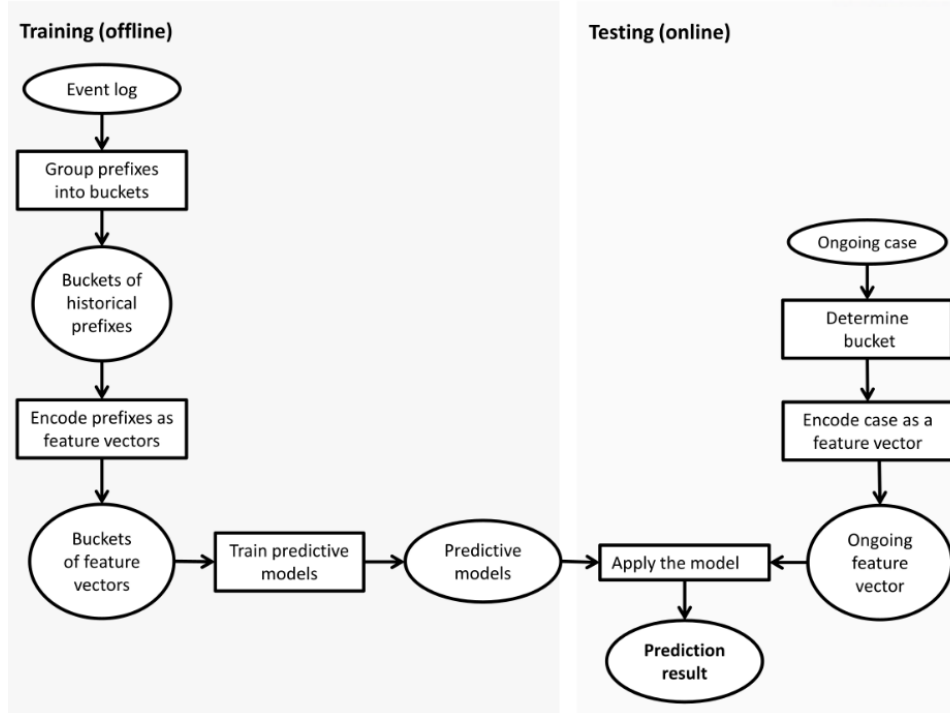


Figure 4: Predictive process monitoring workflow [1]

et al. [19], which adopts inter-case feature encoding to capture case interaction scenarios while predicting the the remaining time of patient treatments.

More recently, Wang et al. [20] have suggested an approach based on deep learning, i.e., bidirectional LSTM with attention mechanism, that outscores other classification algorithms traditionally adopted in this field, such as random forests and gradient boosting machines. In summary, previous works in this field have mainly experimented with different classification algorithms and feature encoding techniques to develop more accurate predictive models, rather than focusing on model interpretability.

2.4 Explainable predictive model in process mining

Regarding interpretable (explainable) predictive models in process mining, Brunk et al. [21] propose a method to predict the next event in a trace based on dynamic bayesian networks, which are an example of inherently interpretable classification techniques. Harl et al. [9] propose to use gated graph neural networks to predict binary process outcomes by transforming an event log into a graph adjacency matrix and feature maps. The trained models in this case is interpretable because each event is scored for its similarity to others in the training set. Therefore, events with higher score are intended to bear a stronger impact on the outcome. Finally, Galanti et al. [10] adopt Shapley value theory to explain factor influence on the outcome of traces predicted using an LSTM model.

2.5 Association rule mining

Another main research theme of this thesis is about association rule mining. Association rule mining is rule-based machine learning method to extract relations between items among transactions based on item frequency and confidence within rules [22].

Transactions	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Table 2: Market basket transactions

Here is an example scenario about how to get association rules. If we would like to know which products in market are bought together by customer, association rule mining can be a method to figure relations between items. Table 2 is a market basket transactions record and each row is items bought by different customers. According to the table 2, multiple items and itemsets are observed in single transaction and the item patterns are also appeared in other transactions. For instance, customer who bought beer and diaper bought bread at the same time and this comply with transaction 2 and 4. This relation can be represented in the rule 1.

$$\{\text{Beer, Diaper}\} \rightarrow \{\text{Bread}\} \quad (1)$$

Left side of the arrow is called antecedents and the right side is consequent of the rule. Considering this rule occurred in 2 transactions among total 5 transactions, we can get *support level* of the rule as $2/5$. Whether to check the rule occurrence is not coincidence, *confidence level* is calculated by dividing frequency of antecedents to consequent. In this case confidence level of the rule is $3/4$ since rule antecedents and consequent are observed 3 and 4 transactions respectively. Support and confidence level are vary by possible rule candidates and the rule which has higher coefficient than threshold would be important rule to remind.

Regarding the application of association rules in event log analysis, we highlight the work by Djenouri et al. [23], which applies frequent itemset mining to extract frequent itemsets in an event log that could approximate the relationship among activities normally captured by a process model. Previously, Maggi et al. [24] have applied association rule mining to extract rules that are then used to build process models. Finally, Böhmer and Rinderle-Ma [25] propose a trace anomaly detection technique that exploits association rules mined from an event log. The

support of association rules, in particular, is used to discriminate between normal and anomalous traces.

Association rule mining has already been adapted to create an inherently interpretable predictive model in other research fields. Mencía et al. [26] propose general methods for multi-label classification using association rule mining. They stress, in particular, that association rules highlight common causal relations in the data, which cater to better explainable predictions. In the health care field, Ji et al. [27] adopted fuzzy rule mining combined with causality rank of infrequent adverse drug reaction to distinguish relationship between target drug and adverse event response probability. In [28], next-day stock price movement as a categorical outcome (e.g., up, down, or hold) has been predicted using association rules extracted from historical stock movements.

In summary, most approaches in the literature for outcome-based predictive process mining are targeting performance improvement. Several recent papers address explainability of process predictive model, but only from a post-hoc perspective, mainly developing an interpretative stage to explain a deep learning predictive model. Association rule mining has been applied to process model approximation and anomaly detection, but not for obtaining inherently interpretable predictive models, which is the focus of the method proposed in this thesis.

III Problem definition and Framework

In this chapter problem definition and framework are discussed to issue two problems and suggest the solution respectively. The problems are about the implementation of association rule mining technique to outcome predictive monitoring and the general framework will be suggested to solve the expected problems which contains pre-processing and outcome prediction phase to create label classification model. This chapter is organized as follows. Section 3.1 introduces the problems about the rule-based predictive monitoring development and performance compared to the baseline models. The rule-based outcome predictive process monitoring framework is discussed in Section 3.2. Section 3.2.1 introduces the pre-processing phase, the first part of the framework, and the classification model creation and logic in the model are discussed in Section 3.2.2.

3.1 Problem definition

An event log EL contains events. An event e is a tuple $e = \langle c, \{(d_i, v_i)\}_{i=1, \dots, D} \rangle$, where c is the case id and (d_i, v_i) are a set of D attributes and their values. We refer \mathcal{E} as the universe of events, and to V_i as the domain of attribute d_i , which can be discrete, $V_i = \{v_{i,j}\}_{j=1, \dots, J}$, or continuous, $V_i = [v_{i,min}, v_{i,max}] \subseteq \mathbb{R}$. The case id, activity and timestamp attributes are normally present in all event log. We refer to them using the letters c , a , and t , respectively, and use a dotted notation to reference them when needed, e.g., $e.c$ to refer to the case id of event e . For instance, the event $e = (45, assess, 2020.1.2, Alice, amount = 1000, type = deep)$ captures the fact that, in a process case associated with loan request number 45 ($e.c$), the resource Alice has executed a deep *assessment* ($e.a$) of a loan request of 1000 USD on January 2nd, 2020 ($e.t$).

Note that any continuous domain V_i can be mapped into a discrete domain $\hat{V}_i = \{\hat{v}_{i,j}\}$. For instance, the loan amount in the example above can be mapped into three 3 levels (high, medium, low) according to the amount requested. We therefore introduce an attribute-specific discretisation function $disc_i : V_i \rightarrow \hat{V}_i$, with $disc(v_i) = \hat{v}_{i,j}$.

The sequence of events generated in a given case form a trace $\sigma = [e_1, \dots, e_n]$, where $\forall i \in [1, n]$, $e_i \in \mathcal{E}$, $\forall i, j \in [1, n]$, $e_i.c = e_j.c$, i.e., all events belong to the same case, and $\forall i \in [1, n-1]$, $e_i.t < e_{i+1}.t$, i.e., events in a trace are ordered in ascending order using the timestamp attribute. The universe of all traces is denoted by \mathcal{S} . Note that attributes of events e_i belonging to a trace σ may be the same $\forall e_i \in \sigma$. We refer to these attributes as case-level attributes. For instance, the amount requested in a loan request process is a case-level attribute. In the remainder, we refer to D_E and D_C as the set of event- and case-level attributes in an event log EL , respectively, and we assume that (i) all events in EL have the same event-level attributes D_E and (ii) all events in EL belonging to the same trace have the same case-level attributes D_C . Given a trace σ and an integer $l < n$, the prefix function pr returns the first l events of σ , that is, $pr(\sigma, l) = [e_1, \dots, e_l]$.

Now, let us assume that all attributes of events in EL are discrete, that is, continuous attributes in an event log have been discretised if needed. For each attribute d_i , we define the

attribute itemizer function $it_i^a : \mathcal{D} \rightarrow \{0, 1\}^J$, which maps an attribute d_i into a sequence of J binary items, obtained by one-hot encoding the value of an attribute within its domain, that is $it_i^a(d_i) = \{dummy_j\}_{j=1, \dots, J}$, with:

$$dummy_j = \begin{cases} 1 & \text{if } d_i = v_{i,j} \\ 0 & \text{if } d_i \neq v_{i,j} \end{cases}$$

The event itemizer $it^e : \mathcal{E} \rightarrow \{0, 1\}^n$ maps an event into the set of items derived from its event-level attributes: $it^e(e) = \bigcup_{d_i \in e} it^a(d_i)$, with $d_i \in D_E$.

The trace itemizer $it^t : \mathcal{S} \rightarrow \{0, 1\}^n$ maps a trace, possibly incomplete, into the set of items derived from its events and from case-level attributes: $it^t(\sigma) = \bigcup_{d_i \in D_C} it^a(d_i) \bigcup_{e_i \in \sigma} it^e(e_i)$.

Given a prefix length L , we define an item matrix T^L containing, in each row, the items generated by a prefix in an event log:

$$T^L = \begin{bmatrix} it^t(pr(\sigma_1, L)) \\ \vdots \\ it^t(pr(\sigma_N, L)) \end{bmatrix}$$

Let us refer to the columns of T^L as $\mathcal{X}_1, \dots, \mathcal{X}_P$. Let us also define a labeling function $y : \mathcal{S} \rightarrow \mathcal{Y}$ mapping a trace $\sigma \in \mathcal{S}$ to its class label $y(\sigma) \in \mathcal{Y}$, with \mathcal{Y} being the domain of the class labels. For outcome predictions, \mathcal{Y} is a finite set of categorical outcomes. Consistently with the literature [16], we consider binary outcomes, i.e. $\mathcal{Y} = \{0, 1\}$. Note that all prefixes generated from a trace σ have the same class label.

