UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Software Engineering Curriculum

Lukas Baltramaitis

Activity-Oriented Causal Process Mining: An End-to-End Approach Utilizing Ylearn

Master's Thesis (30 ECTS)

Supervisor(s): Fredrik Milani, MSc
Mahmoud Shoush, MSc

Tartu 2023

Activity-Oriented Causal Process Mining: An End-to-End Approach Utilizing Ylearn

Abstract:
In recent decades, companies have explored data-driven methods and tools to improve their business processes. More recently, prescriptive business process analysis became popular among data analysts and researchers. There are many studies on the use of prescriptive algorithms for the optimization of a variety of different business processes. Prescriptive algorithms given the historical and/or real-time data try to discover and recommend the best actions to improve the future outcome, e.g. what existing actions in the advertisement process need to be changed to increase the sales. One of the prescriptive methods approaches is Causal Process Mining which uses event logs received from the company's information systems and then analyses them with Causal Inference algorithms to discover and estimate these possible changes (treatments) that would affect the final outcome. However, all event logs can differ by the variables that are logged and the models may become dependent on the data structure. This means that each event log requires separate variables investigation and modeling that would match the event log data structure. Consequently, performing these activities takes time and resources. A more generic and automated approach could be better applicable in different business cases and give useful results without excessive analysis or model building. For this reason, in this study, we investigate the possibility to use only case ID, activity, and timestamp variables of the event log for the causal inference algorithms. We propose the experimentation software artifact that includes data preparation and integrates the existing Ylearn causal inference tool. The approach is evaluated using five real-world event logs. Evaluation results show that causal relationships can be detected between activities of the event log and estimated treatment effects are comparable with other approaches.

Keywords: Causal inference, Uplift modeling, Causal process mining.

CERCS: P170 Computer science, numerical analysis, systems, control.

Tegevusele orienteeritud põhjusliku protsessi kaevandamine: Otsast lõpuni lähenemine Ylearn'iga

Lühikokkuvõte:
Viimastel aastakümnetel on ettevõtted uurinud andmepõhiseid meetodeid ja vahendeid oma äriprotsesside uuendamiseks. Viimasel ajal on andmeanalüütikute ja -uurijate seas populaarseks saanud preskriptiivne äriprotsesside analüüs. Paljud uuringud käsitlevad ettekirjutavate algoritmide kasutamist erinevate äriprotsesside optimeerimiseks. Ettekirjutavad algoritmid püüavad ajalooliste ja/või reaalajas saadud andmete põhjal leida ja soovitada parimaid tegevusi tulevaste tulemuste parandamiseks, nt milliseid olemasolevaid tegevusi reklaamiprotsessis on vaja muuta, et suurendada müüki. Üks ettekirjutavate meetodite lähenemisviis on kasutada põhjusliku protsessi kaevandamist. See lähenemisviis kasutab ettevõtte infosüsteemidest saadud sündmuste logisid ja analüüsib neid seejärel põhjusliku järeldamise algoritmide abil, et avastada ja hinnata neid võimalikke muutusi, mis mõjutaksid lõpptulemust. Kõik sündmuste logid võivad siiski erineda logitavate muutujate poolest ja mudelid võivad muutuda sõltuvaks andmestruktuurist. See tähendab, et iga sündmuse logi nõuab eraldi muutujate uurimist ja modelleerimist, mis vastaks sündmuse logi andmestruktuurile. Seetõttu võtab nende tegevuste teostamine aega ja ressursse. Üldisem ja automatiseeritud lähenemisviis oleks paremini rakendatav erinevate äritegevuste puhul ja annaks kasulikke tulemusi ilma liigse analüüsi või mudeli koostamiseta. Seetõttu uurime käesolevas uuringus võimalust kasutada põhjusliku järeldamise algoritmide jaoks ainult sündmuse ID, tegevuse ja ajatempli muutujaid sündmuste logis. Pakume välja eksperimendi tarkvara artefakti, mis sisaldab andmete ettevalmistamist ja integreerib olemasoleva Ylearni põhjusliku järeldamise tööriista. Lähenemisviisi hinnatakse viie reaalse sündmuse logi abil. Hindamistulemused näitavad, et sündmuse logi tegevuste vahel on võimalik tuvastada põhjuslikke seoseid ja hinnanguline muudatuste efekt on võrreldav teiste lähenemisviisidega.

Võtmesõnad: Põhjuslik järeldus, Tõusu modelleerimine, Põhjusliku protsessi kaevandamine.

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria).

Contents

1 Introduction

Every company consists of business processes which can be defined as a set of activities that fulfills a specific organizational goal and produce an outcome. Business processes could be dependent on the business context (e.g. car inspection process during the roadworthiness test), or generic among businesses (e.g. accounting). Execution of business processes produces a lot of data that could be used for management, accounting, validation, or other purposes. The captured data is saved in different formats, e.g. cheques, logs, documents, digital systems and etc. Analyzing the business process data can help to optimize the process by solving problems, predicting the future of the processes, or even recommending actions that would improve the final outcome [KMND21].

Business process analytics is a set of methodologies that can help to identify and solve business process problems [Bay15]. Business process analytics has three main types: (Descriptive, Predictive, and Prescriptive) which differ by the complexity and the output that they can return [Bay15]. Descriptive analytics is the most popular one [LBAM20]. It focuses on analyzing business process data and identifying the problems in the current or past situation. Predictive analytics tries to construct a possible trend or pattern that can happen in the future from historical data. Usually, the output of Predictive analytics techniques is a possible event and its mathematical probability. Prescriptive analytics goes one step beyond and tries to predict the consequences of future events. Then its goal is to optimize the business process by recommending business decisions that could lead to the best possible outcome [LBAM20].

Causal process mining (CPM) is a field that bridges the gap between Process Mining and Causal Inference, with a focus on prescriptive analytics [SD22]. CPM comprises two main components: Causal discovery and Causal estimation algorithms. Causal discovery analyzes the data to identify potential causal relationships between variables and outcomes, essentially uncovering possible treatments based on the available data. On the other hand, Causal estimation investigates the potential effects of the identified treatments on the outcome, aiming to understand the causal impact. By combining these two components, CPM provides valuable insights into understanding and optimizing processes through causal relationships[1].

In recent studies [SD22], [BTD+20], [TDRM18], researchers have explored CPM algorithms and their application to process event logs. However, none of these studies have presented a comprehensive solution that combines both Causal discovery and Causal estimation in a single framework. One of the reasons for this gap is the inherent uniqueness of business event logs, as they often contain a distinct set of attributes that can vary depending on the specific business context. Developing a custom CPM model tailored to match each company's unique data and processes requires significant

[1]Marked paragraphs were written from a combined effort of the student and ChatGPT (06.04.2023), i.e. a language model whose training is based on a large number of different text sources. ChatGPT is developed by OpenAI. For more information about ChatGPT and OpenAI: `https://openai.com`.

resources, which only a tiny fraction of companies possess. To solve this problem, our approach utilizes an existing causal tool capable of automatically analyzing data from any business process, uncovering potential treatments, and estimating their effectiveness. By doing so, we aim to provide actionable recommendations for process improvement[1].

This thesis focuses on the investigation of a minimal set of attributes, namely *case ID*, *activity*, and *timestamp*, which are commonly found in any process data or event log. By leveraging a causal tool, the aim is to automate the process of discovering causal relationships and estimating their effects based on these fundamental attributes. This approach enables the exploration of causal relationships and their impact without requiring extensive additional information, thereby facilitating the analysis of process data in a more efficient and accessible manner [1].

The main contribution of this thesis is the development of a tool that integrates Ylearn[2] [Yan22], providing an interface for causal algorithms and data preparation functionality. In addition to the tool itself, the thesis presents performance metrics obtained from experiments conducted using the tool. This developed tool is intended to be utilized by researchers and data analysts, enabling them to perform experiments with their own process logs. Furthermore, the captured metrics can serve as a baseline for future benchmark studies in the field. Overall, the tool and its accompanying metrics offer valuable resources for conducting causal analysis and advancing research in this domain.

In the chapters below, we provide background with algorithms and evaluation metrics with an overview of related work; a method chapter with a description of our research process; a summary of the proposed approach and its implementation; a developed tool evaluation and its results, and a conclusion with future work.

[1]Marked paragraphs were written from a combined effort of the student and ChatGPT (06.04.2023), i.e. a language model whose training is based on a large number of different text sources. ChatGPT is developed by OpenAI. For more information about ChatGPT and OpenAI: `https://openai.com`.

[2]`https://ylearn.readthedocs.io/en/latest/index.html`

2 Background and related work

In this chapter, we present principles of the Causal Process Mining (CPM) and Causal Inference (CI) field. We begin with an overview of CPM and CI. After that, we briefly describe the Causal discovery, Causal estimation, and Uplift modeling processes. Then we provide information about the evaluation metrics that were used in the tool evaluation. Finally, we end this chapter with an overview of related work.

2.1 Causal Process Mining

CPM can be described as a sub-field of Process Mining and Causal Inference that investigates possibilities to find causal relationships between variables of the process event log [WPRM22].

Process Mining (PM) is a relatively young research discipline that involves computational intelligence and data mining, process modeling and analysis. The idea of PM is to discover, monitor, and improve real processes by extracting knowledge from the event logs readily available in today's software systems. Process Mining includes automated process discovery (i.e., extracting process models from an event log), conformance checking (i.e., monitoring deviations by comparing model and the log), organizational mining, automated construction of simulation models, model extension, model repair, case prediction, and history-based recommendations. The main data entity used in PM is an event log [KMND21]. An event log can be defined as a collection of business process cases. Every case is a sequence of events that has several variables: a *case ID*, *activity*, a point in time - (*timestamp*), and additional optional variables (e.g. *resources*, *employees*, and etc.). Causal PM specializes in event log investigation using CI algorithms.

2.2 Causal Inference

Causal Inference (CI) is a set of methods that aims to find and estimate the relationships between cause and effect from historical data. Together causal relationships form a Causal model which is the main instrument of the Causal Inference. CI is used in a variety of domains, including business process mining and monitoring [SD22].

The Causal Inference process can be divided into two main sub-processes: Causal discovery (CD) and Causal estimation (CE). The CD is the process that tries to find causal relationships (also referred to as cause-effect relationships between two variables when one directly influences the other - or one event makes another happen) and build a causal model from those discovered relationships. Causal estimation is the process that estimates the size of the treatment effect on the outcome. This process is based on the causal model discovered during the CD process, and also all available historical data [Nea20]. The *outcome* variable in the CE and CD is defined as the variable that has a causal relationship (as an effect end) with *treatment*, *confounder* and *covariate*

variables and is influenced by those variables. The *outcome* variable is the one that Causal Inference tries to improve. The, *treatment* is the variable that has a causal relationship (as a cause end) with the *outcome* variable. The Causal estimation tries to calculate what is the effect of changing the *treatment* values on the *outcome*. The other possible variable types are *confounder*, *covariate*, *instrument*. The *confounder* is the variable that has a causal relationship (cause-end) with both *treatment* and *outcome* variables [Tho23]. The *instrument* variable influences only the *treatment* and *covariate* only the *outcome* (see figure 1).

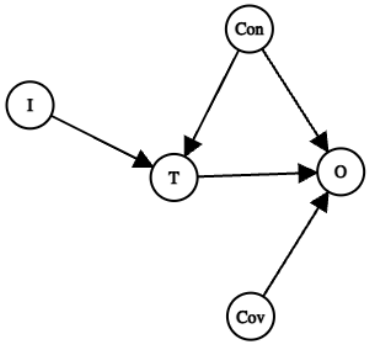


Figure 1. Example of the causal DAG. I - is the *instrument*, T - *treatment*, Con - *confounder*, Cov - *covariate*, O - *outcome*.

2.3 Causal discovery and estimation

In this subsection, we describe Causal discovery and Causal estimation algorithms.

2.3.1 Causal discovery algorithm

Causal discovery (also called Causal search) algorithms try to identify the causal relationships between values and build the causal model. The causal model is usually represented as a DAG (Directed Acyclic Graph) - where nodes are variables, edges are discovered causal relationships, and the direction of the edges points from the cause variable toward the effect variable. As is shown in figure 1, if the historical data about the *outcome* is known, the CD algorithms can directly find and return the set of potential *treatment*

variables (those variables that have the biggest impact on the outcome) and *confounders* - variables that influence both: the *outcome* and the *treatment* variables [NPR+22].

Learning a causal DAG from the data is an NP-hard problem because of the acyclicity requirement that is hard to ensure efficiently [ZARX18]. For this reason, there are a variety of approximate algorithms that tries to optimize the search of the DAG.

2.3.2 Causal estimation algorithm

Causal estimation algorithms try to estimate the *treatment* effect on the *outcome* also considering *confounders*. This could help to predict how good the *treatment* can be before applying it and wasting resources or searching for the best treatment option. The problem that Causal estimation algorithms have to solve is that the object cannot be treated and not treated at the same time, so it is only possible to observe the outcome of only one of those two states. So, instead, the effect is estimated by learning the structure from the historical data (training set) and a causal model. Then estimator can calculate metrics that show how good the treatment could be if applied [CVMP23].

In figure 2, we present the overview of the Causal estimation algorithm components. At first, the causal estimator learns the data structure from its training dataset and causal model (*outcome*, *confounders*, *covariates*, *instruments*, and *treatment*). Then given the test data, the estimator calculates the treatment effect on the outcome. The estimated effect is returned as ATE or ITE (see subsection 2.5) or other causal effect metrics.

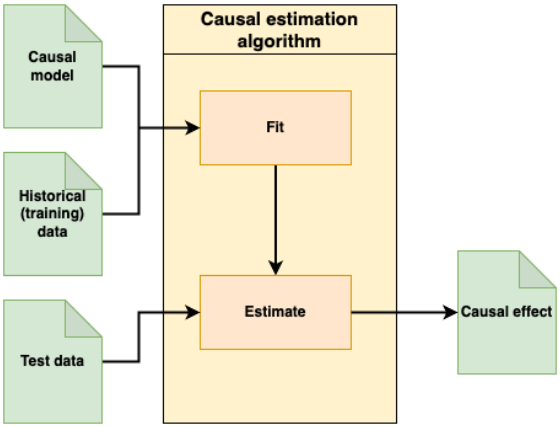


Figure 2. Causal estimation algorithm overview[1].

