International Journal of Machine Learning and Cybernetics
https://doi.org/10.1007/s13042-025-02666-1

ORIGINAL ARTICLE

Causalteshap: discerning predictive from prognostic features for treatment effect analysis

Jarne Verhaeghe[1] · Femke Ongenae[1] · Sofie Van Hoecke[1]

Received: 25 January 2024 / Accepted: 28 April 2025

Abstract
Treatment effect analysis investigates the effect of a treatment or intervention. The variables that will determine the treatment effect are called, predictive variables, while prognostic variables determine the outcome regardless of treatment, based on existing conditions on characteristics. The identification of these predictive factors facilitates understanding the treatment effect and even allows for improving its success. However, in many cases, the predictive factors of a treatment or intervention are unknown. Furthermore, methods to find these predictive factors are limited and only focus on quantifying the predictive performance of a CATE estimator instead of discerning predictive from prognostic variables. Therefore, to find these predictive variables we present *Causalteshap*. *Causalteshap* is a Shapley-based method that leverages multiple statistical tests and treatment effect estimators to discern prognostic from predictive features. The method is benchmarked on multiple fully synthetic datasets and four semi-synthetic datasets. In most of these benchmarks, *Causalteshap* demonstrates high precision and recall performances above 0.9. Subsequently, *Causalteshap* is applied to a real-world ICU use case using the AmsterdamUMCdb dataset. We analyzed the effect of Noradrenaline on Atrial Fibrillation in the ICU to display the potential of *Causalteshap* as a tool for treatment effect analysis. Our results demonstrate that *Causalteshap* has the potential of combining treatment effect estimators with Shapley values and statistical tests to provide a novel method for discerning predictive from prognostic features in treatment effect analysis and making understanding treatment effects more accessible.

Keywords Treatment effects · Causality · Machine learning · Explainability · Meta-learners

1 Introduction

Causal thinking is a vital part of what makes humans inventive and creative. Reasoning about hypothetical worlds and understanding the effects of actions is what makes us unique [1]. However, this cannot yet be said about Artificial Intelligence (AI). Although, efforts are being made to reach this status using causal AI and causal inference, both emerging subdomains within AI and machine learning [2]. Their significance for AI lies in their potential to explain, interpret, analyze, and pave the way for groundbreaking research and applications. The primary goal of causal inference is to unravel causal relationships between variables. In contrast to traditional AI or machine learning (ML), causal AI or causal ML focuses on leveraging machine learning models to estimate and understand causal relationships. This approach enables informed decision-making, accurate predictions, and the ability to answer 'what-if' questions by identifying the underlying mechanisms that drive outcomes [3]. Causal AI is paramount for clinical trials, policy formulation, and marketing strategies by harnessing causal knowledge to generalize predictions, provide insights, and guide decision-making processes [1].

The estimation and analysis of the impact of treatments or actions on a specific outcome is a branch of causal AI, defined as treatment effect analysis [4]. Treatment effect

✉ Jarne Verhaeghe
jarne.verhaeghe@ugent.be

Femke Ongenae
femke.ongenae@ugent.be

Sofie Van Hoecke
sofie.vanhoecke@ugent.be

[1] IDLab, Ghent University - imec, Technologiepark-Zwijnaarde, 9052 Ghent, Belgium

Published online: 13 June 2025

analysis has large importance in healthcare, policy, and marketing, to understand which actions or treatments will be the most beneficial. This involves quantifying the impact of an action, such as the introduction of a policy by a government or the administration of a medication to a patient, on a particular outcome [2]. Fully understanding treatment effects can facilitate providing personalized healthcare, personalized marketing, or fully targeted policy-making.

A key component of understanding and interpreting the treatment effect is identifying predictive features (also called predictive covariates) that can influence it [5]. Predictive features are variables that increase or decrease the treatment effect, and their identification can provide valuable insights into how the system operates and can even be improved. In contrast, prognostic features are features that contribute to the outcome but their contribution is not influenced by treatment. This distinction can be formally described as $y = y_{prog} + T \cdot y_{pred}$, where prognostic features y_{prog} fully characterize the treatment-independent outcome, and predictive features y_{pred} determine the treatment effect, activated only when treatment is present. Predictive features are modelled as parents of mediators that transmit treatment effects, while prognostic features influence outcomes directly, independent of treatment. However, a feature can be both as well. Recognizing this distinction is crucial for effective treatment strategies in various domains.