An outcome classifier $oc : \mathcal{X}_1 \times \dots \times \mathcal{X}_P \rightarrow \mathcal{S}$ is a function mapping the items of a (possibly incomplete) trace into its class label, that is, $oc(item^t(\sigma)) = y$.

A machine learning-based (ML-based) classifier (*mlbc*) is an *oc* developed using machine learning classification techniques, such as a decision tree or an artificial neural network. In other words, a ML-based classifier uses the items \mathcal{X}_p as features to train a model that predicts the outcome label of traces. The literature typically has focused on this type of techniques to solve the outcome prediction task in predictive monitoring [29].

An association rule mining-based (ARM-based) classifier (*armb*) is an *oc* developed using association rule mining techniques. Specifically, given a training set of traces in an event log \mathcal{S}_{train} , developing an *armb* implies first to extract association rules from an item matrix T_{ext}^L extended with items generated from the class labels y . We define 2 label itemizer functions $it_i^{l,0} : \mathcal{S} \rightarrow \{0, 1\}$ and $it_i^{l,1} : \mathcal{S} \rightarrow \{0, 1\}$ that map a trace $\sigma \in \mathcal{S}$ into an item capturing its class label value generated as follows:

$$it_i^{l,0}(\sigma) = \begin{cases} 1 & \text{if } y(\sigma) = 0 \\ 0 & \text{if } y(\sigma) = 1 \end{cases}$$

and

$$it_i^{l,1}(\sigma) = \begin{cases} 1 & \text{if } y(\sigma) = 1 \\ 0 & \text{if } y(\sigma) = 0 \end{cases}$$

The *extended* item matrix is now defined as follows:

$$T_{ext}^L = \begin{bmatrix} it^t(pr(\sigma_1, L)), it^{l,0}(pr(\sigma_1, L)), it^{l,1}(pr(\sigma_1, L)) \\ \vdots \\ it^t(pr(\sigma_N, L)), it^{l,0}(pr(\sigma_N, L)), it^{l,1}(pr(\sigma_N, L)) \end{bmatrix}$$

The association rules extracted from T_{ext}^L are then used to predict the outcome of new incomplete traces in the future. An association rule r (simply a *rule* in the remainder) is a relation between two disjoint set of items in T_{ext}^L , namely the antecedent X and the consequent Y , that is, $r : X \rightarrow Y$.

In this thesis, we tackle the following two problems:

- Devise a method to develop an ARM-based classifier to solve the outcome prediction task in process predictive monitoring;
- Compare the performance of the ARM-based classifier with state-of-the-art ML-based classifiers for the same prediction task.

The first problem is addressed in the next section, while the second one is addressed by Section IV, which presents the experimental evaluation of the proposed method to develop an ARM-based classifier.

3.2 Framework

The framework of this research is depicted in Figure 5. The initial steps concern acquiring an event log and pre-processing it for the outcome prediction task. The pre-processing phase is discussed in detail in Section 3.2.1. The remaining phases are presented in Section 3.2.2.

3.2.1 Pre-processing

A first pre-processing step concerns the discretisation of continuous attributes in an event log. Values of attributes with continuous domains are clustered into 3 groups, usually labelled 'small', 'medium', and 'large', using the 33% and 66% percentiles as thresholds, e.g., a value is mapped to the categorical value 'medium' if it falls in a percentile comprised between the 33% and 66%.

Next, to train and test ML-based predictive models, an event log is pre-processed using prefix-length bucketing and index-based encoding [30]. That is, we create a different model for each prefix length, trained/tested using prefixes pre-processed to obtain a fixed-length features vector. All attributes of an event log are pre-processed using one-hot encoding. Note that in our framework all attributes are discrete.

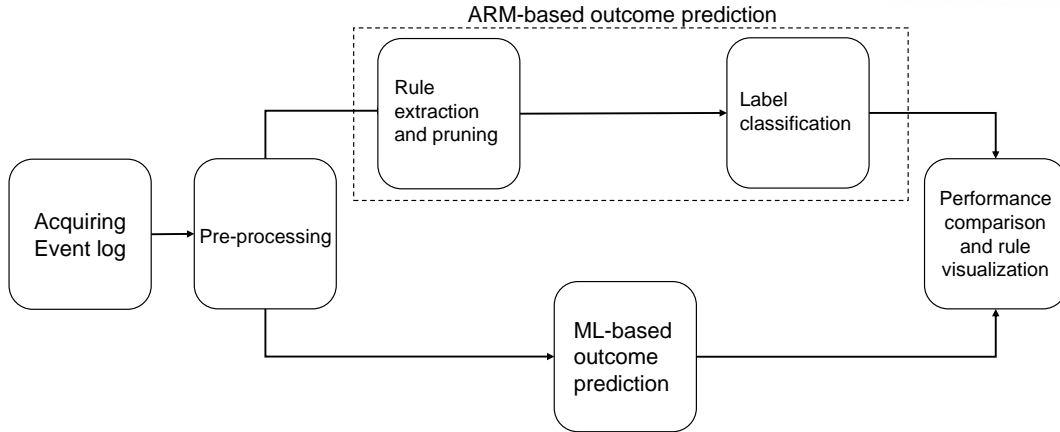


Figure 5: Framework

For ARM-based models, the objective of pre-processing is to obtain an extended item matrix from which rules can be extracted. Similarly to the case of ML-based models, we consider prefix-length bucketing and index-based encoding, that is, we create different item matrices for each prefix length. Figure 6 shows an example of pre-processing an event log where the continuous attributes are first discretised and then attributes are one-hot encoded to obtain an item matrix.

3.2.2 Outcome prediction

As ML-based models, we consider gradient boosting machines (XGboost) and random forests. Both classifiers have shown to perform well in event log predictive monitoring use cases [19]. The details about the hyperparameterisation of these models are given in the next section before introducing the experimental results.

Outcome prediction in the case of ARM-based is comprised of 2 steps, i.e., (i) rule extraction and pruning and (ii) label classification.

Rule extraction and pruning

Rules are extracted from the extended item matrix using the apriori algorithm [31], which is an exact method that, given a minimum support level in input, guarantees that all rules with equal or higher support level are discovered. Let us refer to \mathcal{R} as the set of rules extracted by applying apriori on T_{ext}^L . Association rule extraction algorithms are parameterised using the minimum support level and the confidence level. The support of a rule is defined as the number of observations (or *transactions*) in a dataset for which the rule holds. Let us define the function $sat : \mathcal{S} \times \mathcal{R} \rightarrow \{0, 1\}$ that captures whether a rule $r = X \rightarrow Y$ holds for a given trace σ (for the sake of clarity we omit any reference to the prefix length from now on):

$$sat(\sigma, r) = \begin{cases} 1 & \text{if } X, Y \subseteq item^t(\sigma) \cup item^0(\sigma) \cup item^1(\sigma) \\ 0 & \text{otherwise.} \end{cases}$$

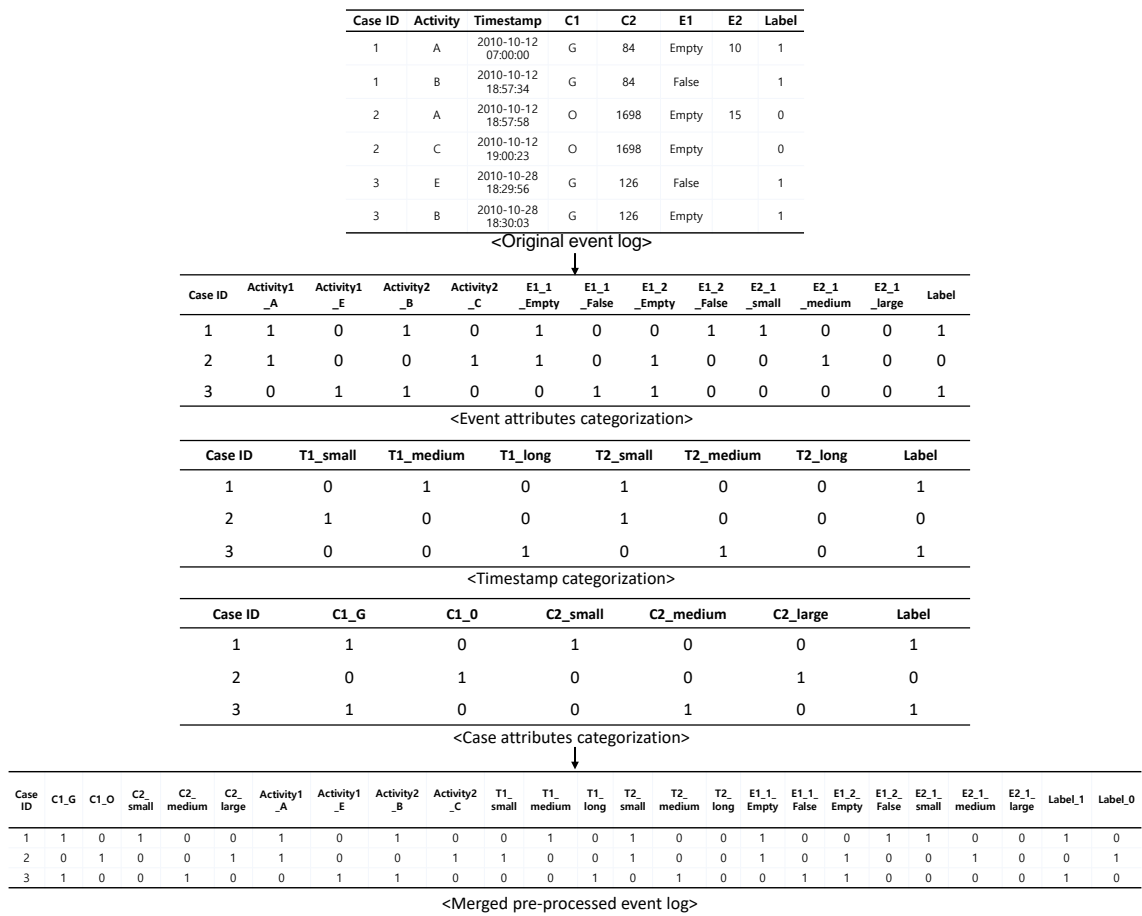


Figure 6: Example of pre-processing to obtain extended item matrix

Then, the support of a rule is defined as:

$$sup(r) = sup(X \cup Y) = \frac{\sum_{\sigma \in EL} sat(r, \sigma)}{|EL|}$$

where $|EL|$ is the number of traces in an event log.

The confidence $conf(r)$ of a rule is the proportion of observations that contain the antecedent X which also contain the consequent Y , that is:

$$conf(r) = sup(r)/sup(X)$$

where $sup(X) = |\{\sigma \in EL : X \subseteq item^t(\sigma)\}|$ is the number of traces for which the antecedent X holds.