[1]The arrows represent the data flow, and green document artifacts indicate inputs and outputs.

2.4 Uplift modelling

In this subsection, we describe Uplift modeling.

In the causal data structure, each observation (in the context of this research we call observation as the case) can be assigned to one of four categories based on the outcome (successful or not successful) and treatment (treated or not treated) dimensions (see figure 3):

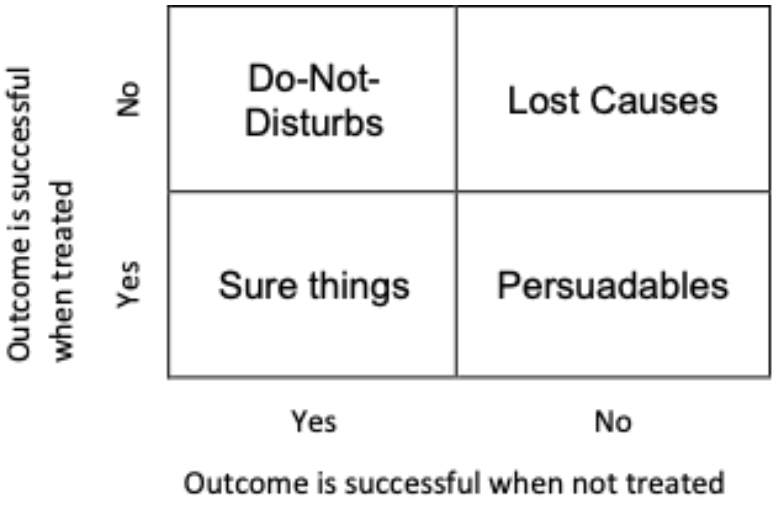


Figure 3. Cases categorization based on two dimensions: outcome and treatment[1].

1. Sure things: cases that end in success independently from treatment. Treating sure things does not improve the rate of successful outcomes but takes additional resources that are spent for the treatment.

2. Lost causes: cases with an unsuccessful end that is independent of treatment. Similar to sure things, treating lost causes will not improve the rate of successful outcomes but would take additional costs or resources.

3. Do-not-disturbs: cases that end unsuccessfully because of treatment and end successfully if not treated. Consequently, treating do-not-disturbs will not improve the successful outcomes rate, on the contrary, the treatment would even worsen the situation and also would take additional costs and resources. When many do-not-disturbs cases are included in the treatment, it could be even better, in terms of the cost of resources, to not apply the treatment at all.

4. Persuadables: cases that end successfully only because they are treated. These are

[1]Figure was adapted from [DMV18].

the cases that should be treated. They increase the rate of successful outcomes if their treatment does not take too many resources. Persuadables are exactly the cases that need to be selected and should be found by uplift modeling.

The main goal of Uplift modeling is to find the category of the case - especially if it is persuadable or do-not-disturb. This categorization depends on the treatment characteristics (e.g. The treatment with x value could leave the case as a lost cause, but with $2x$ (two times higher) treatment value the case could become persuadable). Uplift modeling is applied in marketing or political campaigns, medical treatment and etc.

Uplift modeling similar to CE involves determining the causal effect of a treatment on an outcome, enabling the identification of the most effective treatment to improve the outcome. By simulating potential scenarios, uplift models provide a way to predict the future based on various control variables.

Let's assume that the cases are randomly divided into two groups: treatment and control. A case can be either treated (treatment group) or not treated (control group). Then *Uplift score* is defined as the difference between the probability of a case ending with a successful outcome if treated and the probability of the case ending with a successful outcome if not treated, or:

$$U(X_i) := P(y_i = 1|x_i; t_i = 1) - P(y_i = 1|x_i; t_i = 0)$$

Where X is a vector of independent variables and

$$X = x_1, \ldots, x_n$$

and Y is the binary outcome variable, and $y_i = 1$ means i -th case ended with a successful outcome. T variable shows if the case is in the control or treatment group, and $t_i = 0$ means i -th case is in the control group, and $t_i = 1$ means i -th case is in the treatment group. P is a probability as estimated by a model [DMV18]. Overall, the uplift refers to the difference in the effect of the treatment.

2.5 Evaluation metrics

In this chapter, we provide a summary of the Causal estimation model metrics. Those metrics are used to measure the causal effect of the *treatment* on the *outcome* and also to compare the performance of CE algorithms.

Individual treatment effect (ITE) - the difference between a case outcome when it is treated and when it is not. It is defined:

$$m_1(x) = \mathbb{E}[y_1|x]$$

,

$$m_0(x) = \mathbb{E}[y_0|x]$$

then the function

$$g(x) := \mathbb{E}[y_1 - y_0|x] = m_1(x) - m_0(x)$$

is the expected treatment effect when the case is treated relative to when not treated on an individual unit with characteristics x, or the Individual Treatment Effect. y_1 is the outcome when treated, y_0 is the outcome when not treated [SJS17].

Average treatment effect (ATE) - a commonly used causal effect metric. It is defined as:

$$ATE = \mathbb{E}[y_1 - y_0]$$

where $y_1(i)$ is the outcome value if the case i is treated and $y_0(i)$ is the outcome value if the case is not treated. However, it is impossible to know both values of $y_0(i)$ and $y_1(i)$ because the case cannot be treated and not treated at the same time [Zha17]. Various CE algorithms try to predict the unknown value of y and calculate the ATE. The interpretation of ATE value is the difference in risks that would be measured if every case in the dataset was treated, versus if every case in the dataset was not treated, which is equivalent to the average of all individual treatment effects [NW23].

Qini curve - the Uplift model comparison and validation metric. Firstly, the uplift model sorts cases in descending order by their treatment effect scores (from the best case for treatment to the worst one). Then it plots the cumulative difference between the outcome rates of the treated group and not treated (control) group. The resulting curve is known as the cumulative uplift or *Qini curve*. Usually, there is a *Qini curve* and a random curve drawn in the graph for comparison (see the example in figure 4). The random curve is constructed by giving a uniform outcome to each case from the population and has a linear close to a straight-line incremental outcome. Then the *Qini coefficient* (also known as *Qini score*) is the difference between the area under these curves. A positive *Qini coefficient* represents a good performance of the model, while a value close to the random curve shows that the model did not return any valuable insights. The maximum possible *Qini coefficient* depends on the input data. Among alternative models that process the same data, the method that returns the highest *Qini coefficient* is preferable [GTFNZ$^+$23]. E.g. in figure 4 *model2* is better than *model1* because its *Qini curve* is higher at every point in the graph.

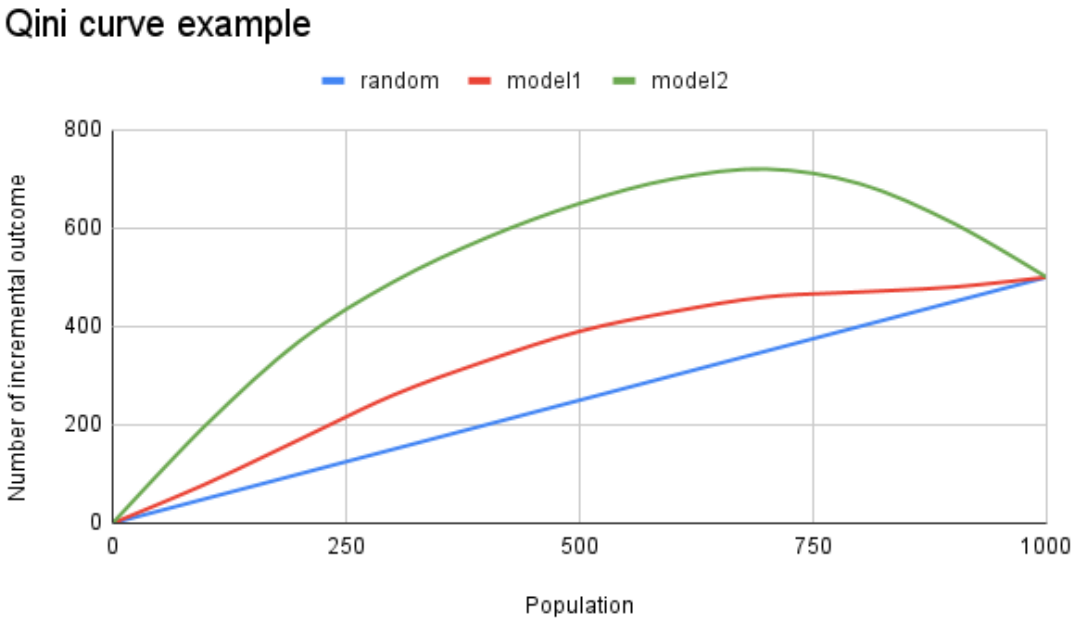


Figure 4. Qini curve example.

2.6 Related work

This subsection is an overview of related scientific studies that were written on the Process mining and Causal inference field.

The idea to combine Process mining and Causal inference methodologies into CPM is investigated in Shoush *et al.* [SD22] and Bozorgi *et al.* [BTD+20] studies.

In Shoush's *et al.* paper [SD22], the authors study the possibility to apply Causal Inference for the Prescriptive process monitoring (PrPM) process. They define PrPM as a branch of process mining techniques. They also state that PrPM is a process during which the prescriptive model observes business process event logs in real time and depending on the situation can recommend or directly intervene and make a decision in order to improve process performance. The authors also included the limitation of resources for interventions and used the Causal Inference approach for the estimation of the interventions to the outcome (if there is a causal relationship between variables X (cause) and Y (effect), then the intervention is a change of variable X value that should change the value of variable Y). Their goal was to maximize a cost function with resource constraints. The evaluation showed that their new approach gives better results than predictive non-causal approaches. In the approach used in this thesis, Causal Inference is also used for the process logs investigation. The difference is that our work

does not consider *resources*, instead, we focus on only *activity* and *timestamp* variables, and our approach works with only historical data and does not aim to optimize the process in real-time.

Teinemaa's *et al.* study [TDRM18] is a review of existing Predictive Process Monitoring methods. The authors try to solve the problem of unclear characteristics, applicability, and poor comparability of different PrPM approaches by presenting a systematic review and the classification of outcome-oriented PrPM techniques and experimental comparison/benchmark of eleven selected methods by using nine real-life event logs. In this thesis, we use events preprocessing methods (*Aggregation* and *One Hot Encoding*) and datasets from their Evaluation chapter [TDRM18] but the difference is that our focus is investigating Causal Inference methods.

Bozorgi *et al.* in their paper [BTD+20] introduce an Action Rule Mining method for examining business process event logs with the goal of producing suggestions for possible treatments that would optimize the probability of receiving the wanted outcome. The Uplift trees were used as a machine learning method for causal rules discovery. The model identifies causal relationships between treatment and outcome while also accounting for the influence of confounding variables. The evaluation of this approach was done by comparing its result with the recommendations of process mining experts. We use a different approach - instead of Action Rule Mining we use Causal discovery to discover possible treatment variables and instead of Uplift trees, we used Causal estimation algorithms for estimating the treatment effect. We used the results of Bozorgi's *et al.* study in our evaluation (see paragraph 5.3.2).

The approach of applying Causal discovery and Causal estimation in one end-to-end pipeline is proposed in Geffner's *et al.* paper [GAF+22]. The authors develop Deep End-to-end Causal Inference (DECI), a single flow-based non-linear additive noise model that takes in observational data and can perform both Causal discovery and estimation, including average treatment effect (ATE) estimation. Their evaluation results show the competitive performance of DECI in comparison with relevant baselines for both Causal discovery and ATE estimation. Their proposed method of Deep end-to-end Causal Inference has six steps: 1) Observe data corresponding to X variables; 2) Learn the causal relationships among all variables; 3) Learn the functional relationships among variables; 4) Select intervention and target variables; 5) Estimate causal quantities such as ATE; 6) Make optimal decisions and take action. The first and fourth step is manual and has to be done by the user of the pipeline while the other steps can be automatically done in their developed DECI model. The principle of having Causal discovery and Causal estimation together in one pipeline is also used in our research. However, the difference between the pipeline in our research and their pipeline is that only the first step (data preparation) in our approach is manual and we also add Uplift modeling instead of their sixth step: "Make optimal decisions and take action step". We chose to not include this tool and the DECI model in further investigation in our research because its repository

with the source code of the model was still being updated during the time we started our research.

The review of other Causal algorithms and tools was done by Guo *et al.* [GCL+18]. The focus of this paper is on examining the impact of having a large amount of data on the capability to understand and learn about causal relationships and effects. The main question that this study tries to answer is: "How does the process of learning about causality differ or resemble the present age of big data, compared to the traditional approach?" The research presents a comprehensive and organized review of the established and cutting-edge techniques for learning causal relationships, in addition to exploring the connections between CI and machine learning [GCL+18]. This paper also provides a link (`https://github.com/rguo12/awesome-causality-algorithms`) to the code repository which contains scientific papers and links to source codes, tutorials, and other information about more than 100 Causal Inference algorithms. This set of algorithms was used in this thesis as a starting point for the selection of the existing Causal Inference tool.

3 Method

In this chapter, we present the method that we used for developing the tool. Our method is divided into four steps that are based on Design science research methodology steps [PTRC07]: 1) Problem identification & Motivation; 2) Objective of the Solution; 3) Requirements gathering and development; 4) Evaluation methods. After describing all these steps we explain how we used the AI language models to assist the writing of this thesis.

3.1 Problem identification & Motivation

The problem was identified as the goal to investigate causal process mining from a more generalizable perspective. E.g. in Bozorgi's *et al.* work [BTD+20], the approach is dependent on the additional dataset variables that could not appear in other business process event logs. The Shoush's *et al.* study [SD22] investigates resource constraints that also could be unknown in simple event logs. While the *activity*, *case ID*, and *timestamp* variables are included in almost event logs, an approach that could give beneficial results from only these variables and has a potential for high applicability. Also having an End-to-End pipeline with Causal discovery combined with Causal estimation algorithms would automate the manual work performed by the data analysts. The requirement to use the generic Causal Inference tool in the study was added because there is a variety of available CI tools [GCL+18]. These tools lack comparison and adaptation research, so we decided to select and use one of them in our solution (the tool selection process is described in the subsection 3.3). Our developed solution tries to cover these points mentioned above by preparing *case ID*, *activity*, *timestamp* event log for the Causal Inference. And then it combines the CD and CE algorithms from the existing tool in order to discover a treatment and its effect.

