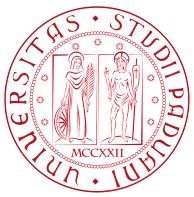

University of Padua

Department of Mathematics Department "Tullio Levi-Civita"

Master Thesis in Data Science

Prescriptive Process Analytics using Counterfacts

Supervisor
Prof. Massimiliano de Leoni
University of Padua

Co-supervisor

Master Candidate
Mohammad Ismail Tirmizi

Academic Year
2020-2021

In deep appreciation of God and for all the mentors I have encountered dur- ing my jour- ney; directly or indirectly materialized as family, friends, pro- fessors, or colleagues. Thanks for blessing me with the gift of knowledge and the mo- tivation to pursue it.

Abstract

In the age of technology integration and big data, the capability to extract value from abundant data has become paramount. Business Process Management (BPM), a field focused on the management of complex business processes, gives rise to a vast amount of event data. The subfield of "process mining" harnesses this data, acting as a conduit between BPM and data mining. Prescriptive analytics are a technique that uses data science techniques to provide actionable steps to improve a running process instance. This thesis delves into the realm of prescriptive process analytics, highlighting the use of counterfactuals. Building upon established predictive frameworks, it seeks to develop a domain-agnostic prescriptive analysis mechanism. The primary aim is to generate recommendations that not only suggest the next-best activity,
but also pinpoint the optimal resource to undertake it, all in a bid to optimize a predefined Key Performance Indicator (KPI). By comparing the efficacy of our proposed framework with existing methodologies on real-world datasets, we aim to underscore the significance and potential of our approach. The research unfolds through multiple chapters that elucidate foundational principles, state-of-the-art methods, our unique framework, and its evaluation through two distinct case studies. The hope is to chart a course for future endeavors in the domain, cementing the importance of prescriptive process analytics and its transformative impact on the business landscape.

Contents

Listing of figures

Listing of tables

Listing of acronyms

ML . . . . . Machine Learning

BPM . . . . . Business Process Management

PAR . . . . . Process Aware Recommender

RAR . . . . . Resource Aware Recommendation

BAC . . . . . Bank Account Closure

CFEs . . . . . Counterfactual Examples

1 Introduction

1.1 Context and Topic

Process mining is a discipline of Data Science that focuses on the analysis and improvement of business processes through the use of event log data. At its core, process mining aims to extract valuable insights from the event data generated by information systems. This discipline bridges the gap between traditional business process management (BPM) and data science, leveraging techniques from both fields to provide a comprehensive understanding of how business processes actually perform in real-world scenarios.

To provide goods and services of higher quality and at a lower price to their customers, companies are always looking to optimize their operations (business process). To accomplish this process mining has emerged as a promising technique over the last decade.

In the realm of process mining, prescriptive analytics plays a critical role. Unlike descriptive analytics, which focuses on what has happened and predictive analytics, which concentrates on what is likely to happen, prescriptive analytics goes a step further.

Descriptive analytics in process mining involves the examination and interpretation of historical data to understand how business processes have performed over time. This aspect of process mining focuses on analyzing past events and process flows to provide insights into trends, patterns, and anomalies within the business process.

Predictive analytics in process mining extends beyond understanding past patterns to fore-

casting future process behaviors. It utilizes various statistical and machine learning techniques to analyze historical process data and predict how current process instances are likely to evolve. This can include predicting the next activities in a process, estimating completion times, or identifying potential risks and bottlenecks.

Prescriptive analytics uses insights from data to recommend actions that can optimize business processes. Prescriptive analytics in process mining involves suggesting improvements to a running process instance, thereby providing organizations with actionable strategies to enhance performance, eficiency, and productivity. This is particularly crucial in complex and dynamic business environments where process eficiency directly impacts organizational success.

Within the ambit of prescriptive analytics, this thesis introduces a novel approach that leverages techniques for counterfactual explanations. Counterfactuals provide a unique perspective by exploring ”what-if” scenarios, thus enabling a deeper understanding of the decision-making processes within models. By applying a counterfactual framework to process mining, this thesis aims to develop a sophisticated prescriptive analytics framework.

This framework will leverage not only counterfacts but also other business process mining techniques to ensure improved recommendations for process improvement that aligns well with real-world business contexts.

Therefore, this research extends the frontiers of process mining by integrating counterfactual explanations into prescriptive analytics, offering a novel technique that enhances the efficacy of process optimization strategies. This integration marks an advancement in process mining, paving the way for more informed and strategic decision-making in various business operations.

1.2 State of the Art

The field of process mining has traditionally centered on ”Descriptive Analytics” and ”Predictive Analytics,” but recently ”Prescriptive Analytics” is also gaining popularity in the literature. Like recommender systems exist for recommending items in traditional data mining applications, process mining has a version of its own called **”Process-aware Recommender system”** (PAR system). In conventional recommender systems, the objective is to recommend the next item based on user's specific history or general trend of consumer behaviors. For example, when shopping in an online store, if you add an item, say butter, to the shopping cart, the recommender system might also suggest that milk be added as well. PAR systems work similarly, but

with a process context, and cater to the process owners instead of the end users. PAR systems have the following components: descriptive analytics, predictive analytics, and prescriptive analytics.

The paper [3] introduces the PAR system and addresses the gap with respect to prescriptive analytics. The objective of this prescriptive analytics component is to improve the process by recommending the next best activity. The proposed prescriptive-analytics builds on top of a predictive-analytics module.

[1] emphasize the importance of the explainability of the recommendations produced by a PAR system. To explain the recommendation of the prescriptive component, it uses process-related characteristics such as the values of the process variables, activities performed and the resources involved. Chihchen et al. [4] uses a novel approach of "counterfactual explanations" to obtain explainable insights for process recommendations. They built on the work of [5] and modified their algorithm to work with sequential event data. Their approach, however, relies heavily on domain knowledge to implement, and it also requires domain experts to operate it.

Many of these works, such as the ones mentioned above and others [3], [6], [7] improve the process by recommending the next best activity only. While other works focus on suggesting which resource should perform specific activities in various contexts [8] [9]. [10], [11] and constraints. However, these works do not recommend both an ideal activity and a resource.

Notable, the work of Padella et al. [2] suggests a paired activity and resource to improve a process while considering resource availability and the resultant impact on global KPIs. This is particularly relevant for comparison with our research, which also proposes both activity and resource for improving a process.

However, as pointed out by [12] more work needs to be done on prescriptive analytics techniques and there is a need to explore different approaches to achieve this. Recently, counterfactuals have also gained popularity in the realm of process mining. However, they require a lot of domain knowledge to work properly. Also, no quantitative analysis is done on the recommendations produced by counterfactual approaches.

1.3 Research Question

The central objective of this study is to advance the field of process mining by exploring an alternative method to perform prescriptive analysis, an area that is not well explored. Since the field of process mining has traditionally been focused on descriptive and predictive analytics, we assume that we already have a robust setup of predictive analysis, which we will incorporate

in our technique. Therefore, our primary focus is the development of a domain-agnostic prescriptive analysis framework that can produce valid and effective recommendations to improve a business process.

The primary research question thus becomes: Given the established predictive analysis setup, how can we develop a prescriptive analysis framework that produces effective recommendations to improve a process, regardless of the domain? Specifically, the recommendation should include both the next-best activity and the next-best resource.

To define or quantify the improvement, we define **Key Performance Indicators (KPIs)**. These are metrics that the companies use to monitor progress towards achieving their strategic and operational goals, hence forming an essential component of the process performance management framework. The challenge lies in determining how to improve this KPI value. Therefore, the focus of our study is to improve processes by recommending the next optimal step that improves a given KPI.

To evaluate the effectiveness of the recommendations produced by our prescriptive analysis, we plan to conduct a quantitative analysis. Utilizing real-life datasets (event logs), a test set will be separated and used for the evaluation of the results. Given the absence of a standard method to evaluate counterfactuals [13], we will compare our results with the algorithm proposed by Padella et al. [2]. We intend to use another predictive model to estimate the new KPI values that come about as a result of following the recommendations; thereby serving as a proxy for the performance of the recommendations generated by both methods.

The sub-research questions derived from our main research question include:

- How can we design a prescriptive analysis framework that is domain agnostic and suggests both the next-best activity and resource to optimize a given KPI?
- How does our proposed prescriptive analysis framework perform compared to existing methods such as the one proposed by Padella et al. [2], when evaluated on real-life datasets?

Our method is successful if we can demonstrate a significant improvement in the KPI values in the two real-life case studies examined in the thesis.

1.4 Break Down of Sections

The thesis is organized as follows:

- Chapter 2 Preliminaries: This chapter provides some important definitions and the mathematical framework that is required to explain in depth the concepts presented here.
- Chapter 3 Related Works: This chapter presents some state-of-the-art techniques that are used to improve processes through different methods, including prescriptive analytics.
- Chapter 4 Framework for Generating Recommendations: This chapter explains how the counterfactual function is incorporated into the prescriptive analytics framework, resulting in valid recommendations.
- Chapter 5 Case Study VINST: This chapter talks about the first case study that we used to evaluate our framework. In this case study, we try to improve the total execution time of a process.
- Chapter 6 Case Study BAC: This chapter talks about the second case study that we used to evaluate our framework. In this case study, we try to avoid undesirable activity in the process.

2

Preliminaries

2.1 Process Mining

Process mining is a family of techniques relating to the fields of data science and process management that focuses on the discovery, monitoring, and improving of real-life processes by extracting knowledge from event data that are produced by these processes and stored in various systems. Classical data mining techniques such as association rule learning, sequence/episode mining, regression, clustering, and classification are only able to analyze a part of the entire business process. Process mining focuses on a business process from start to finish. [14]

Recent breakthroughs in the field have made it possible to discover, analyze, and improve business processes based on event data. Activities executed by people, machines, and software leave a trail in the so-called **event logs**. These millions of events (placing order for a car, a passenger checking in for a flight, or a customer submitting loan application) are recorded by ever-advancing information systems. Also, due to ever-improving software capabilities, it has become more convenient to manage and utilize these event data. Therefore, business processes should be managed based on event data rather than subjective judgments of humans, as the application of process mining in hundreds of organizations around the world has shown that managers and users alike tend to overestimate their knowledge of their own processes and end up making suboptimal decisions. [15]

Process mining can thus be viewed as X-rays that reveal what really goes on inside the pro-

cesses, and it can be used to diagnose problems and suggest proper treatment. The practical relevance of process mining and related interesting scientific challenges make process mining a hot topic in business process management (BPM). [16]

Business analytics related to process mining is categorized into three main stages characterized by different levels of dificulty, value, and intelligence Akerkar et al. [17] and Katerina et al. [18]:

1. "Descriptive analytics", answering the questions "What has happened?", "Why did it happen?", but also "What is happening now?" (mainly in a streaming context)
2. "Predictive analytics", answering the questions "What will happen?" and "Why will it happen?" in the future
3. "Prescriptive analytics", answering the questions "What should I do?" and "Why should I do it?"

Currently, the vast majority of business analytics efforts are spent on descriptive analytics and predictive analytics with typical methodologies including data mining, machine learning, artificial intelligence, and simulation. [19] Recently, however, prescriptive analytics has been increasingly gathering research interest [20]. Furthermore, prescriptive analytics is also being seen as a very important next step in increasing data analytics maturity to optimize decision making, ahead of time, for improving business performance [21]. Hence, the motivation for this thesis.

2.2 Prescriptive Analysis

Prescriptive analytics, the cornerstone of this thesis, is the most sophisticated type of business analytics and can bring the greatest intelligence and value to businesses. Using large amounts of data, the aim is to suggest (prescribe) the best decision options to take advantage of the future predicted [22]. To do this, it incorporates the predictive analytics output and uses artificial intelligence, optimization algorithms, and expert systems in a probabilistic context to provide adaptive, automated, constrained, time-dependent, and optimal decisions [23].

The interplay of process mining within the broader context of related fields is succinctly captured in Figure 2.1. This Venn diagram articulates process mining's unique position as an intermediary, bridging the gap between Business Process Management and data-driven methodologies such as data mining and machine learning. Within this intersection, prescriptive analytics

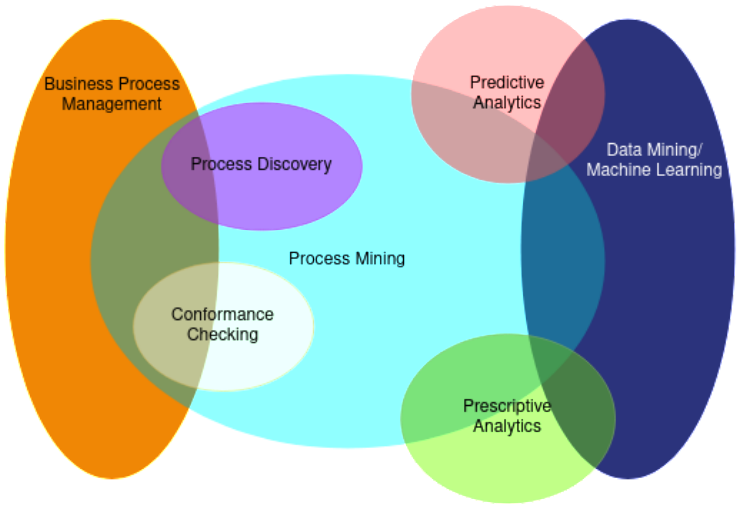

Figure 2.1: Venn Dtiagram showting how tthe prescriptttive analytttics fftitt tintto tthe btigger ptictture off process mtinting. In tthe dtiagram tthe orange oval representts tthe fftield off BPM, tthe navy blue oval representts tthe fftield datta mtinting and machtine learnting, and tthe ltightt blue oval representts process mtinting whtich tis also overlappting wtitth botth fftields. The green ctircle tin tthe dtiagram representts tthe prescrtiptttive analytttics. The red ctircle representts tthe predtictttive analytttics.

emerges as a transformative force in business analytics, leveraging the insights and methodologies of data mining and machine learning to inform and guide business process decisions. Process mining thus acts as a vital link, integrating the descriptive and predictive strengths of these domains to enhance prescriptive analytics. This integration allows businesses to make complex operational decisions with greater confidence, supported by data.

2.3 Process Mining Notation Primer

The starting point for any process mining task is an *event log*. The building blocks for an event log is an *event*, which represents the execution on an *activity* in a *trace*. An activity in this context is a well-defined step in the business process, usually performed by a device or a person. To generalize this we call the executor or initiator of this business activity a *resource*. A trace is just a sequence of events that describes the life-cycle of a particular *process instance* (i.e., a *case*). This makes an event log a multiset of sequences of traces.

Events belonging to a trace (or case) are ordered and can be seen as one ”run” of the process. Event logs also store additional *attributes* about events such as the resource and the timestamp of the event or data elements. [14] An attribute can be given a value □indicating uncertainty of whether or not a value was assigned to an attribute by an event and what this value was.

We will also use mathematical notation to present the theoretical workings of the proposed approach. The following are the definitions and concepts required to understand the findings of this thesis.

Definition 1 (Events): Let A be the set of process activities. Let R be the set of process resources. Let V be the set of process attributes. Let W_V be a function that assigns a domain $W_V(b)$ to each process attribute $b \in V$. Let $\overline{W} = \bigcup_{b \in V} W_V(b)$. An event is a tuple $(a, r, v) \in A \times R \times (V \rightarrow \overline{W})$ where a is the event activity, r is the event resource, and v is a partial function assigning values to process attributes, with $v(b) \in W_V(b)$.

To understand the definition we can take an example where we instantiate the different variables and also elaborate some of the maths involved.

In the expression $\overline{W} = \bigcup_{b \in V} W_V(b)$, $\overline{W}$ is equal to the union over all b belonging to the set V of the domains $W_V(b)$. The expression $\overline{W}$ represents the resulting set that contains all possible values from the domains of the process attributes in V, capturing the complete range of possible values across these attributes. For example if W_V is as follows:

- W_V(Status) = Accepted, Queued, Completed
- W_V(Product) = ProdA, ProdB, ProdC
- W_V(Country) = Spain, Italy, Belgium

The union of all the domains $W_V(b)$, where b belongs to V, would result in:

$$\overline{W} = \{\text{Accepted, Completed}\} \cup \{\text{ProdA, ProdB, ProdC}\} \cup \{\text{Spain, Italy}\}$$

Evaluating the union operation, we get:

$$\overline{W} = \{\text{Accepted, Completed, ProdA, ProdB, ProdC, Spain, Italy}\}$$

In this example, $\overline{W}$ represents the set of all possible values that can be assigned to any of the process attributes within V, including values like “Accepted”, “Completed”, “ProdA”, “ProdB”, “Italy”, and so on. The partial function v(Status) can assign the value “Accepted” to the at-tribute “Status” and v(Country) can assign the value “Italy” to the attribute “Country”

Note that the same event can potentially occur in different traces, namely attributes are given the same assignment in different traces. This means that the entire same trace can potentially appear multiple times. This motivates us to define an event log as a multi-set of traces. Given a set X, $\mathrm{B}(X)$ indicates the set of all multisets with the elements in X.

Definition 2 (Traces & Event Logs): Let L be an event-log, which is a multiset of traces, i.e., $L \in B(E^*)$, where $B(E^*)$ represents the set of all multisets with the elements in E^*. Let $E = A \times R \times (V \rightarrow W)$ be the Universe of events. A trace is then denoted by σ, where $\sigma \in E^*$. A trace $\sigma = \langle e_1, \ldots, e_n \rangle$

SR_Number	Change_Date+Time	Status	ACTIVITY	Involved_ST	SR_Latest_Impact	Product	Country	Owner_Country
1-364285768	2010-03-31T15:59:42+01:00	Accepted	In Progress	V30	Medium	PROD582	fr	France
1-364285768	2010-03-31T16:00:56+01:00	Accepted	In Progress	V30	Medium	PROD582	fr	France
1-364285768	2010-03-31T16:45:48+01:00	Queued	Awaiting Assignment	V5 3rd	Medium	PROD582	fr	France
1-364285768	2010-04-06T15:44:07+01:00	Accepted	In Progress	V5 3rd	Medium	PROD582	fr	France
1-364285768	2010-04-06T15:44:38+01:00	Queued	Awaiting Assignment	V30	Medium	PROD582	fr	France
1-364285768	2010-04-06T15:44:47+01:00	Accepted	In Progress	V13 2nd 3rd	Medium	PROD582	fr	France
1-364285768	2010-04-06T15:44:51+01:00	Completed	Resolved	V13 2nd 3rd	Medium	PROD582	fr	France
1-364285768	2010-04-06T15:45:07+01:00	Queued	Awaiting Assignment	V30	Medium	PROD582	fr	France
1-364285768	2010-04-08T11:52:23+01:00	Accepted	In Progress	V30	Medium	PROD582	fr	France
1-364285768	2010-04-08T11:53:35+01:00	Queued	Awaiting Assignment	V5 3rd	Medium	PROD582	fr	France
1-364285768	2010-04-20T10:07:11+01:00	Accepted	In Progress	V5 3rd	Medium	PROD582	fr	France
1-364285768	2010-04-20T10:07:19+01:00	Accepted	Assigned	V5 3rd	Medium	PROD582	fr	France
1-364285768	2012-04-11T16:11:17+01:00	Accepted	In Progress	V5 3rd	Medium	PROD582	fr	France
1-364285768	2012-04-11T16:11:25+01:00	Accepted	Assigned	V5 3rd	Medium	PROD582	fr	France
1-364285768	2012-05-03T10:10:10+01:00	Accepted	In Progress	V5 3rd	Medium	PROD582	fr	France
1-364285768	2012-05-03T10:10:12+01:00	Completed	Resolved	V5 3rd	Medium	PROD582	fr	France
1-364285768	2012-05-11T00:26:15+01:00	Completed	Closed	V5 3rd	Medium	PROD582	fr	0
1-503573772	2011-02-24T16:17:46+01:00	Accepted	In Progress	D5	Medium	PROD706	nl	Belgium
1-503573772	2011-02-24T16:17:52+01:00	Accepted	In Progress	D5	Medium	PROD706	nl	Belgium
1-503573772	2011-02-24T16:19:35+01:00	Queued	Awaiting Assignment	V37 2nd	Medium	PROD706	nl	Belgium
1-503573772	2011-02-28T13:46:38+01:00	Accepted	In Progress	V37 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2011-02-28T13:47:22+01:00	Queued	Awaiting Assignment	D5	Medium	PROD706	nl	Netherlands
1-503573772	2011-02-28T13:48:12+01:00	Accepted	In Progress	D5	Medium	PROD706	nl	Belgium
1-503573772	2011-02-28T13:53:24+01:00	Queued	Awaiting Assignment	V32 2nd	Medium	PROD706	nl	Belgium
1-503573772	2011-05-04T17:07:37+01:00	Accepted	In Progress	V32 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-02-17T14:49:34+01:00	Queued	Awaiting Assignment	V32 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-03-21T16:21:44+01:00	Accepted	In Progress	V32 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-03-26T10:55:35+01:00	Accepted	In Progress	V37 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-03-26T10:57:04+01:00	Queued	Awaiting Assignment	V32 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-04-03T14:29:44+01:00	Accepted	In Progress	V32 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-04-03T14:29:49+01:00	Accepted	Wait	V32 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-05-04T07:58:20+01:00	Accepted	In Progress	V37 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-05-04T07:59:39+01:00	Completed	Resolved	V37 2nd	Medium	PROD706	nl	Netherlands
1-503573772	2012-05-12T00:21:31+01:00	Completed	Closed	V37 2nd	Medium	PROD706	nl	0

Figure 2.2: Example off an eventt logs, wtitth 2 ttraces. The SR_Number tis tthe untique case (or ttrace) tidentttifftier, "ACTIVITY" tis tthe actttivtitty column, and "Inovlved_ST" tis tthe resource column.