Next, we need to prune the rules discovered to obtain a set of rules that are relevant for outcome prediction and limited in number so that can be handled by human decision makers [32]. In our framework, we distinguish among three types of pruning, which are applied in sequential order, i.e., the *label* pruning, the *cross-label* pruning, and the *intra-label* pruning (see Figure 7).

Label pruning. The label pruning (or *outcome* pruning) addresses the need to retain only relevant rules for outcome prediction. In this case, rules are relevant when they can be used to predict the outcome of process cases. We translate this requirement into a policy that retains only rules containing the items derived from the case outcome label in the consequent, i.e., for which $item^0(\sigma) \in Y$ or $item^1(\sigma) \in Y$. This type of rules, in fact, capture the fact that a combination of values assumed by a given set of attributes (i.e., the antecedent items) imply with a certain level of support the values, possibly among other attributes, of the attribute capturing the value assumed by the outcome label for a case. Let us refer to $\mathcal{R}^0 \subseteq \mathcal{R}$ and $\mathcal{R}^1 \subseteq \mathcal{R}$ as the set of rules containing the item $item^0(\sigma)$ and $item^1(\sigma)$, respectively.

Cross-label pruning. The cross-label pruning addresses the fact that, after having applied label pruning, the sets \mathcal{R}^0 and \mathcal{R}^1 may contain rules that have the same antecedent X . These rules are not significant for the outcome-based predictive monitoring, because in practice they are not able to discriminate between the values assumed by the label. In other words, with this type of rules, a given combination of attribute values in the antecedent can be used to imply both values of the outcome label.

After this pruning, there may still be two or more rules within the same group, i.e., \mathcal{R}^0 or \mathcal{R}^1 , that have the same antecedent. For instance, given r_1 and r_2 having the same antecedent X , r_1 's antecedent may only contain the class label item, e.g., $Y = item^0(\sigma)$, while the consequent of r_2 may also contain additional items, e.g., $Y = item^0(\sigma) \cup A$, with $A \subseteq item^t(\sigma)$. Among the rules having the same antecedent, we retain the one with *conviction* closest to 1. The conviction $conv(r)$ of a rule can be interpreted as the expected frequency at which the antecedent X occurs in a trace without the consequent Y , that is, the frequency at which a rule is likely to make a wrong prediction:

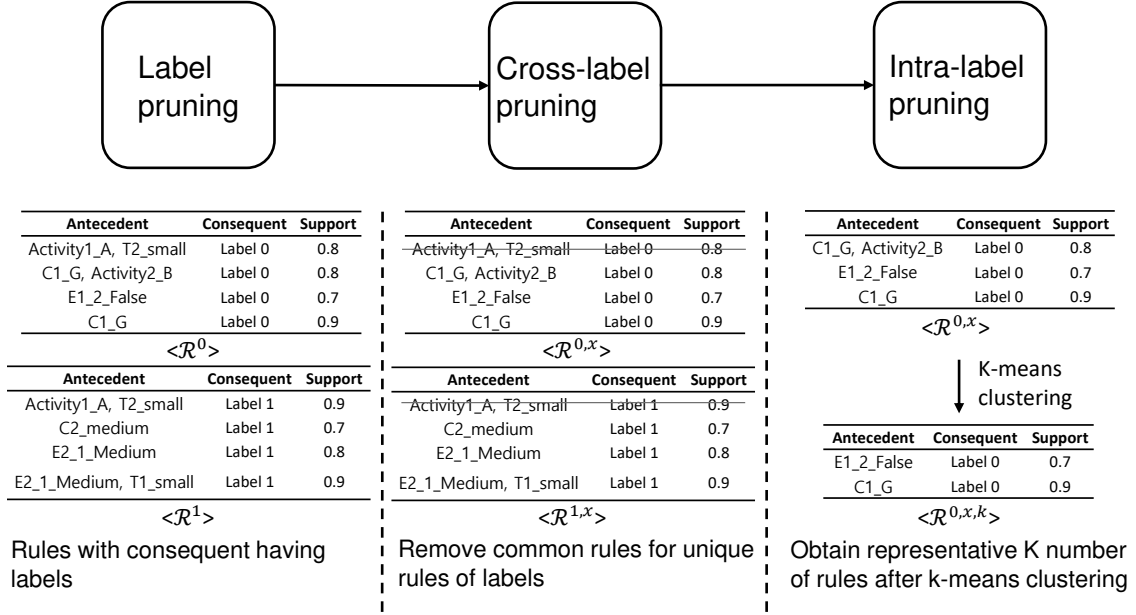


Figure 7: 3-step rule pruning process

$$conv(r) = \frac{1 - sup(Y)}{1 - conf(r)}$$

where $sup(Y) = |\{\sigma \in EL : Y \in item^t(\sigma) \cup item^0(\sigma) \cup item^1(\sigma)\}|$ is the number of traces for which the consequent Y holds. Let us refer to $\mathcal{R}^{0,x}$ and $\mathcal{R}^{1,x}$ as the label 0 and label 1 set of rules obtained after applying cross-label pruning, respectively. Note that, by construction, $\mathcal{R}^{0,x} \cap \mathcal{R}^{1,x} = \emptyset$.

Intra-label pruning. The final pruning step is optional and is applied in those cases where the number of rules retained after cross-label pruning is still high. The intra-label pruning applies k-means clustering [33], an unsupervised clustering algorithm, to obtain k clusters of rules. Within each cluster obtained, the rule with the highest support is retained as cluster representative. We refer to $\mathcal{R}^{0,x,k}$ and $\mathcal{R}^{1,x,k}$ as the label 0 and label 1 rules obtained after applying the intra-label pruning, respectively.

In the experiment, we consider $k = 20$, that is, we prune the number of rules to 20. We argue, in fact, that 20 is a number of rules that can be understood by a human decision maker. In Section IV we also propose a way to visually depict these 20 rules organically using network graphs, to improve their understanding and interpretability for human decision makers.

Label classification.

After having obtained a final set of rules for each label through rule extraction and pruning, the next step is to determine how these rules are used to classify the outcome of process cases.

We propose two methods ,namely SBOP (Support-Based Outcome Prediction) and RSOP (Rule Satisfaction-based Outcome Prediction).

For the sake of simplicity, when not strictly necessary, in the remainder we refer to $\mathcal{R} = \mathcal{R}^0 \cup \mathcal{R}^1$ as the pruned set of rules obtained after applying all 3 pruning steps, written as the union of rules containing the label 0 (\mathcal{R}^0) and label 1 (\mathcal{R}^1) item in the consequent. Given a trace σ , let us also refer to $\mathcal{R}_\sigma \subseteq \mathcal{R}$ as the set of rules that hold for σ , i.e., $\mathcal{R}_\sigma = \{r \in \mathcal{R} : sat(r, \sigma) = 1\}$. When needed, we will also refer to $\mathcal{R}_\sigma^0 \subseteq \mathcal{R}^0$ and $\mathcal{R}_\sigma^1 \subseteq \mathcal{R}^1$ as the set of rules that hold for σ having label 0 and 1 in their consequent, respectively.

In SBOP, the predicted label of a trace σ is the label $y \in \{0, 1\}$ characterised by the highest rule satisfaction average weighted using rule support:

$$y^{SBOP}(\sigma) = \begin{cases} 1 & \text{if } \frac{\sum_{r \in \mathcal{R}_\sigma^1} sup(r)}{\sum_{r \in \mathcal{R}^1} sup(r)} > \frac{\sum_{r \in \mathcal{R}_\sigma^0} sup(r)}{\sum_{r \in \mathcal{R}^0} sup(r)} \\ 0 & \text{otherwise} \end{cases}$$

The SBOP method fails when no pruned rules are found for one specific label, i.e., $\mathcal{R}_\sigma^0 = \emptyset$ or $\mathcal{R}_\sigma^1 = \emptyset$. In this case, in fact, the denominator of one of the two terms in the condition of y^{SBOP} collapses to 0.

RSOP addresses this limitation by determining the predicted label only considering one set of rules at a time. There are therefore two versions of RSOP, i.e. RSOP0 and RSOP1, depending on which set of rules is considered. For RSOP0, given a threshold T , a trace σ is assigned the predicted label 0 if the average number of rules in \mathcal{R}^0 that hold for σ weighted by their support is greater than T . The label 1 is assigned otherwise:

$$y^{RSOP0}(\sigma) = \begin{cases} 0 & \text{if } \frac{\sum_{r \in \mathcal{R}_\sigma^0} sup(r)}{|\mathcal{R}^0|} > T \\ 1 & \text{otherwise} \end{cases}$$

Similarly, in RSOP1 the label 1 is assigned to a trace if the average number of rules that hold for σ in \mathcal{R}^1 weighted by their support is greater than T , and label 0 is assigned otherwise:

$$y^{RSOP1}(\sigma) = \begin{cases} 1 & \text{if } \frac{\sum_{r \in \mathcal{R}_\sigma^1} sup(r)}{|\mathcal{R}^1|} > T \\ 0 & \text{otherwise.} \end{cases}$$

In the experiments, we use the value $T = 0.7$, which signifies that a trace is assigned a given label if it satisfies at least 70% of the rules containing that label in the consequent. This value has been set empirically after pilot experiments.

IV Evaluation

The evaluation of the proposed predictive monitoring framework and visualizing rule-based monitoring are discussed in this chapter. Section 4.1 presents the datasets used in our evaluation, along with the hyperparameters of ML-based classifiers and the parameter values used for ARM. The performance comparison between rule-based and traditional outcome predictive models is presented in Section 4.2. Finally, Section 4.3 discusses rule visualisation and interpretation, suggesting, in particular, a visualisation approach for rules based on network graphs and process instance on a dependency graph.

4.1 Datasets, model parameters and hyperparameters

We consider 5 event logs publicly available at <https://data.4tu.nl/>: the BPIC 2011, BPIC 2015_1,2,3¹, and the Sepsis event logs. The BPIC 2011 is an event log about medical area information in a Dutch academic hospital. BPIC 2015 are logs from a process of building permit requests at different Dutch municipalities. The Sepsis event log refers to a process of sepsis diagnosis and treatment in a hospital. These logs have been chosen because they contain an outcome label and have been used by previous research on outcome-based process predictive monitoring [16]. Table 3 shows the descriptive statistics of these event logs. Note that event logs differ widely for number of activities, number of attributes and label class proportion.

	# activities	# cases	# variants	# categorical Case attributes	# continuous case attributes	# categorical event attributes	# continuous event attributes	# outcome 0 cases	# outcome 1 cases
BPIC 2011	677	1143	981	4	6	5	0	683	460
BPIC 2015_1	289	1199	1100	8	1	3	0	910	289
BPIC 2015_2	304	832	828	8	1	3	0	670	162
BPIC 2015_3	277	1409	1349	8	1	3	0	1122	287
Sepsis cases	16	846	846	23	1	2	3	671	378

Table 3: Descriptive statistics of event logs used for evaluation

In all logs, given a temporal linear temporal logic constraint predicated on the order and occurrence of activities, the outcome label is set to 1 for cases satisfying the constraint and to 0 for other cases. In the BPIC 2015 event logs, the constraint is that activity 'send confirmation receipt' is eventually followed by activity 'retrieve missing data'. In BPIC11, the constraint is that either the activity 'ca-19.9 tumormarker' or 'ca-125 mbv meia' occur in a trace. In the Sepsis event log, the constraint is simply that the activity 'Release A' does not occur in a trace.