3.2 Objective of the Solution

The solution objective is to facilitate Causal Inference experiments with real-world event logs. The existing Causal Inference tool provides generic interfaces for the experiments with a selected causal discovery or estimation algorithms, and the data preparation step is left to the users of this tool. Our solution integrates those interfaces and adds a data preparation feature for the *activity* variable encoding. It also expands the functionality by allowing to perform experiments with several selected or all CE and CD algorithms from the generic tool in one pipeline.

3.3 Requirements gathering and development

Functional requirements for the solution were gathered from the problem identification.

Based on the solution objective we divided the development process into three main steps: 1) Selection of the Causal Inference tool; 2) Integration of the selected Causal Inference tool; 3) Development of the code for the data preparation step. The selection of the CI tool started from the causal algorithms and tools source [GCL+18], then inclusion criteria were introduced based on gathered requirements and sources from the related work (see subsection 2.6). While the related work was found using supervisors' knowledge. Then algorithms from the source were investigated based on the introduced inclusion criteria, and the best tool was selected. The integration was done based on the interfaces that were discovered in the selected tool. We wrote the code needed for the integration and data preparation step in the same programming language and technologies as the selected Causal Inference tool in order to maintain consistency and avoid interoperability problems that occur integrating several programming languages in one project. The techniques for the data preparation were selected based on the problem identification and the source [TDRM18] from the related work (see subsection 2.6). The development process was organized using the iterative-incremental model of two weeks cycles for the duration of two months. The feedback was received from the supervisors and improvement based on the feedback was done in the next cycle. The choice of this development organization model was made because of the need to perform initial experiments before the final version of the solution.

3.4 Evaluation methods

The sources for the evaluation technique selection were the problem identification and studies from the related work (see section 2.6). The evaluation step was divided into four substeps: 1) Dataset selection; 2) Metrics selection; 3) The execution of experiments; 4) Results analysis. Dataset selection was started by constructing requirements for the event logs. Then datasets from the related studies [TDRM18, BTD+20] were selected based on those requirements. Metrics selection was done based on the problem identification and requirements for the solution. The execution of experiments was performed using the datasets from the first substep on the developed solution. During the execution metrics from the second substep were calculated. Results analysis consisted of metrics results analysis and comparison with the other studies' findings.

3.5 The use of AI

The AI models: ChatGPT [1], and Grammarly[2], were used to assist in the writing of this thesis. These models were used only to improve the quality of the thesis style and do not contribute to the results of the experiments or the analysis. The use of ChatGPT was done by writing the initial version of paragraphs by ourselves and posting the written paragraph with the query: "Rewrite this paragraph in academic style: <paragraph>". All paragraphs that were written with the assistance of the ChatGPT are marked with a

footnote. Grammarly was utilized in the Overleaf[3] tool in order to check the grammar of the text and fix mistakes. Overleaf was used for the writing and formatting[4] of this thesis.

[1]ChatGPT - a language model whose training is based on a large number of different text sources. ChatGPT is developed by OpenAI. For more information about ChatGPT and OpenAI: https://openai.com.

[2]Grammarly - an online writing tool that provides automated grammar, spelling, and punctuation checking. For more information about Grammarly: https://www.grammarly.com/.

[3]Overleaf - an online platform for collaborative writing and publishing of scientific and technical documents using LaTeX. For more information about Overleaf: https://www.overleaf.com/.

[4]Document format used for this thesis was taken from the template: https://www.overleaf.com/latex/templates/unitartucs-thesis-template/hvprxmvkzywk.

4 Implementation

In this chapter, we present an overview of the proposed evaluation tool. The chapter is divided into two subsections: the Requirements subsection is the design and functionality description of the approach, and the Implemented solution subsection is the overview of the evaluation tool that was implemented to perform experiments and evaluate the approach.

4.1 Requirements

In this subsection, we present the pipeline for the utilization of our End-to-End Causal Inference approach. The pipeline consists of four steps: 1) Input data preparation; 2) Causal discovery; 3) Causal effect estimation; 4) Uplift modeling (see figure 5). In the following paragraphs, we describe in detail each of the steps.

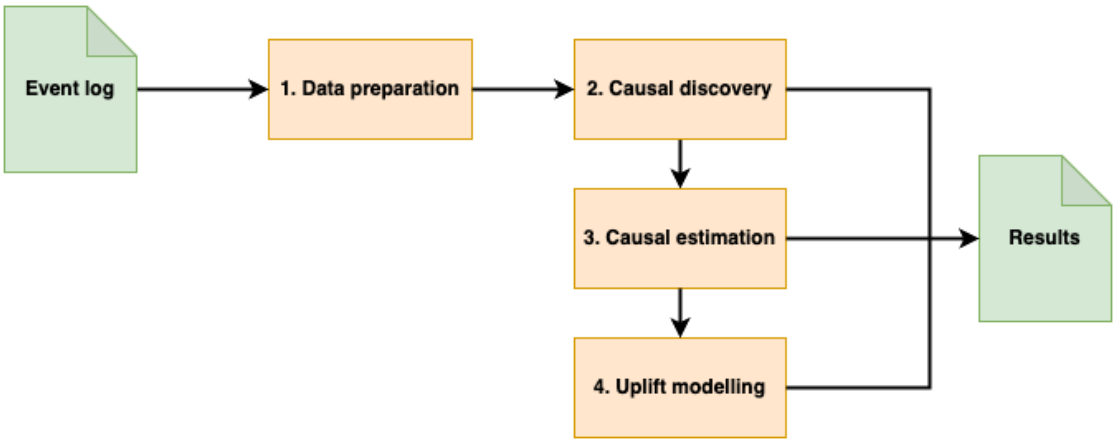


Figure 5. Pipeline steps diagram[1].

4.1.1 Data preparation

This step consists of the data preparation activities, to prepare the data for the next steps of the pipeline. The input of this step is the event log; the output is the train and test data for Causal Inference steps. This step has three substeps: 1.a Event log attributes preparation; 1.b Data encoding; 1.c Data split (see figure 6).

[1] Arrows represent the data flow.

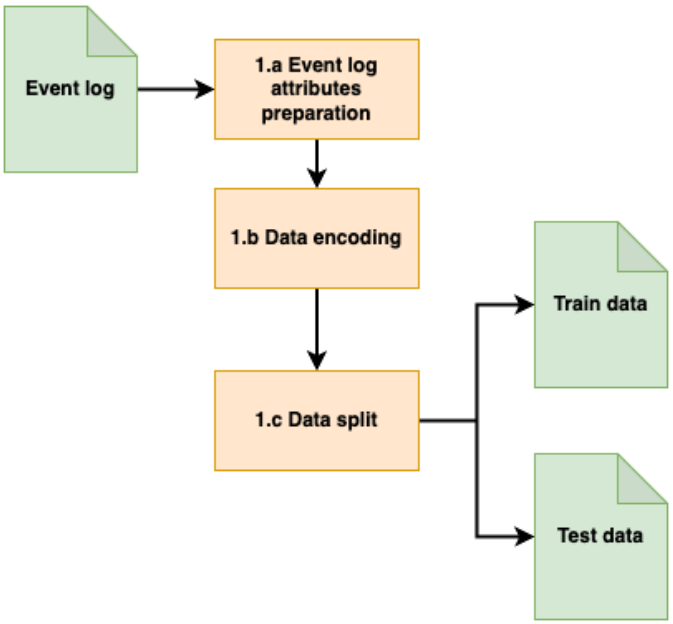


Figure 6. Overview of the Data preparation step[1].

Event log - event log is the initial input of the whole approach. Requirements for the event log were formulated according to the approach:

1. Event log should consist of cases - sequences of activities (events).
2. Every event in the event log should have these attributes (other attributes are optional) and types: *Case ID* (string); *Activity* (string); *Timestamp* (datetime).
3. Event log should be saved in .csv format.

It is worth noting that the absolute majority of all event logs hold these requirements. Additional preparation might be needed to transform the event log to the .csv format. The template of the event log that corresponds to these requirements can be seen in table 1.

[1] Arrows represent the data flow. All data (green document artifacts) is saved in .csv format

Table 1. The template of the event log for the data preparation step.

Case ID	**Activity**	**Timestamp**
CASE_1	ACTIVITY_1_1	TIMESTAMP_1_1
	...	
CASE_1	ACTIVITY_1_M	TIMESTAMP_1_M
	...	
CASE_N	ACTIVITY_N_1	TIMESTAMP_N_1
	...	
CASE_N	ACTIVITY_N_K	TIMESTAMP_N_K

Event log attributes preparation substep - during this substep the *outcome* of the case should be formulated as a binary variable and added to the event log. If the outcome value for case c is 1, then when adding it to the event log, it should be 1 next to all activities of case c. Because the fact that the *outcome* is dependent on the event log, this substep should be implemented separately for each event log. The output of this substep should be the event log enriched with the *outcome* variable. The example of an event log with the *outcome* is visualized in table 2.

Table 2. The template of the event log after the addition of the outcome.

Case ID	**Activity**	**Timestamp**	**Outcome**
CASE_1	ACTIVITY_1_1	TIMESTAMP_1_1	0
	...		
CASE_1	ACTIVITY_1_M	TIMESTAMP_1_M	0
	...		
CASE_N	ACTIVITY_N_1	TIMESTAMP_N_1	1
	...		
CASE_N	ACTIVITY_N_K	TIMESTAMP_N_K	1

Data encoding substep - data encoding substep is needed to encode the categorical *activity* variable. This substep is divided into four smaller phases:

1. Sort the event log.
2. Encode the *activity* variable.
3. Add activities duration to encoded variables by multiplying them by the calculated duration.
4. Aggregate all variables by *case ID*.

In the first phase, the event log should be sorted by *case id*(as the first level of the sort, order: ascending) and *timestamp* (as the second level of the sort; order: ascending).

For the activity encoding phase, the One Hot Encoding method is used. One Hot Encoding is a commonly utilized encoding scheme [PPP17]. It compares each level of the categorical variable to a fixed reference level. One hot encoding transforms a single categorical variable X with n elements and d unique categories, to d new binary variables with n elements each. Each new variable represents one category from the initial X variable. Each element in the new variable specifies the presence (1) or absence (0) of the category in the previous variable X [PPP17].

In the third phase, the activity duration is calculated by subtracting two timestamps of consecutive activities of the same case, the last activity in the case (which does not have the next timestamp to calculate the duration) is set to have a default 0.01 sec. duration. Then encoded variables are multiplied by the calculated activity duration in order to get the duration $d_k > 0.0$ in the encoded activity variable k when the activity k was observed in the case and $d_k = 0.0$ in the encoded activity variable k when the activity k was not observed in the case.

In the fourth phase, we use the activity aggregation method from Teinamaa's *et al.* paper [TDRM18]. We group activities by *case id*, then remove *case id*, *timestamp* variables, and other optional variables that depend on the event, and then aggregate the activities into one row by performing a sum of each case encoded activities durations. After this phase, each column (except the *outcome*) represents distinct activity and each row represents a distinct case. Then if the value in the cell is equal to d and is more than 0.0 it means that the activity took d time in the case and if the d is equal to 0.0, then the activity was not observed in the case.

The example of the end result of the encoding substep is visualized in table 2.

Table 3. The template of the event log after the encoding substep when the dataset has k distinct activities and n distinct cases.

Outcome	**Activity_1**	**...**	**Activity_k**
0	d_{11} [1]	...	0.0 [2]
...			
1	0.0 [2]	...	d_{kn} [3]

Data split - this substep is required to separate the prepared data into training and testing parts, in order to validate the performance of the estimation models. The data

[1] d_{11} - The sum of Activity_1 duration in the first case.

[2] 0.0 - means that the activity in that particular case was not observed.

[3] d_{kn} - The sum of Activity_k duration in the n-th case.

should be shuffled before the split and the proportion for the train and test size is selected depending on the resulting event log size (number of cases).

4.1.2 Causal discovery step

The Causal discovery step is required to learn the causal model of the data by getting a causal graph - especially to capture possible treatment variables. The input of this step is the train data and outcome variable name from the Data preparation step. Then based on the settings one or many Causal discovery algorithms (in the latter option we use the execution of many CD algorithms to compare them) are executed and return the causal graph(-s) as the output of this step (see figure 7). The causal graph is represented as the adjacency matrix of the graph. Where the adjacency matrix is a square matrix G that has v rows and columns, where v is the number of variables from the train data and each variable is represented by the number from 1 to d. Then the matrix element $G_{ij} = 0$ if i and j variables do not have a causal relationship or $G_{ij} > 0$ if i causes j to happen $(1 \leq i, j \leq v)$ [Big93].

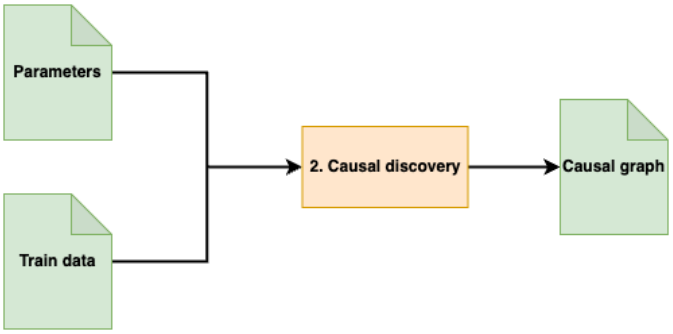


Figure 7. Overview of the Causal discovery step[1].

4.1.3 Causal estimation step

The Causal estimation step is needed to gather the causal effect metrics from the discovered possible treatment. The step is divided into two substeps: a) Fit (or estimator training); b) Estimate (prediction of the trained estimator) (see figure 8).

The input of the Fit substep is the causal graph from the Causal discovery step to give the estimator potential treatment variables and train data with outcome variable name to get the outcome from train data. The Fit substep ends with a fitted estimator that is ready to estimate the causal effect between treatment and outcome and will also be used in Uplift modeling.