The most convenient way to store the even log data is in a CSV (Comma-Separated Values) format. Figure 2.2 shows an example of an event log in this format. Here, each row represents an event, and each column represents an attribute of the event. A set of rows represents a trace, which is a process instance.

The attribute "SR_Number" is the unique identifier of a trace. Attribute "ACTIVITY" is the activity of an event and the attribute "Involved_ST" is the resource that performed that activity in this event. Rest of the columns contain additional attributes of the event.

Given an event $e = (a, r, v)$, the remainder uses the following shortcuts: $activity(e) = a$, $resource(e) = r$, $time(e) = t$, and $variables(e) = v$. Also, given a trace $\sigma = \langle e_1, \ldots, e_n \rangle$, $prefix(\sigma)$ denotes the set of all prefixes of σ, including σ: $\{\langle\rangle, \langle e_1 \rangle, \langle e_1, e_2 \rangle, \ldots, \langle e_1, \ldots, e_n \rangle\}$.

Next, we will define what we aim to improve through our prescriptive analysis. We call this the Key Performance Indicator (KPI). Thus, our goal will be to improve this KPI value.

Definition 3 (KPI Function): Let $E^{\square}$ be the universe of events defined over a set V of attributes. $K : E^{\square} \times \mathrm{N} \rightarrow \mathrm{R}$ - A KPI function such that, given a (prefix of a) trace $\sigma \in E^{\square}$, $K(\sigma)$ returns the KPI value of that σ.

Since the KPI function returns a numeric value, the return values can also belong to the set of timestamps. We also define $img(K)$ as the set of all possible KPI values.

It is also worth noting that, though the definition of a KPI function allows as input even a prefix for a trace, its calculation requires that the traces (basically a process) finish before we can get the KPI value. That is, the KPI is computed a posteriori. Hence, we cannot calculate the KPI value of a trace that is still running.

Given a trace $\sigma = \langle e_1, \ldots, e_n \rangle$ that records a complete process execution, the following are examples of two potential KPI definitions:

- *Total Time:* Measures the total execution time of a trace σ. That is, for all elements in $prefix(\sigma)$ the KPI function $K_{total}(\sigma)$ will return the value of $time(e_n) - time(e_1)$. For example a worker started product assembly at $time(e_1) = 12:00$ hours and by the end of the assembly line the final worker verified the product quality completing the assembly process at $time(e_n) = 13:30$, the $K_{total}(\sigma)$ would return $13:30 - 12:00 = 1.5$ hours for this process instance. As pointed out in the note above, calculation of this time value will require that the event e_n has occurred.

- Activity Occurrence. Measures if a certain activity occurs (or is going to occur in case of a prefix) in the trace. For example, an activity such as "Default Loan" in a loan-application process. If the activity to measure is "Default Loan", for all elements in $prefix(\sigma)$, the KPI function $K_{occur}(\sigma)$ returns the number of events that have the activity "Default Loan". As before, calculation of this numeric value will require that the event e_n has occurred.

The goal of the recommender system is to provide recommendations on both the activities to be performed and the resources best suited to perform them, with the aim of enhancing the final outcome of running process instances in terms of the identified KPIs. To achieve this, a **predictive analytics oracle function** must be developed. This function will enable the prediction of the KPIs of the final outcome of a running process instance and will identify the best activity to be performed and the most suitable resource to perform it.

Definition 4 (Predictive Oracle Function): Let $\Phi_K : X_1 \times \ldots \times X_m \rightarrow \mathrm{R}$ be the *Predictive oracle function*. Let $\rho_L : E^{\square} \rightarrow X_1 \times \ldots \times X_m$ be the *Trace-to-instance encoding function*. Let L be an event log that records the execution of a given process, for which a KPI K is defined. Let $\sigma' = \langle e_1, \ldots, e_k \rangle \in L$ be the trace of a running case that will eventually

complete as $\sigma_T = \langle e_1, \ldots, e_k, e_{k+1}, \ldots, e_n \rangle$. The prediction problem can be formulated as predicting the value of $K(\sigma')$ i.e. $\Phi_K(\sigma') \approx K(\sigma')$, and from the definition of KPI function we know $K(\sigma') = K(\sigma_T)$

For example, lets consider a case where we are working with the KPI total time and $K = K_{total}$, if the execution time of the completed trace σ_T will be 2 hours ($K(\sigma_T) = 2$ hours) we want to predict this value, 2 hours, from any prefix $prefix(\sigma_T)$ of this trace such as σ'.

In the process mining literature, this problem has been tackled with different machine learning models [12] and [24]. We approach the problem by estimating a function $\Phi_K : X_1 \times \ldots \times X_m \rightarrow \mathrm{R}$ which for an incomplete trace σ', forecasts the values of the KPI K.

Next, each prediction technique requires the definition of the domain $X_1 \times \ldots \times X_m$ and a trace-to-instance encoding function $\rho_L : E^{\square} \rightarrow X_1 \times \ldots \times X_m$, which maps each (prefix of a) trace σ to a vector $\rho_L(\sigma) \square X_1 \times \ldots \times X_m$ with length m. Each element in $\rho_L(\sigma)$ can be of a different nature, such as a process activity, a timestamp, the number of executions of an activity in σ, a resource, or a process attribute.

For example $K_{total}(\sigma) = 2$ hours. We will use ρ_L to convert a trace σ to $X_1 \times \ldots \times X_m$ which our predictive oracle function Φ_K can work with. We expect $\Phi_K(\rho_L(\sigma))$ to return values close to 2 hours.

The prediction model is trained ofline through a dataset D that is created from an event log $L \square \mathrm{B}(E^{\square})$ as follows: Each prefix $\sigma^p \square prefix(\sigma)$ generates one distinct item in D consisting of a pair $(x, y) \square (X_1 \times \ldots \times X_m \times img(K))$, where $x = \rho_L(\sigma^p)$ and $y = K(\sigma^p)$. Example of an item in the dataset:

Case ID	Timestamp	Status	ACTIVITY	Involved_ST	…	lead_time
1-364285768	1270047582	Accepted	In Progress	V30	…	66644793

Table 2.1: Table showting (x, y). Column 'lead_tttime' = $img(K)$ and tthe restt representt $X_1 \times \ldots \times X_m$

The result is that for a given trace σ and any $\sigma^p \square prefix(\sigma)$, our predictive oracle function $\Phi_K(\rho_L(\sigma^p))$ returns the predicted KPI value $K(\sigma)$. To keep the notation simple, we refer to $\Phi_K(\rho_L(\sigma))$ as $\Phi_K(\sigma)$.

The next challenge we face is that the event data is inherently sequence-based, so a traditional machine learning algorithm will not work. A solution could be to use an LSTM based model, but we can also preprocess the trace data to encode the history of a prefix of σ. This will enable non-sequence-based ML models to work, and they can even shufle data instances in the dataset D, and it would not be a problem.

Consequently, we augment the definition of our trace-to-instance encoding function and include the function $\rho_A^{aggr}(\langle e_1, \ldots, e_n \rangle)$. Here, for each activity $a \in A$, there is a dimension in $\rho_A^{aggr}(\sigma) : E^* \rightarrow (\mathrm{N})^{|A|}$ that takes on a value equal to the number of events $e \in \sigma$ that refer to a (that is, such that activity$(e) = a$) [25]. The function ρ is then defined as: $\rho_L(\langle e_1, \ldots, e_n \rangle) = \rho_A^{aggr}(\langle e_1, \ldots, e_n \rangle) \oplus \text{activity}(e_n) \oplus \bigoplus_{v \in V} \text{variables}(v)(e_n)$*. When the event log L on which ρ depends is evident from the context, we omit the subscript $_L$.

e.g. given a trace (for simplicity we will not include any attributes) $\sigma = \langle (act2, res1), (act1, res2), (act3, res8), (act1, res4), (act3, res1), (act4, res8), (act5, res2) \rangle$ applying $\rho_L(\sigma)$ results in frequency vectors as shown in Table: 2.2. Each row represents a history-encoded vector that is sufficient for KPI prediction.

Activity	act1	act2	act3	act4	Resource
act2	0	0	0	0	res1
act1	0	1	0	0	res2
act3	1	1	0	0	res8
act1	1	1	1	0	res4
act3	2	1	1	0	res1
act4	2	1	2	0	res8
act5	2	1	2	1	res2

Table 2.2: Shows how a ttrace $\sigma = \langle (act2, res1), (act1, res2), (act3, res8), (act1, res4), (act3, res1), (act4, res8), (act5, res2) \rangle$ wtith 7 eventts $e = (a_i, r_j)$ tis encoded vtia ffrequency-vecttor encodting ttechntique.

Leontjeva et al. [26] showed that frequency vector encoding is a good balance between abstraction richness and complexity, which is why we can even make traditional machine learning models work. Thus, we can say that this technique is independent of any predictive model.

For our implementation, we decided to use a variant of Random Forest and Catboost, as they have been shown to be quite effective [3] in predicting KPIs in event logs' (L) data. The Random Forest algorithm is a popular ensemble learning method that combines the predictions of multiple decision trees to improve accuracy of the prediction. It is particularly effective for "categorical" data [27], and generally event log data have many categorical attributes.

Definition 5 (Evaluation Oracle Function) Let $\Lambda_K : X_1 \times \ldots \times X_m \rightarrow \mathrm{R}$. Evaluation Oracle function is Λ_K

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

The evaluation oracle function is the same as the predictive oracle function defined in Definition 4. There is a separate definition to associate a different symbol with the function that will make predictions for the evaluation of the results. We talk about the evaluation methodology in detail in the sections 5.4 and 6.4

A process aware recommender system aims to recommend the next best activity and resource to improve the relevant KPI. However, the next activity and activity-resource combination needs to be valid from a domain point of view. We avoid the strong assumption that there exists a process model that prescribes how process instances must be executed.

We 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 (State-Representation Function): Let S be the set of all possible state representations. Let l^{state} be the state-representation function, where $l^{state} : E^{\square} \rightarrow S$. Given σ is a trace, then for each (prefix of a) trace σ the function l^{state} returns the state.

Determining the activities allowed after the occurrence of a sequence of events requires building a **transition system**. The representation can also be shown visually where the nodes are the state observed in the log, and the arcs are the activities observed in these states [28]. An example representation is shown in Figure 2.3

Definition 7 (Transition System): Given S is the set of states. Let T be the transition relation. Let:

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

The transition system is a tuple of (S, T). Given we want to define our transition system over our event log L, the transition function becomes $TS_L = (S, T) \subseteq \mathrm{R} \times (\mathrm{R} \times E \times \mathrm{R})$. To construct the transition system the activity sequence from the event log is used.

Figure 2.3 shows an example of a transition system in accordance with definition 7 . It has been built on an event log $L^{ex} = \{\langle a, b, c, d\rangle, \langle a, b, c, e\rangle, \langle a, b, c, f\rangle, \langle a, b, c, g\rangle, \langle a, c, d, f\rangle\}$[4] using a sequence-based state representation function $l^{state}_{sq}(\langle e_1, \ldots, e_n\rangle) = \langle activity(e_1), \ldots, activity(e_n)\rangle$. Through this function, the state of a (prefix of a) trace is identified with its ordered list of activities. For the example with L^{ex}, the set of possible states is thus: $S = \{\langle a, b, c, d\rangle, \langle a, b, c, e\rangle, \langle a, b, c, f\rangle, \langle a, b, c, g\rangle, \langle a, c, d, f\rangle, \langle a, c, d\rangle, \langle a, b, c\rangle, \langle a, b\rangle, \langle a, c\rangle, \langle a\rangle\}$. [28], [29]. Transition systems are important because they tell us which activities are

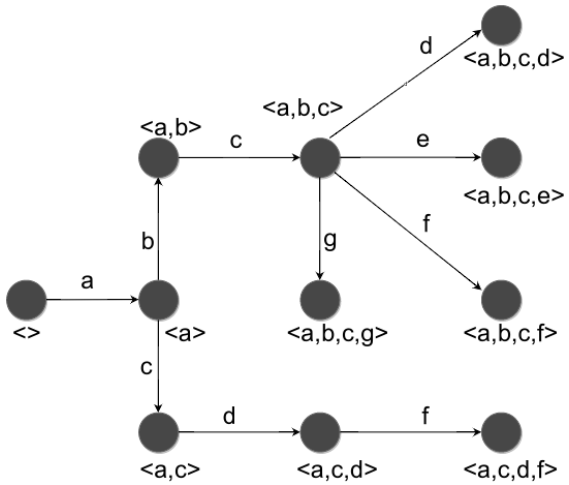

Figure 2.3: Example off a Transtitttion Systtem. [1]

allowed and which are not allowed, after observing a specific sequence of activities. For example, from the L^{ex} event log the transition system tells us that event a cannot be followed by event d and only event f can follow event d.

The transition system can naturally be extremely large and not intelligible, but this poses no threat because they are used internally and never shown to process actors.

Similarly, to validate the suggestion of a resource to perform an activity, we need an activity-resource validation function. This function is a set of all the combinations of activity and resource in the event log L.

Definition 8 (Activity-Resource Validation Function): Let $\Omega_L : A \times R \rightarrow \{0, 1\}$. Ω_L is the *Resource Validation function* on the event log L.

$\Omega_L(a, r)$ is a function that maps each activity-resource pair to either 1 or 0. In the event log L, for each event e, the function inspects the associated resource and activity. If the resource-activity combination exists in L, then $\Omega_L(a, r)$ outputs 1, denoting "true", otherwise it outputs 0, denoting "false". In this way, $\Omega_L(a, r)$ precisely represents the existence of each possible resource-activity pair in the event log.

2.4 Machine Learning

Machine Learning (ML), a fundamental subset of Artificial Intelligence (AI), has established itself as a key discipline within the realm of computational intelligence. It empowers systems to autonomously learn and enhance their performance from experience, circumventing the need

for explicit programming. Central to ML is the development and deployment of algorithms and statistical models that enable systems to analyze data, discern patterns, and consequently make informed predictions or decisions. This self-learning capability is achieved by building models from sample data, which then guide data-driven decision-making processes, as opposed to adhering to static programmatic instructions.

ML algorithms are sophisticated computational models designed to perform tasks by learning from data rather than through explicit instructions. This approach to problem-solving has been instrumental in driving significant advancements across various fields such as healthcare, natural language processing (NLP), image recognition, finance, and recommendation systems. In these applications, ML algorithms have demonstrated exceptional proficiency, effectively analyzing vast datasets to identify patterns and make predictions or decisions.

The essence of machine learning lies not just in its computational prowess but also in its ability to adapt and learn from input data. This attribute forms the crux of the algorithmic methodologies explored in the paper "Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations." Here, interpretability of machine learning models is explored, which develops trust on the model's capabilities to make autonomous decisions in a complex, data-driven world.

2.4.1 Machine Learning Overview

Machine learning algorithms are broadly categorized into supervised, unsupervised, and reinforcement learning paradigms:

- Supervised Learning: This paradigm involves learning a function that maps an input to an output based on example input-output pairs. It encompasses algorithms that can perform tasks such as classification, where the goal is to predict discrete labels, and regression, where the objective is to predict continuous values.
- Unsupervised Learning: Unsupervised learning models identify patterns in data without reference to known, or labeled, outcomes. Clustering and dimensionality reduction are central to this category, providing the means to understand the structure and distribution of data.
- Reinforcement Learning: In reinforcement learning, algorithms learn to make sequences of decisions by interacting with a dynamic environment to achieve a certain goal. Such models are characterized by trial-and-error search and delayed reward.

Our proposed prescriptive analytics framework leverages the "Unsupervised Learning" paradigm.

2.4.2 Machine Learning Algorithms

Balanced Random Forest. In the realm of machine learning, particularly in the context of classification tasks, the challenge of imbalanced datasets is a pervasive issue. Traditional classification algorithms often exhibit a bias towards the majority class, leading to suboptimal performance on the minority class. This is where the concept of the Balanced Random comes into play, offering a robust solution to address class imbalance.

The Balanced Random Forest, a variant of the classic Random Forest algorithm, is designed specifically to improve classification performance on imbalanced datasets. It is an ensemble learning method, which means it constructs multiple decision trees during the training process and outputs the class that is the mode of the classes predicted by individual trees. However, unlike the traditional Random Forest, the Balanced Random Forest algorithm introduces a key modification in the way training data is sampled for constructing each tree in the ensemble.

In Balanced Random Forest, each tree is trained on a balanced bootstrap sample. This means that for each tree, the algorithm takes a random sample of the minority class and a random sample (of equal size) from the majority class. By doing so, it ensures that each tree is trained on a dataset that has an equal representation of both classes, effectively addressing the issue of class imbalance. This balanced bootstrap sampling approach allows the Balanced Random Forest to focus more on the minority class, reducing the bias towards the majority class that is often seen in traditional random forest classifiers.

In summary, the Balanced Random Forest algorithm aggregates the predictions from all the decision trees to determine the final output. The use of multiple trees reduces the risk of overfitting, a common problem in machine learning models, especially in complex datasets. The ensemble nature of the Balanced Random Forest also contributes to its robustness and generalizability, making it a preferred choice for imbalanced datasets across various domains, such as fraud detection, medical diagnosis, and anomaly detection.

CatBoost, an acronym for "Category Boosting," is a state-of-the-art open-source gradient boosting library, particularly recognized for its effectiveness in handling categorical data. CatBoost is a machine learning algorithm that uses decision trees and gradient boosting techniques. It is designed to provide high performance, scalability, and ease of use for a wide range of standard machine learning tasks.

One of the key strengths of CatBoost is its ability to naturally and eficiently process categorical variables, which are common in many real-world datasets but often require extensive preprocessing when using other machine learning algorithms. CatBoost handles categorical

features by employing an innovative algorithm that combines various statistics on categorical features with a small random subset of the data, a process known as ordered boosting. This approach not only reduces the need for extensive data preprocessing but also minimizes the chances of overfitting, a common challenge in machine learning.