For example, in clinical trials, predictive features could include a patient's age, gender, and medical history, which influence the effectiveness of a treatment. Prognostic features in the same context might be demographic factors like baseline health status or genetic predispositions that affect outcomes regardless of treatment. It is also possible for features to be both predictive and prognostic, such as Age. An example Directed Acyclic Graph (DAG) or Causal graph is presented in Fig. 1. In this example, it might be, for example, more beneficial to give the treatment to patients with a specific medical history as the treatment will be more effective compared to those who do not have this medical history. Similarly, in marketing campaigns, predictive features might encompass customer demographics, buying habits, and preferences that determine the success of a promotion, while prognostic features relate to consistent purchasing behaviour unaffected by marketing efforts. A feature can be both predictive and prognostic, therefore, it is important to know if there is a predictive component as only the predictive features can help tailor treatments to individual characteristics, in turn optimizing the chances of favourable outcomes. Additionally, access to predictive features in causal AI can enhance the interpretation and impact of causal AI models.

Therefore, in this paper, we present a method, named *Causalteshap* (Causal Treatment Effect Shapley values), that fully focuses on identifying predictive features for treatment effect models. Our contributions are as follows:

- We propose a novel method leveraging Shapley values, based on meta-learners and combined statistical tests to differentiate predictive features from purely prognostic ones, providing a model-agnostic method that applies to any gradient-boosting model while generalizing to other ML models supporting Shapley values.
- We demonstrate the effectiveness of *Causalteshap* through extensive experiments and compare them to other methods on newly proposed and expanded synthetic and semi-synthetic datasets for identifying predictive features.
- We showcase a real-world use case using *Causalteshap*, highlighting the benefit of discerning predictive features from prognostic and how this can impact future research.

With this work, we aim to position *Causalteshap* as a valuable method for distinguishing predictive features from purely prognostic ones in causal AI and treatment effect analysis.

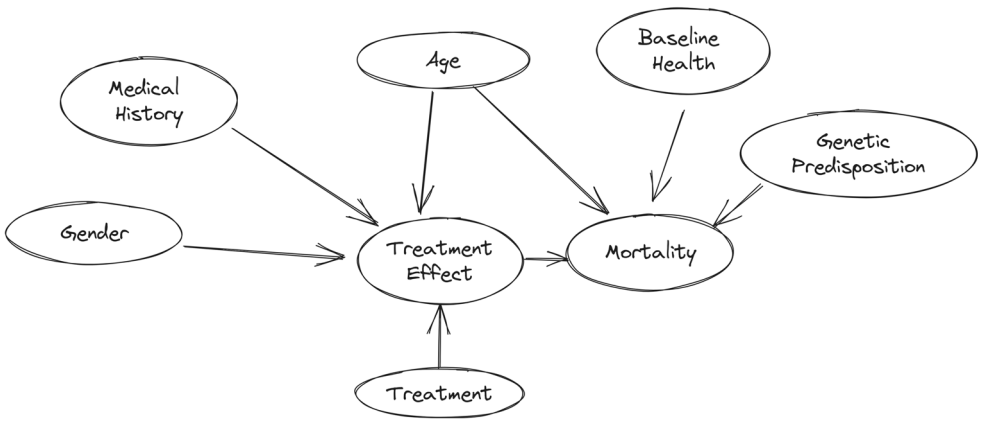


Fig. 1 Directed Acyclic graph or causal graph of the example. The predictive features are Gender, Medical History, and Age. The prognostic features are Age, Baseline Health, and Genetic Predisposition. This is merely an illustrative example and does not necessarily reflect the real world