[1] Arrows represent the data flow.

The input of the Estimate substep is test data that is prepared in the Data preparation step. Also, it uses the fitted estimator for the causal effect estimation. The output of this substep is Causal effect metrics: ITE and/or ATE (see subsection 2.5).

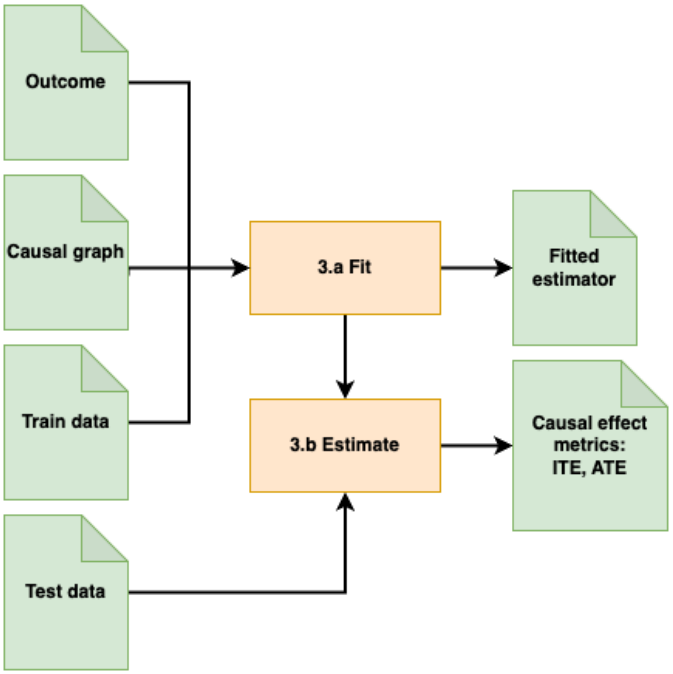


Figure 8. Overview of the Causal estimation step[1].

4.1.4 Uplift modeling step

The Uplift modeling step is needed to draw a *Qini curve* for model validation and comparison. The step is divided into two substeps: a) Select one treatment; b) Perform Uplift modeling (see figure 9).

The input of the "Select one treatment" substep is the causal graph from the Causal discovery step. The causal graph provides the set of treatment variables and during this substep, one discrete treatment variable should be selected (because Uplift modeling is only performed on one discrete treatment variable at the time).

The input of the "Perform Uplift modeling" substep is a Fitted estimator from the Causal estimation step and test data prepared during the Data preparation step.

The input of the estimate step is test data that is prepared in the Data preparation step. The Uplift model uses the Fitted estimator to calculate the *uplift score* and then the

[1] Arrows represent the data flow.

Qini curve for the test data. The output of this substep is Uplift metric: *Qini curve* (see subsection 2.5). After this substep, if there are more possible discrete treatment variables from the causal graph, then the substeps can be repeated again with the new treatment variable until all discovered discrete treatment variables are used in the Uplift modeling.

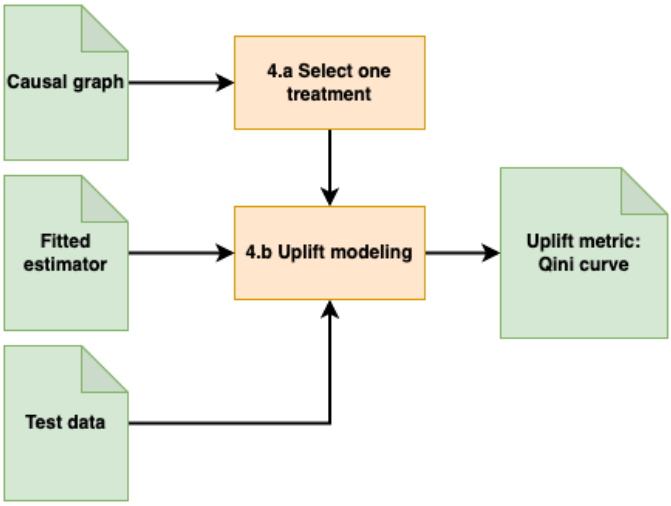


Figure 9. Overview of the Uplift modeling step[1].

4.2 Developed solution

To evaluate the End-to-End pipeline proposed in subsection 4.1 we developed an Evaluation tool that consists of two main components: The Data preparation module and the Causal Inference toolbox. The Data preparation module facilitates the Data encoding and data split substeps from the Data preparation step as the outcome preparation substep should be implemented by the users of the tool for each event log separately. The Causal discovery, Causal estimation, and Uplift modeling steps were implemented using the Ylearn Causal inference library [Yan22]. Finally, the Evaluation tool collects the results from the Causal Inference toolbox, saves them, and returns them to the user (see figure 10). The implemented tool corresponds to the design and functionality description in subsection 4.1.

[1] Arrows represent the data flow.

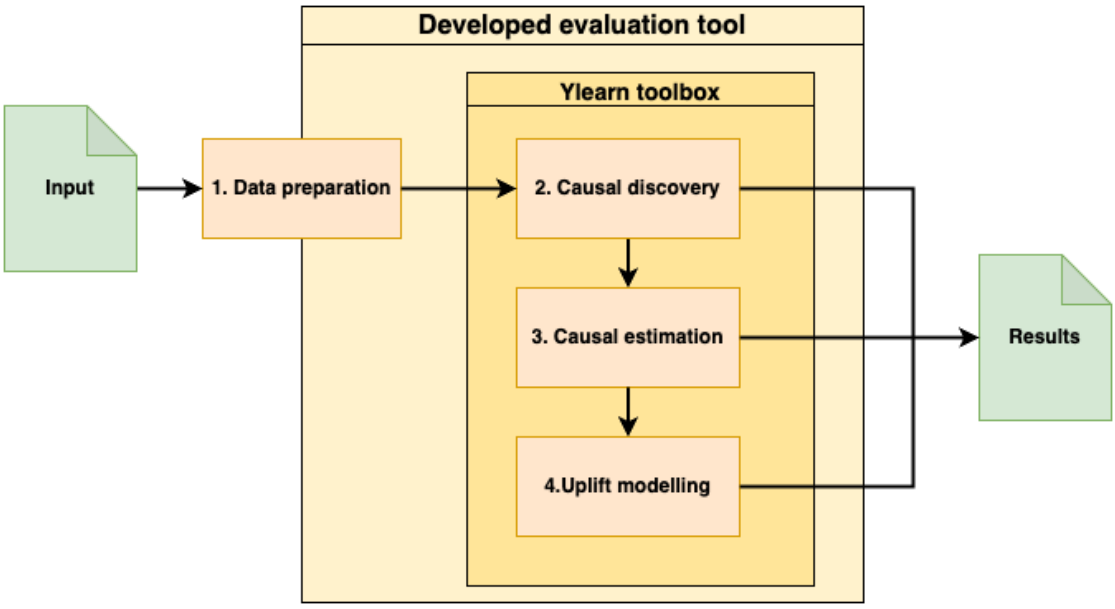


Figure 10. Implemented pipeline components diagram [1].

4.2.1 Technology selection

In this part of the chapter, we list the technology used in the Evaluation tool and the reasons for each selection. Then we describe selected Ylearn tool features.

Causal inference tool selection - to evaluate the approach we needed the tool that would provide the automatic pipeline for the Causal Inference steps: Causal discovery, Causal estimation, and Uplift modeling. Based on the design states in subsection 4.1, we formulated these inclusion criteria for the tool:

1. The tool is not dedicated to investigating only one specific problem - should provide several different options for the same problem.
2. The tool should have Causal Inference (can find the causal graph), Causal estimation (can estimate ATE), and Uplift modeling features.
3. The priority should be given to well-documented tools that provide a clear description of functions and classes and give use examples.
4. The priority should be given to the tools with straightforward installation and setup (provides installation and environment setup instructions).

[1] Arrows represent the data flow.

The main source for the tool search was "A Survey of Learning Causality with Data: Problems and Methods" [GCL+18] which has a list of more than 100 causal algorithms/tools/methods. After the first review of that list, 8 tools that match the first inclusion criterion were selected for further investigation: Trustworthy AI, YLearn, DoWhy, EconML, Uber CausalML, JustCause, WhyNot, scikit-uplift. Then the comparison against the remaining inclusion criteria of those 8 tools was done (see table 4).

Table 4. Comparison of the Causal Inference tools.

Tool name	**Study**	**Link to source code**	**Inclusion criteria**		
			2	**3**	**4**
Trustworthy AI	[ZZK+21]	Python	-	-	+
YLearn	[Yan22]	Python	+	+	+
DoWhy	[KS18]	Python	-	+	+
EconML	[SLO+21]	Python	-	+	+
Uber CausalML	[CHL+20]	Python	-	+	+
JustCause	-	Python	-	-	+
WhyNot	[MHT+20]	Python	-	+	+
scikit-uplift	[MS20]	Python	-	+	+

As presented in the table 4 the Ylearn tool was the only one that matched all inclusion criteria and was selected for the use in Evaluation tool.

Evaluation tool technologies - the evaluation tool was written in Python3.8.16[1] programming language. The selection was done to ensure interoperability with the Ylearn tool. In table 5, we present other Python libraries and tools that were used in the Evaluation tool:

[1] https://www.python.org/downloads/release/python-3816/

Table 5. The overview of the Python libraries or tools used for the Evaluation tool.

Tool or library name	**Reason for use**	**Link**
Conda	Packages management	Documentation
Jupyter notebook	Data preparation and experiments	Documentation
pandas	Data preparation and results	Documentation
numpy	Data preparation	Documentation
scikit-learn	Data encoding and split	Documentation
NetworkX	Causal graphs	Documentation
Matplotlib	Results	Documentation
pytest	For data encoding unit tests	Documentation
Ylearn	Causal discovery, Causal inference, Uplift modeling algorithms	Documentation

4.2.2 Ylearn toolbox

The Ylearn (a pun for "learn why"), is a Python package for CI that supports various aspects of Causal Inference ranging from causal effect identification, estimation, causal graph discovery, etc. It is freely available for use. Release 0.2.0 was used in this research. The documentation can be found: `https://ylearn.readthedocs.io/en/latest/`. The package provides algorithms for Causal discovery and estimation also Uplift modeling and other feature related to Causal Inference. The tool also provides an All-in-One feature, to perform Causal Inference algorithms while only using one Class. This flexible feature was used in our solution. In table 6 we list the Ylearn All-in-One feature methods that were used in our solution.

Table 6. The use of YLearn tool methods in our solution.[1]

Package path	**Method name**	**Use in our solution**
ylearn._why	Why()	Object initialization method to use other interface methods.
ylearn._why.Why	identify()	Used for the Causal discovery step - to discover causal relationships.
ylearn._why.Why	causal_graph()	Used to receive the adjacency matrix of the causal graph.
ylearn._why.Why	fit()	Used to train the estimator.
ylearn._why.Why	causal_effect()	Used to estimate the causal effect.
ylearn._why.Why	uplift_model	Used to get fitted Uplift model
ylearn._model.Uplift-Model	get_qini()	Used to get qini scores for the *Qini curve*.

The Ylearn has a variety of Causal discovery and Causal estimation algorithms to choose from for the experiment. They all were used in our developed solution. In tables 7 and 8 we list the set of CD algorithms that the Ylearn tool provides.

[1]Used Ylearn methods documentation: `https://ylearn.readthedocs.io/en/latest/sub/why.html`

Table 7. The overview of Causal discovery algorithms that Ylearn tool provides (part: I).

Algorithm name	**Description**	**Study**
Notears NotearsNonlinear	NOTEARS (Non-combinatorial Optimization via Trace Exponential and Augmented lagRangian for Structure learning) algorithm introduces a Gradient-based optimization approach and novel characterization of acyclicity for learning the DAG structure. NotearNonlinear is the algorithm version for the non-linear data.	[ZARX18]
ANMNonlinear	Nonlinear function-based causal discovery algorithm with additive noise models.	[HJM$^+$08]
GES	A score-based Greedy Equivalence Search algorithm.	[Chi03]
DirectLiNGAM ICA-LiNGAM	A direct learning function-based and An ICA (Independent component analysis) algorithm for linear non-Gaussian acyclic model (LiNGAM)	[SIS$^+$11, SHHK06]
PC	A classic constraint-based causal discovery Peter-Clark (PC) algorithm which uses conditional independence tests for the DAG structure learning. PC algorithm is useful for problems with a huge number of nodes (variables).	[KB05]
DAG_GNN	DAG gradient based Structure Learning with Graph Neural Networks	[YCGY19]
RL CORL	A Reinforcement Learning-based algorithm that can work with flexible score functions. A CORL- and order-based RL algorithm that improves the efficiency and scalability of the RL-based approach.	[ZNC20, WDZ$^+$21]
GOLEM	A more efficient gradient-based version of NOTEARS that can reduce the number of optimization iterations.	[NGZ21]
MCSL	Masket Causal Structure Learning (MCSL) - a gradient-based algorithm for non-linear additive noise data by learning the binary adjacency matrix.	[NZF$^+$22]
GAE	A gradient-based algorithm using Graph AutoEncoder (GAE) to model non-linear causal relationships	[NZCF19]

Table 8. The overview of Causal discovery algorithms that Ylearn tool provides (part: II).

Algorithm name	**Description**	**Study**
Hill Climb Search	Optimization algorithm that applies local search to estimate the DAG structure that has an optimal score, according to the provided scoring method. Starts with the initial DAG model and does graph modifications step-by-step until a local maximum is reached.	[KF09]
Chow–Liu Tree search	The method uses the Chow Liu algorithm for the estimation of the tree structure, then the DAG is constructed based on tree search learning.	[CL68]
MmhcEstimator	Estimates a BayesianNetwork for the data set, using the Max-Min Hill climb(MMHC) algorithm. First estimates a graph skeleton using Max-Min Parents and Children (MMPC) algorithm and then orients the edges using a score-based local search (hill climbing).	[TBA06]

In tables 9 and 10 we list existing Causal estimation algorithms that can be accessed using the Ylearn tool.

Table 9. The overview of Causal estimation algorithms that Ylearn tool provides (part I).