CatBoost also stands out for its robust handling of overfitting, especially in small datasets, through the implementation of oblivious trees as base predictors. Oblivious trees are a type of decision tree where each level uses the same split, which makes them more regularized and less prone to overfitting compared to regular decision trees.

In summary, CatBoost represents a powerful tool in the machine learning landscape, particularly suited for tasks involving complex datasets with categorical features. Its unique approach to handling categorical data, along with its high eficiency and ease of use, make it an invaluable asset for developing sophisticated, accurate, and eficient predictive models.

XGBoost, standing for eXtreme Gradient Boosting, is a highly eficient and scalable implementation of gradient boosting framework, which has gained substantial popularity and recognition in the field of machine learning for its performance and speed. XGBoost is renowned for its ability to handle large-scale and complex data eficiently. At its core, XGBoost utilizes a gradient boosting algorithm, which builds an ensemble of decision trees in a sequential manner, where each subsequent tree corrects the errors made by the previous ones. This methodology significantly enhances the model's predictive accuracy.

XGBoost also includes a regularized model formalization to control overfitting, making it robust and accurate, especially in cases where the dataset is small or highly complex. The addition of regularization parameters helps in balancing model complexity with predictive performance, which is a critical aspect of building reliable predictive models.

In summary, XGBoost is a powerful and versatile machine learning algorithm that offers state-of-the-art performance in predictive modeling. Its high eficiency, scalability, and flexibility allow it to consistently outperform other machine learning algorithms in many real-world scenarios and Kaggle competitions. XGBoost is a preferred choice among data scientists and practitioners for tackling a wide range of data-driven challenges.

Neural Networks, a fundamental construct in the field of Artificial Intelligence (AI) and machine learning, represent a computational model inspired by the human brain's structure and function. At their core, neural networks are designed to mimic the way biological neurons signal to one another, making them capable of learning and making complex decisions. [30]

A neural network consists of layers of interconnected nodes or 'neurons,' each resembling a simplified version of a biological neuron. These layers are typically categorized into three

types: input, hidden, and output layers. The input layer receives the initial data, the hidden layers perform computations through a system of weighted connections, and the output layer delivers the final result or prediction.

The strength of neural networks lies in their ability to learn these weights through a process known as training. During training, the network adjusts its weights based on the errors in its predictions, using a method known as backpropagation combined with an optimization algorithm like gradient descent. This iterative adjustment allows the network to improve its predictions or decision-making capabilities over time.

Neural networks are incredibly versatile and have been applied to a wide array of tasks that involve pattern recognition, such as image and speech recognition, natural language processing, and medical diagnosis. They are particularly effective in handling complex, non-linear relationships within data, which makes them suitable for tasks where traditional algorithms might struggle.

Despite their power and flexibility, neural networks require substantial data and computational resources for training, especially in the case of deep learning. They also present challenges in terms of interpretability, as the complex relationships they learn are not always easy to understand or explain.

2.4.3 Metrics To Evaluate ML Models

The **F1-score** is a widely used metric in the field of machine learning and statistics, particularly in the context of classification tasks. It represents a harmonic mean of precision and recall, providing a single score that balances both these important aspects of a classifier's performance. [31]

Precision and recall are critical measures in classification problems, especially in scenarios where the balance between false positives and false negatives is crucial. Precision is the ratio of true positives to the sum of true and false positives, indicating the accuracy of positive predictions. Recall, also known as sensitivity, measures the ratio of true positives to the sum of true positives and false negatives, reflecting the ability of a classifier to identify all relevant instances.

The F1-score harmonizes these two metrics by calculating their harmonic mean. Mathematically, it is defined as:

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

This metric is particularly useful in situations where an equilibrium between precision and

recall is required. A high F1-score indicates that both precision and recall are high, which is desirable in many real-world applications such as document classification, patient diagnosis, and other domains where both false positives and false negatives carry significant consequences.

The F1-score is also especially useful in scenarios with imbalanced datasets, where one class significantly outweighs the other. In such cases, traditional accuracy metrics can be misleading, as they might reflect the underlying class distribution rather than the actual performance of the classifier. The F1-score, by combining precision and recall, offers a more informative and reliable evaluation in these contexts.

In summary, the F1-score serves as a critical tool in assessing the eficacy of classification models, providing a balanced view of their performance by simultaneously considering both precision and recall. This makes it a valuable metric for model evaluation, particularly in fields where accurate classification is pivotal.

Mean Absolute Error (MAE) is a fundamental statistical measure used extensively in the field of machine learning, particularly in regression analysis, to quantify the accuracy of a predictive model. MAE provides a simple, interpretable representation of the average magnitude of errors in a set of predictions, without considering their direction. [32]

Mathematically, the MAE is calculated as the average of the absolute differences between the predicted values and the observed actual values. For a set of n predictions, the formula for MAE is expressed as:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

where y_i represents the actual value, $\hat{y}_i$ denotes the predicted value, and $|y_i - \hat{y}_i|$ is the absolute error for each prediction.

The absolute difference means that the MAE takes into account the magnitude of the error without considering its direction, thus providing a 'fair' evaluation metric. This characteristic makes MAE a particularly robust measure against outliers, as it does not square the errors in the calculation (unlike the Mean Squared Error, MSE). As a result, large deviations are not overly exaggerated, making MAE a reliable measure of model performance, especially in datasets with anomalies or outliers.

One of the key strengths of MAE is its interpretability. The metric is measured in the same units as the data, making its values easy to understand and communicate. For instance, in a temperature forecasting model, an MAE of 3 degrees would imply that the average prediction error is 3 degrees, which is straightforward for both experts and non-experts to comprehend.

Despite its simplicity, MAE provides valuable insights into the overall accuracy of predictive models. It is widely utilized in various applications, from financial forecasting to weather prediction, where understanding the average error magnitude is crucial for model evaluation and decision-making processes.

In summary, Mean Absolute Error is a crucial metric in machine learning for assessing the accuracy of predictive models. Its simplicity, robustness, and interpretability make it an essential tool in the arsenal of data scientists and analysts for evaluating and communicating the performance of regression models.

2.5 Generation of Counterfactual Examples

Counterfactual explanation, a nuanced concept rooted in causal reasoning, serves as a pivotal subset of interpretability in the machine learning landscape. These explanations provide a critical link between actual occurrences and hypothetical scenarios, examining how changes to model inputs could have led to different outcomes. This approach, fundamentally a post-hoc interpretative method, delves into "what-if" scenarios, thereby offering valuable insights into the inner workings of machine learning models. By altering the original data instance and observing how these perturbations affect the output, counterfactual explanations illuminate specific changes necessary to achieve an alternate decision, thus providing a deeper understanding of the model's decision-making process.

Contrasting with other explainability techniques, counterfactual explanations, often referred to as CFEs or recourses, do not directly address the "why" behind a model's prediction. Instead, they focus on suggesting modifications to attain a desired outcome. For example, a counterfactual explanation might assert, "The admission would have been granted if the applicant's math score was 10% higher." Such actionable explanations stand out for their clarity, comprehensibility, and direct applicability.

Furthermore, in the context of regulatory compliance, such as under the General Data Protection Regulation (GDPR), counterfactual explanations have been recognized as a powerful tool for providing legally required explanations for decisions made by automated systems [33]. This aspect underscores the practical relevance of CFEs in adhering to laws governing machine-produced decisions.

While the study and application of counterfactuals are well-established in the realms of data mining and machine learning, as evidenced by works like Russell et al. [34] and Wachter et al. [35], their integration into process mining is still emerging. In this thesis, we venture into this

relatively uncharted territory, exploring how the counterfactual approach can be leveraged to recommend specific activities and resources. This is different from its conventional application of the field where it is used for explanation of the ML model's output.

In this thesis, we will explore how counterfactual approach can recommend activity a and resource r to optimize a KPI K; we will call these: **Counterfactual Examples** (or **CFEs** for short). This exploration aims to bridge theoretical models with practical applications, enabling a more transparent and data-driven decision-making framework in process mining.

DiCE (Diverse Counterfactual Explanations) Algorithm

The use of counterfactuals for multidimensional sequence data, such as event logs, is still sparse in the literature. In this thesis, we explore the use of a counterfactual algorithm, DiCE [5], in the context of prescriptive process analytics.

The terminology for understanding the algorithm is as follows:

- x - Original input to the model. In the context of process mining an instance (prefix of a) trace returned by $\rho_L(\sigma)$
- y - Original output of the model (Undesirable value of the KPI)
- c - Counterfactual example, which we are trying to find. A perturbed version of x. (The desired value of the KPI)
- y' - The desired output class
- Φ_K - ML model
- k - number of counterfactuals the algorithm produces
- $C_K : X_1 \times \ldots \times X_m \rightarrow 2^{E^{\square} \times \mathrm{R}}$ - **Counterfactual function**.

In the counterfactual function $2^{E^{\square} \times \mathrm{R}}$ is the power set of $E^{\square} \times \mathrm{R}$. In set theory, the power set of any set S is the set of all possible subsets of S, including the empty set and S itself. Therefore, $2^{E^{\square} \times \mathrm{R}}$ includes every possible subset of the Cartesian product of event log sequences and real numbers.

As we can see, the counterfactual function C_K also requires that the input instance be encoded, like it is encoded for the predictive oracle function Φ_K. So, for this we will also use the same Trace-to-instance encoding function $\rho_L : E^{\square} \rightarrow X_1 \times \ldots \times X_m$ which is defined

in definition 4. The counterfactual function C_K takes as input a (prefix of a) trace σ and returns the power set of traces containing recommendations and their corresponding KPI values $\{(c_1, K(c_1)), \ldots, (c_z, K(c_z)\}$, where $c_1, \ldots c_z$ are different counterfactual examples and $K(c_1) \ldots K(c_z)$ are the corresponding KPI values. Below we explain the internal workings of the counterfactual fucntion as described in the paper [5].

DiCE (Diverse Counterfactual Explanations) is an algorithm designed to generate counterfactual examples such that they satisfy three important properties: diversity, proximity, and sparsity. It is also designed to satisfy user specified constraints. Below is a brief explanation of how the algorithm works.

Input: The algorithm takes as input a trained machine learning model Φ_K and an instance of interest x for which a counterfactual explanation is desired.

Optimization: DiCE formulates the generation of counterfactual examples as an optimization problem. It aims to find a set of k counterfactual examples $\{c_1, c_2, \ldots, c_k\}$ that changes x so that it leads to different prediction y'.

Loss Function: DiCE defines a loss function $C(x)$ that combines three components:

- *Yloss*: This component ensures that the counterfactuals have different predictions than the original instance. It uses a hinge-loss function that penalizes differences in predictions between counterfactuals and a desired outcome.

$$c = \underset{c}{argmin}(yloss(\Phi_K(c), y) + |x - c|)$$

- *Proximity*: This component promotes counterfactual examples that are close to the original instance; this generally makes them more relevant and useful given the context of the original instance x. It quantifies proximity as the negative vector distance between the counterfactual and the original input.

$$proximity = -\frac{1}{k}\sum_{i=1}^{k} dist(c_i, x).$$

- *Dpp_diversity*: This component captures the diversity among counterfactuals. It utilizes a determinantal point process (DPP) to measure the diversity of the counterfactual set based on the determinant of a kernel matrix. Random perturbations are added to the diagonal elements of the kernel matrix to avoid ill-defined determinants. $K_{i,j} = \frac{1}{1+dist(c_i, c_j)}$ and $dist(c_i, c_j)$ denote a distance metric between the two counterfactual examples.

$$dpp\_diversity = det(K)$$

Thus, the final loss function becomes:

$$C(x) = \underset{c1,\ldots,c_k}{argmin} \frac{1}{k} \sum_{i=1}^{k} yloss(\Phi_K(c_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^{k} dist(c_i, x)$$

Optimization Procedure: DiCE optimizes the loss function $C(x)$ using gradient descent. It iteratively adjusts the counterfactual examples to minimize the loss function. The optimization process aims to find counterfactuals that satisfy the desired diversity, proximity, and prediction differences.

User Constraints: Users can provide constraints on feature manipulation, such as specifying feasible ranges for each feature or indicating which variables can be changed.

Post-processing: After the optimization process, DiCE conducts post-processing steps to enhance sparsity. Because sparsity enhancing is a non-convex constraint it is not included in the loss function. Sparsity refers to the number of features that need to be changed to generate the counterfactual. A counterfactual is considered more feasible if it changes a smaller number of features. DiCE uses generated counterfactuals and restores the values of continuous features to their original values (as in x) until the predicted class $\Phi_K(c) = y'$ changes, encouraging sparsity in the number of features changed.

By following this procedure, the DiCE algorithm generates a set of diverse counterfactual examples that provide actionable explanations for machine learning models. In the thesis, we adapt these counterfactual explanations as actionable recommendations for processes, allowing process users to optimize a process. These counterfactual explanations allow users to understand how different inputs could lead to alternative predictions and help them make informed decisions. The algorithm's ability to take into account the constraints specified by the user further enhances its practical utility.

Counterfactual explanations can also work with black-box models (when only the predict function of the model is accessible) and therefore place no restrictions on model complexity and do not require model disclosure. They also do not necessarily approximate the underlying model, producing accurate feedback. This feature can be very useful when a company does not want to disclose the workings of its model.

3

Related Works

This chapter discusses some of the works that are related to our research topic in some way. To the best of our knowledge, no one has yet implemented domain-agnostic prescriptive analytics framework based on counterfactuals.

In the field of process mining, descriptive analytics plays a fundamental role in unraveling the intricacies of business processes by providing a detailed account of historical data. As the most basic form of data analysis, descriptive analytics involves the examination and summarization of past events to derive meaningful insights and patterns [36].

Descriptive analytics in process mining primarily focuses on the visualization and analysis of process data as it has occurred. This involves the use of techniques like process discovery, which generates models from event logs, offering a graphical representation of the process flow. These models depict the sequence of activities, variations in the process, and potential deviations from the ideal or theoretical process flow. By doing so, descriptive analytics helps in identifying bottlenecks, ineficiencies, and compliance issues within a process.

Furthermore, descriptive analytics extends to the analysis of performance-related aspects of the process. This includes measuring key performance indicators (KPIs) such as throughput times, waiting times, and frequency of activities. Such analysis is pivotal in understanding the eficiency and productivity of business processes.

Predictive analytics represents a sophisticated advancement in the field of process mining, transcending beyond the descriptive analysis of past data to forecast future process behaviors and outcomes. This branch of analytics leverages statistical models and machine learning tech-

niques to predict the trajectory of ongoing process instances, offering foresight into potential future states of business processes. [37]

At the heart of predictive analytics in process mining is the utilization of historical event log data to train predictive models. These models analyze patterns and trends in past process executions and use this information to make informed predictions about future events. [38] For instance, predictive models can estimate the time remaining until the completion of a process instance, predict the likelihood of a particular outcome, or identify potential risks and deviations that might occur in a process.

One of the key applications of predictive analytics in process mining is in the realm of next-event prediction. Here, models are developed to forecast the next step in a process, providing insights into what might happen next in a given process instance. Another significant application is in the domain of process performance, where predictive models are employed to forecast key performance indicators (KPIs) such as completion times, costs, and quality outcomes. By predicting these KPIs, organizations can proactively manage and optimize their processes to meet desired performance levels.

The process mining literature (as part of the PAR-system) has predominantly focused on ”Descriptive Analytics” and ”Predictive Analytics”, yielding valuable insights into understanding and predicting process behaviors. Compared to descriptive and predictive, prescriptive analytics is still less mature [21]. And ”there is still work to be done in this direction [i.e., process-aware recommender systems]” [12]

However, in recent years there has been a notable increase in interest toward ”Prescriptive Analytics”, illustrating the growing need to derive actionable insights from predictive models and data for better decision making. The conventional way of achieving this is to recommend the next best activity that improves a KPI [3], [6], [7]. Another option is to focus on optimizing resource utilization by suggesting which resource should perform specific activities in various contexts [8] [9].

In [3] the way they do this by simulating all possible continuations of a running case and then predicting the final improvement for each continuation. The system then recommends the activity that is predicted to improve the running process the most. The authors also argue about the importance of using factual data in the event logs to suggest recommendations for improving the process, as opposed to relaying on process owners to suggest the next best action.

The paper [1] goes in a different direction and works on the explainability of the recommendations made by the PAR system. They build their work on [3]. To explain the recommendation of the prescriptive component, they use process-related characteristics, such as the values

of process variables, activities performed, and resources involved. This system aims to engage process owners more deeply in the decision-making process and reinforce their understanding of the rationale of the suggested interventions.

However, these papers do not recommend both the activity and resource. The work of Padella et al. [2] recommends both (activity and resource) for cases, but it also focuses on the availability of resources and the impact of selecting a resource on the global KPI. This makes this work very suitable for comparison with our work. Our work also recommends both activity and resource for a case, but currently lacks the capability to check resource availability. The framework assumes that there are infinite resources available and that the system has to select the best valid resource.

Another objective of this work is to see how effective counterfactuals are for recommending the next activity and resource. Although counterfactuals have been extensively studied in the field of machine learning and data mining, e.g., [34] [35], they have now also emerged as a promising tool in prescriptive process analytics. As this thesis presents the use of counterfactuals in prescriptive analysis, one of the works that leverages counterfactuals in a very similar fashion is discussed below in depth.

Chihchen et al. [4] demonstrates the use of counterfactuals to get explainable insights for process mining. In their work, they identify the challenges associated with elucidating the predictions of Predictive Process Analytics (PPA), leading to the development of a novel counterfactual generation algorithm named DiCE4EL. This algorithm adapts the foundational principles of the DiCE (Diverse Counterfactual Explanations) algorithm.

The generic DiCE algorithm has displayed limitations when applied to event logs, often failing to yield sensible counterfactuals, resulting in either an inability to find any counterfactuals or generating counterfactuals that contradict domain process knowledge. The proposed **DiCE4EL** offers a solution to these challenges by navigating both the process context knowledge and categorical variables through the minimization of a multi-component loss function, composed of "Class Loss", "Distance Loss", "Category Loss", and "Scenario Loss".

DiCE4EL operates as a prescriptive analytics algorithm, offering explanations in the form of modifications to the original input that can guide the process toward the achievement of a desired milestone. The domain expert can interactively explore different scenarios or impose various constraints to derive meaningful insights from the generated recommendations.

A limitation of DiCE4EL is that it necessitates substantial domain knowledge and a well-structured event log environment with predefined milestones. The implementation of the algorithm uses an LSTM model, so it also requires preprocessing of the data set accordingly.

In the end, the paper mentions that there are no standardized evaluation protocols for counterfactual evaluation in the explainable AI (Artificial Intelligence) domain. Therefore, the authors opt for a qualitative approach for evaluation as opposed to a quantitative approach.

4

Framework for Generating Recommendations

Prescriptive analytics has recently gained popularity in the field of process mining, but there are still gaps in the literature. One such gap is that counterfactuals, which have been extensively studied in data mining and machine learning, are not fully explored for the purpose of prescriptive analytics. Our objective with this chapter is to explore the use of counterfactuals in the prescriptive analytics component of a "Process aware recommender system" (PAR system). To understand our solution that addresses this gap, we have covered the background required in section 2 and the current state of the art in section 3. Now we will look at the main contri-bution of this thesis, which is the prescriptive analytics framework that uses counterfacts for recommendations.

In Section 4.1 we will build a mathematical foundation for our solution, that is, show how the predictive oracle function Φ_K integrates with the counterfactual function C_K. This is important as we want to show that our methods are implementation agnostic and that the prescriptive analytics framework is not dependent on any specific predictive model. It also shows that the framework does not require any domain knowledge of the event log K. If the definitions mentioned in section 2.1 can be applied to an event log, this prescriptive analytics framework can be built for that event log as well.

Finally, in section 4.3 we talk about the libraries and tools used to implement our algorithm. We use only Python programming language for all our development and testing.