2 Background

2.1 The potential outcome framework for treatment effects

The potential outcomes framework, introduced by Neyman and Rubin, provides a formalized framework for inference of treatment effects in causal analysis [6]. The framework is based on the idea of potential outcomes, which are the outcomes that would be observed under each possible treatment assignment. Therefore, each individual has two possible or potential outcomes (can be binary or continuous), one under treatment $Y(1)$ and one without treatment $Y(0)$. We can never observe both at the same time as only the performed action can be observed. Given the framework, it is possible to estimate the Conditional Average Treatment Effect (CATE) for an individual with covariates X = x [5]:

$$\begin{aligned} CATE(x) =& \tau(x) = E[Y(1) - Y(0)|X = x] \\ =& \mu_1(x) - \mu_0(x) \end{aligned} \tag{1}$$

With $\mu_T(x) = E[Y(T)|X = x]$ the expected potential outcome for treatment T, or in this case, treatment ($T = 1$) or control ($T = 0$). Setting T to either 0 or 1 is defined as intervening and tries to mimic random assignment because T is changed while other variables are not. Furthermore, only one of the outcomes can be observed, therefore the other outcome should be inferred for a given patient. Here is where machine learning can fit in. This CATE estimation requires three standard assumptions [6]:

- **Unconfoundedness**: $Y(t) \perp\!\!\!\perp T|X, \forall t \in T$. This assumption implies that, given the observed covariates, the treatment assignment is independent of the potential outcomes. In simpler terms, all confounding factors affecting both the treatment and the outcome have been accounted for, with no hidden variables influencing both.
- **Overlap** or **positivity**: $0 < P(T = t|X = x) < 1, \forall t \in T$ with $x \in X$. The overlap assumption guarantees that, for every covariate value x, there is a non-zero chance of receiving each treatment option. This is essential for reliably estimating treatment effects across all treatment groups.
- **Consistency**: $Y = Y(t)$ with probability 1. This assumption connects the observed outcomes to the potential outcomes, stating that the observed outcome matches the potential outcome associated with the treatment actually received.

Meta-learners in causal AI are frameworks in which machine learning models are used to perform CATE estimation [4, 7]. These meta-learners are popular and already widely available for CATE estimation use cases [4]. Examples are T-learners, S-learners, X-learners, and R-learners. The simplest of these meta-learners is the S-learner or Shared-model. This meta-learner only trains a single model M and predicts the outcome y, however, the treatment is added as a feature. After training, a patient is put through the single model twice: once with $T = 1$ and once with $T = 0$. The CATE can then be estimated as: $CATE(x) = M(X, T = 1) - M(X, T = 0)$. However, if the treatment effect is small or weak, the S-learner could learn to disregard the treatment variable as it is not informative enough during training. Each meta-learner has its advantages and disadvantages and varying complexity [7].

2.2 Predictive and prognostic features

In classic machine learning problems, some features are informative to predict the outcome. Likewise, as there are variables that are informative for a specific outcome, there are variables that are informative to explain the treatment effect. Variables that model the prediction outcome without the influence of any treatment are called prognostic variables. On the other hand, variables that model the treatment effect of a specific treatment are called predictive variables. This is modeled as follows [8]:

$$y = y_{prog} + T \cdot y_{pred} \tag{2}$$

Here, the predictive features fully characterize the treatment effect. The prognostic variables fully characterize the treatment-independent outcome. These predictive features can be interpreted as determinants of the treatment susceptibility that only get activated when the treatment is given. This interpretation is visualized using a Directed Acyclic Graph, called a causal graph, in Fig. 2. In the graph, only the *prognostic*, *predictive*, *Y*, and *treatment* variables are observable. Do note that there can also be a direct path from *predictive* to *Y* as variables can be both predictive and prognostic, this causal graph simply visualizes the prognostic versus predictive variable theory. Another interpretation of the graph is that giving the treatment or not is separated from its treatment effect. This treatment effect can be seen as the mediator through which the treatment achieves its effect. This effect is also dependent on variables that determine the treatment susceptibility, which can be deemed the predictive variables. Hence, predictive variables are thus defined as parents of mediators.