Algorithm name	**Description**	**Link to the study**
DR	The doubly robust (DR) method estimates the causal effects when the treatment is discrete. Training a doubly robust model is composed of three steps.	[JFWW$^+$11]
ML	Meta-Learners (ML) aim to estimate the ATE when the treatment is discrete by using machine learning models. It provides flexibility in choosing a machine learning model.	[KSBY19]
S-learner	The branch oh Meta-Learners. S(single)-learner uses one machine learning model to estimate the causal effects.	[KSBY19]
T-learner	The branch oh Meta-Learners. T(two)-learner uses two machine learning models to estimate the causal effects.	[KSBY19]
X-learner	T-Learner does not use all data efficiently. This issue can be addressed by the X-learner which utilizes all data to train several models. Training an X-learner is composed of three steps.	[KSBY19]
Causal Tree	Causal Tree is a data-driven approach to grouping the data into parts that differ by the size of their causal effects.	[AI16]
GRF	Generalized Random Forest (GRF) - is a random forest algorithm adaption on causal effect estimation. Performs better in highly flexible non-parametric causal effect estimation.	[AW20]
Approximation bound (Bound)	Many estimator models require the unconfoundedness condition which is not testable. The bound approach builds the upper and lower bounds of causal effects before diving into specific estimations.	[Nea15]

Table 10. The overview of Causal estimation algorithms that Ylearn tool provides (part: II).

Algorithm name	**Description**	**Link to the study**
IV DIV	Instrumental Variables (IV) models are dedicated to the case of estimating causal effects in the presence of unobserved confounding variables that simultaneously have effects on the treatment and the outcome (instrumental variables). Deep Instrumental Variables (DIV) is the model which uses deep learning for instrumental variables estimation.	[New13, HLLBT17]
DML	The double machine learning (DML) model can be applied when all confounders of the treatment and outcome, variables that simultaneously influence the treatment and outcome, are observed.	[CCD$^+$17]

5 Evaluation

The evaluation process was divided into four steps: 1) Dataset selection; 2) Metrics selection; 3) The execution of experiments; 4) Results analysis. In the following subsections, we present the evaluation process and obtained results.

5.1 Dataset and metrics

To select the datasets we formulated these requirements:

1. The event log is about the real-world business process.
2. The event log has *case ID*, *activity*, and *timestamp* variables.
3. The event log has a defined binary *outcome*.
4. The event log has more than 9 but less than 51 distinct activities (to have a variety of possible activities but having too many of them would be unfeasible for the algorithm's execution time).

The datasets that would match those requirements were found in Teinemaa's and Bozorgi's studies [TDRM18, BTD+20]. The selected event logs are listed in the table with the name of their business process and the *outcome* = 1 definition:

Table 11. The list of selected datasets.

Dataset name	**Bussiness process**	**Outcome = 1 definition**
BPIC2017	Loan application	If the case has O_Accepted activity.
BPIC2012	Loan application	If the case has O_ACCEPTED_-COMPLETE activity.
Production	Manufacturing process	The number of rejected work orders is larger than zero.
Sepsis_cases	Patient cases from hospital	The patient returns to the emergency room within 28 days from the discharge.
Traffic_fines	Fines for the violated traffic rules management process	Whether the fine is repaid in full or is sent for credit collection.

To test the Causal discovery results we introduced the Consistency metric that is calculated and interpreted as follows:

1. Causal discovery was done with the prepared dataset.

2. Causal discovery was done a second time with the same data but randomly rearranged variable positions.

3. If Causal discovery returned treatment variables were the same in both tryouts, then the Consistency metric is equal to 1.

4. If the Causal discovery returned treatment variables that did not match in both tryouts, the model just returned random results and the Consistency metric is equal to 0.

If the context of the dataset was known, the Causal discovery results were additionally interpreted according to that context.

To measure Causal estimation we used an ATE metric (see subsection 2.5). To measure the Uplift effect we used the *Qini curve* (see subsection 2.5). After initial tests of the algorithms, we discovered the unequal duration of algorithms execution, so also added the duration in seconds as the metric for algorithms.

5.2 Experimental setup

The experiments were done using the developed tool. Each dataset (see table 11) was executed in the pipeline once. The pipeline of one experiment was divided into these steps: 1) Data preparation; 2) Causal discovery; 3) Causal estimation and Uplift modeling; 4) Results gathering and analysis. All experiments were done on Macbook Air 10,1 M1 chip, 2020, 8 processors, 16 GB RAM, macOS Ventura 13.1 laptop. For the data preparation step, 80% for train and 20% for test split was used. Causal discovery was done with all algorithms available in the Ylearn tool (see table 12) two times (one time with prepared data, and another with the same data but randomly switched variables), in order to check the consistency of the results. The timeout of the algorithm functionality was introduced because some algorithms did not finish - the timeout was chosen to equal 1600 seconds. Causal estimation and Uplift modeling were done once after the results of the Causal discovery were received. Only results with a Consistency metric equal to 1 were tested further with all CE algorithms (see table 12) that are available in the Ylearn tool. The timeout function also was used for 1600 seconds. Then all results were saved (numerical in .csv format, graphs in .png format) and analyzed by drawing *Qini curve* graphs and comparing the performances.

Table 12. List of Causal discovery and Causal estimation algorithms that were used in the evaluation.

CD algorithms		CE algorithms
External package	**Name**	**Name**
gcastle[1]	ANMNonlinear GES DirectLiNGAM ICALiNGAM PC Notears DAG_GNN RL CORL NotearsNonlinear GOLEM MCSL GAE	S-learner T-learner X-learner Causal tree GRF (Generalized Random Forest) Bound (Approximation bound) IV (Instrumental Variables) Deep div (Instrumental Variables) DML (Double Machine Learning) DR (Doubly Robust)
pgmpy [2]	ExhaustiveSearch HillClimbSearch TreeSearch MmhcEstimator PC	
-[3]	Notears	

5.3 Results

In the following subsections, we present the results of the experiments.

5.3.1 BPIC2017

Firstly, individual preparation was done - to prepare the dataset to the one that has Case ID, Activity, and Timestamp columns and is in .csv format. The BPIC2017 has 26 unique activities, the average duration of the case is 38 activities. The data was encoded, prepared, and split into train and test parts, then put into the Causal discovery step which

[1]gcastle - the external Python package that the Ylearn tool uses for its CD algorithms [ZZK+21].
[2]pgmpy - the external Python package that the Ylearn tool uses for its CD algorithms [AP15].
[3]Notears CD algorithm version that is the only one built-in the Ylearn tool.

was done two times to check if the position of variables does not affect the result. We present discovery results in table 13.

Table 13. BPIC2017 Causal discovery step results.

Discovery algorithm	**Treatment result A**	**Treatment result B**	**C**[1]	**T**[2]
DirectLiNGAM				20.7
ICALiNGAM		Activity_O_Sent (mail		3.9
gcastle-Notears	Activity_A_Complete	and online)		5.3
NotearsNonlinear	Activity_A_Accepted	Activity_O_Create Offer	**0**	153.6
GOLEM	Activity_A_Cancelled			1122.9
pgm-PC				71.5
Notears	Activity_W_Call incomplete files, Activity_W_Call after offers, Activity_W_Validate application	Activity_W_Call incomplete files, Activity_W_Call after offers, Activity_W_Validate application	**1**	809.9
TreeSearch	Activity_W_Call after offers	Activity_W_Call after offers	**1**	1.5
ANMNonlinear GES DAG_GNN RL gcastle-PC CORL MCSL GAE ExhaustiveSearch MmhcEstimator HillClimbSearch	Terminated after timeout	Terminated after timeout	**-**	**-**

The results showed that 11 of 19 algorithms were terminated because of the timeout. Another 6 showed inconsistent results and the Notears (which is the only built-in Ylearn discovery algorithm) and TreeSearch algorithms were consistent by testing the

[1]Consistency metric.

[2]Average duration of algorithm execution in seconds.

data with shuffled variable positions. The Notears returned possible treatments in "Activity_W_Call incomplete file", "Activity_W_Call after offer" and "Activity_Validate application". The "TreeSearch" returned "Activity_W_Call after offer". Resulting causal graphs are presented in Appendix I.

The BPIC2017 dataset was analyzed by other researchers and their study provides additional information about this dataset. Rodrigues' *et al.* in their paper [RAS+17], did a descriptive analysis of the BPIC2017 dataset and presented recommendations, on how to improve the business process. Their analysis showed that more interaction with the customer results in more accepted offers. Activities that involve interaction with the customer are "Activity_W_Call incomplete files" and "Activity_A_Call after offers". "Activity_W_Validate". According to the authors [RAS+17] those activities are dependent on the customer as the bank waits for the applicant to send the requested documents before starting the validation step. This results in the longest durations that influence the case time and also the success rate. So, the discovery of these activities as possible treatments seems logical. However, the data analyst still has to manually check the discovery step results if it is possible to make interventions in discovered treatments because a bigger part of algorithms returned inconsistent results.

After the Causal discovery step. We performed the Causal estimation step with found consistent treatment variables. In table 14 and figures 11 and 12 we present BPIC2017 Causal estimation results.

Table 14. BPIC2017 Causal estimation step results.

Estimator	**Treatment name**	**ATE**	**Avg time (sec)**[1]
T-learner	Activity_W_Call incomplete files Activity_W_Call after offers Activity_W_Validate application	0.035 0.038 0.036	887.9
Causal tree	Activity_W_Call incomplete files Activity_W_Call after offers Activity_W_Validate application	0.426 0.415 0.423	5.5
GRF	Activity_W_Call incomplete files Activity_W_Call after offers Activity_W_Validate application	0.056 0.047 0.051	270.8
X-learner Bound DML IV DIV	Terminated after error	-	-
S-learner DR	Terminated after timeout	-	-

Five algorithms terminated because of the error. The reasons for the error are that additional input was needed for the model or that the error happened inside the package. Two algorithms terminated because of the timeout (1600 seconds). T-learner, Causal tree, and GRF (Generalized Random Forest) algorithms returned ATE results for each treatment. The best ATE was measured by the Causal tree model while T-learner and GRF showed similar results. All three models returned the biggest ATE for "Activity_W_Call after offer" treatment, then "Activity_W_Validate", and lowest to "Activity_W_Call incomplete files". We interpret these results as the following: positive ATE of these activities shows that having those activities in the case would give a slightly better outcome than without them.

[1] Average duration of algorithm execution

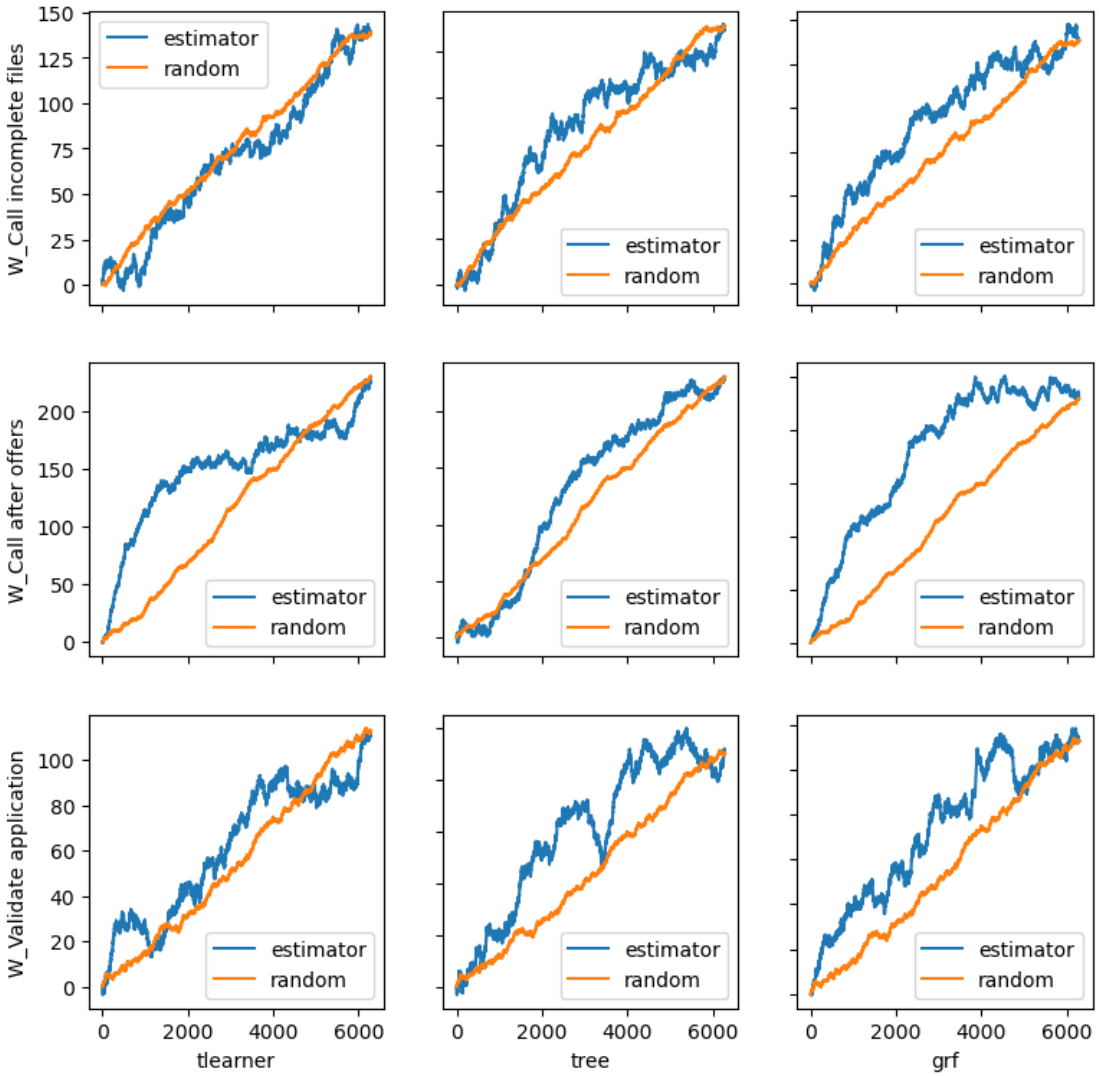


Figure 11. BPIC2017 data Uplift modeling Qini curve results.