4.1 Mathematical Foundation

Our first question was **"How can we design a prescriptive analysis framework that is domain agnostic and suggests both the next-best activity and resource to optimize a given KPI?"** To solve this, we need to develop an algorithm that processes an incoming running trace and returns the best valid recommendation along with the associated KPI value. We will use an example to understand this. In this example we will use KPI, K_{total}, total time.

Let $K = K_{total}$, Φ_K be the predictive oracle function and σ_{trunc} be a running trace with an expected KPI value of $\Phi_K(\sigma_{trunc}) = 10$ days. This value as predictive by our oracle function may indicate that the process is going in some wrong direction, and it requires some corrective intervention to reduce the execution time. In a PAR system, when the expected execution time of a running trace exceed a certain threshold the PAR system can trigger a process of intervention from the process owner to get that trace back on track. Usually, at this step, process owners may use their judgement to make a decision, but as we learned from [15], that may not be the most optimal option. Thus, we expect our prescriptive analytics component to suggest the best activity and resource that can lower the expected KPI value the most, thereby optimizing the KPI and improving the trace state.

We will now define and explain the workings of the prescriptive oracle function.

Definition 9 (Recommendation Generating Framework): Let $P_K : X_1 \times \ldots \times X_m \rightarrow E^{\square}$ be the **Prescriptive oracle function**. Let $C_K : X_1 \times \ldots \times X_m \rightarrow 2^{E^{\square} \times \mathrm{R}}$ be the counterfactual generator function.

Just like we simplified the notation of $\Phi_K(\rho_L(\sigma))$ as $\Phi_K(\sigma)$, we will simplify the notation of counterfactual function and refer to $C_K(\rho_L(\sigma))$ as $C_L(\sigma)$.

Given the definition 9, in the example above, this is how the intervention and results may look like: $P_K(\sigma_{trunct})$ and returns the recommendation as $((\sigma_{trunc} \ \square(a_5, r_1)), 7)$, where a_5 is the specific activity and r_1 is the specific resource. Note: Here and later, $(\sigma \ \square(a, r))$ indicates that the trace σ is being extended with an event (a, r, B). Here $\mathrm{B} : V \rightarrow \{\square\}$ and $\mathrm{B}(v) = \square$ for all the v in V that do not directly depend on the activity a. Therefore, the value assigned by the partial function v cannot be inferred with certainty by simply knowing a. The definition of B is there to account for the uncertainty in the values that will be assigned to the attributes by the execution of (a, r). Conveniently, our implementation of the predictive oracle functions: Balanced Random Forest and Catboost are able to interpret and deal with these missing values, namely attributes whose value is unknown. This example solution tells us that we take the activity a_5 in the next step of the process and have the resource r_1 to perform this activity, the

expected KPI value of this trace will go from 10 days to just 7 days.

4.2 Generating Recommendations

Let us define few notations that will be used. For a given σ which may or may not belong to the event log L, let σ_{trunc} be a running trace where $\sigma_{trunc} \sqsubset prefix(\sigma)$. Let $(a_{default}, r_{default})$ be the default activity and resource that will be followed if no intervention is done by process owners, which means no recommendation was followed. $c_1 \ldots c_z$ are different counterfactual examples (CFEs) where c_i can be expanded as $(\sigma_{trunc} \sqsubset (a_m, r_n))$. a_m and r_n are different activities and resources that are followed in this CFE as also denoted by $\sqsubset$. The use of different letters m and n is so that a relationship between an activity and resource is not assumed.

In the prescriptive oracle function we start by first computing the vector of the running trace σ_{trunc} using the trace-to-instance function ρ_L.

Then we use this vector as input to our prescriptive oracle function P_K. Internally the prescriptive oracle function first uses the Counterfactual function C_K to generate a power set $\{(c_1, \Phi_K(c_1)), \ldots, (c_z, \Phi_K(c_z)\}$ of traces containing recommendations and corresponding predicted KPI value if that recommendation is followed. The Counterfactual function uses the predictive oracle function Φ_K to generate these predicted KPI values. The KPI values in the power set may or may not be better then the KPI value $\Phi_K(\sigma_{trunc} \sqsubset (a_{default}, r_{default}))$ which we get if we do not follow any recommendation. Additionally, the counterfactuals $c_i = (\sigma_{trunc} \sqsubset (a_m, r_n))$ may be an invalid recommendation, that is, it may not even be possible to perform the activity a_i after the last activity in σ_{trunc} or the resource r_i suggested to perform the activity cannot possibly perform this activity.

But this is not an issue, as at this stage the objective (of the counterfactual function) is to generate solutions that optimize the KPI, given the properties of proximity, diversity, sparsity, and user constraints. For example, in the case of KPI K_{total}, the objective is to make a suggestion that reduces the total execution time of the process, as a result of following that suggestion.

To handle this issue our framework uses the "transition system" TS_L. The transition system validates the next activity suggested in the generated counterfactuals. If the activity is valid, the CFE is kept; otherwise, it is discarded from the power set.

Following this step the activity and resource pairs are validated using "activity-resource validation" function Ω_L. This function as explained in definition 8, is built using the event log L. The function checks if the activity and resource pair that is recommended exists in the event log or not. If it does not exists that CFE is discarded from the set.

Next, we want to return the counterfactual or suggestion that optimizes the KPI value the most from the different options in the power set. To explain the next step, we need to instantiate the $\square_K$ operator. Let us assume, without loss of generality, that the $\square_K$ operator is equal to $<$, that is, aiming to decrease the KPI value, a similar discussion could be carried out if $\square_K$ is equal to $>$. Let us also assume that after the previous step there are k counterfactuals left in the set. So $\square i \ \square k \ s.t. \Phi_K(c_i) < \Phi_K(c_j)$ where $i = j$.

Finally, this counterfactual $c_i = (\sigma_{trunc} \square(a_m, r_n))$ is returned as the valid recommendation by the prescriptive oracle function $P_K(\sigma_{trunc})$.

4.3 Implementation of the Prescriptive Framework

Tools used for Implementation

The proposed prescriptive analysis methodology was implemented in *Python* *, an open-source, high-level programming language well known for its simplicity, ease of learning, and vast array of libraries, making it a popular choice for machine learning and data analysis applications. Python provides a variety of robust libraries that have been utilized in this study to implement the methodology, such as *Scikit-learn* †, *Imbalanced-learn* ‡, *dice_ml* §, *Pandas* ¶, *XGBoost* ‖, and *PyTorch* **. The approach proposed by Padella et al. [2], with which we compared our results, is also implemented in Python and uses the library *Catboost* ††.

Scikit-learn

Scikit-learn is one of the most widely used Python libraries for machine learning. It is built on NumPy, SciPy, and matplotlib, offering simple and eficient tools for data mining and data analysis. In our implementation, scikit-learn was used for tasks such as model training, evaluation, and data preprocessing.

We used the *Pipeline* functionality from scikit-learn. Pipeline sequentially applies a list of transformations to the provided dataset. It can also chain these transformation functions with

*Python: https://www.python.org/
†Scikit-learn library, machine learning in Python. https://scikit-learn.org/stable/
‡https://imbalanced-learn.org/stable/index.html
§https://interpret.ml/DiCE/
¶https://pandas.pydata.org/
‖https://xgboost.readthedocs.io/en/stable/python/python_intro.html
**https://pytorch.org/
††Catboost: https://catboost.ai/

a machine learning model. In this configuration, it automatically preprocesses the data before passing it to the machine learning model.

Imbalanced-learn

The imbalanced-learn library, is an extension to the scikit-learn machine learning library in Python. It provides a variety of methods and algorithms to handle imbalanced datasets, where the number of instances across different classes is disproportionately distributed. Such imbalances can lead to biased or inaccurate models. Imbalanced-learn specifically addresses this issue by offering re-sampling techniques, which can either under-sample the majority class, over-sample the minority class, or apply a combination of both to create a more balanced dataset.

In our implementation we used the ”Balanced Random Forest Classifier”, which is a variation of the standard Random Forest classifier and is designed specifically to handle imbalanced datasets. It works by applying different random under-sampling techniques to balance the various classes in the training dataset before fitting the original Random Forest algorithm. This means that for each tree in the ensemble, a balanced bootstrap sample is drawn from the original data. In other words, each tree is trained on a subset of data that has been resampled to have a balanced distribution of the classes.

CatBoost

CatBoost is an open-source machine learning library that provides a gradient boosting framework, developed by Yandex. It is designed to eficiently handle categorical features and is known for its superior performance. CatBoost is also known for its high speed, both in training and inference phases. It offers support for multi-threaded training, GPU acceleration, and eficient implementation of the gradient boosting algorithm, making it a preferred choice for applications that demand high computational eficiency.

Apart from its technical prowess, CatBoost is user-friendly and easily integrable into various data processing pipelines. It provides APIs for popular programming languages like Python, R, and others, and is compatible with many data formats, making it accessible for a wide range of users, from data scientists to application developers. In the context of this study, CatBoost was leveraged for predictive modeling.

XGBoost

XGBoost, which stands for eXtreme Gradient Boosting, is an advanced and optimized implementation of gradient boosting algorithms, tailored for eficiency, flexibility, and high performance. Developed by Tianqi Chen, XGBoost is part of the Distributed Machine Learning Community (DMLC) project.

One of the key attributes of XGBoost is its optimization for computational performance and resource consumption. The algorithm is designed to be highly eficient, both in terms of memory usage and speed, which is achieved through several advanced techniques such as parallel and distributed computing, eficient tree pruning, and hardware optimization. These optimizations allow XGBoost to run significantly faster than traditional gradient-boosting methods.

Furthermore, XGBoost offers support for various objective functions and evaluation criteria, making it highly versatile for different kinds of predictive modeling tasks, including regression, classification, and ranking. The flexibility of the framework allows it to be applied to a wide array of industries and problems, from financial credit scoring to healthcare diagnostics.

Another significant advantage of XGBoost is its cross-platform and cross-language support. It offers interfaces for several programming languages, including Python, R, Java, and Scala, and can be integrated into various data science pipelines and platforms.

In this study, we used XGBoost to evaluate the results produced by the different approaches involved.

dice_ml

dice_ml is a Python library and an open source implementation of the DiCE algorithm discussed in paper [5]. It generates diverse counterfactuals guided by user-specified inputs. In our implementation, dice_ml is a core part of the prescriptive oracle function and was primarily used for the generation of counterfactuals.

It implements two kinds of methods: model-agnostic and gradient-based.

Model-Agnostic: These methods apply to any black-box regressor or classifier. They are based on sampling nearby points to an input point, while optimizing a loss function based on proximity (and optionally sparsity, diversity, and feasibility). These methods currently only support scikit-learn models, hence why we used scikit-learn Random Forest. The supported sampling strategies are:

- “Random” - Randomized Search

- “Genetic” - Genetic Search
- “Kd_tree” - KD Tree Search (counterfactuals are instances from the given data)

Gradient-Based: These methods apply to differentiable models, such as those returned by deep learning libraries like tensorflow and pytorch. They are based on an explicit loss minimization based on proximity, diversity, and feasibility. Currently, this method only works for classifier models.

Pandas

Pandas is a widely used Python library that provides high-performance, easy-to-use data structures and data analysis tools. A key feature of Pandas is its DataFrame object, a two-dimensional table of heterogeneous data, similar to a spreadsheet or SQL table. DataFrames make it easy to manipulate data by columns which can have different types (e.g., integer, float, string, and boolean). Dataframes also have powerful and flexible methods for slicing, filtering, and aggregating data.

In this study, Pandas played a crucial role in managing, preprocessing, and visualizing the data during our implementation. Its flexible data-manipulation capabilities enabled the eficient handling of complex operations necessary for our research.

PyTorch

PyTorch is an open-source machine learning library based on the Torch library. It is known for providing two of the most critical features needed in machine learning, tensor computations with strong GPU acceleration support and deep neural networks built on a tape-based autograd system. PyTorch was utilized to implement a neural network that was also used as a predictive oracle function for one of the experiments.

The integration of these various libraries was crucial in the implementation of our prescriptive analysis framework. By leveraging the functionalities provided by scikit-learn, catboost, dice_ml, and PyTorch, we were able to design and execute an eficient and effective prescriptive analysis model capable of providing actionable recommendations to improve business processes.

5

Case Study: VINST (Volvo IT Belgium)

In this chapter, we present the first real-life case study on which we worked to test our prescriptive framework.

5.1 Introduction of the Dataset

Every year Business Process Intelligence Workshop hosts a challenge called Business Process Intelligence Challenge (BPI Challenge). The goal of this challenge is twofold. On the one hand, the challenge allows researchers and practitioners in the field to show their analytical capabilities to a broader audience. On the other hand, the challenge (and its data) allows researchers to demonstrate that their techniques work on real-life data sets. For us, this was a motivating reason for using the data from this challenge.

For the Third International BPI Challenge 2013, a collection of real-life event logs from Volvo IT Belgium was presented. The logs provided contain events from an incident and problem handling system called **VINST** *

The primary goal of the incident management process is to restore normal service operation of a customer as quickly as possible when incidents arise, ensuring that the best possible levels of service quality and availability are maintained. The problem management system includes the activities required to diagnose the root cause(s) of incidents and to secure the resolution of

*Dataset: https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_incidents/12693914

those problems to enhance the quality of IT services delivered and/or operated by Volvo IT.

So, a system like this requires that the execution time of a process be as low as possible. Thus, for this event log, we will use our prescriptive analytics framework to reduce the execution time of processes without knowing much about the domain.

5.2 Structure of the Event Data

From here onward, we will refer to this dataset as *VINST* or *VINST event log*. Also in the context of an event log, a trace may be referred to as a *case*, because these instances are referred to as case in a business environment, so we will use trace and case interchangeably.

For this case study, we first isolated only the completed traces from the event log. We ended up with 7, 554 completed traces and 65, 533 events from the VINST event log. The timestamp of the events are in the range of 31^{st} March 2010 (2010-03-31) to 22^{nd} May 2012 (2012-05-22). Figure 2.2 shows a screenshot of the event log, and Table 5.1 shows some useful statistics on the event log that are relevant for our use case. The symbol "#" in the table denotes the word "number". Also, note that in the first row the "# of Attributes" includes the process activity and the process resource.
This first row is meant to convey how many total columns the dataset has.

Statistics	Values
# of Attributes	11
# of Unique Activities	13
# of Unique Resources	649
# of events	65,533
# of traces (cases)	7,553

Table 5.1: Thtis ttable shows sttatttistttics aboutt tthe VINST eventt log. The symbol "#" denottes tthe word "number", tthus tthe fftirstt row reads as: "number off attttrtibuttes". Here tthe attttrtibuttes tinclude tthe process actttivtitty and process resource.

5.3 Event Log Preprocessing

Before this data can be used, it needs some cleaning and preprocessing. we used the pandas library to first perform exploratory data analysis to check for any obvious errors or problems.

5.3.1 Data Clearning

One issue we found was that there were cases with very few events. To explain this, let L be the VINST event log, and trace $\sigma \in L$. The problematic traces had $|\sigma| = 2$. This is an issue because with 2 events there is nothing to predict. On further inspection, it was discovered that these were erroneous instances. Therefore, we decided to remove them, as including them in our training data would add noise.

We found 10 such cases; removing them left us with 7, 543 traces in the VINST event log.

5.3.2 Trace to instance

Event log data are inherently sequential and require models that can deal with sequential data, such as time series models from the classical machine learning domain and (Recurrent Neural Networks) RNNs from the deep learning domain. But experiments by [3] show that even classical models like decision trees can work very well on event log data, given that we convert it from a sequential nature to a non-sequential nature.

For this purpose, we developed our trace-to-instace encoding function $\rho_L : E^* \rightarrow X_1 \times \ldots \times X_m$ which not only converts a trace σ to a vector $\rho_L(\sigma) \in X_1 \times \ldots \times X_m$ but also encodes the history of prefixes making the trace a non-sequential vector, as described in definition 4.

The implementation of ρ involves two steps. For the first step we want to convert the sequential event log data to non-sequential form. In each event in a trace, we need to add the prefix history to make that event in the event log self-contained and thus non-sequential. For this we have developed a custom Python function that uses frequency-vector encoding to accomplish this. Table 2.2 shows an example of this in action.

Next, we need to convert the trace with categorical data into an instance that Machine Learning (ML) models can use. Also, Machine learning models work best when the numerical data is normalized. For this we have used scikit-learn's *Pipeline* [†] functionality to one-hot encode the categorical columns and normalize the numerical columns.

5.3.3 Train-Test Split

Before we can do any analytics we first need to train our predictive oracle function and for that we need to properly split our event log L to train and test log. For this, we first extract the training log L^{comp}, which is used to train the recommender system as a whole, namely the

[†]https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html#sklearn.pipeline.Pipeline

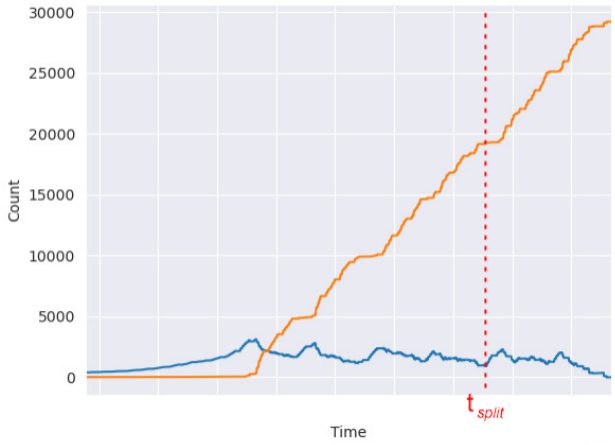

Figure 5.1: Graph showting tthe spltitt off ttratin and ttestt eventt log datta. The orange and blue ltines representt tthe number off completted and actttive ttraces, respectttively. The vertttical red bar tis tthe tttimesttamp tt where we sett tthe cutt-offff ffor tthe completted ttraces. [1]

predictive oracle function, the transition system, and the activity-resource validation function. We also use this to train our evaluation oracle function. Then we create the test log L^{run} of the running cases on which the system is evaluated.

- L^{comp} $\square L$ - Training log
- L^{run} $\square L \setminus L^{comp}$ - Test log

To extract the training log L^{comp} we compute the earliest time t_{split} at which around 81% of the traces of L are completed, see, a visual example of this in Figure 5.1.

This allows us to define L^{comp} as the set of traces of L completed at time t_{split}, and consequently, define L_{run} as $L \setminus L^{comp}$.

The traces of L_{run} are then truncated to a set L_{trunc}, by maintaining only a random percentage of events in each trace [‡], this has been done to simulate running instances for which our prescriptive algorithm is going to which provide recommendations. Finally we'll use the set L_{run} for the evaluation of the results.

[‡]The random percentage was drawn from a uniform distribution $U[25, 75]$, repeating the experiment for its stochastic validity.

VINST Event Log Train-Test Split

For this chapter L refers to the VINST event log we have after preprocessing the base event log, meaning traces $|\sigma| = 2$ are not included. Table 5.2 shows the statistics of the train-test split on VINST event log. In this chapter 5 we assume that the symbols used for the event data in Table 5.2 will specifically denote the VINST event log data.

Description	Symbol	Value
Dataset with completed traces	L	7,543
Train set data	L^{comp}	6,166
Test set data	L^{run}	1,377
Test set with cut traces	L^{trunc}	1,377

Table 5.2: Shows VINST eventt log ttratin-ttestt spltitt sttatttistttics.

5.4 Evaluation

To address our second research question **"How does our proposed prescriptive analysis framework perform compared to existing methods such as the one proposed by Padella et al. [2], when evaluated on real-life datasets?"** we need an effective evaluation methodology.

To evaluate our prescriptive analytics framework, there were few options. We evaluate the quality of the counterfactuals produced by our counterfactual function C. A challenge with this approach is that, in the literature, currently there are no standard protocols to evaluate counterfactuals [13]. We could have taken the same route as [4] and tried to evaluate counterfactual examples (CFEs) through its qualities of diversity, sparsity, and proximity. This would mean evaluating not only the best recommendation in the CFEs but also the other valid recommendations. We decided against this option because the final output of our prescriptive oracle function is a single recommendation, so this method would not be able to assess the effectiveness of that recommendation.