Knowing these predictive features facilitates explaining and understanding the success or failure of treatment. Furthermore, if these variables are modifiable, these can also increase the chance of success for specific treatments, enable targeting a sub-population that benefits more from

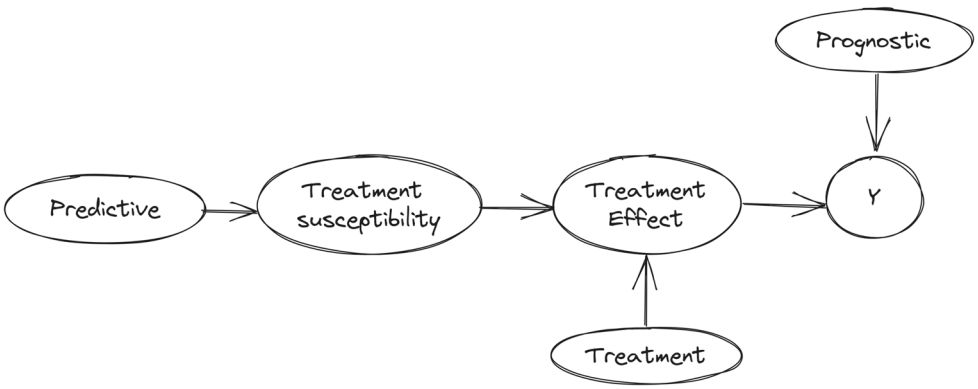


Fig. 2 Theoretic causal graph of the predictive and prognostic feature theory. Y is the measured outcome

it, or avoid a sub-population that suffers from the treatment.

2.3 Shapley values

Determining whether a feature is predictive or prognostic requires quantifying its feature contribution when $T = 1$ and $T = 0$. There are various methods to find and explain the contributions of variables to the predictions of machine learning models. A popular technique to explain model predictions is SHAP [9]. SHAP aims at quantifying the marginal contribution of each feature to the output prediction. This is defined as the average marginal contribution of a feature to all possible combinations of features. The so-called Shapley values measure how much each feature contributes to the prediction when considered in combination with other features [9]. Mathematically Shapley values try to reconstruct the prediction using their contributions. Given a trained estimator M on data X, with K features and N samples, and $M(X_i)$ being the prediction of M on X_i, we first define the baseline Shapley value S_b as the mean prediction over X:

$$S_b = \frac{1}{N}\sum_{i=1}^{N} M(X_i) \tag{3}$$

We then define the Shapley values for each sample and each feature $S(X_i^k, M)$ as the marginal additive contribution of feature k in sample X_i in the prediction $M(X_i)$ compared to the mean prediction of baseline Shapley value S_b. Now, given S_b and $S(X_i, M)$ we can define the local accuracy of Shap:

$$M(X_i) = S_b + \sum_{j=1}^{K} S(X_i^k, M) \tag{4}$$

The method is model-agnostic and available for various models, e.g., linear, kernel-based, deep learning, and tree-based models. Although SHAP suffers from shortcomings, such as the TreeExplainer variant (i.e. the SHAP method to explain ensemble tree models) providing non-zero Shapley values to noise features, it is technically strong and very popular [10].

2.4 Related work

In the causal machine learning field, only limited work is available to provide interpretations to treatment effect estimators [5]. Even though the literature on finding predictive variables is limited in the causal machine learning field, the concept is more explored in the statistical field under interaction effects and subgroup analysis [11–13]. Several studies have attempted to identify subgroups with differential treatment effects, mentioning distinguishing between prognostic and predictive variables.

In early work, Bonetti and Gelber [14] introduced the STEPP (Subpopulation Treatment Effect Pattern Plot) method, which identifies subgroups with distinct treatment effects on the outcome. Their approach is mainly for clinical trials, requiring predefined subpopulation regions, making it heavily parameter-dependent. Similarly, Lipkovich et al. [15] proposed the SIDES (Subgroup Identification based on Differential Effect Search) method, which recursively partitions data to determine subgroups with significant treatment differences, extending ideas from interaction trees. The features used in splits could be seen as predictive features. However, their method does not scale well with the number of features and data.