ARM requires to specify a *support* threshold and a rule *confidence* threshold. Since each event log has different complexity in terms of number of activity labels or variants, different values of these parameters must be considered for each event log. For the BPIC 2011 event log, we consider 0.8 as minimum support threshold and 0.7 as confidence threshold. The minimum support threshold for BPIC 2015 is 0.6 and the confidence threshold is set to 0.7. For the Sepsis

¹Note that there are 5 different instances of this event log, with data from different municipalities. We consider the instance numbered as 1, 2, and 3.

event log, we consider 0.9 as both minimum support and confidence threshold.

Note that a higher support/confidence threshold is likely to lead to the discovery of a lower number of more meaningful rules for a predictive model. However, in some cases too high thresholds may lead to discover no rules at all. At the same time, lowering the support threshold increases the memory requirements of the rule extraction algorithms. The threshold values that we have chosen aim at striking a balance between number of rules discovered, meaningfulness of rules, and memory requirements of rule discovery. In one particular case (Sepsis), which was particularly demanding in respect of the memory required for rule extraction, we were not able to successfully complete the rule extraction task with a threshold lower than 0.9.

As far as hyperparameters of machine learning classifiers are concerned, for Random Forest (RF) we consider the implementation in the Python package 'Scikit-learn' [34], with the number of estimators set to 100. For XGBoost (XGB), we consider 500 estimators, 0.1 learning rate, and 4 as maximum tree depth.

We consider prefixes between 2 and 10 for all event logs. For example, if prefix length is equal to 5, then our proposed approach, RF and XGB are trained and tested on a dataset containing all the prefixes of length 5 in the event log, that is, partial traces considering the first 5 events of traces with at least 5 events. To cope with sampling bias, we consider 10-fold cross-validation. The proportion of train and test set sampling in every replication of the experiment is 70% to 30%. The code to reproduce the experiments is publicly available at <https://github.com/ghksdl6025/rule-based-predictive-monitoring>.

4.2 Results and discussion

Before comparing the performance of ML-based and ARM-based classifiers, we compare the performance of the proposed ARM-based classifier with and without the optional intra-label pruning discussed in Section 3.2. This allows us to validate the proposed pruning policy and also to elect the best performing method for each event log. Table 4 shows the average number of rules extracted after label pruning and cross-label pruning for all event logs, i.e., the size of $\mathcal{R}^{0,x}$ and $\mathcal{R}^{1,x}$. It can be seen that in some cases the number of rules retained is still well above 20.

Figure 8 and 9 compare the performance (precision, recall, and F1-score) obtained by ARM-based classifiers that consider the rules retained before and after the intra-label pruning. Note that, since our ARM-based method separates rule extraction by label, we also present the results separated by label. We also remind here that after the intra-label pruning, only $k = 20$ pruned rules are considered for all methods of label classification.

It can be noted that the 2 corresponding lines (with/without intra-label pruning) for a given method, i.e., RSOP0, RSOP1, or SBOP, often overlap, particularly at lower prefixes. Generally, it is not possible to elect whether intra-label pruning policy works better for all methods. In some cases, for instance for the BPIC 2015 event logs, the intra-label pruned rules achieve a better performance than the corresponding group of rules not pruned. This can be explained

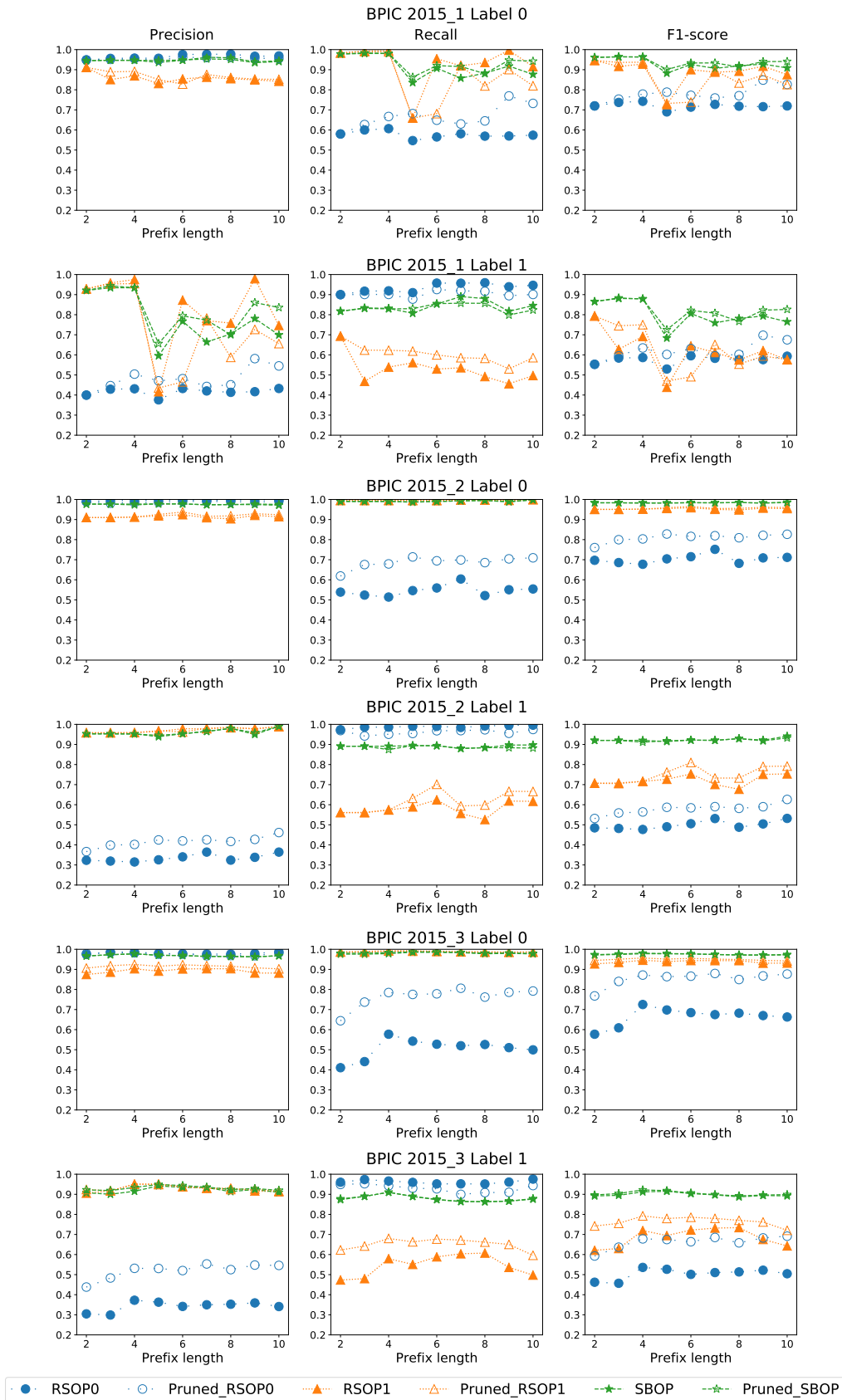


Figure 8: Performance with/without intra-label pruning - BPIC 2015 event logs

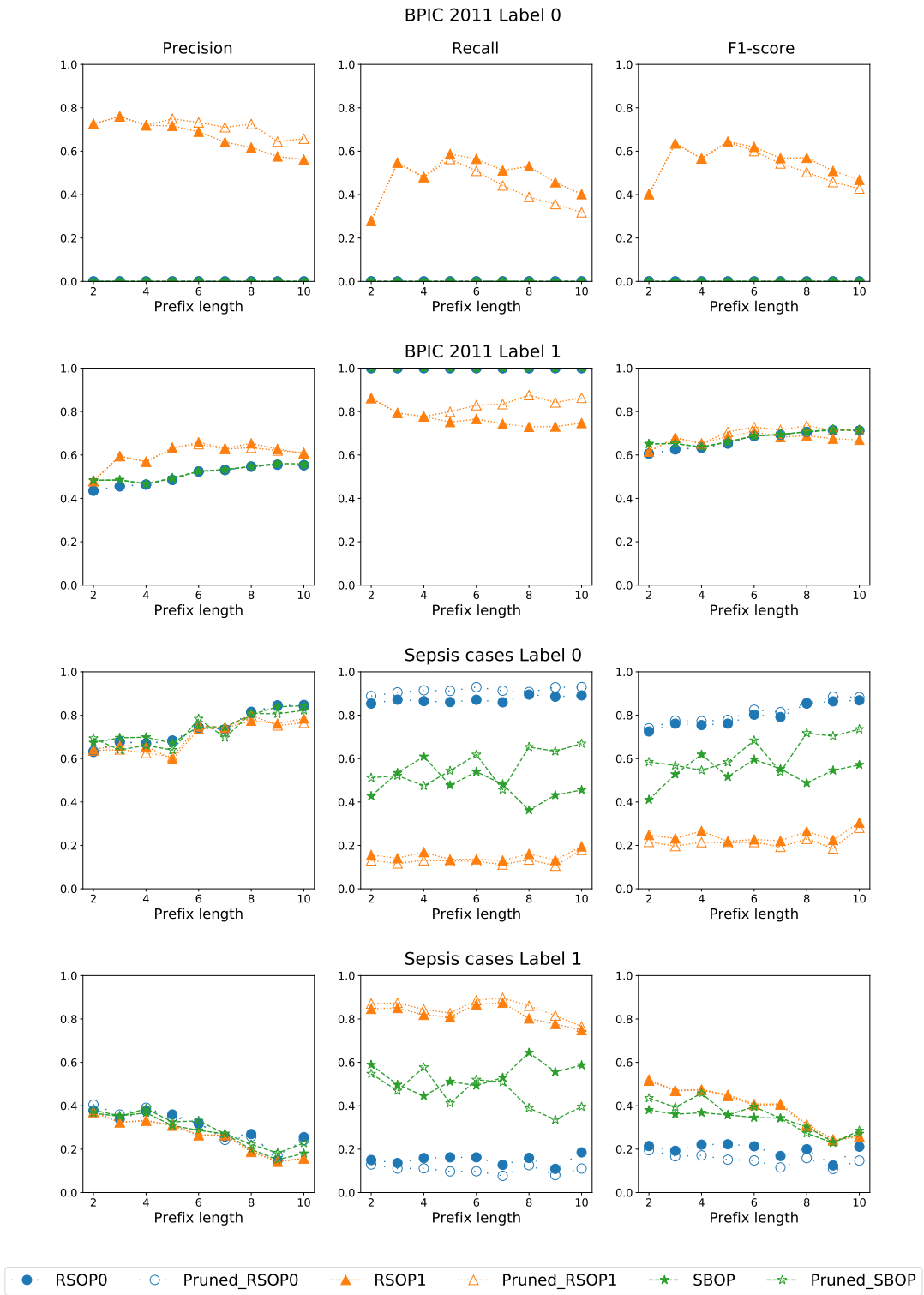


Figure 9: Performance with/without intra-label pruning - BPIC 2011 and Sepsis event logs

Event log	Label	Prefix length								
		2	3	4	5	6	7	8	9	10
BPIC 2011	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	1	4.0	9.4	17.6	79.2	148.8	557.2	1,375.8	4,484.2	6,606.4
BPIC 2015_1	0	21.0	34.6	32.4	40.8	56.4	59.6	60.2	77.6	85.6
	1	8.0	25.6	24.8	92.4	106.6	103.2	121.6	140.4	131.0
BPIC 2015_2	0	65.8	128.2	143.4	149.4	202.2	185.4	208.6	208.4	207.2
	1	15.0	20.4	19.8	20.4	30.2	31.8	39.4	38.6	34.0
BPIC 2015_3	0	61.8	109.8	155.8	185.4	239.2	226.4	222.6	268.0	282.0
	1	24.0	46.8	57.0	58.4	56.4	60.4	58.6	82.2	108.0
Sepsis cases	0	1,517.6	5,654.8	4,412.0	7,153.6	9,606.4	9,084.8	9,339.2	16,116.8	14,343.2
	1	2,709.6	1,948.4	2,846.4	406.4	413.6	244.8	610.4	1,222.4	1,375.2

Table 4: Average number of unique rules per event log and outcome label before intra-label pruning

considering that, before intra-label pruning, the set of retained rules may also contain spurious rules with lower support and/or conviction, which may perturb the prediction task, while the 20 rules extracted after intra-label pruning are more effective in leading to a correct prediction.