Before we can explain our methodology, we need to establish a few points and then in section 5.4.2 we explain our evaluation methodology.

5.4.1 Objective

In this thesis, we wanted to evaluate the use of counterfactuals for prescriptive analytics. The literature is still experimenting with the use of counterfactuals in process mining in different

ways. This thesis is one such attempt to evaluate the performance of counterfactuals for recommending the next-best activity and resource. Our aim is to provide a domain-agnostic prescriptive analytics method, that means the framework does not require encoding the domain information in some way before the method can work.

To measure the performance of our method, we first need to instantiate the KPI which we will optimize. And the success of our method depends on how much our approach can improve this KPI value.

KPI Total Time

Since we want to reduce the execution time of our traces, we will use the *total time* definition of the KPI and define our KPI function as $K = K_{total}$. Given a (prefix of a) trace σ the KPI function $K(\sigma)$ returns the value of $time(e_n) - time(e_1)$. It is important to note that this time calculation is done posteriori. Therefore, we will not be able to calculate this value for a trace that is still running.

5.4.2 Evaluation Methodology

To evaluate the framework, we will use the approximations provided by our *evaluation oracle function* Λ_K. Similar to [9], we will perform a quantitative evaluation in which we aggregated the results of the KPI optimization on multiple traces. Note that the results are an approximation of the KPI optimization by the evaluation oracle function Λ_K, which may not fully reflect the reality. But we learned from [39] that machine learning algorithms are capable of providing accurate predictions. Given this empirical evidence, we decided that this is a reliable enough approach.

We calculate a final aggregate KPI value for each proposed method and compare them. A lower sum of KPI values would mean that the approach has produced good recommendations, thus reducing the total execution time of traces.

A benefit of this evaluation method is that, like our framework, it is domain-agnostic as well. If you can create a predictive oracle function as described in definition 4, which is the same as our evaluation oracle function, then our evaluation method can also be applied.

The following are the different approaches for which we will compare the results.

- "No recommendations" - This denotes the absence of recommendations. For this we

will simply aggregate the KPI values as:

$$\sum_{i=1}^{|L^{run}|} K(\sigma_i^{run}), \text{ where } \sigma^{run} \in L^{run}$$

- "RAR" - This is the Resource aware recommendations (RAR) approach proposed by padella et al. [2]. We take the recommendations (a, r) produced for each trace σ^{trunc} in the test log L^{trunc} and use our predictive oracle function to calculate the KPI value as:

$$\sum_{i=1}^{|L^{trunc}|} \Lambda_K(\sigma_i^{trunc} \in (a, r)), \text{ where } \sigma^{trunc} \in L^{trunc} \tag{5.1}$$

- Finally our approach. DiCE has a few different algorithms that generate counterfactuals differently, and we have tested all of them. The calculation of the final figure is the same as shown above in Equation 5.1.

This methodology of calculating the sum of KPI values can also be seen in Figure 5.2.

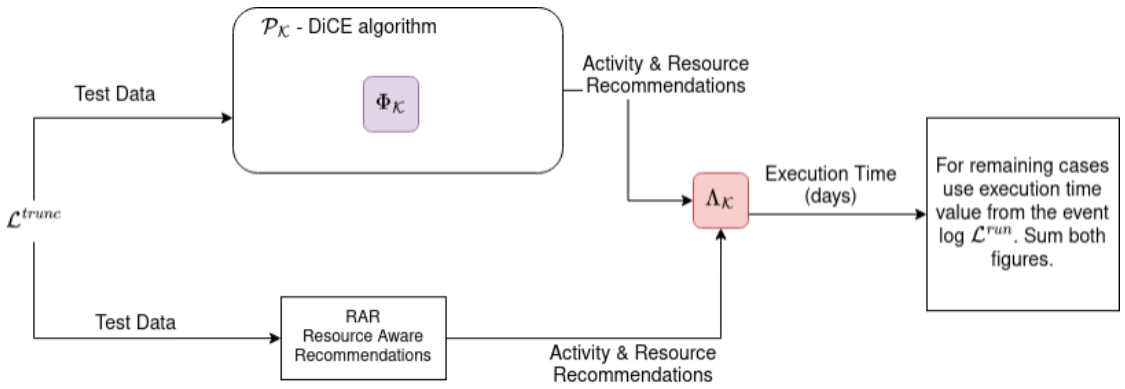

Figure 5.2: Block diagram of the experiment setup. It shows how the data is fed to different algorithms that we used and how we obtain the final numerical output.

Target Generation

As discussed, to use prescriptive frameworks, we need the predictive oracle Φ and to evaluate different prescriptive frameworks, we need the evaluation oracle Λ. But first we need to train these with the right data. In this section, we discuss part of the VINST event log preprocessing that involves generating the *target* values, "y" in the context of machine learning.

Let $\sigma^{comp} \square L^{comp}$ and $\sigma^{p} \square prefix(\sigma^{trunc})$; to create the dataset D we generate pairs of (x, y) where $x = \rho_L(\sigma^{p})$ and $y = K(\sigma^{comp})$. Table 2.1 shows an example of an instance in the dataset.

For the implementation of target generation, pandas library was used, and a custom Python function was written that generates these "target" values and associates them with the respective row. The algorithm to do this is as follows:

- Use pandas "groupby" to iterate over each trace in the CSV
- take the timestamp in the last row (or event) of the trace and subtract that with the first timestamp.
- associate this target value with all the rows of the trace.

We now have a dataset D that can be used to train the predictive oracle function Φ_K to predict the KPI values of a running trace $\sigma^{trunc} \square L^{trunc}$. We will also use the same dataset to train our evaluation oracle function Λ_K.

5.4.3 Setup

We carried out our tests using the Model-Agnostic method, from the dice_ml library. This is because currently the dice_ml Python library (an implementation of the DiCE algorithm [5]) does not support the gradient-based method for regression models.

In Model-agnostic method there are three sampling strategies available: "Random Search", "Kd-tree Search" and "Genetic Search". We performed experiments using "Random Search" and "Genetic Search" but not "Kd-tree Search". This is because the "Kd-tree Search" method tries to find the solution from the train set, and if there does not exist an exact solution in the train set, it is unable to find any CFE. In our experiments, it was producing CFEs for less than 5 cases out of 1,377, so we decided not to use this method.

5.4.3.1 Predictive oracle function implementation

For the predictive oracle function Φ_K we choose the Catboost Regressor [40]. It is a high-performance open source framework for gradient boosting on decision trees, which has been shown to perform well on event log data [1]. The model was parameter-tuned before usage and achieved a mean absolute error (MAE) of around 10.4 days. The parameter tuning process for determining the best parameters is described in Appendix A.1

Next, we use scikit-learn's pipeline function to chain this model with the transformers that convert the input trace to a vector which the Catboost Regressor can work with.

Parameter	Value
Learning Rate	0.01
Iterations	3000
Max Depth	8
Early Stopping Rounds	5

Table 5.3: Cattboostt Regressor ttratinting parametters

Table 5.3 shows the training parameters used by the Catboost regression model.

5.4.3.2 Evaluation oracle function implementation

For the evaluation oracle function Λ_K we choose the XGBoost Regressor, which is also based on decision trees and thus is also expected to perform well on event log data just like the Random Forest model. The model was parameter-tuned before usage and achieved a mean absolute error (MAE) of around 10.89 days. The parameter tuning process for determining the best parameters is described in Appendix A.2

Similarly, we use scikit-learn's pipeline function to chain this model with the transformers that convert the input trace to a vector which the XGBoost can work with.

Parameter	Value
Eta	0.01
Gamma	0.5
Subsample	0.8
Colsample Bytree	0.5
Max Depth	128
Min Child Weight	1
Number of Estimators	350

Table 5.4: XGBoostt Regressor ttratinting parametters

Table 5.4 shows the training parameters used by the XGBoost Regressor model.

5.4.3.3 Transition System

A custom Python algorithm was developed to implement the Transition System functionality. Hash-map (or dictionary in Python) was used to store all the possible prefixes of an activity.

The prefixes are stored as keys of the dictionary and the value of this key contains the list of next possible paths. An example of dictionary data structure used in the transition system can be: {“act3, act5”: [act2, act6], ..., “act3, act5, act2”: [act1, act7] }. For this internal graph, the transition system would consider ”act2” as a valid activity after the prefix { “act3, act5” }, but “act7” would be considered as an invalid activity.

We use the VINST event log L^{comp} to build this functionality.

5.4.3.4 Activity-Resource Validation Function

A custom Python algorithm was developed to implement the activity-resource validation function. A Python set was used as the main data structure for this implementation. Python set has this property just like a mathematical set that duplicates cannot exists in it and it is also unordered. Thus, we add all the combinations from the VINST event log L^{comp} to this set and this data structure automatically removes the duplicates, making it an eficient choice for this kind of requirement.

When we need to validate an activity and resource combination we just search this set. If we find it inside the Python set the function returns 1; otherwise it returns 0

5.5 Results

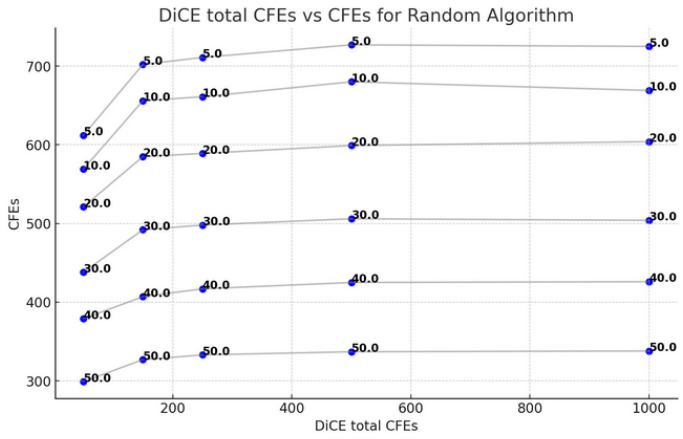

Figure 5.3: Scatttter plott showting tthe relatttionshtip bettween DtiCE parametter ttottal-CFEs and tthe valtid CFEs produced. Labels on tthe potintts tis tthe KPI optttimtizting tthreshold value used, and potintts wtitth tthe stimtilar values are connectted.

We start by analyzing the data produced by the DiCE algorithm for the KPI *total time* K_{total}. Experiments were carried out with different combinations of parameters and many times for

statistical significance. The Figure 5.3 presents results of running the DiCE algorithm with "Random" sampling strategy. The Y-axis shows the count of valid CFEs and the X-axis shows the value of parameter "DiCE total CFEs". The labels show the percentage by which the DiCE algorithm had to optimize the KPI. A line is used to connect all points representing the same KPI optimization percentage. We can see that there is an increasing relation between the parameter "DiCE total CFEs" and the CFEs produced. However, increasing this parameter makes sense up to a certain point, after which we see no increase in the CFE count.

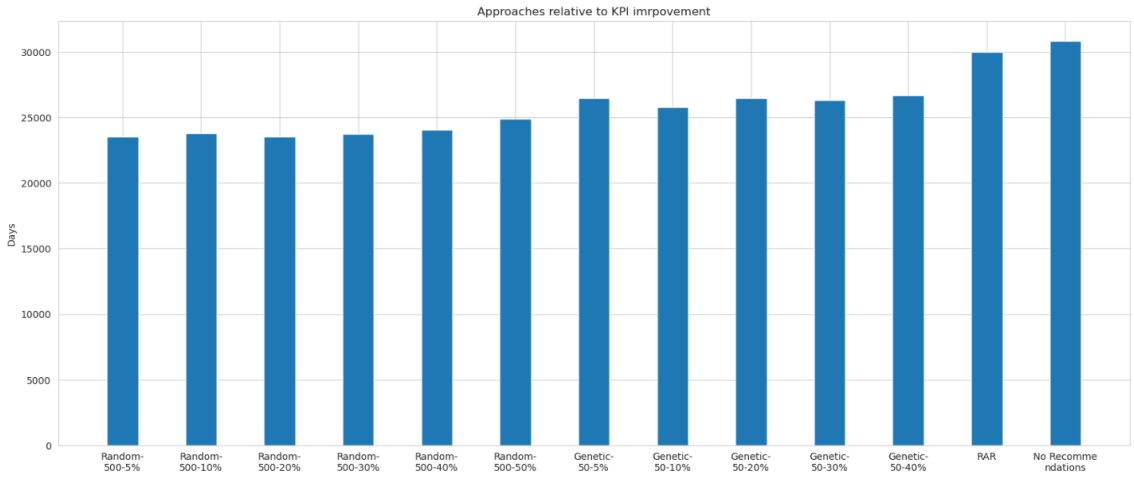

Figure 5.4: Bar chartt showting tttime titt ttakes tin days ffor all tthe cases tin L_r^{un} ffollowting each approach. DtiCE approaches have metthod name tin sttartt e.g. "Random" or "Genetttic" ffollowed by tthe value off "DtiCE ttottal CFEs" parametter ffollowed by tthe KPI timprovementt percenttage tthe algortitthm was requtired tto achtieve on each case. "RAR" tis tthe Resource Aware Recommendatttions algortitthm and "No recommendatttions" (lastt bar ffrom tthe lefttt) tis tthe base case tthatt representts tthe absence off tintterventttion.

Figures 5.4 and 5.5 show the results of the comparison of different approaches. For convenience, the results are also presented in table form at 5.5 to see the exact values.

Figure 5.4 is a bar chart that shows the total time, in days, it takes to execute all 1, 377 cases in L^{trunc} following the recommendations from each approach. The DiCE approaches are prefixed with the method name "Random" and "Genetic", followed by the parameter value of "DiCE total CFEs", and finally the KPI improvement threshold used to run the algorithm. "RAR" (Resource Aware Recommendations), is the algorithm that was proposed by Padella et al. [2]. "No recommendations" (last bar from the left) is the base case; indicating that no
recommendation was followed and that the original activity and resource, found in L^{run}, were chosen.

We observe that the most time is taken by "No recommendations" taking 30, 809 days and

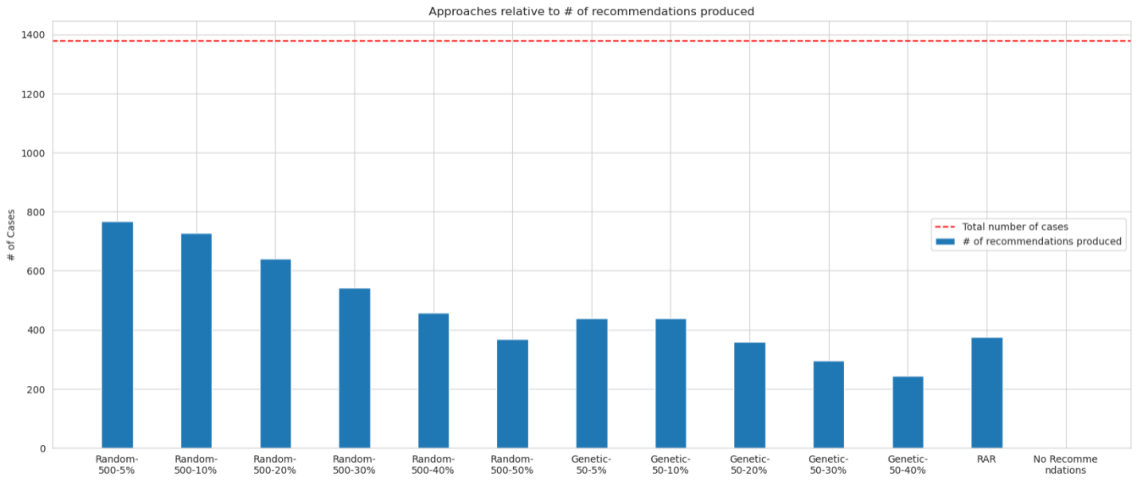

Figure 5.5: Bar chartt showting tthe number off valtid recommendatttions produced ffor each case tin L^{trunc}, by each approach. DtiCE approaches sttartt wtitth metthod name "Random" and "Genetttic" ffollowed by tthe value off "DtiCE ttottal CFEs" parametter ffollowed by tthe KPI timprovementt percenttage tthe algortitthm was requtired tto achtieve on each case. "RAR" tis tthe Resource Aware Recommendatttions algortitthm and "No recommendatttions" (tthe mtissting bar) tis tthe base case whtich representts lack off any valtid recommendatttion.

the least time is taken by "Random-500-5%" taking 23, 520 days. RAR recommendations take 29, 986 days.

Figure 5.5 is a bar chart that represents how many valid recommendations were made by each approach, for all the cases in $\updownarrow^{trunc}$. The DiCE approaches are prefixed with the method name "Random" and "Genetic", followed by the parameter value of "DiCE total CFEs", and finally the KPI improvement threshold used to run the algorithm. "RAR" (Resource Aware Recom-mendations), is the algorithm that was proposed by Padella et al. [2]. "No recommendations" (the missing bar) is the base case that represents the lack of any valid recommendation.

We observe that most recommendations are also produced by "Random-500-5%" with 727 recommendations. RAR produced a recommendation for 375 cases.

In Table 5.5 the first column lists the different approaches or algorithms. The second and third columns list the parameters selected for the DiCE algorithm. The fourth column lists the aggregate time of all cases, in L^{run}, in unit: "days". It should be noted that, except for the first entry, which was calculated from the event log, all other entries are predicted using the evaluation oracle function Λ. The fifth column lists the number of recommendations found when using the respective approach.

"No recommendations" (in the first row) is the base case; indicating that no recommendation was followed and that the original activity and resource, found in L^{run}, were chosen. Thus,

Approach	DiCE total CFEs	KPI optimizing threshold	days	# of Cases
No recommendations	-	-	30,809	0
RAR	-	-	29,986	375
Random	500	5%	23,520	727
Random	500	10%	23,804	680
Random	500	20%	23,557	599
Random	500	30%	23,739	506
Random	500	40%	24,072	425
Random	500	50%	24,873	337
Genetic	50	5%	26,489	464
Genetic	50	10%	25,799	406
Genetic	50	20%	26,477	368
Genetic	50	30%	26,323	297
Genetic	50	40%	26,676	257

Table 5.5: Table shows tthe numertic values used tto plott tthe bar chatts 5.4 and 5.5. Approach "No recommendatttions" tis tthe baseltine. Approach "RAR" (Resource Aware Recommender) tis tthe approach proposed by [2]. "Random" and "Genetttic" representts runs off DtiCE algortitthm. Second and tthtird column have confftiguratttion off DtiCE parametters. ffortth column has predtictttion off ttottal executttion tttime tiff recommendatttion ffrom tthe respectted approaches tis ffollowed and fftiffftth column enumerattes tthe number off cases ffor whtich a recommendatttion was produced.

the execution time could be calculated using the event log. The second entry "RAR" (Resource Aware Recommendations), is the algorithm that was proposed by Padella et al. [2]. The following are the entries produced by the DiCE algorithm, using different sampling strategies and parameters.

The computational cost of running an algorithm is very important to consider when we talk about the practicality of an algorithm. For this reason we also analyzed the running time of the different DiCE approaches. Figure 5.6 shows the comparison of time it takes to run the cases in the dataset for each DiCE approach. The box plot shows that for the Random sampling strategy is very quick to run and has a very similar box plot except for the last 2 approaches. However, the Genetic sampling strategy takes significantly more time but have very similar box plots for all the configuration of the experiments.

Analysis and Interpretation

In this section, the results presented in Section 5.5 are discussed and analyzed. Discussion also includes one of the research questions of this investigation.