Foster et al. [16] explicitly mentioned the distinction between prognostic and predictive roles in subgroup analysis and tried to find them using a Virtual Twins framework. This method, an early form of a meta-learner, estimates treatment effects via a two-step modelling approach. In the first step, they train an S-learner with either a random forest or a linear regression model and then fit a new model on the pseudo outcomes. However, their focus was mainly on finding subgroups, not on finding the predictive variables. Cai et al. [17] also mentioned that finding predictive features is an important issue for covariate selection, however, the conclusions depend on

the underlying model (mis)specification. Doove et al. [18] developed QUINT, a tree-based method that partitions patients into three groups: those who benefit more from treatment A, those who benefit more from treatment B, and those with no significant difference. However, QUINT is computationally expensive and only applicable to randomized controlled trials (RCTs) without handling confounders. A solution for high-dimensional settings was proposed by Guo et al. [19] which applied bootstrapped sparse logistic regression models to identify vulnerable subgroups in electronic health record data.

Instead of directly searching for subgroups, several other studies have focused on statistical and machine-learning methods for identifying treatment-interaction effects. For example, Tian et al. [20] proposed a modified covariate method (MCM) that transforms the features to better estimate the treatment-interaction effects in linear models. Consequently, the coefficients of these models can then be interpreted to understand which features influence the treatment. Park et al. [21] proposed a solution with sparse additive models (SAM) for high-dimensional problems to find treatment-interaction effects. The estimated nonzero components in the SAM model can then be seen as predictive features.

Within causal machine learning, Hermansson et al. [8] analyzed causal trees and virtual twins for ranking predictive features using gradient boosting trees but did not propose a method to separate predictive from prognostic variables. Crabbe et al [5] introduced the ITErpretability framework, evaluating treatment effect models on their accuracy and ability to attribute prognostic and predictive contributions. However, they focused on attribution measurement rather than directly separating predictive variables. Besides these works, the literature for finding predictive variables is limited in causal machine learning.

While the above methods focus on distinguishing prognostic and predictive features, another line of work has focused on selecting the minimal adjustment set required for unbiased treatment effect estimation. These methods aim to control for confounding rather than identifying features that modify treatment response. For example, De Luna and Waernbaum [22] addressed covariate selection for nonparametric treatment effect estimation, providing minimal adjustment sets in the Neyman-Rubin framework. Similarly, Cheng et al. [23] refined treatment effect estimation by using local search algorithms to determine optimal adjustment sets even in the presence of hidden variables. However, these methods primarily target valid treatment effect estimation rather than identifying predictive variables for treatment heterogeneity. Unlike these approaches, our work focuses on discovering predictive features that influence treatment response, independent of their role as confounders.

Building upon these works, *Causalteshap* introduces a novel approach that explicitly separates predictive and prognostic variables by leveraging Shapley values in combination with statistical testing. Unlike prior methods that focus on either interaction-based subgroup discovery or tree-based importance attribution, *Causalteshap* provides a model-agnostic framework that applies to any gradient-boosting model while generalizing to other ML models supporting Shapley values. This enables more precise identification of predictive features while avoiding the scalability and parametric limitations of prior approaches. Compared to treatment subgroup methods, *Causalteshap* focuses solely on identifying predictive features in a computationally efficient manner. Unlike previous approaches that either rely on linear models, tree-based importance measures, or strong parametric assumptions, our method remains flexible and scalable, making it applicable to a wider range of machine-learning-based meta-learners.

3 Methods

3.1 Causalteshap

Causalteshap leverages meta-learners, such as an S-learner, to find predictive features. The current algorithm employs an S-learner and is extendable to other meta-learners with the requirement that the baseline Shapley value should be the same. First, we train an S-learner M on the data. Given Eq. 2, the T^1-model models the potential outcome as $y_{T^1} = y_{prog} + y_{pred}$ while the T^0 models it as $y_{T^0} = y_{prog}$. Therefore, subtracting $y_{T^1} - y_{T^0}$ leaves us only with y_{pred} in the ideal case. To find these features we leverage Shapley values using the SHAP library [9] to explain the attribution of these features. For this, we define the Predictive Shapley features $S_{pred}(X)$ as follows, with S representing the function for Shapley value calculation:

$$S_{T^0}(X) = S(X, T = 0, M) \tag{5}$$

$$S_{T^1}(X) = S(X, T = 1, M) \tag{6}$$

$$S_{pred}(X) = S_{T^1}(X) - S_{T^0}(X) \tag{7}$$

The $T = 1$ or $T = 0$ represents performing an intervention on the data or not respectively. To avoid bias, these Shapley values are calculated on a validation set of unseen data, i.e., data that was not used to train the model M.