The performance of the proposed ARM-based classifiers compared against the one of ML-based classifiers is shown in Figure 10 and 11. Note that, for each ARM-based method, in these figures we consider the best performing set of rules, i.e., before or after intra-label pruning, as highlighted by the results of Figure 8 and 9.

Before diving into the interpretation of individual experiment results, we highlight the following two general issues:

- ML-based classifiers (RF and XGB) generally represent an upper bound for the performance of the classification model. This should not surprise since, as we stressed in the Introduction, the proposed ARM-based classification approach trade-offs the interpretability of a rule-based model for performance in the classification task;
- It does not make sense to compare and discuss the performance of different models when this is low. In this regard, let us consider the Sepsis event log label 1 experiment (bottom-right in Figure 11). In this case, the classification performance is low for all approaches at all prefix lengths, about 50% lower on average than the corresponding performance on label 0. Such a situation simply signals that building a high performing outcome predictive model for label 0 traces in this event log is very challenging and that both the ML- and ARM-based approaches considered in this thesis fail at this task. Therefore, in the remainder of this section we will not focus further on this particular experiment.

Based on these results, we highlight the following insights regarding our proposed approach:

1. Except the case of Sepsis label 1 discussed above, in all experiments there is at least one ARM-based label classification method showing comparable performance with ML-based approaches;

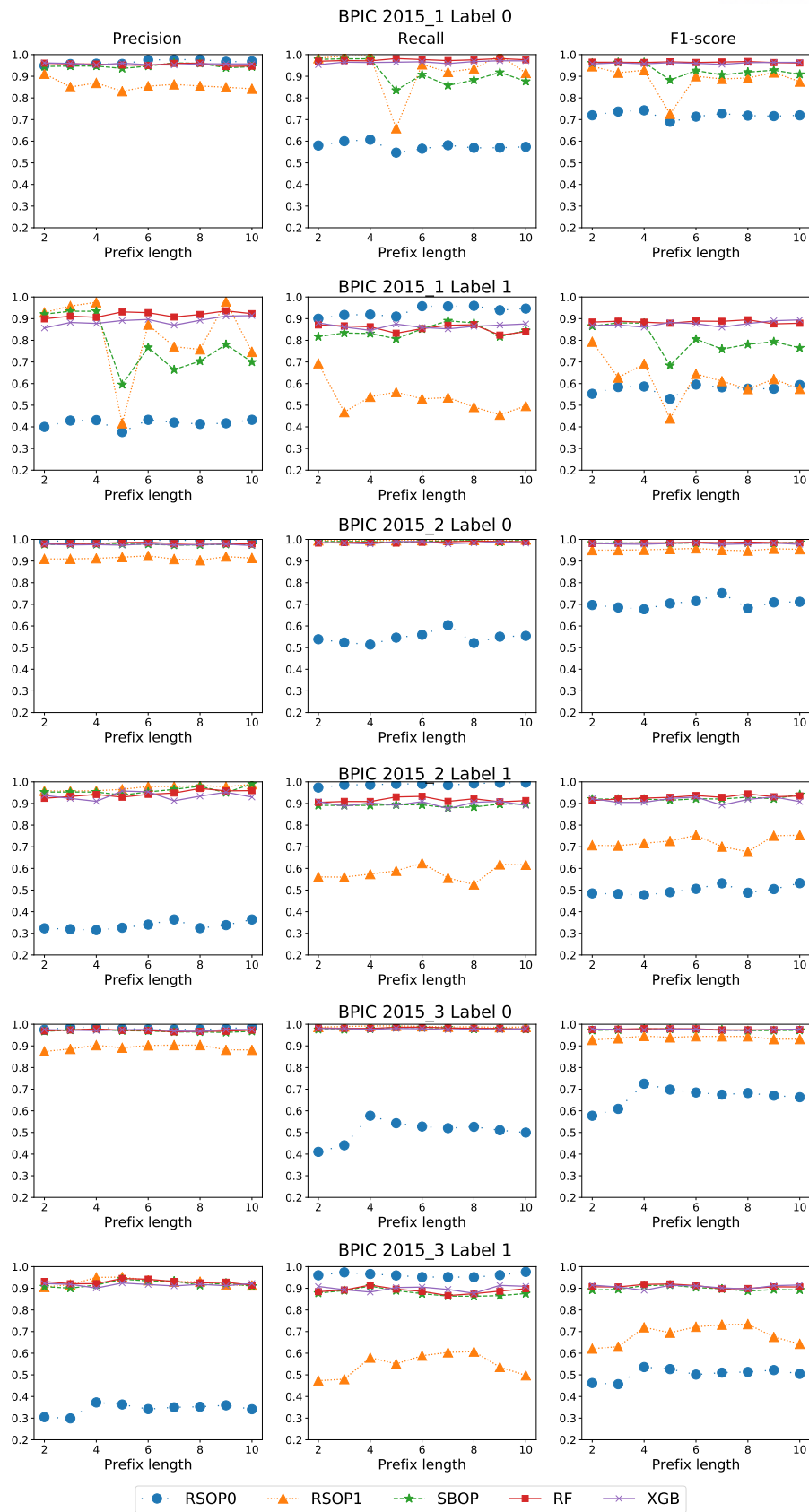


Figure 10: ARM-based and ML-based outcome classification performance - BPIC 2015 event logs

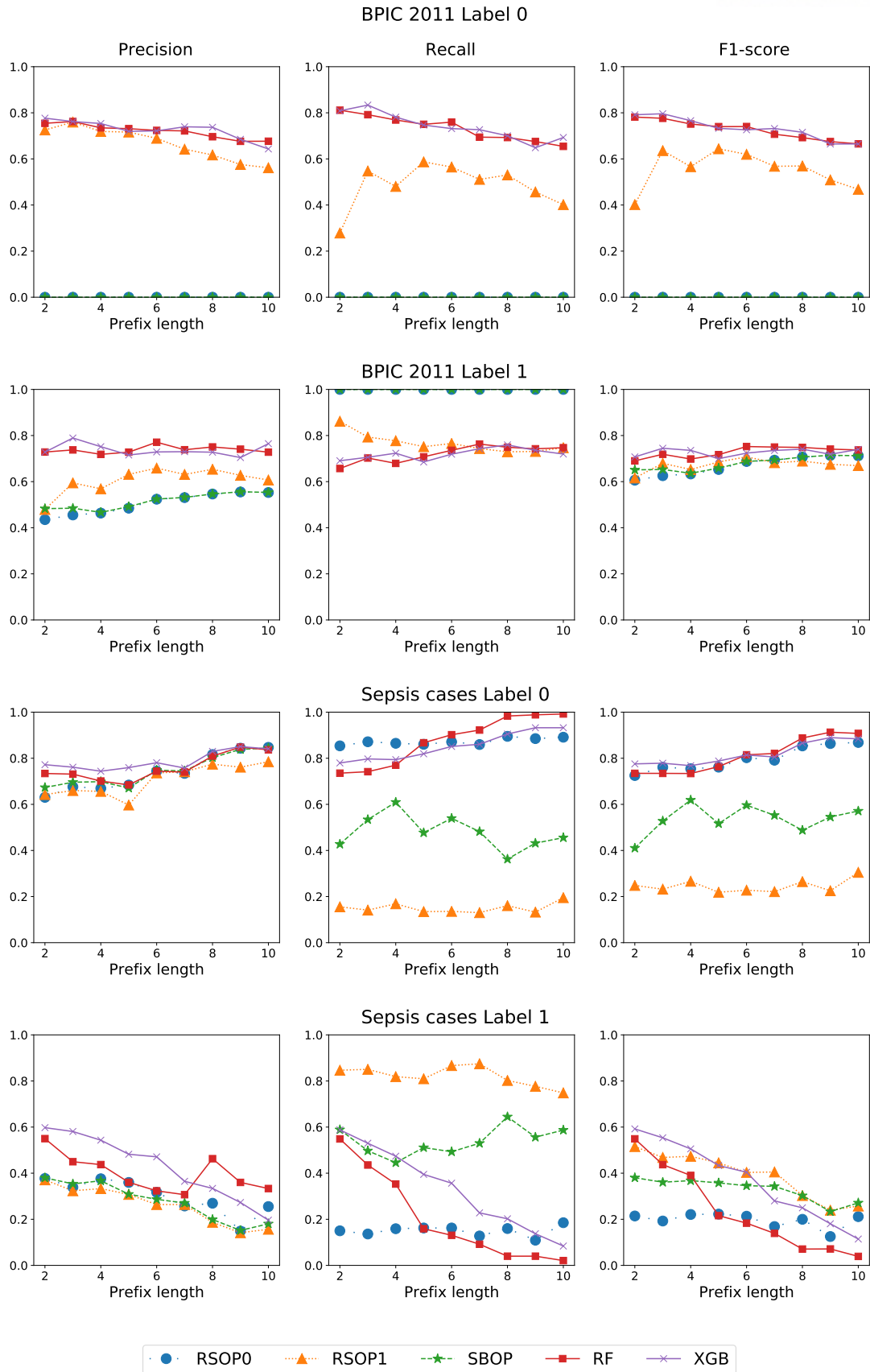


Figure 11: ARM-based and ML-based outcome classification performance - BPIC 2011 and Sepsis event logs

2. Even at early prefix lengths, at least one of the ARM-based label classification methods shows a performance level comparable with ML-based approaches;
3. The SBOP label classification method, when applicable, shows generally a better performance than the RSOP methods.