Qini curve results also confirm this observation. In the figure 11 "W_Call incomplete files" activity *Qini curve* of all estimators is close to the random model while the best results can be seen in "W_Call after offers" activity estimations. In figure 12 all estimators are compared with each other. GRF estimator showed the best *Qini scores* in "W_Call incomplete files" and "W_Call after offers" and was between the two best in "Activity_W_Validate" treatment.

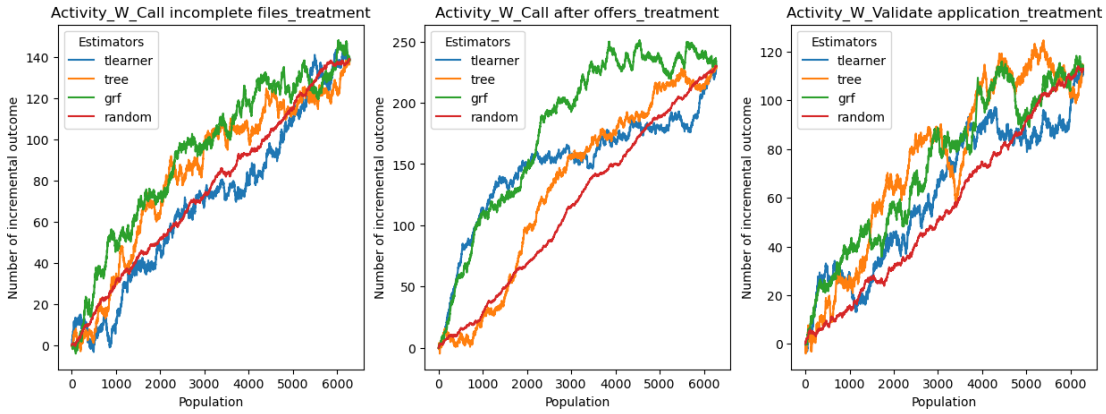


Figure 12. BPIC2017 data estimator comparison by Qini curve.

5.3.2 Comparison with the "Process mining meets causal machine learning" study results

Bozorgi *et al.* also evaluated BPIC2017 data in their study. Their approach consisted of: 1) Identifying candidate treatments; 2) Causal Rules Discovery; 3) Ranking Rules Using a Cost-Benefit Model. In the end, their approach suggests eight actions to improve the BPIC2017 dataset outcome. The actions correspond to treating the withdrawal amount and a number of terms variables based on sub-populations of the cases. We decided to evaluate their approach by performing Causal discovery with our evaluation tool on their variables and constructing a treatment variable that would match their recommended actions. Then we calculated the ATE value and compared it to our result. To achieve this goal we took a BPIC2017 dataset with Case ID, Outcome, Loan Goal, Application Type, Credit Score, Offered Amount, Number Of Terms, Monthly Cost, and First Withdrawal Amount variables. Then we constructed continuous treatment values "First Withdrawal Amount treatment" and "Number of terms treatment" which correspond to the recommended actions from Bozorgi's *et al.* study [BTD+20]. Then we evaluated the treatment in Ylearn continuous treatment models: DML (Double Machine learning) and IV (Instrumental variables) because other Ylearn algorithms do not support the treatment with continuous values. We calculated the results of ATE metric and presented them in the table 15.

Table 15. ATE results from applying recommended actions from Bozorgi's *et al.* paper [BTD+20] on Ylearn tool's CE algorithms.

Estimator	**Withdrawal amount ATE**	**Number of terms ATE**
DML	0.0030	-0.069
IV	0.0012	-0.024

We could not draw the *Qini curve* for the models because of the limitation of the Ylearn tool (it cannot apply the uplift model on continuous treatment). The best ATE that we received by applying the treatment from Bozorgi's *et al.* approach was 0.003 while from our approach it was 0.426. Respectively, the worst ATE was -0.069 while from our approach it was 0.035.

5.3.3 Results of other datasets

Causal discovery consistency results of other datasets are presented in Appendix II. Causal estimation results and *Qini curves* of other datasets are presented in Appendix III.

5.4 Threats to validity

The validation described above comes with threats to internal validity. The results may be influenced by data preparation or the limited knowledge about the event log context (e.g. meaning and the use of the activities). This threat is reduced by comparing part (BPIC2017 data) of the experiments with other studies which provided more information about the data. By the use of *activity*, *case ID* and *timestamp* variables that are commonly found in most event logs our approach has a good generalizability. We acknowledge that there could be a potential threat to reliability because our evaluation tool was largely dependent on the Ylearn tool. To address this issue and combat the reproducibility crisis we provide a software artifact that allows other researchers to recreate our evaluation process and test their own data. The developed evaluation tool is saved in the repository (`https://github.com/lukasbaltramaitis/CI_Experiments`). The instructions on how to recreate the experiments are described in the README.md file.

6 Conclusion

Our proposed approach aims to enhance business processes by automatically discovering potential treatments from event logs and estimating their effects. This approach includes both Causal discovery and Causal estimation into a unified solution. To achieve this, we select a minimal set of variables that are commonly available in any business process event log, namely *case ID*, *activity*, and *timestamp*. The approach integrates the existing causal tool Ylearn, adapting it to the context of Causal process mining. Using this developed tool enables researchers and data analysts to apply causal analysis techniques to process logs, leading to valuable insights for process improvement. To validate the effectiveness of our approach, we conducted an empirical evaluation using six real-life event logs from diverse business domains. This evaluation provides tangible evidence of the applicability and benefits of our approach in real-world settings, further reinforcing its potential for improving business processes.

While our work has made significant progress in automating the discovery and estimation of causal relationships from event logs, it still requires human intervention to validate the results of Causal discovery. This limitation opens up opportunities for future research to explore ways to reduce or eliminate the need for manual validation[1].

One potential direction for future research is to incorporate human feedback into the pipeline, enabling real-time evaluation of the discovered treatments. This can be achieved by integrating reinforcement learning techniques, allowing the system to learn from human judgments and continuously improve the effectiveness of the discovered treatments[1].

Furthermore, expanding the scope of our analysis by exploring alternative causal tools would be valuable. Comparing different causal analysis tools with the adapted Ylearn tool can help determine which tool provides better results in terms of improving business processes. This comparative analysis would contribute to a better understanding of the strengths and limitations of various tools and guide the selection of the most appropriate tool for different contexts and datasets[1].

Finally, the validation of the tool is only done by the authors of this thesis and does not have the feedback of the possible users. Doing more extensive testing and validation with new data configurations could be another future step of this work.

[1]Marked paragraphs were written from a combined effort of the student and ChatGPT (06.04.2023), i.e. a language model whose training is based on a large number of different text sources. ChatGPT is developed by OpenAI. For more information about ChatGPT and OpenAI: `https://openai.com`

References

[AI16] Susan Athey and Guido Imbens. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27):7353–7360, jul 2016.

[AP15] Ankur Ankan and Abinash Panda. pgmpy: Probabilistic graphical models using python. In *Proceedings of the 14th Python in Science Conference (SCIPY 2015)*. Citeseer, 2015.

[AW20] Susan Athey and Stefan Wager. Policy learning with observational data, 2020.

[Bay15] Tuncay Bayrak. A review of business analytics: A business enabler or another passing fad. *Procedia - Social and Behavioral Sciences*, 195:230–239, 2015. World Conference on Technology, Innovation and Entrepreneurship.

[Big93] N. Biggs. *Algebraic Graph Theory*. Cambridge University Press, 2 edition, 1993. Definition 2.1.

[BTD+20] Zahra Dasht Bozorgi, Irene Teinemaa, Marlon Dumas, Marcello La Rosa, and Artem Polyvyanyy. Process mining meets causal machine learning: Discovering causal rules from event logs, 2020.

[CCD+17] Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. Double/debiased machine learning for treatment and causal parameters, 2017.

[Chi03] David Maxwell Chickering. Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3(null):507–554, mar 2003.

[CHL+20] Huigang Chen, Totte Harinen, Jeong-Yoon Lee, Mike Yung, and Zhenyu Zhao. Causalml: Python package for causal machine learning, 2020.

[CL68] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

[CVMP23] Martin Cousineau, Vedat Verter, Susan A. Murphy, and Joelle Pineau. Estimating causal effects with optimization-based methods: A review and empirical comparison. *European Journal of Operational Research*, 304(2):367–380, jan 2023.

[DMV18] Floris Devriendt, Darie Moldovan, and Wouter Verbeke. A literature survey and experimental evaluation of the state-of-the-art in uplift modeling: A stepping stone toward the development of prescriptive analytics. *Big Data*, 6(1):13–41, 2018. PMID: 29570415.

[GAF+22] Tomas Geffner, Javier Antoran, Adam Foster, Wenbo Gong, Chao Ma, Emre Kiciman, Amit Sharma, Angus Lamb, Martin Kukla, Nick Pawlowski, Miltiadis Allamanis, and Cheng Zhang. Deep end-to-end causal inference, 2022.

[GCL+18] Ruocheng Guo, Lu Cheng, Jundong Li, Paul Hahn, and Huan Liu. A survey of learning causality with data: Problems and methods, 09 2018.

[GTFNZ+23] Daniela Galatro, Rosario Trigo-Ferre, Allana Nakashook-Zettler, Vincenzo Costanzo-Alvarez, Melanie Jeffrey, Maria Jácome, Jason Bazylak, and Cristina Amon. Framework for evaluating potential causes of health risk factors using average treatment effect and uplift modelling. *Algorithms*, 16:166, 03 2023.

[HJM+08] Patrik O. Hoyer, Dominik Janzing, Joris Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, page 689–696, Red Hook, NY, USA, 2008. Curran Associates Inc.

[HLLBT17] Jason Hartford, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy. Deep IV: A flexible approach for counterfactual prediction. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1414–1423. PMLR, 06–11 Aug 2017.

[JFWW+11] Michele Jonsson Funk, Daniel Westreich, Chris Wiesen, Til Stürmer, M Brookhart, and Marie Davidian. Doubly robust estimation of causal effects. *American journal of epidemiology*, 173:761–7, 03 2011.

[KB05] Markus Kalisch and Peter Buehlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm, 2005.

[KF09] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*, pages 811–824. The MIT Press, 2009.

[KMND21] Kateryna Kubrak, Fredrik Milani, Alexander Nolte, and Marlon Dumas. Prescriptive process monitoring: Quo vadis?, 12 2021.

[KS18] Emre Kiciman and Amit Sharma. Tutorial on causal inference and counterfactual reasoning. In *ACM KDD International Conference on Knowledge Discovery and Data Mining*, August 2018.

[KSBY19] Sören R. Künzel, Jasjeet S. Sekhon, Peter J. Bickel, and Bin Yu. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the National Academy of Sciences*, 116(10):4156–4165, feb 2019.

[LBAM20] Katerina Lepenioti, Alexandros Bousdekis, Dimitris Apostolou, and Gregoris Mentzas. Prescriptive analytics: Literature review and research challenges. *International Journal of Information Management*, 50:57–70, 2020.

[MHT+20] John Miller, Chloe Hsu, Jordan Troutman, Juan Perdomo, Tijana Zrnic, Lydia Liu, Yu Sun, Ludwig Schmidt, and Moritz Hardt. Whynot, 2020.

[MS20] Irina Elisova Maksim Shevchenko. User guide for uplift modeling and casual inference. `https://www.uplift-modeling.com/en/latest/user_guide/index.html`, 2020.

[Nea15] Brady Neal. Introduction to causal inference. 2015.

[Nea20] Brady Neal. Introduction to causal inference. 2020.

[New13] Whitney K. Newey. Nonparametric instrumental variables estimation. *The American Economic Review*, 103(3):550–556, 2013.

[NGZ21] Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the role of sparsity and dag constraints for learning linear dags, 2021.

[NPR+22] Ana Nogueira, Andrea Pugnana, Salvatore Ruggieri, Dino Pedreschi, and João Gama. Methods and tools for causal discovery and causal inference. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 12, 03 2022.

[NW23] Ashley I Naimi and Brian W Whitcomb. Defining and Identifying Average Treatment Effects. *American Journal of Epidemiology*, 01 2023. kwad012.

[NZCF19] Ignavier Ng, Shengyu Zhu, Zhitang Chen, and Zhuangyan Fang. A graph autoencoder approach to causal structure learning, 2019.

[NZF+22] Ignavier Ng, Shengyu Zhu, Zhuangyan Fang, Haoyang Li, Zhitang Chen, and Jun Wang. Masked gradient-based causal structure learning, 2022.

[PPP17] Kedar Potdar, Taher Pardawala, and Chinmay Pai. A comparative study of categorical variable encoding techniques for neural network classifiers. *International Journal of Computer Applications*, 175:7–9, 10 2017.

[PTRC07] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, and S. Chatterjee. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24:45–77, 01 2007.

[RAS+17] Ariane Moraes Bueno Rodrigues, Cassio F. P. Almeida, Daniel D. G. Saraiva, Benítez Felipe, Moreira, George Miguel Spyrides, Guilherme Varela, Gustav Krieger, Tsarkov Igor, Peres, Leila Figueiredo Dantas, Mauricio Lana, Odair E. Alves, Rafael França, Ricardo, A. Neira, Sonia F. Gonzalez, William Fernandes, Simone Diniz Junqueira Barbosa, Marcus Poggi, and Hélio Côrtes Vieira Lopes. Stairway to value : mining a loan application process. 2017.

[SD22] Mahmoud Shoush and Marlon Dumas. Prescriptive process monitoring under resource constraints: A causal inference approach. In Jorge Munoz-Gama and Xixi Lu, editors, *Process Mining Workshops*, pages 180–193, Cham, 2022. Springer International Publishing.

[SHHK06] Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-gaussian acyclic model for causal discovery. 7:2003–2030, dec 2006.

[SIS+11] Shohei Shimizu, Takanori Inazumi, Yasuhiro Sogawa, Aapo Hyvarinen, Yoshinobu Kawahara, Takashi Washio, Patrik O. Hoyer, and Kenneth Bollen. Directlingam: A direct method for learning a linear non-gaussian structural equation model, 2011.

[SJS17] Uri Shalit, Fredrik D. Johansson, and David Sontag. Estimating individual treatment effect: generalization bounds and algorithms, 2017.