Ideally, a purely prognostic feature X will have $S_{pred}(X) = 0$. However, the SHAP library, especially tree-SHAP, tends to attribute non-zero importance to noise, as noted in prior studies [10]. This occurs due to model-induced randomness, overfitting tendencies in complex

models (e.g., gradient boosting), and the inherent variance in Shapley value estimation. Consequently, simply comparing $S_{pred}(X)$ to zero is not reliable, as random fluctuations can lead to false positives even when no true predictive effect exists.

To address this, *Causalteshap* employs a two-part approach designed to mitigate the impact of such noise. The first part tests whether the variable's effect differs between treatment groups, a necessary condition for a feature to be predictive. The second part further controls for false positives by comparing the variable's Shapley contributions to those of a known random (non-informative) feature. This composite strategy ensures that identified predictive variables are robust to both model-related noise and the stochastic nature of SHAP explanations. The two parts are as follows:

1. If the feature is purely prognostic, then the $S_{T^0}(X)$ and $S_{T^1}(X)$ distribution should have the same variance and same mean.
2. When these distributions are different and the feature X is truly prognostic, then $|S_{pred}(X_{noisy})|$ of a known noise variable X_{noisy} that contains no information should be larger or equal compared to $|S_{prog}(X)|$. This covers the cases where these differences would be caused by noise.

The following subsections explain how each part is addressed by *Causalteshap*.

3.1.1 Part 1: test whether the prognostic features have the same variance & mean

To address the first part we use Welch's t-test, i.e. student t-test with unequal variance, to check for different means and the Fligner test to check for different variances. The combination of these tests compares $S_{T^1}(X)$ with $S_{T^0}(X)$ to test the following null-hypothesis:

"The Shapley values $S_{T^1}(X_{prog})$ of a prognostic feature X_{prog} when $T = 1$ should have the same mean and same variance as the Shapley values $S_{T^0}(X_{prog})$ of the feature when $T = 0$."

If the p-value of either Welch's t-test or the Fligner test is below our predefined threshold α, the hypothesis is rejected. A mathematical explanation of the Welch's t-test can be found in the appendix A.1. Welch's t-test assumes that the mean of the sampled distribution is normally distributed. Given the central limit theorem and the size of the tested datasets ($N > 100$), this can be assumed to be true [24]. Therefore, the test can be applied.

The second test is the Fligner test, or the Fligner–Killeen test of homogeneity of variances, which is a distribution-free test of variances [25]. A mathematical explanation of Welch's t-test can be found in the appendix A.2.

3.1.2 Part 2: test whether the difference in distributions of a predictive feature is due to noise

To check whether the difference is caused by noise, we add a random feature sampled from a uniform distribution to the dataset. This feature provides a baseline of what $S_{pred}(X)$ should be for a pure prognostic feature and can therefore be used as a comparison. Shapley values are signed, therefore, we take the absolute value of the difference as we are only interested in the amplitude of the difference. Now, given the predictive Shapley values of the features and those of the random feature, we can statistically test whether they are predictive. This part is built on the following null hypothesis:

"The Cumulative Distribution Function (CDF) of the absolute predictive Shapley values $|S_{pred}(X_p)|$ of a prognostic feature X_p should be lower or equal to the CDF of the absolute predictive Shapley values $|S_{pred}(X_r)|$ of the random feature X_r."

To verify this hypothesis we use the Kolmogorov–Smirnov test (KS-test). As already stated, this KS-test is mainly a method to compensate for the importance attribution to the noise of TreeShap. A mathematical explanation of the Welch's t-test can be found in the appendix A.3.