The last insight listed above focuses on the different level of performance achieved by ARM-based label classification methods. In general, it is not possible to elect a best-performing method. For the BPIC 2011 event log, the performance of the two methods is comparable for label 1, but for label 0 only RSOP can be applied, because the number of rules found for label 0 is always 0. In other words, it is only possible to classify as label 1 those cases that do not satisfy any of the rules extracted for label 1. For the BPIC 2015 event logs, SBOP outcores RSOP0 and RSOP1 at all prefixes for both labels, showing a remarkable performance at early prefixes. For Sepsis (label 0), RSOP0 appears to outscore SBOP at all prefixes, with a performance level aligned to those of ML-based approaches at all prefixes.

4.3 Visualising association rules

4.3.1 Predictive model

The last step in our evaluation is to propose an approach for visualising the rules used for outcome prediction in a more interpretable way. While association rules are inherently interpretable and a basic interpretation of rules can be operated by skimming a printed list of rules, having a low number of meaningful rules that can be visualised effectively can dramatically increase their interpretability by human decision makers [32]. In the proposed framework, the low number of meaningful rules is guaranteed by the rule pruning discussed earlier, whereas an effective technique for rule visualisation, based on network graphs, is presented next.

The visualisation of rules that we propose is based on the following principles:

- Two network graphs are built for a specific event log at a given prefix length, one for the rules predicting label 0 ($\mathcal{R}^{0,x,k}$) and one for the rules predicting label 1 ($\mathcal{R}^{1,x,k}$);
- Owing to the fact that each graph represents all the rules for a given label, each graph is built only with information related to the antecedents X of the rules; In particular, each node is an item that appears in at least one rule and edges connect two items (nodes) that appear together in the antecedent of a rule;
- The thickness of nodes represents the frequency of appearance of an item in rules relative the total number of items in the rules;
- Four types of items are identified and different colours are assigned to the corresponding nodes in a graph: items derived from an *activity* label, from *timestamps*, from other *case-level attributes*, or from other *event-level attributes*;

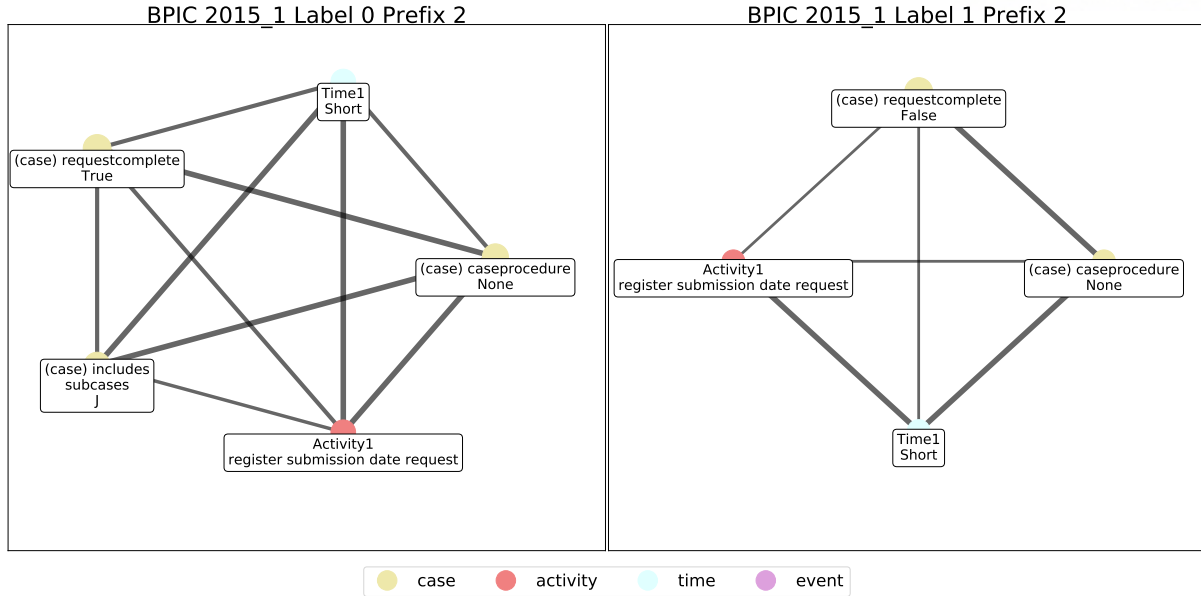


Figure 12: BPIC 2015_1 event log network graph for different labels

- The thickness of edges represents the relative frequency at which the two nodes linked by an edge appear together in one rule, relative to the total number of rules.

A first straightforward way to interpret the rules obtained is to compare the graphs obtained for label 0 and label 1, respectively, i.e., rules in $\mathcal{R}^{0,x,k}$ and $\mathcal{R}^{1,x,k}$, for a given event log at a given prefix length. A different analysis involves comparing graphs obtained for a given event log at different prefix lengths. The former allows to identify which items are more relevant in determining a certain type of outcome. The latter allows to identify, for a given event log, the *importance* of items in determining a certain outcome as more information about the execution of a process cases becomes available.

Figure 12 shows an example of the first approach, comparing the graphs obtained for label 0 and label 1 for the BPIC 2015_1 event log with prefix length equal to 2. Note that items derived from event-level attributes in this graph are named as *attribute_value*. Every item in an item matrix, in fact, is obtained from a specific value $v_{i,j}$ of a given attribute d_i in an event log. The name of items derived from case-level attributes is prefixed by *(case)*.

It looks clear that the case-level attribute *(case) includes subcases_J* is a good predictor of label 0 outcome, since it appears in the graph for label 0, while it is not present in the one for label 1. The graphs also signal that the value of the case attribute '*(case) requestcomplete*' is important to predict the outcome, since an item associated with the label 0 outcome is the one in which this attribute has value True, while the label 1 outcome is associated with the value False of this attribute. As far as edges in the graph are concerned, it is possible to notice that for label 0 all the item combinations seem to share a similar relative frequency in the rules, since all edges have similar thickness. On the contrary, edges' thickness in the label 1 graph is notably different. For instance, the edge connecting *(case) requestcomplete_False* and *(case) caseprocedure_None*

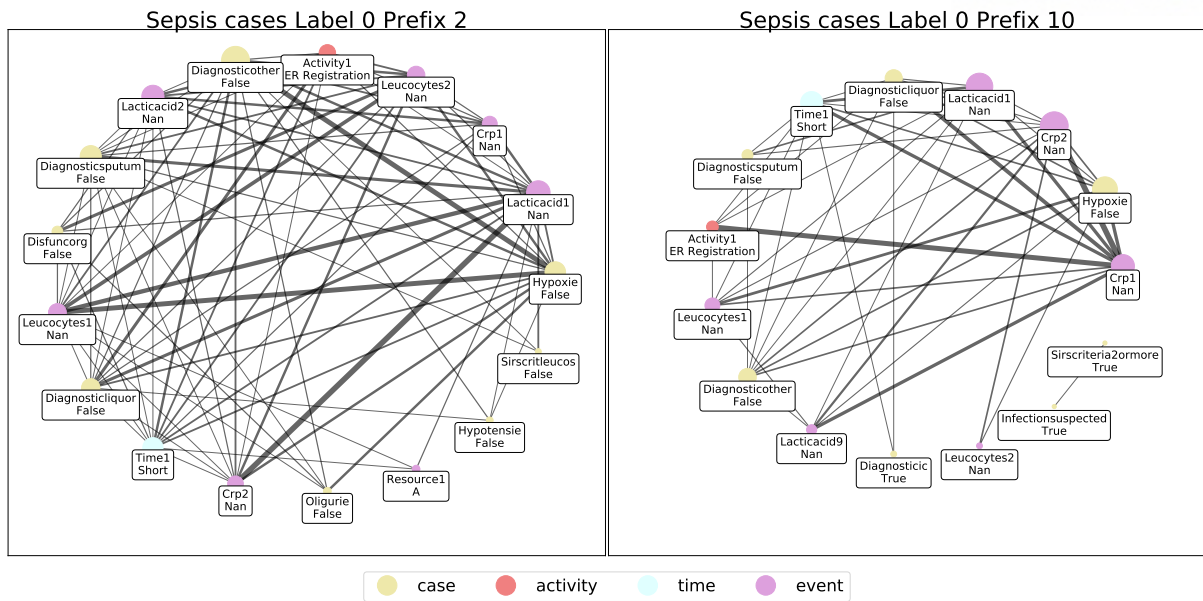


Figure 13: Sepsis event log network graph (label 0) for different prefix lengths

has heavier thickness than others, pointing at a higher support of prediction rules involving these 2 items in the antecedent.

Figure 13 shows an example of graphs built for the Sepsis event log, label 0, at two different prefix lengths (2 and 10). First, it has to be noticed that outcome prediction using the Sepsis log is more challenging than for the BPIC2015_1 logs. In the proposed ARM-based model, this translates into a higher number of rules involving a higher number of items in the case of the Sepsis log. To this end, note that the graphs built for the Sepsis log are structurally more complex than the ones of the BPIC2015_1 log.

At the level of items (nodes in the graph), we can see that 12 items appear in both graphs. These can be seen as the more robust items for outcome prediction, since they remain relevant as more events in a case are executed. It also interesting to note that the importance of nodes varies by prefix length, as highlighted by the size of the nodes in the graph. For example, the size of item *Crp1_Nan* increases at prefix 10, i.e., this item appears more frequently among all rules at longer prefix length. At the level of edges in the graph, it can be noticed that connections between nodes become less as the prefix length increases, but, at the same time, the longer prefix length graph has fewer thick edges. This signals that, as the prefix length increases, the combination of items determining the antecedents of rules becomes less. Yet, however, all combinations show a relatively low frequency, i.e., there are no clear-cut groups of items more likely to appear in the rules to predict label 0. Finally, it is worth noticing that, particularly for prefix length 10, the item *Crp1_Nan* appears to be central in the graph, since it is connected to many other nodes by thick edges. This signals that this item appears frequently in the rules and it is therefore a good predictor of the label 0 outcome.

4.3.2 Case prediction

Section 4.3.1 introduces association rules and one visualization method which has network graph form with rule itemsets and connection frequency as nodes and edges. With this method, decision maker could prospect relations between items in rule antecedents section regarding to predictive monitoring in view of holistic association rules. On the other hand this section presents single process instance and particular satisfied rule visualization method through emphasizing case and event level attributes in process instance. Figure 14 and Figure 15 shows visualization of case prediction which has process outcome as label 1 and label 0 respectively.

Process instance and rule visualization consists of 4 main sections, 1) *case information*, 2) *trace with event level attributes*, 3) *case level attributes*, and 4) *association rules by label*. Case information segment shows case id, predicted outcome, and actual outcome of the case. Case ID is an identifier to indicate following trace images and information are belong to which process instance. The predicted outcome label derived from rule-based predictive monitoring model is mentioned in predicted outcome. Actual outcome shows outcome label which satisfies linear temporal logic constraints.