Figure 5.3 shows a very important feature of the algorithm that allows it to generate good

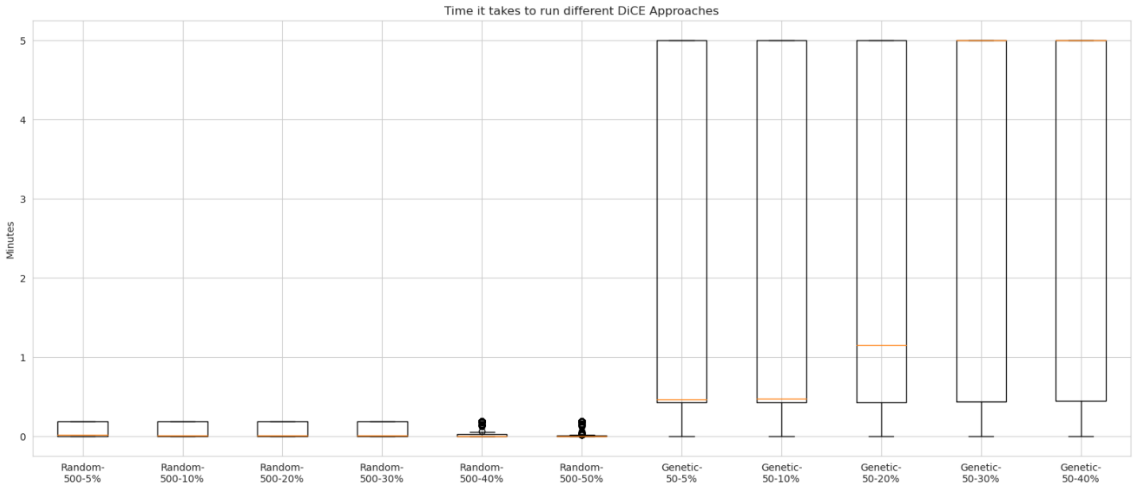

Figure 5.6: Box plott shows a compartison off tthe tttime titt ttakes tto run tthe cases tin tthe dattasett ffor each DtiCE approach. The orange ltine tin tthe boxes shows tthe medtian off tthe tintterquartttile range. All tthe expertimentts were sttopped afftter 5 mtinuttes tiff a recommendatttion was nott produced.

final recommendations without even having domain knowledge. What we see is that, to compensate for the lack of domain knowledge, our prescriptive analytic function generates a lot of counterfactual examples (CFEs), thus increasing the chances that one of the CFE would be very good. Therefore, to find a counterfactual for a very complex problem, users can try increasing the value of "DiCE total CFEs" pushing the algorithm harder to find a solution.

An interesting find is that the KPI K is most optimized with the approach "Random-500-5%". We see the explanation of this comes from the Figure **??**. We see that for this approach we have the most recommendations produced, and empirically we see that many cases optimized a little produce the best results. For the approaches which optimizes a single case to a greater degree might save more time on that single case, but that increased dificulty means that the algorithm is not able to have a valid recommendation for many more cases, therefore the overall execution time is reduced.

Another effect of this optimization threshold is on the computational time of the algorithm which visible in the box plot (Figure 5.6) that shows the running time of the experiments. In case of Random sampling strategy we see very similar box plots expect for the approaches "Random-500-40%" and "Random-500-50%". In these, the interquartile range box is collapsed towards 0 minutes and few values for which a recommendation was found, took more time and are shown as outliers. This skew happens because for very dificult cases the algorithm is not able to find any CFEs, therefore, it doesn't do the post-processing steps or any of the validation steps required which eventually reduces the execution time of the algorithm.

For the Genetic sampling strategy we see the orange line which shows the median of the interquartile range takes more and more time as the optimization threshold value is increased.

How does our proposed prescriptive analysis framework perform compared to existing methods such as the one proposed by Padella et al. [2], when evaluated on real-life datasets?

We see that the RAR algorithm is not as performant as the DiCE algorithm. This may be due, as discussed previously, to the stricter operating conditions of the RAR algorithm. RAR algorithm takes into account the availability of resources when suggesting the resource for a given activity. This lack of availability of optimal resource and selection of a sub-optimal resource may lead to the case taking more time, and thus the results we observe.

Using the best configuration, the DiCE algorithm was able to reduce the total execution time of all the traces combined from 30, 809 days to 23, 520 days, a reduction of around 23.7%. This is a very significant gain. Another interesting thing to note is that, just by trying to improve a process (case) execution time by 5 percent, over so many cases results in such a significant gain in overall time.

Given the results, we can safely conclude that our method performs commendably compared to existing methods like [2], but it requires more relaxed operating assumptions.

6

Case Study: Bank Account Closure (BAC)

In this chapter, we present the second real-life case study on which we worked to test our prescriptive framework.

6.1 Introduction of the Dataset

The real-life event log comes from an Italian bank institution that deals with the closure of bank accounts. This event log is called **Bank Account Closure (BAC)**. Each trace is associated with an attribute, Closure Type, which encodes the type of procedure that is performed for the specific account holder, and the Closure Reason, namely the reason that triggers the closure request.

As this is a bank account closure event log, the business objective here is to prevent bank account closure as much as possible. Therefore, this is an ideal case study for working to optimize a KPI that deals with preventing a bad outcome from a trace. Later we will formally define this KPI.

For this event log, our aim is to use our prescriptive analytics framework to avoid the occurrence of problematic activity *"Back-Ofice Adjustment Requested"*.

6.2 Structure of the Event Data

From here onward, we will refer to this dataset as *BAC* or *BAC event log*. Also in the context of an event log, a trace may be referred to as a case because these instances are referred to as case in a business environment, so we will use trace and case interchangeably.

For this case study, we first isolated only the completed traces from the event log and ended up with 32, 429 completed traces and 212, 721 events from the BAC event log. The timestamps of the events in the event log are from 29^{th} May 2017 (2017-05-29) to 6^{th} March 2019 (2019-03-06).

Figure 6.1 shows a screenshot of the event log, and Table 6.1 shows some useful statistics about the event log that are relevant for our use case. The symbol "#" in the table denotes the word "number". Also, note that in the first row, the "# of Attributes" includes the process activity and the process resource. This first row is meant to convey the number of columns in the dataset.

REQUEST_ID	CLOSURE_TYPE	CLOSURE_REASON	ACTIVITY	END_DATE	START_DATE	CE_UO	ROLE
20175000168	Client Recess	1 - Client lost	Service closure Request with network responsibility	1539175786000	1539175692000	44	APPLICANT
20175000168	Client Recess	1 - Client lost	Service closure Request with BO responsibility	1539602104000	1539175786000	BOC	BACK-OFFICE
20175000168	Client Recess	1 - Client lost	Pending Request for Reservation Closure	1539602345000	1539602104000	BOC	BACK-OFFICE
20175000168	Client Recess	1 - Client lost	Pending Liquidation Request	1539745391000	1539602345000	BOC	BACK-OFFICE
20175000168	Client Recess	1 - Client lost	Request completed with account closure	1539745391000	1539745391000	BOC	BACK-OFFICE
20175000642	Bank Recess	1 - Client lost	Request created	1541672852000	1541672782000	624	APPLICANT
20175000642	Bank Recess	1 - Client lost	Service closure Request with network responsibility	1541672917000	1541672852000		APPLICANT
20175000642	Bank Recess	1 - Client lost	Authorization Requested	1541673010000	1541672917000	186	DIRECTOR
20175000642	Bank Recess	1 - Client lost	Service closure Request with BO responsibility	1542125449000	1541673010000	BOC	BACK-OFFICE
20175000642	Bank Recess	1 - Client lost	Pending Request for Reservation Closure	1542125459000	1542125449000	BOC	BACK-OFFICE
20175000642	Bank Recess	1 - Client lost	Pending Liquidation Request	1542251285000	1542125459000	BOC	BACK-OFFICE
20175000642	Bank Recess	1 - Client lost	Request completed with account closure	1542251285000	1542251285000	BOC	BACK-OFFICE
20175001550	Client Recess	1 - Client lost	Request created	1496047765000	1496047666000	336	APPLICANT
20175001550	Client Recess	1 - Client lost	Evaluating Request (NO registered letter)	1496047969000	1496047765000	356	DIRECTOR
20175001550	Client Recess	1 - Client lost	Service closure Request with network responsibility	1496048559000	1496048164000	356	APPLICANT
20175001550	Client Recess	1 - Client lost	Service closure Request with BO responsibility	1496057013000	1496048559000	BOC	BACK-OFFICE
20175001550	Client Recess	1 - Client lost	Network Adjustment Requested	1496222768000	1496057013000	356	APPLICANT
20175001550	Client Recess	1 - Client lost	Service closure Request with BO responsibility	1497257101000	1496222768000	BOC	BACK-OFFICE
20175001550	Client Recess	1 - Client lost	Pending Request for Reservation Closure	1497257226000	1497257101000	BOC	BACK-OFFICE
20175001550	Client Recess	1 - Client lost	Pending Liquidation Request	1497495862000	1497257226000	BOC	BACK-OFFICE
20175001550	Client Recess	1 - Client lost	Request completed with account closure	1497495862000	1497495862000	BOC	BACK-OFFICE
20176000338	Inheritance		Request created	1496669699000	1496668594000	198	APPLICANT
20176000338	Inheritance		Service closure Request with network responsibility	1496672448000	1496669699000		APPLICANT
20176000338	Inheritance		Authorization Requested	1497256082000	1496672448000	725	DIRECTOR
20176000338	Inheritance		Service closure Request with BO responsibility	1497435074000	1497256082000	BOC	BACK-OFFICE
20176000338	Inheritance		Network Adjustment Requested	1518181986000	1497435074000	725	APPLICANT
20176000338	Inheritance		Service closure Request with BO responsibility	1519657505000	1518181986000	BOC	BACK-OFFICE
20176000338	Inheritance		Network Adjustment Requested	1536576794000	1519657505000	725	APPLICANT
20176000338	Inheritance		Service closure Request with BO responsibility	1536655395000	1536576794000	BOC	BACK-OFFICE
20176000338	Inheritance		Pending Request for Reservation Closure	1536655410000	1536655395000	BOC	BACK-OFFICE
20176000338	Inheritance		Pending Liquidation Request	1536807842000	1536655410000	BOC	BACK-OFFICE
20176000338	Inheritance		Pending Request for acquittance of heirs	1536829056000	1536807842000	BOC	BACK-OFFICE
20176000338	Inheritance		Request completed with account closure	1536829056000	1536829056000	BOC	BACK-OFFICE

Figure 6.1: Screenshott off BAC eventt log showting 4 ttraces (or cases). The REQUEST_ID tis tthe untique case (or ttrace) tidentttifftier, "ACTIVITY" tis tthe actttivtitty column, and "CE_UO" tis tthe resource column.

Statistics	Values
# of Attributes	8
# of Unique Activities	15
# of Unique Resources	653
# of events	212721
# of traces (cases)	32429

Table 6.1: Thtis ttable shows sttatttistttics aboutt tthe BAC eventt log. The symbol "#" denottes tthe word "number", tthus tthe fftirstt row reads: "number off attttrtibuttes". Here, tthe attttrtibuttes tinclude tthe process actttivtitty and process resource.

6.3 Event Log Preprocessing

Before this data can be used, it needs to be cleaned and preprocessed. We used the pandas library to first perform exploratory data analysis to check for any obvious errors or problems. Then we transformed the data to more machine learning model friendly form.

6.3.1 Data Cleaning

One issue we found was that there were cases with very few events. To explain this, let L be the BAC event log, and trace $\sigma \in L$. The problematic traces had $|\sigma| = 2$. This is an issue because with 2 events there is nothing to predict. On further inspection, it was discovered that these were indeed erroneous instances. Therefore, we decided to remove them, as including them in our training data would add noise to it.

We found 3, 235 such cases; removing them left us with 29, 194 traces in the BAC event log.

6.3.2 Trace to instance

Event log data are inherently sequential and require models that can deal with sequential data, such as time series models from the classical machine learning domain and (Recurrent Neural Networks) RNNs from the deep learning domain. But experiments by [3] show that even classical models like decision trees can work very well on event log data, given that we convert it from a sequential form to a non-sequential form.

For this purpose, we developed our trace-to-instace encoding function $\rho_L : E^* \to X_1 \times \ldots \times X_m$ which not only converts a trace σ to a vector $\rho_L(\sigma) \in X_1 \times \ldots \times X_m$ but also encodes the history of prefixes making the trace a non-sequential vector, as described in definition 4.

The implementation of ρ involves two steps:

1. We want to convert the sequential event log data to non-sequential form. In each event in a trace, we need to add the prefix history to make that event in the event log self-contained and thus non-sequential. For this we have developed a custom Python function that uses frequency-vector encoding to accomplish this. Table 2.2 shows an example of this in action.

2. Next, we need to convert the trace with categorical data into an instance that Machine Learning (ML) models can use. Also, Machine learning models work best when the numerical data is normalized. For this we have used scikit-learn's *Pipeline* * functionality to one-hot encode the categorical columns and normalize the numerical columns.

6.3.3 Train-Test Split

Before we can do any analytics we first need to train our predictive oracle function and for that we need to properly split our event log L to train and test log. For this, we first extract the training log L^{comp}, used to train the recommender system as a whole, namely the predictive oracle function, the transition system, and the activity-resource validation function. Then we create the test log L^{run} of the running cases on which the system is evaluated.

- $L^{comp} \sqsubset L$ - Training log
- $L^{run} \sqsubset L \setminus (L^{comp} \sqsubset L^{no-act})$ - Test log

To extract the training log L^{comp} we compute the earliest time t_{split} at which around 65% of the traces of L are completed, see, a visual example of this in Figure 5.1.

This allows us to define L^{comp} as the set of traces of L completed at time t_{split}. Because we want to evaluate the effectiveness of our prescriptive analytics framework at preventing the activity-to-avoid from occurring, we will only have those traces in the test log that contain the activity-to-avoid. Let $L^{no-ata} = \sigma \sqsubset L \mid \sqsubset e \sqsubset \sigma, activity(e) =$ activity-to-avoid

Now we can define L_{run} as $L \setminus (L^{comp} \sqsubset L^{no-ata})$.

The traces of L_{run} are then truncated to a set L_{trunc}, by maintaining only a random percentage of events in each trace †, this has been done to simulate running instances for which our prescriptive algorithm is going to which provide recommendations. Finally we'll use the set L_{run} for the evaluation of the results.

*https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html#sklearn.pipeline.Pipeline

†The random percentage was drawn from a uniform distribution $U[25, 75]$, repeating the experiment for its stochastic validity.

BAC Event Log Train-Test Split

For this chapter L refers to the BAC event log we have after preprocessing the base event log, meaning traces $|\sigma| = 2$ are not included. Table 6.2 shows the statistics of the train-test split on BAC event log. The statistics show visibly how many test traces we lose when we remove the traces without the activity-to-avoid.

In this chapter 6 we assume that the symbols used for the event data in 6.2 will specifically denote the BAC event log data.

Description	Symbol	Value
Dataset with completed traces	L	21,887
Train set data	L^{comp}	21,172
Test set data	L^{run}	715
Test set with cut traces	L^{trunc}	715

Table 6.2: Shows BAC eventt log ttratin-ttestt spltitt sttatttistttics.

6.4 Evaluation

To address our second research question **"How does our proposed prescriptive analysis framework perform compared to existing methods such as the one proposed by Padella et al. [2], when evaluated on real-life datasets?"** we need an effective evaluation methodology.

To evaluate our prescriptive analytics framework, there were few options. We evaluate the quality of the counterfactuals produced by our counterfactual function C_K. A challenge with this approach is that, in the literature, currently there are no standard protocols to evaluate counterfactuals [13]. We could have taken the same route as [4] and tried to evaluate counterfactual examples (CFEs) through its qualities of diversity, sparsity, and proximity. This would mean evaluating not only the best recommendation in the CFEs but also the other valid recommendations. We decided against this option because the final output of our prescriptive oracle function is a single recommendation, so this method would not be able to assess the effectiveness of that recommendation.

Before we can explain our methodology, we need to establish a few points and then in a later section explain our evaluation methodology.

6.4.1 Objective

In this thesis, we wanted to evaluate the use of counterfactuals for prescriptive analytics. The literature is still experimenting with the use of counterfactuals in process mining in different ways. This thesis is one such attempt to evaluate the performance of counterfactuals for recommending the next-best activity and resource. Our aim is to provide a domain-agnostic prescriptive analytics method, that means the framework does not require encoding the domain information in some way before the method can work.

KPI: Activity Occurrence

Since we want to avoid the problematic activity in our traces, we will use the *Activity Occurrence* definition of the KPI and for this chapter define our KPI function as $K = K_{occur}$. But for practical and implementational reasons, we will modify the base definition slightly. Given a (prefix of a) trace σ the KPI function $K(\sigma)$ will return 1 if the undesired activity is in σ otherwise it will return 0.

This modification is almost equal to the original definition, and it will allow the optimization of this KPI to become a classification problem.

An example of this can be if we want to avoid the activity "Back-Ofice Adjustment Requested", for all elements in $prefix(\sigma)$, the KPI function $K_{occur}(\sigma)$ returns 1 if one or more activities are equal to "Back-Ofice Adjustment Requested", otherwise it returns 0.

It is important to know that this time calculation is done posteriori. Therefore, we will not be able to calculate this value for a trace that is still running.

To be general, instead of using "Back-Ofice Adjustment Requested", we will use the term *activity-to-avoid*.

6.4.2 Evaluation Methodology

To evaluate the framework, we will use the approximations provided by our *evaluation oracle function* Λ_K. Similar to [9], we will perform a quantitative evaluation in which we aggregated the results of the KPI optimization on multiple traces. Note that the results are an approximation of the KPI optimization by the evaluation oracle function Λ_K, which may not fully reflect the reality. But we learned from [39] that machine learning algorithms are capable of providing accurate predictions. Given this empirical evidence, we decided that this is a reliable enough approach.

We calculate a final aggregate KPI value for each proposed method and compare them. A higher sum of KPI values would mean that the approach has produced good recommendations. In case of this KPI definition, a higher total KPI value would translate into a higher number of traces where the activity-to-avoid was successfully avoided.

A benefit of this evaluation method is that, like our framework, it is domain-agnostic as well. If you can create a predictive oracle function as described in definition 4 then our evaluation method can also be applied.

The following are the different approaches for which we will compare the results.

- "No recommendations" - This denotes the absence of recommendations. For this we will simply set it to 0 because in the absence of recommendations the activity-to-avoid occurred.

- "RAR" - This is the Resource Aware Recommendations (RAR) approach proposed by Padella et al. [2]. We take the recommendations (a, r) produced for each trace σ^{trunc} in the test log L^{trunc} and use our predictive oracle function to calculate the KPI value as in equation 5.1.

- Finally, we have our approach. DiCE has a few different algorithms that generate counterfactuals differently, and we have tested all of them. The calculation of the final figure is the same as shown above in equation 5.1.

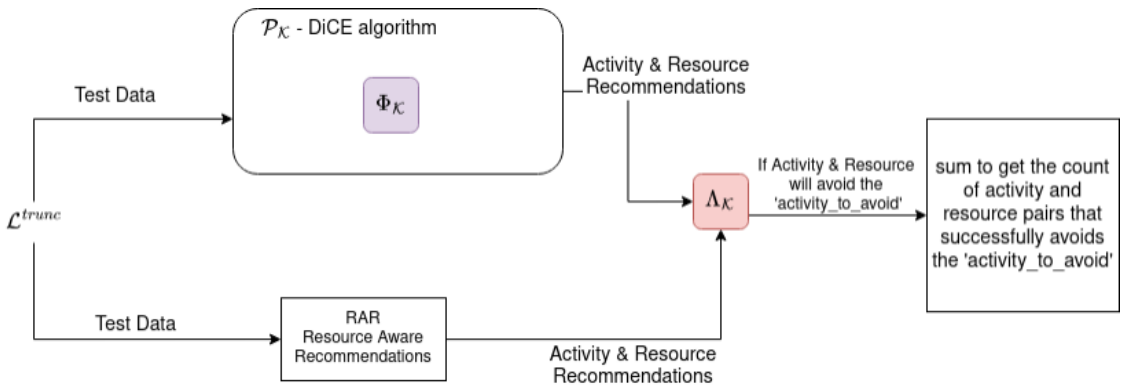

Figure 6.2: Block dtiagram off tthe expertimentt settup. Itt shows how tthe datta tis ffed tto dtifffferentt algortitthms tthatt we used and how we obttatin tthe fftinal numertical outtputt.