If a feature passes both parts, i.e. significant result on both the KS-test and either the t-test or Fligner test (that tests whether either the mean and or variance is different), we determine the feature to be predictive. In the case any of the parts fail, the feature is flagged as prognostic. Additionally, while these specific tests are well-suited to our hypotheses, they are not the only options: Any test that fulfils similar requirements, e.g. testing comparable null hypotheses with fewer assumptions, could of course be substituted.

3.1.3 Main algorithm

We can now present the *Causalteshap* algorithm. The complete algorithm can be found in Algorithm 1. The function returns the predictive features, given an S-Learner M, treatments $\mathbf{T}$ for n samples, data $D^{n\times m}$ with n samples and m features, all feature names $\mathbf{F}_{set}$, and the significance threshold α. *Causalteshap* is implemented in Python as an open-source plug-and-play *sklearn* compatible component[1] to enable direct usage in meta-learner machine learning pipelines [26].

[1] The code, documentation, and more benchmarks can be found using the following link: https://github.com/predict-idlab/causalteshap.

This section describes the *Causalteshap* algorithm. The algorithm begins by generating an array D^n_{random} of length n, sampled from a uniform distribution over the interval $[-1, 1]$. Next, we construct the dataset $\mathbf{D}^{n\times(m+2)}$, which includes the original m features, the treatment variable T, and the random variable D^n_{random}. The dataset $\mathbf{D}$ is then split into a training set $\mathbf{D}_{train}$ and a validation set $\mathbf{D}_{val}$ based on $Q_K(\alpha)$ (Appendix A.3) and at least one of the p-values (p_{Wt} or p_{Fl}) is less than or equal to the significance threshold α. Otherwise, the feature is marked as prognostic ($\mathbf{P}[i] = 0$). Finally, the algorithm returns all features F_i for which $\mathbf{P}[i] = 1$, indicating their predictive relevance.

Algorithm 1 *Causalteshap* loop

function CAUSALTESHAP($M \leftarrow$ S-Learner, $\mathbf{T} \leftarrow t_1, ..., t_n$, $\mathbf{D}^{n\times m} \leftarrow$ Data, $\mathbf{F}_{set} \leftarrow F_1, \ldots, F_m$, $\alpha \leftarrow$ threshold, split size ω):
  $D^n_{random} \leftarrow$ Sample n times from RandomUniform $\in [-1, 1]$
  $\mathbf{D}^{n\times m+2} \leftarrow \mathbf{D}^{n\times m} \cup T^n \cup D^n_{random}$
  $\mathbf{D}^{(1-\omega)n\times m+2}_{train}, \mathbf{D}^{\omega n\times m+2}_{val} \leftarrow$ split $(\mathbf{D}, \omega)$
  $M \leftarrow$ Fit $M(\mathbf{D}_{train})$
  $\mathbf{D}^{T^1}_{val} \leftarrow T = 1$
  $\mathbf{D}^{T^0}_{val} \leftarrow T = 0$
  $\mathbf{S}_{T^1} \leftarrow \text{SHAP}(M, \mathbf{D}^{T^1}_{val})$
  $\mathbf{S}_{T^0} \leftarrow \text{SHAP}(M, \mathbf{D}^{T^0}_{val})$
  $\mathbf{S}_{pred} \leftarrow S_{T^1} - S_{T^0}$
  $\mathbf{P} \leftarrow$ size m
  for $i \leftarrow 1, 2, \ldots, m$ **do**
    $D^+ \leftarrow \sqrt{n} \cdot sup_x(F_i(\mathbf{S_{pred}}[...][\mathbf{i}]) - F_R(\mathbf{S_{pred}}[...][\mathbf{m+2}]))$
    $p_{Wt} \leftarrow WelchsTTest(\mathbf{S_{T^1}}[...][i], \mathbf{S_{T^0}}[...][i])$
    $p_{Fl} \leftarrow Fligner(\mathbf{S_{T^1}}[...][i], \mathbf{S_{T^0}}[...][i])$
    if $D^+ > Q_K(\alpha)$ **AND** $(p_{Wt} \leq \alpha$ **OR** $p_{Fl} \leq \alpha)$ **then**
      $\mathbf{P}[i] \leftarrow 1$
    else
      $\mathbf{P}[i] \leftarrow 0$
    end if
  end for
  return $[F_i, \forall\ i : \mathbf{P}[i] = 1]$
end function