Trace section which starts with start node (top circle node) depicts series of events with activity and event level attributes. The first row of the event node is an activity and rest of the rows are attributes related with the event. The left column of the even node table is attribute names and the right column is the value of the attribute. For example, in figure 14, which has 9431949 as case id, first event has 3 event level attributes, 'Resource', 'Time', and 'monitoringResource' and values of the attributes are 560872, Short, and 560872 respectively.

The next section is about case level attributes table, i.e., attributes regarding case information not limited to an event. Like in representing event level attributes in event node, first column describes the name of the case attributes and the other column shows according value of it. In case of example case, 9431949, there are 9 case level attributes from '(case) includessubcases' to '(case) SUMlegs'.

Regarding last section of visualization method, association rules by label of the case are listed at the right bottom of the image. After intra-label pruning process, 20 representative rules are selected to build predictive monitoring model. The association rule antecedents consist the rule list which has equal consequent. Association rule list could also be changed by different prefix length.

After getting association rules for predictive monitoring and classification process, there are rules which comply with information about given case. Case and event level attributes in accordance with satisfied association rules are colored in blue. Association rules in rule list section which are observed in the case are colored in yellow. By coloring cells process instance visualization method emphasize which items and attributes are critical to predict outcome. With this visualization method inspectors could anticipate predictive monitoring model explainability through scanning colored attributes in the case and the association rules in the image.

Case Information

Case ID: 9431949 Predicted Outcome: 1 Actual Outcome: 1

register submission date request	
Resource	560872
Time	Short
monitoringResource	560872

Case level attributes	
(case) includessubcases	None
(case) responsibleactor	560464.0
(case) caseprocedure	Uitgebreid
(case) casestatus	O
(case) lastphase	Besluit onherroepelijk
(case) parts	Bouw
(case) requestcomplete	False
(case) termname	None
(case) SUMleges	Large

term 14 or 26 weeks	
Resource	560872
Time	Long
monitoringResource	2670601

first phase affected	
Resource	560872
Time	Long
monitoringResource	2670601

start phased application	
Resource	560872
Time	Long
monitoringResource	2670601

first or second phase	
Resource	560872
Time	Long
monitoringResource	2670601

read field phased application	
Resource	560872
monitoringResource	2670601

Label 1 association rules	
Time2_Long	
(case) caseprocedure_None, Activity1_register submission date request	
(case) requestcomplete_False, Time1_Short	
Activity1_register submission date request, Time1_Short	
Activity1_register submission date request, Time5_Long	
Activity1_register submission date request	
Support, Time1_Short	
(case) requestcomplete_False	
(case) caseprocedure_None	
Time2_Long, Time4_Short	
(case) requestcomplete_False, Time2_Long	
Time1_Short, Time2_Long	
Activity1_register submission date request, Time2_Long	
Time4_Short	
Time1_Short, Time5_Long	
Time1_Short, Time4_Short	
Time2_Long, Time5_Long	
Time5_Long	
Activity1_register submission date request, Time4_Short	
(case) requestcomplete_False, Activity1_register submission date request	

Figure 14: BPIC 2015 event log process instance and rule visualization: Case outcome label 1

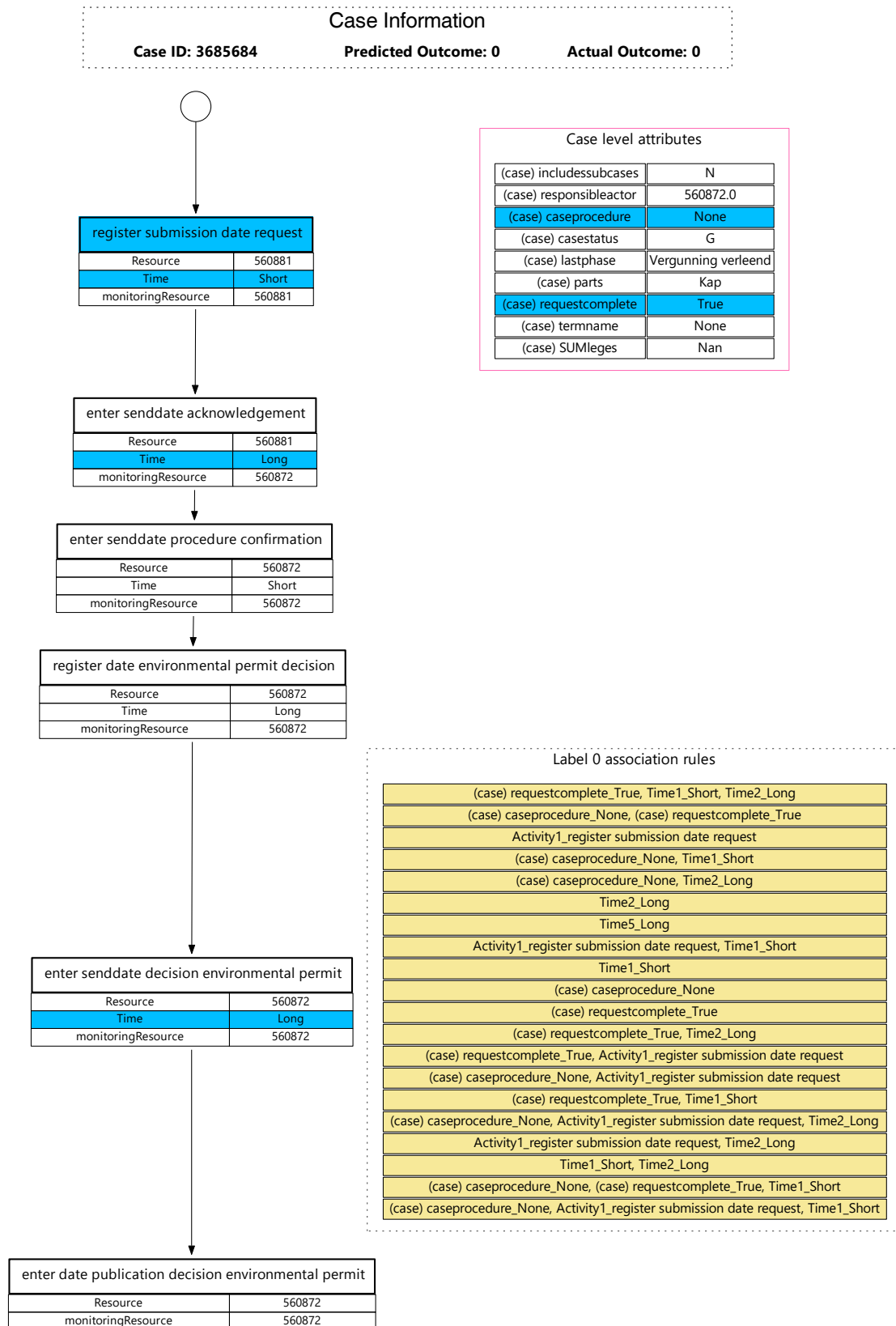


Figure 15: BPIC 2015 event log process instance and rule visualization: Case outcome label 0

Figure 14 and Figure 15 show two process instances with different outcome label from BPIC 2015 event log. These cases are already pre-processed with filtered by prefix length 6. There are 3 aspects to interpret the process instance and outcome relation with association rules. The first possible approach is intra-case level inspection, i.e., selecting colored case and event level attributes. For example, the first figure, 9431949 as case id, has 4 event level attributes and 1 case level attribute in blue color and these are the emphasized attributes.

Case ID: 9431949

- Event level attributes
 - Event 1 Activity: register submission date request
 - Event 1 Time: Short
 - Event 2 Time: Long
 - Event 5 Time: Long
- Case level attributes
 - (case) requestcomplete: False

Decision makers are able to catch the important attributes for outcome prediction by scanning single time and know which rules are more useful to explain itemsets and outcome relations.

The second approach is inter-case level interpretation. The second figure, 3685684 as case id, has actual and predicted outcome label 0. And followings are highlighted attributes.

Case ID: 3685684

- Event level attributes
 - Event 1 Activity: register submission date request
 - Event 1 Time: Short
 - Event 2 Time: Long
 - Event 5 Time: Long
- Case level attributes
 - (case) requestcomplete: True
 - (case) caseprocedure: None

The event level attributes are commonly appeared in both first and second figures which have different outcome label as 1 and 0 respectively. On the other hand, the value of case level attribute, '(case) requestcomplete' is different. In case 9431949, which outcome label is 1, has False value for the attribute. However, the value of the same attribute is True in outcome label

0 case, 3685684. As well as divergent value at the equal attribute, the other case level attribute is found. Through comparison between process instances with different outcome label, decision makers may recognize turning point which leads to case outcome.

The last approach of interpretation from process instance and rule satisfaction is analysis on attributes importance of prefix variance under same case. Figure 16 shows same case's trace and case information of figure 14 but different prefix length. This figure represents the attributes and satisfied rules when the case prefix length is 3. Although case level attributes are served since the start of the case, new event level attributes are possible to be colored after case progressed. For example, case id 9431949 in prefix length 6 shows additional colored event attributes compare to prefix length 3 trace. 'Time' attribute of 5th event is colored. By connecting newly observed attributes and satisfied rules, decision makers are able to figure decisive items which appear from early prefix length or outcome divergence point as the process continues.