[SLO+21] Vasilis Syrgkanis, Greg Lewis, Miruna Oprescu, Maggie Hei, Keith Battocchi, Eleanor Dillon, Jing Pan, Yifeng Wu, Paul Lo, Huigang Chen, Totte Harinen, and Jeong-Yoon Lee. Causal inference and machine learning in practice with econml and causalml: Industrial use cases at microsoft, tripadvisor, uber. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery Data Mining*, KDD '21, page 4072–4073, New York, NY, USA, 2021. Association for Computing Machinery.

[TBA06] Ioannis Tsamardinos, Laura Brown, and Constantin Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 10 2006.

[TDRM18] Irene Teinemaa, Marlon Dumas, Marcello La Rosa, and Fabrizio Maria Maggi. Outcome-oriented predictive process monitoring: Review and benchmark, 2018.

[Tho23] Lauren Thomas. Confounding Variables | Definition, Examples 038; Controls. *Scribbr*, 4 2023.

[WDZ+21] Xiaoqiang Wang, Yali Du, Shengyu Zhu, Liangjun Ke, Zhitang Chen, Jianye Hao, and Jun Wang. Ordering-based causal discovery with reinforcement learning, 2021.

[WPRM22] Philipp Waibel, Lukas Pfahlsberger, Kate Revoredo, and Jan Mendling. Causal process mining from relational databases with domain knowledge, 2022.

[Yan22] Bochen LyuXuefeng LiJian Yang. YLearn: A Python Package for Causal Inference. https://github.com/DataCanvasIO/YLearn, 2022. Version 0.2.x.

[YCGY19] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. Dag-gnn: Dag structure learning with graph neural networks, 2019.

[ZARX18] Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. Dags with no tears: Continuous optimization for structure learning, 2018.

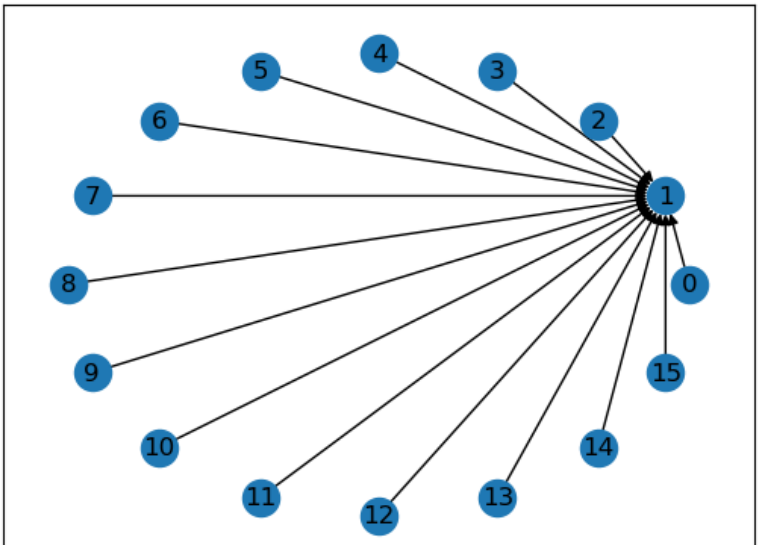
[Zha17] Peter Zhang. Estimation of average treatment effects honors thesis. 2017.

[ZNC20] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. Causal discovery with reinforcement learning, 2020.

[ZZK+21] Keli Zhang, Shengyu Zhu, Marcus Kalander, Ignavier Ng, Junjian Ye, Zhitang Chen, and Lujia Pan. gcastle: A python toolbox for causal discovery, 2021.

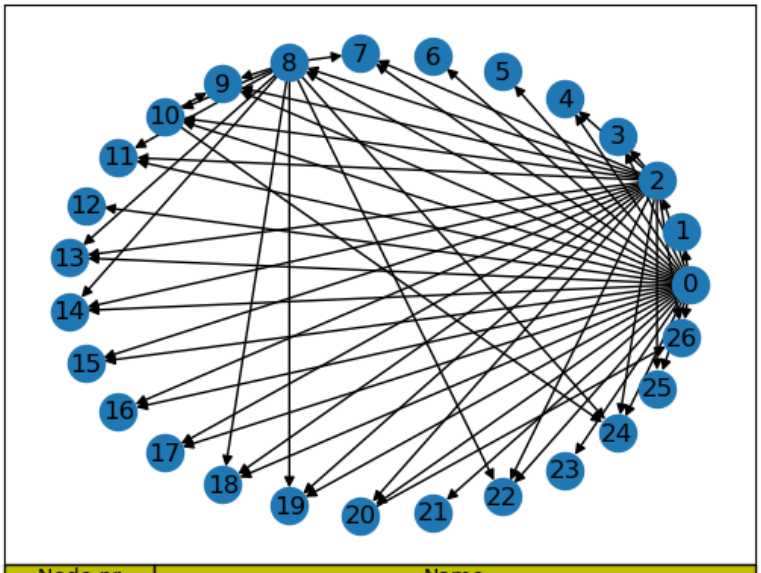
Appendices

I BPIC2017 Causal graphs



Node nr	Name
0	Activity_O_Accepted
1	Outcome
2	Activity_A_Accepted
3	Activity_A_Cancelled
4	Activity_A_Complete
5	Activity_A_Concept
6	Activity_A_Create Application
7	Activity_A_Denied
8	Activity_A_Incomplete
9	Activity_A_Pending
10	Activity_A_Submitted
11	Activity_A_Validating
12	Activity_O_Accepted
13	Activity_O_Cancelled
14	Activity_O_Create Offer
15	Activity_O_Created
16	Activity_O_Refused
17	Activity_O_Returned
18	Activity_O_Sent (mail and online)
19	Activity_O_Sent (online only)
20	Activity_W_Assess potential fraud
21	Activity_W_Call after offers
22	Activity_W_Call incomplete files
23	Activity_W_Complete application
24	Activity_W_Handle leads
25	Activity_W_Personal Loan collection
26	Activity_W_Shortened completion
27	Activity_W_Validate application

Figure 13. BPIC2017 dataset Notears algorithm discovered causal graph.



Node nr	Name
0	Activity_O_Accepted
1	Activity_A_Accepted
2	Activity_A_Cancelled
3	Activity_A_Complete
4	Activity_A_Concept
5	Activity_A_Create Application
6	Activity_A_Denied
7	Activity_A_Incomplete
8	Activity_A_Pending
9	Activity_A_Submitted
10	Activity_A_Validating
11	Activity_O_Accepted
12	Activity_O_Cancelled
13	Activity_O_Create Offer
14	Activity_O_Created
15	Activity_O_Refused
16	Activity_O_Returned
17	Activity_O_Sent (mail and online)
18	Activity_O_Sent (online only)
19	Activity_W_Assess potential fraud
20	Activity_W_Call after offers
21	Activity_W_Call incomplete files
22	Activity_W_Complete application
23	Activity_W_Handle leads
24	Activity_W_Personal Loan collection
25	Activity_W_Shortened completion
26	Activity_W_Validate application
27	Outcome

Figure 14. BPIC2017 dataset TreeSearch algorithm discovered causal graph.

II Datasets Causal discovery results

Table 16. Discovery step consistency and duration results of all (except BPIC2017) datasets.

Discovery algorithm	**BPIC2012**		**Production**		**Sepsis cases**		**Traffic fines**	
	C[1]	**T**[2]	**C**[1]	**T**[2]	**C**[1]	**T**[2]	**C**[1]	**T**[2]
DirectLiNGAM	0	12.3	0	1.9	0	0.7	0	8.3
ICALiNGAM	0	1.3	0	0.26	0	0.4	0	3.7
gcastle-PC	0	16.0	**1**	0.08	0	0.2	0	19.2
gcastle-Notears	0	5.0	0	1.5	0	1.6	0	5.6
NotearsNonlinear	0	78.8	0	3.5	0	12.7	0	84.5
GOLEM	0	511.4	0	97.0	0	75.0	0	1364.0
pgm-PC	-[3]	-[3]	0	49.1	0	81.6	**1**	1644.8
TreeSearch	**1**	0.7	**1**	0.4	**1**	0.2	**1**	0.4
Notears	**1**	425.8	0	39.6	**1**	26.8	0	535.0
ANMNonlinear GES DAG_GNN RL CORL MCSL GAE ExhaustiveSearch MmhcEstimator HillClimbSearch	Terminated after timeout							

[1]Consistency between two experiments with different variable positions.
[2]Average execution time in seconds.
[3]Terminated because of the timeout.

Table 17. Discovered treatments.

Dataset	**Discovered treatments**
BPIC2012	Activity_W_Nabellen offertes-START, Activity_W_Nabellen offertes-SCHEDULE, Activity_O_SENT-COMPLETE, Activity_W_Nabellen offertes-COMPLETE
Production	Activity_Turning & Milling - Machine 9, Activity_Turning & Milling - Machine 4, Activity_Flat Grinding - Machine 11, Activity_Turning & Milling Q.C
Sepsis cases	Activity_CRP, Activity_Admission NC
Traffic fines	Activity_Add penalty, Activity_Send Fine, Activity_Create Fine

III Datasets Causal estimation results

Table 18. BPIC2012 dataset Causal estimation step results.

Estimator	**Treatment name**	**ATE**	**T**[1]
S-learner	Activity_W_Nabellen offertes-START	0.010	
	Activity_W_Nabellen_offertes-SCHEDULE	0.030	359.5
	Activity_O_SENT-COMPLETE	-0.013	
	Activity_W_Nabellen_offertes-COMPLETE	-0.006	
T-learner	Activity_W_Nabellen offertes-START	0.021	
	Activity_W_Nabellen_offertes-SCHEDULE	-0.004	359.5
	Activity_O_SENT-COMPLETE	-0.011	
	Activity_W_Nabellen_offertes-COMPLETE	0.061	
Causal tree	Activity_W_Nabellen offertes-START	0.513	
	Activity_W_Nabellen_offertes-SCHEDULE	0.459	0.8
	Activity_O_SENT-COMPLETE	0.454	
	Activity_W_Nabellen_offertes-COMPLETE	0.491	
GRF	Activity_W_Nabellen offertes-START	0.044	
	Activity_W_Nabellen_offertes-SCHEDULE	0.056	62.0
	Activity_O_SENT-COMPLETE	0.031	
	Activity_W_Nabellen_offertes-COMPLETE	0.023	
DR	Activity_W_Nabellen offertes-START	-0.035	
	Activity_W_Nabellen_offertes-SCHEDULE	0.061	1663.9
	Activity_O_SENT-COMPLETE	0.005	
	Activity_W_Nabellen_offertes-COMPLETE	0.028	
X-learner Bound DML IV DIV	Terminated after error	-	-

[1] Average duration of algorithm execution in seconds.

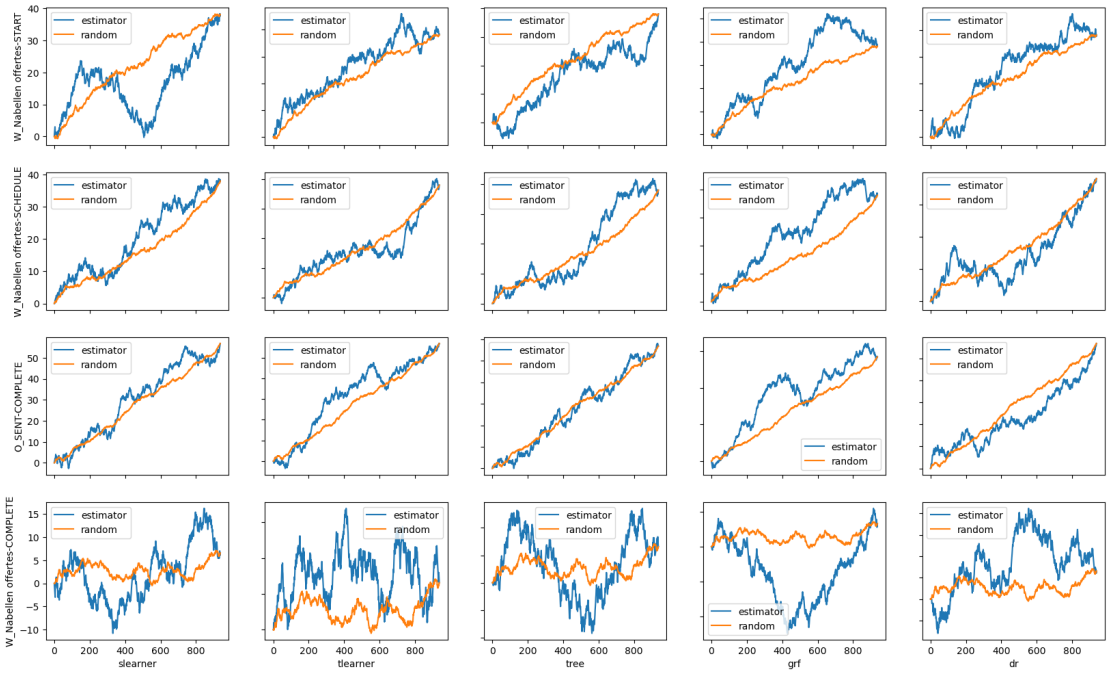


Figure 15. BPIC2012 dataset Uplift modeling Qini curve results.

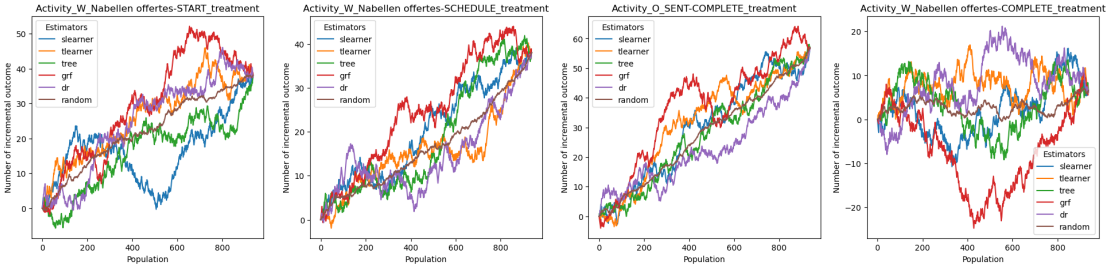


Figure 16. BPIC2012 dataset estimator comparison by Qini curve.