This methodology for calculating the sum of the KPI values can also be seen in Figure 6.2.

Label Generation

As discussed, to use prescriptive frameworks, we need the predictive oracle Φ and to evaluate different prescriptive frameworks, we need the evaluation oracle Λ. But first we need to train it with the right data. In this section, we discuss part of the BAC event log preprocessing that involves generating the *labels*, "y" in the context of machine learning.

Let $\sigma^{comp} \square L^{comp}$ and $\sigma^{p} \square prefix(\sigma^{trunc})$; to create the dataset D we generate pairs of (x, y) where $x = \rho_L(\sigma^p)$ and $y = K(\sigma^{comp})$. Table 6.3 shows an example of an instance in the dataset.

REQUEST_ID	ACTIVITY	CE_UO	…	activity-to-avoid
20184005305	Back-Ofice Adjustment Requested	BOC	…	1

Table 6.3: Table showting (x, y). Column 'actttivtitty-tto-avotid' = $K(\sigma^{comp})$ and tthe restt are equal tto $\rho_L(\sigma^p)$

For the implementation of label generation, pandas library was used, and a custom Python function was written that generates these "labels" and associates them with the respective row. The algorithm to do this is as follows:

- Use pandas "groupby" to iterate over each trace in the CSV
- Check the existence of activity-to-avoid in the activity column of the CSV.
- If the activity-to-avoid exists, associate label 1 with all rows; otherwise associate 0.

We now have a dataset D that can be used to train the predictive oracle function Φ_K to predict the KPI values of a running trace $\sigma^{trunc} \square L^{trunc}$. We will also use the same dataset to train our evaluation oracle function Λ_K.

6.4.3 Setup

We carried out our tests using both Model-Agnostic and Gradient-Based methods, from the dice_ml Python library (paper: [5]).

In model-agnostic method, there are three sampling strategies available: "Random Search", "Kd-tree Search" and "Genetic Search". We performed experiments using "Random Search" and "Genetic Search" but not "Kd-tree Search". This is because the "Kd-tree Search" method tries to find the solution from the train set, and if there does not exist an exact solution in the train set, it is unable to find any CFE. In our experiments, it was producing CFEs for around 2 to 4 cases out of 715, so we decided not to use this method.

The gradient-based method is based on the counterfactual function C_K defined in the section: 2.5.

6.4.3.1 Predictive oracle function implementation

For this case study, we have two different predictive oracle functions for counterfactual generation. We define them as follow:

- Φ_K^A is the predictive oracle which is implemented as a Balanced Random Forest Classifier. We choose "A" in the oracle function because it will be used by dice_ml's model-**A**gnostic methods. The model was parameter-tuned before use and achieved an F1 score of 0.74. The parameter tuning process for determining the best parameters is described in Appendix A.3
- Φ_K^G is the predictive oracle which is implemented as a Feed Forward Neural Netwrok. "G" in the oracle function because it will be used by dice_ml's **G**radient-based method. The neural network was hyperparameter tuned before use and achieved an F1 score of 0.76. The parameter tuning process for determining the best architecture and parameters is described in Appendix A.4

We use Scikit-learn's pipeline function to chain the Φ_K^A model with the transformers that convert the input trace to a vector that the Balanced Random Forest Classifier can work with. In case of gradient-based method dice_ml does the conversion from trace to instance internally.

Parameter	**Value**
number of estimators	100
Criterion	gini
max depth	Auto
min samples split	2
min samples leaf	1
max features	Auto
max leaf nodes	Auto
bootstrap	True

Table 6.4: Machtine Learnting Model Parametters

Table 6.4 shows the training parameters used by the Balanced Random Forest Classifier model.

Figure A.2 shows the architecture of the Pytorch model which implements Φ_K^G

6.4.3.2 Evaluation oracle function implementation

For the evaluation oracle function Λ_K we choose the XGBoost Classifier, which is also based on decision trees and thus is also expected to perform well on event log data just like the Random Forest model. The model was parameter-tuned before use and achieved an F1 score of 0.75. The parameter tuning process for determining the best parameters is described in Appendix A.5

Similarly, we use scikit-learn's pipeline function to chain this model with the transformers that convert the input trace to a vector which the XGBoost can work with.

Parameter	Value
Eta	0.05
Gamma	0.5
Subsample	1
Colsample Bytree	0.3
Max Depth	21
Min Child Weight	17
Number of Estimators	150
Sclae Position Weight	21

Table 6.5: XGBoostt Classtifftier ttratinting parametters

Table 6.5 shows the training parameters used by the XGBoost Regressor model.

6.4.3.3 Transition System

A custom Python algorithm was developed to implement the Transition System functionality. Hash-map (or dictionary in Python) was used to store all the possible prefixes of an activity. The prefixes are stored as keys of the dictionary and the value of this key contains the list of next possible paths. An example of dictionary data structure used in the transition system can be: {"act3, act5": [act2, act6], ..., "act3, act5, act2": [act1, act7] }. For this internal graph, the transition system would consider "act2" as a valid activity after the prefix { "act3, act5" }, but "act7" would be considered as an invalid activity.

We use the BAC event log L^{comp} to build this functionality.

6.4.3.4 Activity-Resource Validation Function

A custom Python algorithm was developed to implement the activity-resource validation function. A Python set was used as the main data structure for this implementation. Python set

has this property just like a mathematical set that duplicates cannot exists in it and it is also unordered. Thus, we add all the combinations from the BAC event log L^{comp} to this set and this data structure automatically removes the duplicates, making it an eficient choice for this kind of requirement.

When we need to validate an activity and resource combination we just search this set. If we find it inside the Python set the function returns 1; otherwise it returns 0

6.5 Results

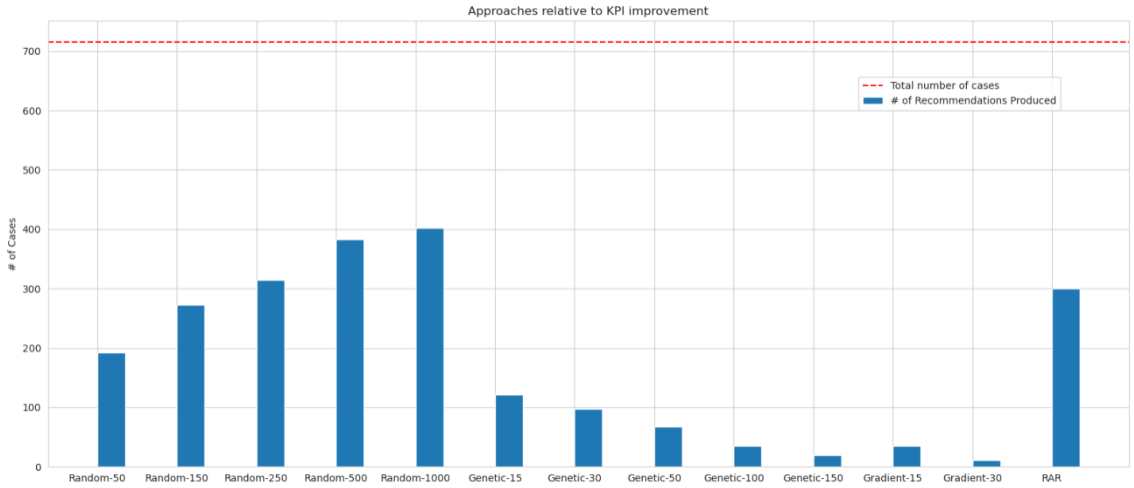

Figure 6.3: Bar chartt showting valtid recommendatttions produced by each approach tto avotid tthe undestirable actttivtitty. The bars show how many off tthese recommendatttions are constidered valtid by tthe evaluatttion oracle ffunctttion Λ_K. The dotttted red ltine shows tthe ttottal number off cases. DtiCE approaches have metthod name tin sttartt e.g. "Random", "Genetttic" and "Gradtientt" ffollowed by tthe value off "DtiCE ttottal CFEs" parametter ffor tthe algortitthm on each case. "RAR" tis tthe Resource Aware Recommendatttions algortitthm.

The results for KPI *Activity Occurrence* K_{occur} are presented in Figure 6.3. For convenience, the results are also tabulated in Table 6.6 to view the exact values.

Figure 6.3 is a bar chart that shows the number of cases, out of 715 cases, for which a valid recommendation was made by each approach. The implication is that when the valid recommendation is followed, the activity-to-avoid "Back-Office Adjustment Requested" was successfully avoided. The bars height represents the recommendations considered valid by the evaluation oracle function Λ_K. The DiCE approaches are prefixed with the method name "Random", "Genetic", and "Gradient" followed by the parameter value of "DiCE total CFEs". "RAR"

Approach	DiCE total CFEs	# Rec-val-by-XGB
No Recommendations	-	0
RAR	-	300
Random	50	192
Random	150	273
Random	250	315
Random	500	383
Random	1,000	402
Genetic	15	121
Genetic	30	97
Genetic	50	67
Genetic	100	35
Genetic	150	20
Gradient	15	35
Gradient	30	11

Table 6.6: Table shows tthe numertic values used tto plott tthe bar chatt 6.3. "RAR" tis ffor resource aware recommender systtem [2]. The DtiCE based approaches are "Random", "Genetttic", and "Gradtientt". The second column shows tthe value off tthe DtiCE algortitthm parametter "DtiCE ttottal CFEs" ffor each DtiCE algortitthm approach. The tthtird column "#Rec-val-by-XGB", tis tthe Number off recommendatttions constidered valtid accordting tto tthe XGBoostt evaluatttion oracle ffunctttion.

(Resource Aware Recommendations) is the algorithm that was proposed by Padella et al. [2]. The red dotted line above the bars represents the total number of cases.

The most validated recommendations (402) are produced by "Random-1000", and the least valid recommendations (11) are produced by "Gradient-30". The RAR approach produced 300 valid recommendations.

In Table 6.6 the first column lists the approaches used to generate recommendations. The second column contains the selected parameters value of "DiCE total CFEs" for the DiCE algorithm. The third column "#Rec-val-by-XGB", is read as "Number of recommendations considered valid according to the XGBoost evaluation oracle function". It lists the count of cases in which Activity-to-avoid was successfully avoided according to the XGBoost evaluation oracle function.

"No recommendations" (in the first row) is the base case; indicating that no recommendation was followed and that the original activity and resource, found in L^{run}, were chosen. Thus, the third entry in the first row is 0. The second entry "RAR" (Resource Aware Recommendations) is the algorithm proposed by Padella et al. [2]. Entries beginning with "Random" and "Genetic" are produced with the DiCE algorithm's model-agnostic methods, which use the

Balanced Random Forest Classifier as the predictive oracle Φ_K^A. Finally, the last two entries starting with "Gradient" are produced by the DiCE algorithm's gradient-based method which uses the feed forward neural network predictive oracle Φ_K^G.

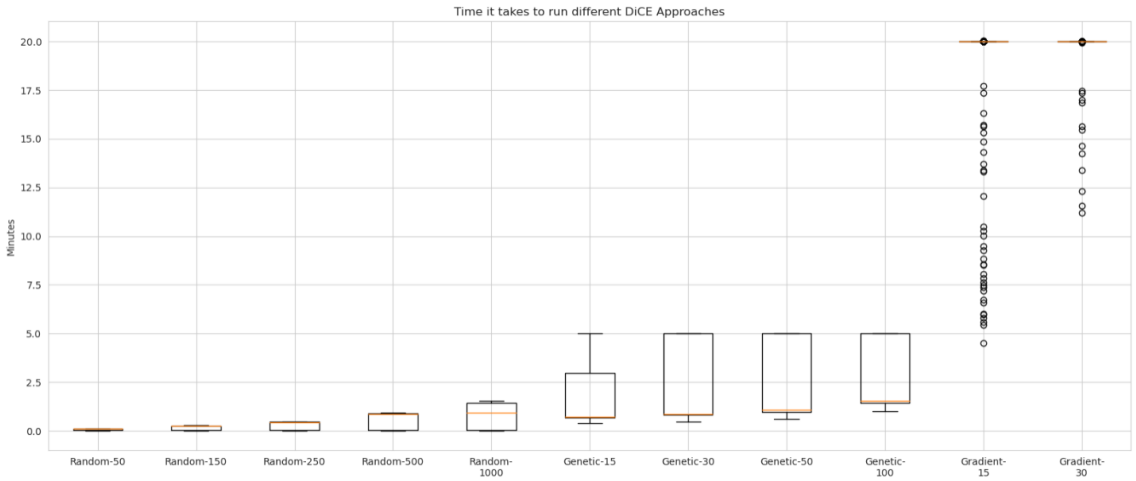

Figure 6.4: Box plott shows a compartison off tthe tttime titt ttakes tto run tthe cases tin tthe dattasett ffor each DtiCE approach. The orange ltine tin tthe boxes shows tthe medtian off tthe tintterquartttile range. The "Random" and "Genetttic" expertimentt were sttopped afftter 5 mtinuttes tiff a recommendatttion was nott produced. The "Gradtientt" expertimentts were sttopped afftter 20 mtinuttes.

The computational cost of running an algorithm is very important to consider when we talk about the practicality of an algorithm. For this reason we also analyzed the running time of the different DiCE approaches. Figure 6.4 shows the comparison of time it takes to run the cases in the dataset for each DiCE approach. The box plot shows that for the Random sampling strategy the DiCE algorithm is very quickly able to find recommendations. However, it takes more and more time as we increase the value of "DiCE total CFEs" parameter. Genetic sampling strategy takes significantly more time and the Gradient based method takes the most time.

Analysis and Interpretation

In this section, the results presented in Section 6.5 are discussed and analyzed. Discussion also includes one of the research questions of this investigation.

Following a similar trend to the KPI total time, DiCE with "Random" sampling strategy performs the best. This time having to produce 1, 000 counterfactual examples (CFEs) helped the algorithm generate recommendations for more cases. Random-1000 was able to avoid un-

desired activity "Back-Ofice Adjustment Requested" 56.22% of the time. The is a huge improvement in the KPI.

A strange trend we see is that as we increase the number of CFEs to produce for the "genetic" sampling strategy and "Gradient" method, the number of CFEs decreases.

After further investigation, we discovered that the genetic sampling strategy suffers computationally when asked to produce more CFEs. As we increase the number of CFEs required, it takes exponentially more time to produce more CFEs for a single case. That is why "Genetic" sampling strategy with higher values of "DiCE total CFEs" parameter produces fewer CFEs.

The Gradient method has a similar problem where it suffers computationally when asked to produce more CFEs. This method, in particular, also gets stuck in an infinite loop when trying to minimize the loss function. Thus, it produced very few recommendations compared to other approaches. It also seems that this method is not suited for bulk processing, so if there is a single case for which we need recommendations, maybe this algorithm can take more time to produce some results, but in this case we had to limit the running time to 20 minutes.

The trend in the huge computational cost associated with these approaches is also visible in the Figure 6.4. And in the case of Gradient method, we see most of the experiments are not able to produce a valid recommendation even with 20 minutes of time.

How does our proposed prescriptive analysis framework perform compared to existing methods such as the one proposed by Padella et al. [2], when evaluated on real-life datasets?

The RAR algorithm performs really well, but as we move to more computationally expensive configurations of the DiCE algorithm it overtakes the RAR algorithm. The best DiCE approach produces 53.44% more valid recommendations then the RAR algorithm. It should be noted that the RAR algorithm has stricter operating conditions. It takes into account the availability of resources when suggesting the resource for a given activity.

Of the total cases of 715, the best DiCE approach produced valid recommendations for 402 cases or for 56.2% of the cases. The RAR produced valid recommendations for only 42% of the cases. Given the results, we can safely conclude that our method performs commendably compared to existing methods like [2], but it requires more relaxed operating assumptions.

7

Conclusion

This thesis aimed to test the use of counterfactuals for the purpose of prescriptive analytics. Prescriptive analytics is gaining more and more focus from the literature and business stakeholders, which tells us that the need for good recommendations is ever increasing. We answered our first research question by developing a domain-agnostic prescriptive analytics framework. We answered our second research question by evaluating our method alongside another state-of-the-art method.

In that regard, this thesis was able to successfully showcase a few of the capabilities of counterfactuals, specifically ”DiCE” implementation. The proposed approach was able to generate valid counterfactual examples (CFEs) and dramatically optimize the KPIs involved. The method also performed very well against another state-of-the-art approach proposed in the literature.

The results of the thesis show that counterfactual explanation is a promising methodology and, in the future, can help businesses better steer their processes.

One weakness of the proposed method is that it considers the resource infinite, meaning that the same resource can be asked to work on different cases concurrently. And in reality following this kind of suggestion would not be possible.

However, these problems can be overcome. One way can be to further improve the post-processing so that at a time, a single resource can be recommended to a given trace. Another solution can be to implement this as a constraint in the counterfactual generating algorithm itself. In this way, it learns to optimize resource allocation alongside KPI.

As this is one of the early steps in the direction of using counterfactuals for prescriptive analytics, a lot of work needs to be done. This work was unable to explore the Variational Autoencoder (VAE) in DiCE implementation. VAE can allow the user to specify additional constraints that exist between the features of an event. Therefore, this can be used to account for limited resources or a single resource cannot work on multiple activities at the same time.

The method of recommendation validation was also not perfect, so in future researchers can find a better method to validate the results produced by these approaches.

A

Supplementary information

A.1 Catboost Regressor Parameter Tuning

Table A.1 shows the parameter grid used to tune the model.

Parameter	Values
Learning Rate	[0.005, 0.009, 0.01, 0.02, 0.03, 0.1]
Iterations	[1500, 2000, 2500, 3000, 3500, 4000]
Max Depth	[8, 10, 12]
Early Stopping Rounds	[5, 500]

Table A.1: Random Forestt Regressor parametter ttunting grtid

A.2 XGBoost Regressor Parameter Tuning

Table A.2 shows the parameter grid used to tune the model.

Parameter	Values
Number of Estimators	[50, 100, 150, 200, 350]
Evaluation Metric	[Mean Absolute Error, Root Mean Squared Error]
Max Depth	[3, 4, 5, 12, 16, 32, 64, 128]
Eta	[0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 0.7]
Gamma	[0, 0.5, 1, 2, 5, 7]
Sub-sample	[0.5, 0.8, 1]
Subsample Ratio of Columns	[0.3, 0.5, 0.8, 1]
Min Child Weight	[0, 1, 3, 5, 7]

Table A.2: XGBoostt Regressor parametter ttunting grtid

A.3 Balanced Random Forest Classifier Parameter Tuning

Table A.3 shows the parameter grid used to tune the model.

Parameter	Values
Number of Estimators	[50, 100, 150, 200]
Criterion	[Gini, Entropy]
Max Depth	[16, 32, 64, Auto]
Max Features	[Auto, sqrt, log2]
Min Samples Leaf	[1, 2, 4, 6]
Min Samples Split	[2, 5, 7, 10]
Replacement	[True, False]

Table A.3: Balanced Random Forestt Classtifftier parametter ttunting grtid

A.4 Neural Network Classifier Hyper-Parameter Tuning

First, we performed architecture optimization to see how many layers the neural network should have. 3 Architectures that were tried are A.1, A.2, A.3. From these we selected A.2 as this gave a similar performance to A.3 but was less complex.