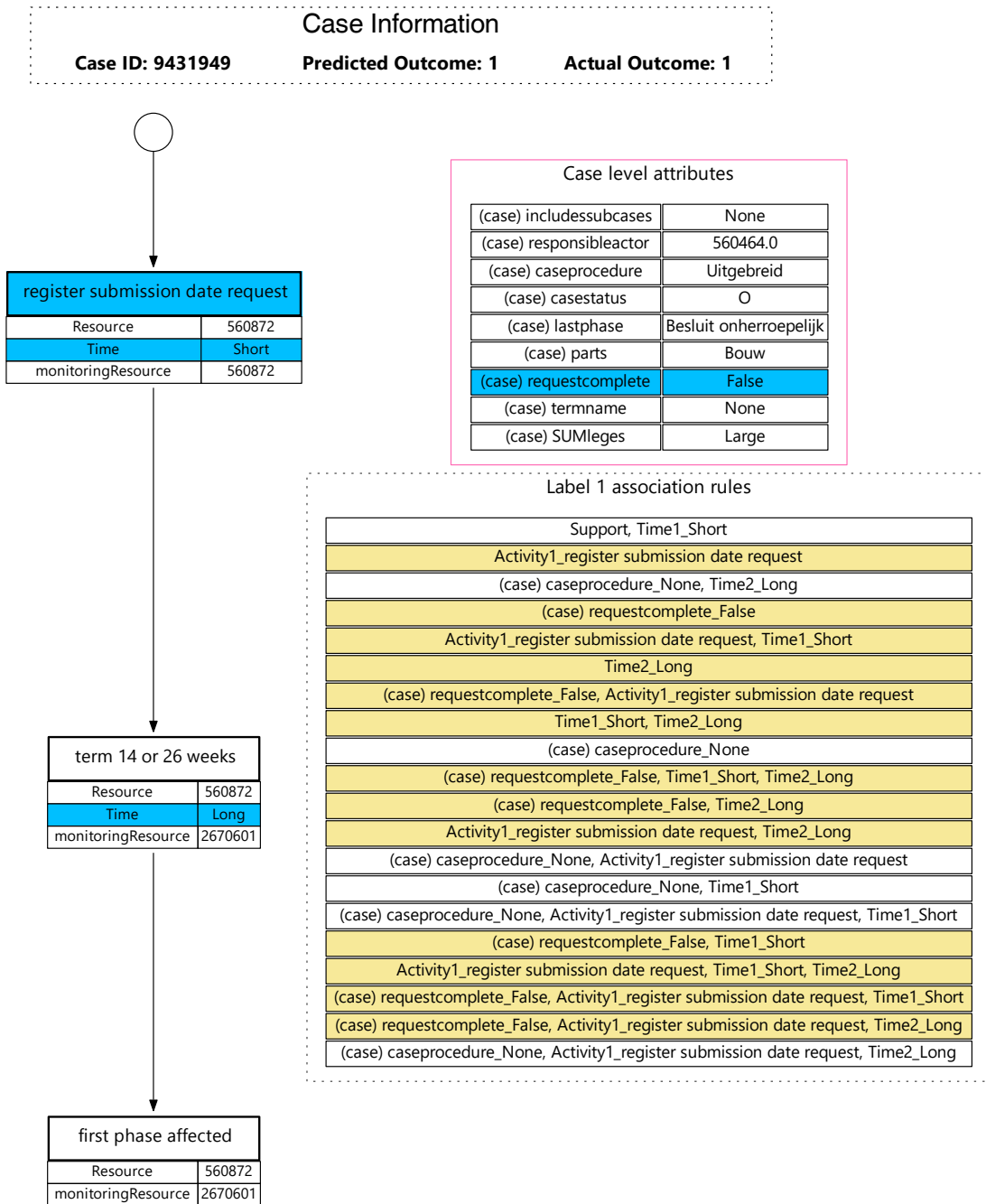


Figure 16: Process instance and rule visualization with different prefix length

V Conclusion

This thesis has introduced an interpretable predictive process outcome monitoring framework to understand classification logic inside of the predictive model and in-depth analysis with visualization method. The framework starts from the pre-processing stage on how to modify event log attributes through continuous attributes discretization and one hot-encoding. After predictive model creation, two visualization methods are presented for model interpretability: predictive model and predicted case outcome prediction. The first visualization method delivers inter-feature relation in frequency measurement from obtained association rules. Directed follow graph of trace visualization proposes analysis on case outcome prediction with satisfied association rules along with case and event-level attributes of the particular case.

The proposed framework has been evaluated, all other conditions being equal, against predictive models built using extreme gradient boosting machine and random forest. The performance obtained by the proposed framework is in most cases only slightly lower than the one of machine learning-based classifiers. However, a slight decrease in performance can be acceptable in return for a model that is interpretable by human decision makers. Regarding the rule pruning process to limit the number of rules, the predictive model with summarized association rules has slightly lower performance compared to the model that considers all the rules. This peculiar feature of the rule pruning process is fundamental to create visualized graphs and lead user comprehension level by deleting redundant and low-impact rules.

The suggested visualization method has been designed to facilitate analysis of the rule-based predictive model in a comprehensive and detailed specific view by providing association rule network graph and dependency graph by certain case respectively. The rule network graph offers a holistic point of analysis on association rules for the predictive model with distinctive itemsets and feature connections. Distinguishable node and edge attributes in the rule network graph differ itemsets importance on predicting outcome label and emphasize prominent model changes across prefix length variation. Predicted case explanation in dependency graph depicts case and event-level attributes as well as label-rule list. Features related to the outcome label are highlighted in different colors in both attributes and label-rule list. Users are expected to look in the specific case and obtain knowledge of each label or various prefix lengths from both visualization methods.

Future work will pursue multiple perspectives, in terms of both application and extension. First, we plan to apply the proposed framework to the more general use case of classification. That is, we intend to use our framework with data different than event logs, studying in particular to what extent the type of phenomenon that the data represent impacts the applicability and performance of our framework. Regarding extension, we are planning to develop different visualization techniques and to evaluate them concerning usefulness and ease of use with pseudo-experts decision makers, such as university students of BPM subjects. To extend the prospect of the visualization section in the framework, in particular concerning contextual variables which

include inter case features or external information out of an event log, developing a comprehensive process visualization method is the other direction of future work. Additional features such as inter-case or resource-aware features within the event log are anticipated to enhance model performance and understating the correlation between attributes. Besides, contextual variables are expected to provide insight to users beyond the limited amount of data from a single event log.

References

- [1] I. Verenich, M. Dumas, M. L. Rosa, F. M. Maggi, and I. Teinemaa, “Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 4, pp. 1–34, 2019.
- [2] M. Dumas, M. L. Rosa, J. Mendling, and H. A. Reijers, “Fundamentals of business process management,” 2018.
- [3] V. market research. Global process analytics market size by process mining type, by application, by geographic scope and forecast. [Online]. Available: <https://www.verifiedmarketresearch.com/product/process-analytics-market/>
- [4] F. M. Maggi, C. Di Francescomarino, M. Dumas, and C. Ghidini, “Predictive monitoring of business processes,” in *International conference on advanced information systems engineering*. Springer, 2014, pp. 457–472.
- [5] J. Evermann, J.-R. Rehse, and P. Fettke, “A deep learning approach for predicting process behaviour at runtime,” in *International Conference on Business Process Management*. Springer, 2016, pp. 327–338.
- [6] A. Rogge-Solti and M. Weske, “Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays,” in *International conference on service-oriented computing*. Springer, 2013, pp. 389–403.
- [7] M. Du, N. Liu, and X. Hu, “Techniques for interpretable machine learning,” *Communications of the ACM*, vol. 63, no. 1, pp. 68–77, 2019.
- [8] Q. Zhang, Y. Nian Wu, and S.-C. Zhu, “Interpretable convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8827–8836.
- [9] M. Harl, S. Weinzierl, M. Stierle, and M. Matzner, “Explainable predictive business process monitoring using gated graph neural networks,” *Journal of Decision Systems*, pp. 1–16, 2020.
- [10] R. Galanti, B. Coma-Puig, M. de Leoni, J. Carmona, and N. Navarin, “Explainable predictive process monitoring,” *International Conference on Process Mining 2020*, 2020.

- [11] B. Liu, W. Hsu, Y. Ma *et al.*, “Integrating classification and association rule mining.” in *KDD’98: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, vol. 98, 1998, pp. 80–86.
- [12] M. Weske, “Business process management architectures,” in *Business Process Management*. Springer, 2012, pp. 333–371.
- [13] W. M. Van der Aalst, “Business process management: a comprehensive survey,” *International Scholarly Research Notices*, vol. 2013, 2013.
- [14] W. Van der Aalst, T. Weijters, and L. Maruster, “Workflow mining: Discovering process models from event logs,” *IEEE transactions on knowledge and data engineering*, vol. 16, no. 9, pp. 1128–1142, 2004.
- [15] W. Van der Aalst, A. Adriansyah, and B. van Dongen, “Replaying history on process models for conformance checking and performance analysis,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 182–192, 2012.
- [16] I. Teinemaa, M. Dumas, M. L. Rosa, and F. M. Maggi, “Outcome-oriented predictive process monitoring: Review and benchmark,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 2, pp. 1–57, 2019.
- [17] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, “Predictive business process monitoring with lstm neural networks,” in *International Conference on Advanced Information Systems Engineering*. Springer, 2017, pp. 477–492.
- [18] C. Di Francescomarino, M. Dumas, F. M. Maggi, and I. Teinemaa, “Clustering-based predictive process monitoring,” *IEEE transactions on services computing*, 2016.
- [19] A. Senderovich, C. Di Francescomarino, C. Ghidini, K. Jorbina, and F. M. Maggi, “Intra and inter-case features in predictive process monitoring: A tale of two dimensions,” in *International Conference on Business Process Management*. Springer, 2017, pp. 306–323.
- [20] J. Wang, D. Yu, C. Liu, and X. Sun, “Outcome-oriented predictive process monitoring with attention-based bidirectional lstm neural networks,” in *2019 IEEE International Conference on Web Services (ICWS)*. IEEE, 2019, pp. 360–367.
- [21] J. Brunk, M. Stierle, L. Papke, K. Revoredo, M. Matzner, and J. Becker, “Cause vs. effect in context-sensitive prediction of business process instances,” *arXiv preprint arXiv:2007.07549*, 2020.
- [22] R. Agrawal, T. Imieliński, and A. Swami, “Mining association rules between sets of items in large databases,” in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, 1993, pp. 207–216.

- [23] Y. Djenouri, A. Belhadi, and P. Fournier-Viger, “Extracting useful knowledge from event logs: a frequent itemset mining approach,” *Knowledge-Based Systems*, vol. 139, pp. 132–148, 2018.
- [24] F. M. Maggi, R. J. C. Bose, and W. M. van der Aalst, “Efficient discovery of understandable declarative process models from event logs,” in *International Conference on Advanced Information Systems Engineering*. Springer, 2012, pp. 270–285.
- [25] K. Böhmer and S. Rinderle-Ma, “Mining association rules for anomaly detection in dynamic process runtime behavior and explaining the root cause to users,” *Information Systems*, vol. 90, 2020.
- [26] E. L. Mencía, J. Fürnkranz, E. Hüllermeier, and M. Rapp, “Learning interpretable rules for multi-label classification,” in *Explainable and Interpretable Models in Computer Vision and Machine Learning*. Springer, 2018, pp. 81–113.
- [27] Y. Ji, H. Ying, J. Tran, P. Dews, A. Mansour, and R. M. Massanari, “A method for mining infrequent causal associations and its application in finding adverse drug reaction signal pairs,” *IEEE transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 721–733, 2012.
- [28] G. Attanasio, L. Cagliero, and E. Baralis, “Leveraging the explainability of associative classifiers to support quantitative stock trading,” in *Proceedings of the Sixth International Workshop on Data Science for Macro-Modeling*, 2020, pp. 1–6.
- [29] M. De Leoni, W. M. van der Aalst, and M. Dees, “A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs,” *Information Systems*, vol. 56, pp. 235–257, 2016.
- [30] A. Leontjeva, R. Conforti, C. Di Francescomarino, M. Dumas, and F. M. Maggi, “Complex symbolic sequence encodings for predictive monitoring of business processes,” in *International Conference on Business Process Management*. Springer, 2016, pp. 297–313.
- [31] R. Agrawal, R. Srikant *et al.*, “Fast algorithms for mining association rules,” in *Proc. 20th int. conf. very large data bases, VLDB*, vol. 1215, 1994, pp. 487–499.
- [32] Y. Djenouri, H. Drias, and A. Bendjoudi, “Pruning irrelevant association rules using knowledge mining,” *International Journal of Business Intelligence and Data Mining*, vol. 9, no. 2, pp. 112–144, 2014.
- [33] G. Hamerly and C. Elkan, “Alternatives to the k-means algorithm that find better clusterings,” in *Proceedings of the eleventh international conference on Information and knowledge management*, 2002, pp. 600–607.

- [34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, “Scikit-learn: Machine learning in python,” *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