Table 19. Production dataset Causal estimation step results.

Estimator	**Treatment name**	**ATE**	**T**[1]
S-learner	Activity_Turning & Milling - Machine 9	0.01	
	Activity_Turning & Milling - Machine 4	0.034	11.4
	Activity_Flat Grinding - Machine 11	-0.020	
	Activity_Turning & Milling Q.C	0.089	
Causal tree	Activity_Turning & Milling - Machine 9	0.810	
	Activity_Turning & Milling - Machine 4	0.540	0.1
	Activity_Flat Grinding - Machine 11	0.182	
	Activity_Turning & Milling Q.C	0.503	
GRF	Activity_Turning & Milling - Machine 9	0.049	
	Activity_Turning & Milling - Machine 4	0.065	2.4
	Activity_Flat Grinding - Machine 11	-0.120	
	Activity_Turning & Milling Q.C	0.048	
X-learner Bound DML IV DIV	Terminated after error	-	-
T-learner DR	Terminated after timeout	-	-

[1]Average duration of algorithm execution in seconds.

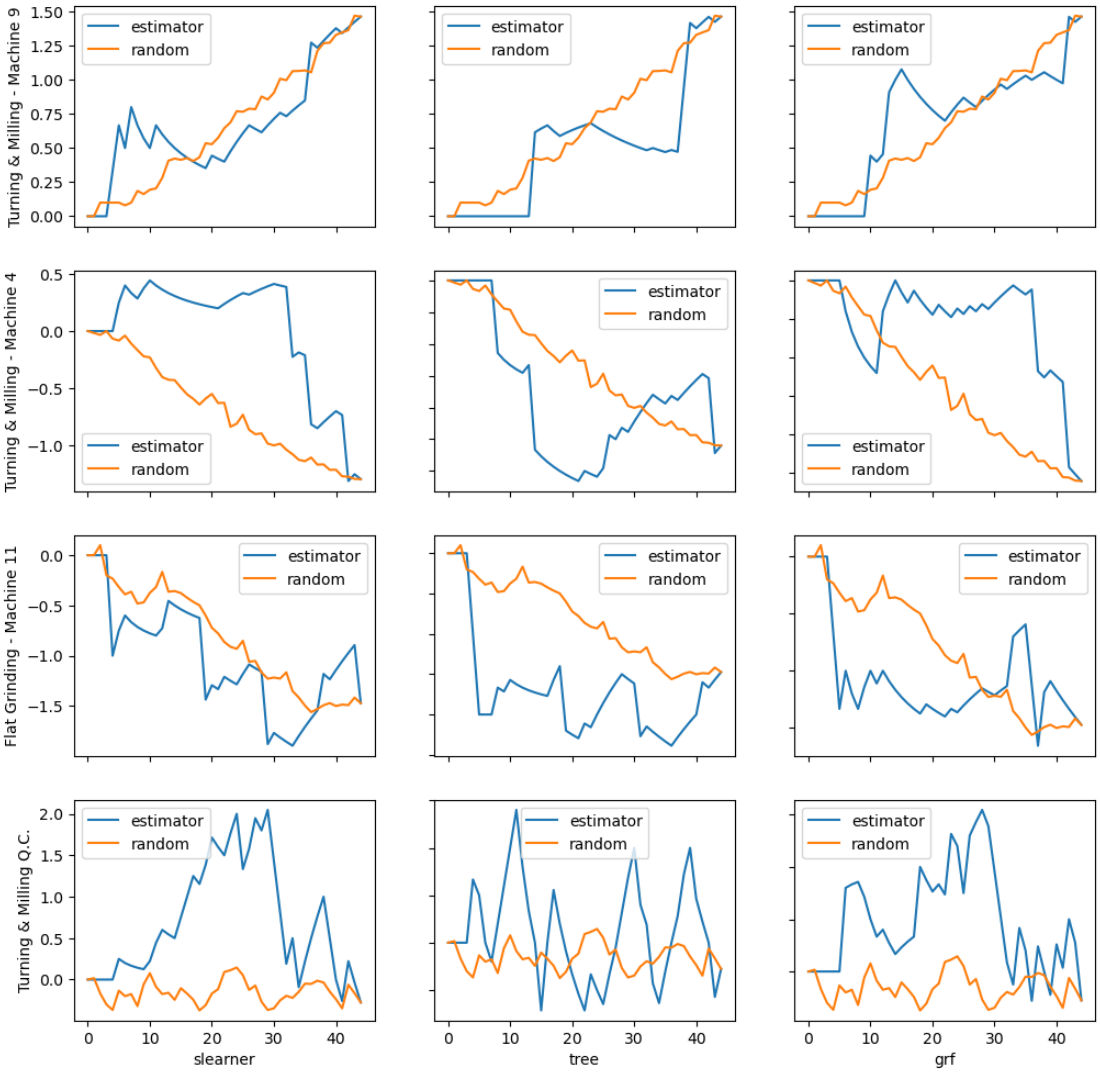


Figure 17. Production dataset Uplift modeling Qini curve results.

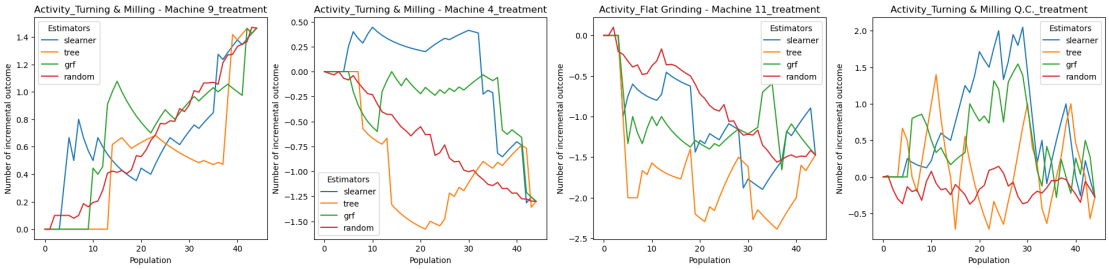


Figure 18. Production dataset estimator comparison by Qini curve.

Table 20. Sepsis cases dataset Causal estimation step results.

Estimator	**Treatment name**	**ATE**	**T**[1]
S-learner	Activity_CRP Activity_Admission NC	-0.012 -0.030	5.1
T-learner	Activity_CRP Activity_Admission NC	0.001 0.014	2.9
Causal tree	Activity_CRP Activity_Admission NC	0.140 0.134	0.1
GRF	Activity_CRP Activity_Admission NC	0.026 0.021	1.6
DR	Activity_CRP Activity_Admission NC	0.000 -0.003	8.3
X-learner Bound DML IV DIV	Terminated after error	-	-

[1]Average duration of algorithm execution in seconds.

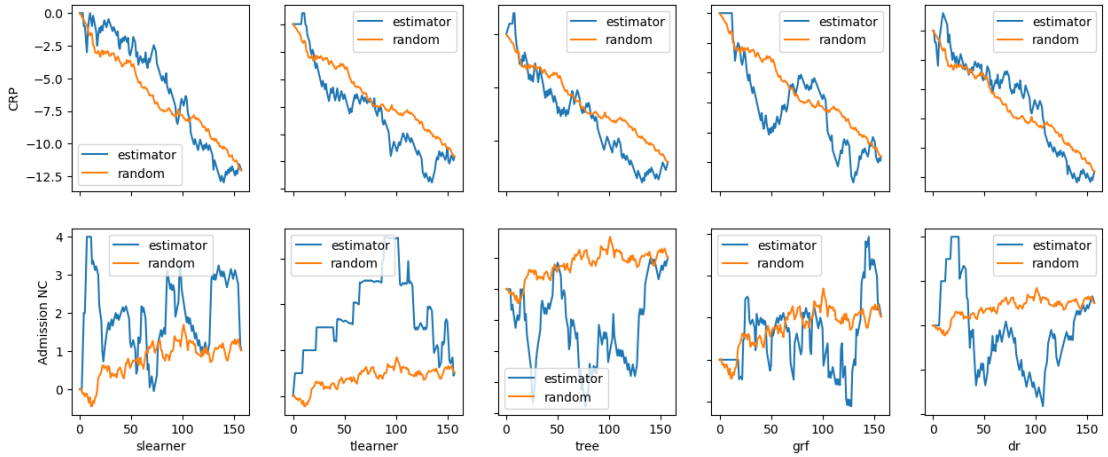


Figure 19. Sepsis cases dataset Uplift modeling Qini curve results.

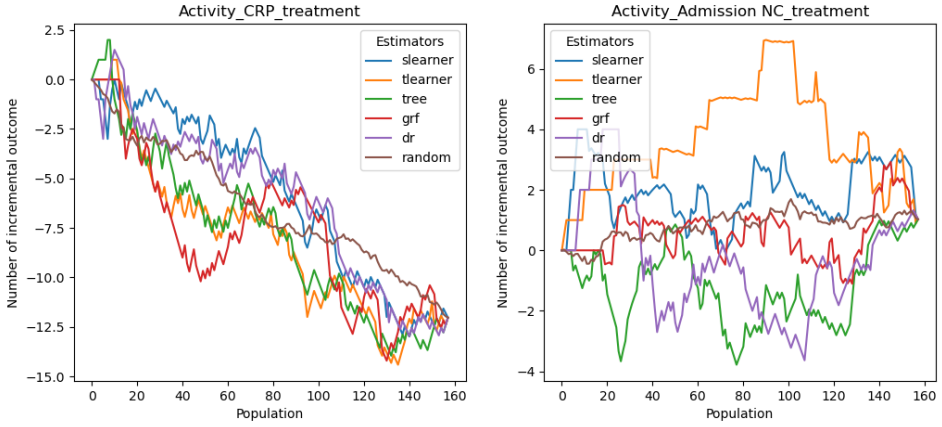


Figure 20. Sepsis cases dataset estimator comparison by Qini curve.

Table 21. Traffic fines dataset Causal estimation step results.

Estimator	**Treatment name**	**ATE**	**T**[1]
S-learner	Activity_Add penalty Activity_Send Fine Activity_Create Fine	0.040 0.086 0.073	2023.1
T-learner	Activity_Add penalty Activity_Send Fine Activity_Create Fine	0.035 0.087 0.074	699.0
Causal tree	Activity_Add penalty Activity_Send Fine Activity_Create Fine	0.466 0.476 0.478	1.8
X-learner Bound DML IV DIV	Terminated after error	-	-
GRF DR	Terminated after timeout	-	-

[1]Average duration of algorithm execution in seconds.

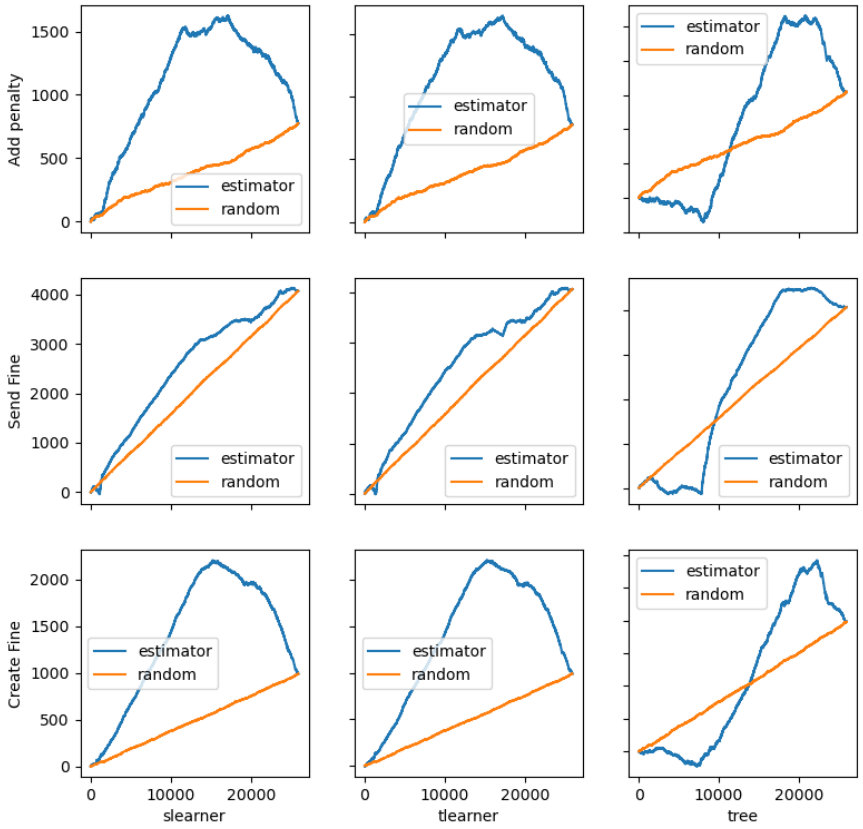


Figure 21. Traffic fines dataset Uplift modeling Qini curve results.

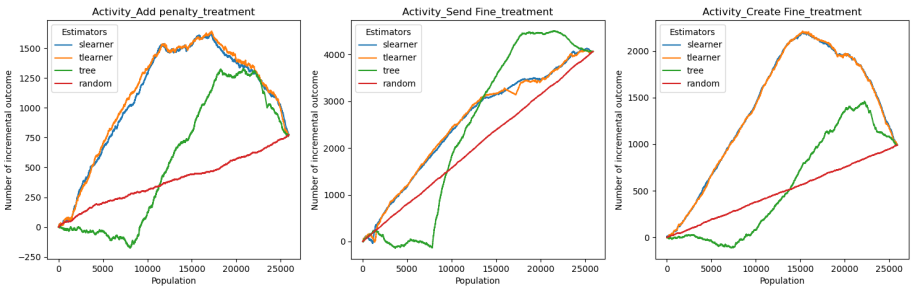


Figure 22. Traffic fines dataset estimator comparison by Qini curve.

IV Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Lukas Baltramaitis**,
*(*author's name)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Activity-Oriented Causal Process Mining: An End-to-End Approach Utilizing Ylearn,
   *(*title of thesis)

   supervised by Fredrik Milani and Mahmoud Shoush.
   *(*supervisor's name)

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Lukas Baltramaitis
09/05/2023