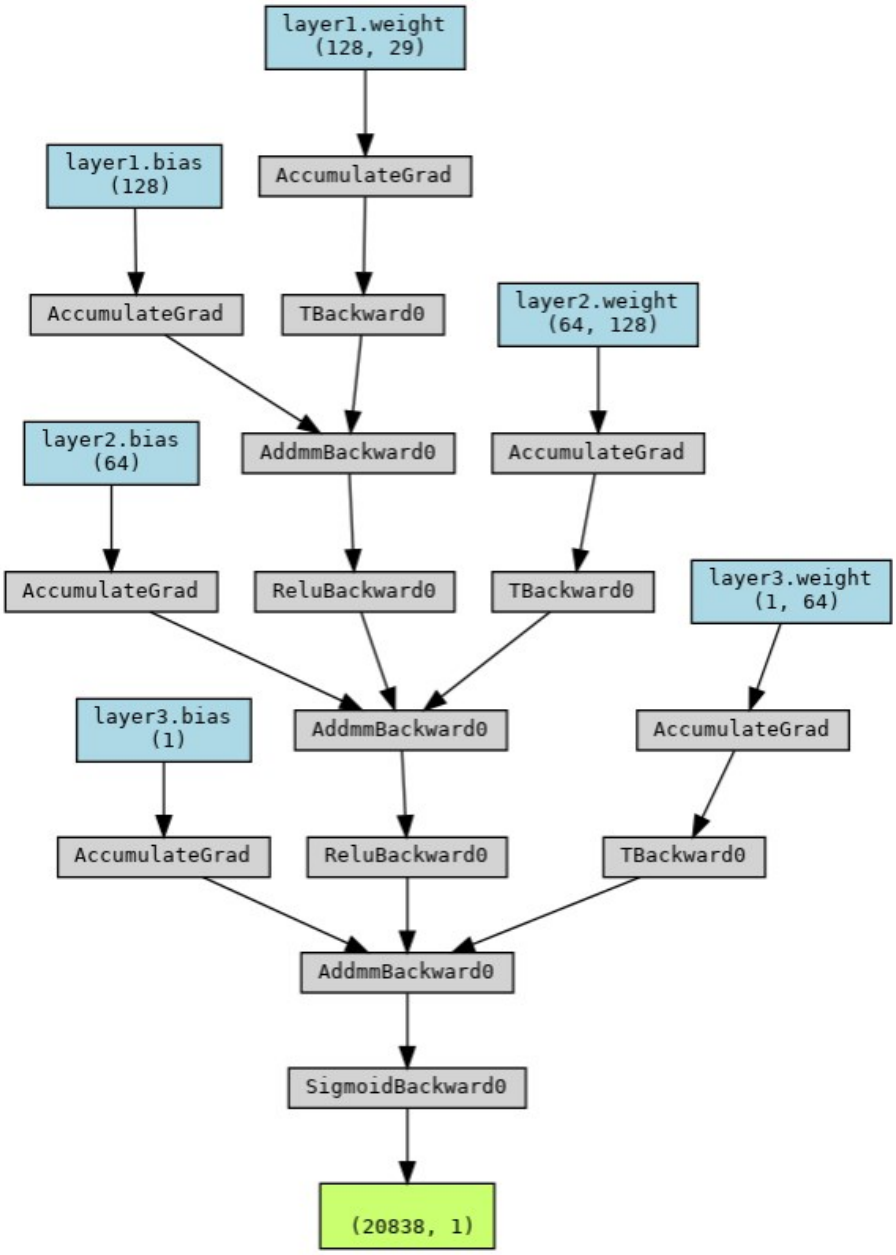

Figure A.1: PyTorch model archtittectture ffor tthe small model.

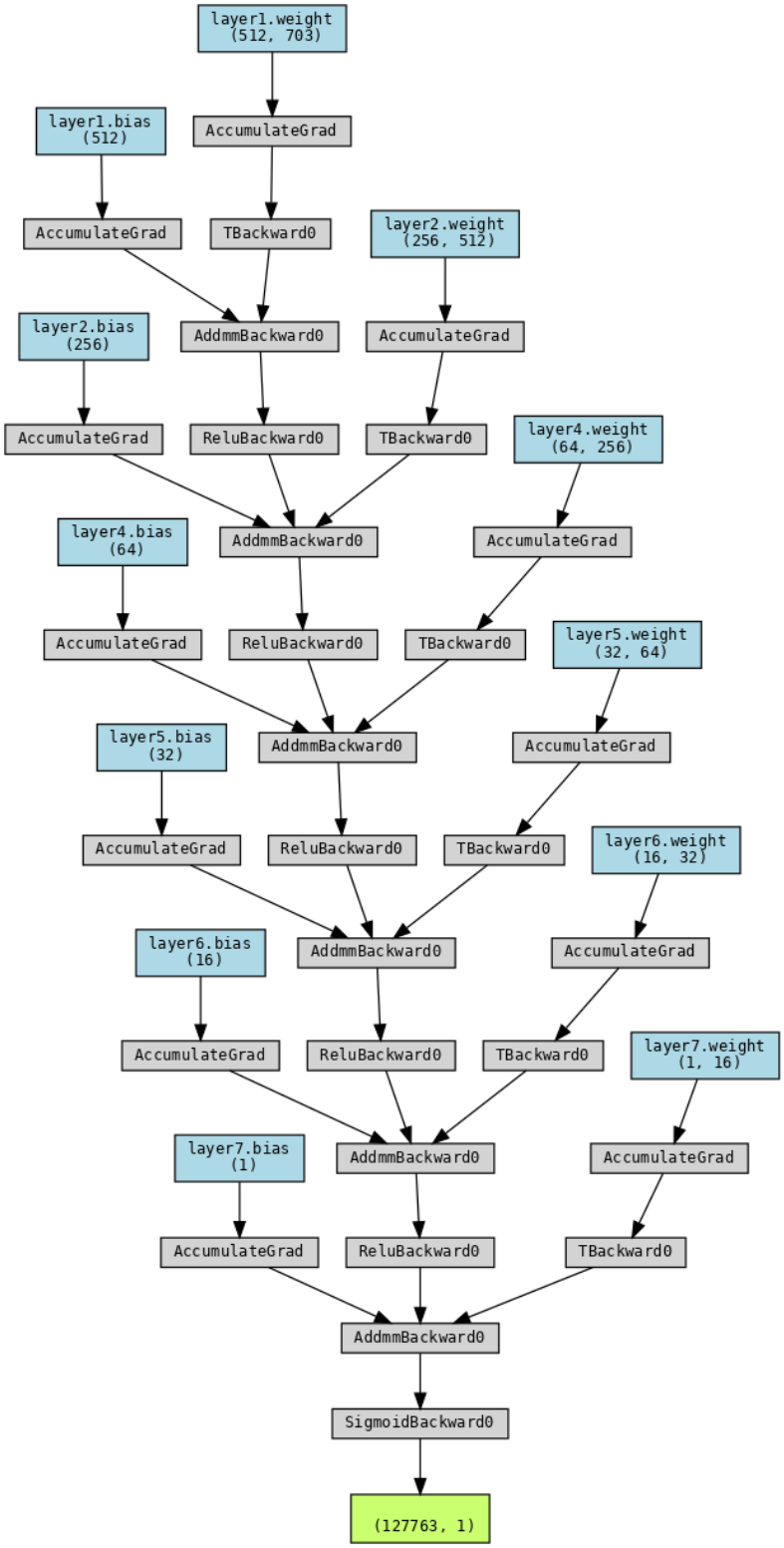

Figure A.2: PyTorch model archtittectture ffor tthe medtium model. Thtis tis an timplementtatttion off predtictttive oracle ffunctttion Φ_K^G

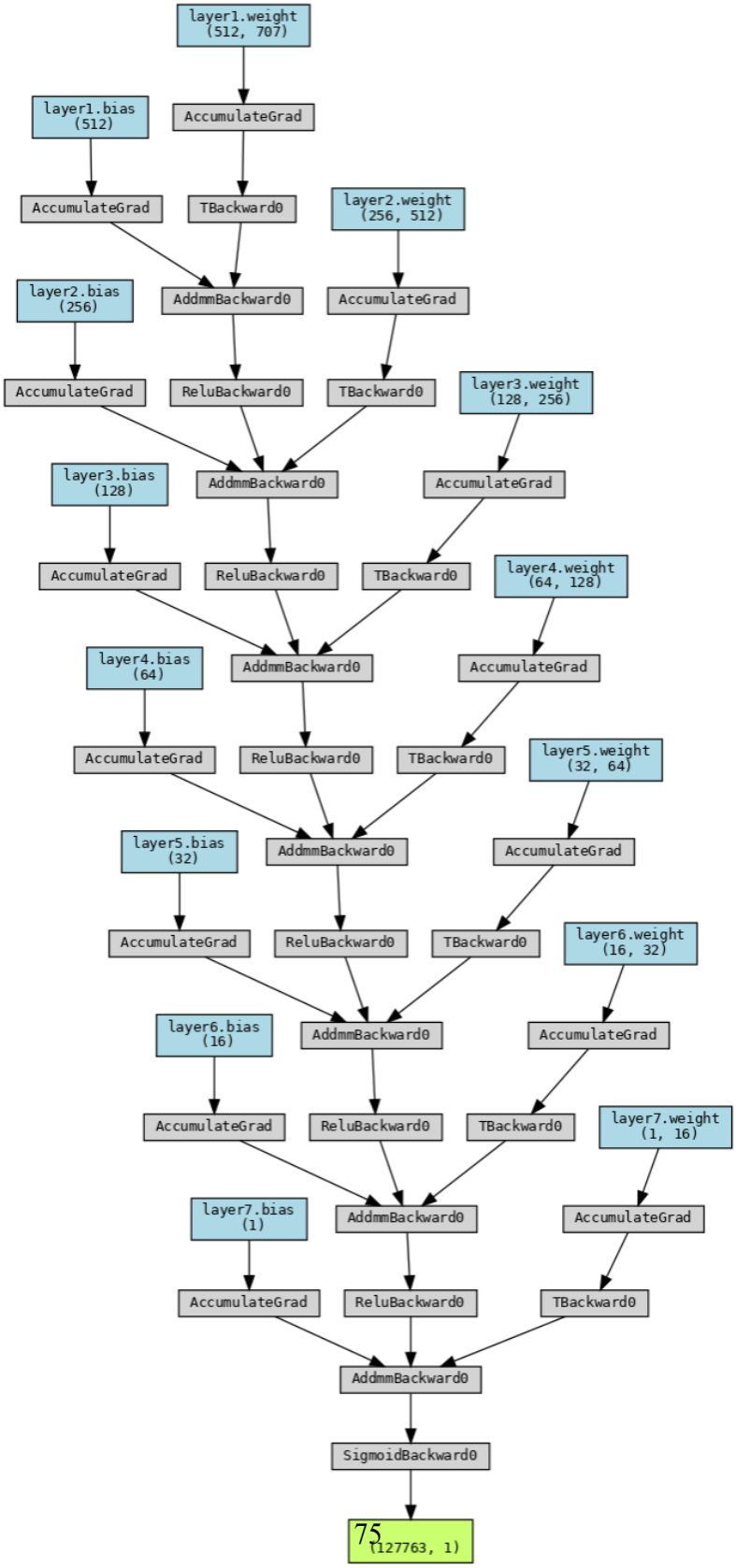

Figure A.3: PyTorch model archtittectture ffor tthe large model.

Table A.4 shows the parameter grid used to tune the model.

Parameter	Values
Batch Size	[32, 64, 256, 512]
Learning Rate	Random from 0.1 to 0.001
Batch Normalization	[True, False]
Regularization	[0.001, 0.004, 0.005, 0.006, 0.009]

Table A.4: Neural Nettwork Classtifftier hyperparametter ttunting grtid

A.5 XGBoost Classifier Parameter Tuning

Table A.5 shows the parameter grid used to tune the model.

Parameter	Values
Number of Estimators	[50, 100, 150, 200, 350]
Evaluation Metric	[Log loss, Error]
Max Depth	[3, 4, 5, 12, 16, 32, 64, 128]
Eta	[0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 0.7]
Gamma	[0, 0.5, 1, 2, 5, 7]
Sub-sample	[0.5, 0.8, 1]
Subsample Ratio of Columns	[0.3, 0.5, 0.8, 1]
Min Child Weight	[0, 1, 3, 5, 7, 10, 14]

Table A.5: XGBoostt Classtifftier parametter ttunting grtid

XGBoost was not optimized to satisfactory level with this grid, so after careful analysis of the test scores and its relation to different parameters another run of focused grid was run to further fine tune the XGBoost Classifier. The grid in Table A.6

Parameter	Values
Number of Estimators	[100, 150, 200, 250, 300, 400, 500, 800]
Evaluation Metric	[Log loss]
Max Depth	[16, 18, 20, 21, 24, 28]
Eta	[0.04, 0.05, 0.06,]
Gamma	[0, 0.5, 1]
Sub-sample	[0.9, 1]
Subsample Ratio of Columns	[0.3, 0.5, 0.7]
Min Child Weight	[7, 10, 13, 15, 17]

Table A.6: XGBoostt Classtifftier parametter ttunting grtid ffocused

References

[1] A. Padella, M. de Leoni, O. Dogan, and R. Galanti, "Explainable process prescriptive analytics," in *2022 4th International Conference on Process Mining (ICPM)*. IEEE, 2022, pp. 16–23.

[2] A. Padella and M. de Leoni, "Resource allocation in recommender systems for global kpi improvement."

[3] M. d. 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, pp. 9–16.

[4] C. Hsieh, C. Moreira, and C. Ouyang, "Dice4el: Interpreting process predictions using a milestone-aware counterfactual approach," in *2021 3rd International Conference on Process Mining (ICPM)*, 2021, pp. 88–95.

[5] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, ser. FAT* '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 607–617. [Online]. Available: https://doi.org/10.1145/3351095.3372850

[6] S. Weinzierl, S. Dunzer, S. Zilker, and M. Matzner, "Prescriptive business process monitoring for recommending next best actions," in *International conference on business pr o-cess management*. Springer, 2020, pp. 193–209.

[7] A. Metzger, T. Kley, and A. Palm, "Triggering proactive business process adaptations via online reinforcement learning," in *International Conference on Business Process Management*. Springer, 2020, pp. 273–290.

[8] K. Żbikowski, M. Ostapowicz, and P. Gawrysiak, "Deep reinforcement learning for resource allocation in business processes," in *International Conference on Process Mining*. Springer, 2022, pp. 177–189.

[9] W. Zhao, L. Yang, H. Liu, and R. Wu, "The optimization of resource allocation based on process mining," in *Advanced Intelligent Computing Theories and Applications: 11th International Conference, ICIC 2015, Fuzhou, China, August 20-23, 2015. Proceedings, Part III 11*. Springer, 2015, pp. 341–353.

[10] M. Shoush and M. Dumas, "When to intervene? prescriptive process monitoring under uncertainty and resource constraints," in *International Conference on Business Process Management*. Springer, 2022, pp. 207–223.

[11] ——, "Prescriptive process monitoring under resource constraints: a causal inference approach," in *International Conference on Process Mining*. Springer, 2021, pp. 180–193.

[12] X. Sun, Y. Ying, S. Yang, and H. Shen, "Remaining activity sequence prediction for ongoing process instances," *International Journal of Software Engineering and Knowledge Engineering*, vol. 31, no. 11n12, pp. 1741–1760, 2021.

[13] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics*, vol. 8, no. 8, p. 832, 2019.

[14] W. van der Aalst, "Process mining: Overview and opportunities," *ACM Trans. Manage. Inf. Syst.*, vol. 3, no. 2, jul 2012. [Online]. Available: https://doi.org/10.1145/2229156.2229157

[15] M. Dees, M. de Leoni, W. M. van der Aalst, and H. A. Reijers, "What if process predictions are not followed by good recommendations?(technical report)," *arXiv preprint arXiv:1905.10173*, 2019.

[16] W. M. Van der Aalst, "Using process mining to bridge the gap between bi and bpm." *Computer*, vol. 44, no. 12, pp. 77–80, 2011.

[17] R. Akerkar, "Advanced data analytics for business," *Big data computing*, vol. 377, no. 9, 2013.

[18] K. Lepenioti, A. Bousdekis, D. Apostolou, and G. Mentzas, "Prescriptive analytics: Literature review and research challenges," *International Journal of Information Management*, vol. 50, pp. 57–70, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0268401218309873

[19] D. den Hertog and K. Postek, "Bridging the gap between predictive and prescriptive analytics-new optimization methodology needed," *Tilburg Univ, Tilburg, The Netherlands*, 2016.

[20] D. Larson and V. Chang, "A review and future direction of agile, business intelligence, analytics and data science," *International Journal of Information Management*, vol. 36, no. 5, pp. 700–710, 2016.

[21] J. Hagerty, "Planning guide for data and analytics," *Gartner Inc*, p. 13, 2017.

[22] L. Šikšnys, T. B. Pedersen, L. Liu, and M. Özsu, "Prescriptive analytics," *Encyclopedia of database systems*, pp. 1–2, 2016.

[23] A. Basu, "Five pillars of prescriptive analytics success," *Analytics magazine*, vol. 8, p. 12, 2013.

[24] L. Lin, L. Wen, and J. Wang, "Mm-pred: A deep predictive model for multi-attribute event sequence," in *Proceedings of the 2019 SIAM international conference on data mining*. SIAM, 2019, pp. 118–126.

[25] 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.

[26] A. Leontjeva, R. Conforti, C. Di Francescomarino, M. Dumas, and F. Maggi, "Complex symbolic sequence encodings for predictive monitoring of business processes," vol. 9253, 08 2015.

[27] H. Sharma, S. Kumar *et al.*, "A survey on decision tree algorithms of classification in data mining," *International Journal of Science and Research (IJSR)*, vol. 5, no. 4, pp. 2094–2097, 2016.

[28] W. M. P. van der Aalst, *Process Mining: Data Science in Action*, 2nd ed. Heidelberg: Springer, 2016.

[29] W. M. P. van der Aalst, M. Pesic, and M. Song, "Beyond process mining: From the past to present and future," in *Advanced Information Systems Engineering*, B. Pernici, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 38–52.

[30] M. Islam, G. Chen, and S. Jin, "An overview of neural network," *American Journal of Neural Networks and Applications*, vol. 5, no. 1, pp. 7–11, 2019.

[31] Y. Sasaki, "The truth of the f-measure," *Teach Tutor Mater*, 01 2007.

[32] T. Chai and R. R. Draxler, "Root mean square error (rmse) or mean absolute error (mae)," *Geoscientific model development discussions*, vol. 7, no. 1, pp. 1525–1534, 2014.

[33] P. Regulation, "Regulation (eu) 2016/679 of the european parliament and of the council," *Regulation (eu)*, vol. 679, p. 2016, 2016.

[34] C. Russell, "Eficient search for diverse coherent explanations," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2019, pp. 20–28.

[35] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual explanations without opening the black box: Automated decisions and the gdpr," *Harv. JL & Tech.*, vol. 31, p. 841, 2017.

[36] L. Duan and Y. Xiong, "Big data analytics and business analytics," *Journal of Management Analytics*, vol. 2, no. 1, pp. 1–21, 2015.

[37] S. Pravilovic, A. Appice, and D. Malerba, "Process mining to forecast the future of running cases," in *New Frontiers in Mining Complex Patterns: Second International Work-shop, NFMCP 2013, Held in Conjunction with ECML-PKDD 2013, Prague, Czech Re-public, September 27, 2013, Revised Selected Papers 2*. Springer, 2014, pp. 67–81.

[38] J. Lu, W. Chen, Y. Ma, J. Ke, Z. Li, F. Zhang, and R. Maciejewski, "Recent progress and trends in predictive visual analytics," *Frontiers of Computer Science*, vol. 11, pp. 192–207, 2017.

[39] M. Polato, A. Sperduti, A. Burattin, and M. d. Leoni, "Time and activity sequence prediction of business process instances," *Computing*, vol. 100, pp. 1005–1031, 2018.

[40] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: gradient boosting with categorical features support," *arXiv preprint arXiv:1810.11363*, 2018.

Acknowledgments

I would like to thank the Almighty God, the most gracious and the most loving and caring.