the specified ratio ω. This results in $(1-\omega)n$ samples in $\mathbf{D}_{train}$ and ωn samples in $\mathbf{D}_{val}$. The S-learner model M is subsequently trained on $\mathbf{D}_{train}$. After training, we create two interventional datasets, $\mathbf{D}^{T^1}_{val}$ and $\mathbf{D}^{T^0}_{val}$, by setting the treatment covariate to 1 and 0, respectively. The Shapley values, $\mathbf{S}_{T^1}$ and $\mathbf{S}_{T^0}$, are then computed using the SHAP library [9] for both interventional datasets. For each feature i in $\mathbf{F}_{set}$, we then calculate the test p-values. First, representing Part 1, we calculate the Kolmogorov–Smirnov test statistic D^+ to compare the distribution of $\mathbf{S}_{pred}[i]$ with that of the random variable $\mathbf{S}_{pred}[m+2]$. Then, representing Part 2, we compute the p-values for both Welch's t-test (p_{Wt}) and the Fligner test (p_{Fl}), comparing $\mathbf{S}_{T^1}[i]$ and $\mathbf{S}_{T^0}[i]$. A feature is marked as predictive ($\mathbf{P}[i] = 1$) if the Kolmogorov–Smirnov test statistic D^+ exceeds the threshold value

3.2 Experiments

To objectively evaluate *Causalteshap* it is not possible to benchmark the method on real-world data as it is hard to know the true relationships of the prognostic or predictive features in many of these datasets. Therefore, *Causalteshap* is first benchmarked on fully synthetic data with predefined cases each testing different possible treatment effect relations. Afterwards, the methods are benchmarked on semi-synthetic data using a randomly generated Data Generating Process (DGP) inspired by the ITErpretability Benchmark [5]. Only predictive or prognostic features will be simulated. Non-informative features will not be included as finding which variables are informative and which ones are not is a feature selection problem and not in the scope of

Table 1 The different cases for the evaluation on synthetic data

Case	Relationship
M1	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot (I(x_6 > 0) + I(x_7 > 0)) + \epsilon$
M2	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot I(x_1 > 0) + \epsilon$
M3	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot I(x_{19} > 0) + \epsilon$
M4	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + \epsilon$
M5	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) +$ $T \cdot (x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14}) + \epsilon$
M6	$y = -1 + 3 \cdot (I(x_0 > 0) + I(x_1 > 0) + I(x_2 > 0) + I(x_3 > 0) + I(x_4 > 0)) +$ $T \cdot I(x_4 > 0) + \epsilon$
M7	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9) +$ $T \cdot I(x_{19} > 0) + \epsilon$
M8	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9) +$ $T \cdot I(x_0 > 0) + \epsilon$
M9	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot sin(x_0) + \epsilon$
M10	$y = -1 + 10 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot I(x_0 > 0) + \epsilon$
M11	$y = -1 + (x_0 + x_1 + x_2 + x_3 + x_4)^3 + T \cdot x_0^3 + \epsilon$
M12	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot x_0 + \epsilon$
M13	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot x_{19} + \epsilon$
M14	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + 50 \cdot T \cdot (I(x_6 > 0) + I(x_7 > 0)) + \epsilon$
M15	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + 50 \cdot T \cdot x_0 + \epsilon$
M16	$y = -1 + 3 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) +$ $T \cdot (I(x_0 > 0) + 10 \cdot I(x_1 > 0) + 5 \cdot I(x_7 > 0)) + \epsilon$
M17	$y = -1 + 10 \cdot (x_0 + x_1 + x_2 + x_3 + x_4) + T \cdot I(x_5 > 0) + \epsilon$
M18	$y = -1 + (x_0 + x_1 + x_2 + x_3 + x_4)^3 + T \cdot I(x_5 > 0)^3 + \epsilon$

this paper. As predictive and prognostic features are both informative features to predict the outcome, we advise first using some high-sensitivity feature selection algorithms such as PowerShap [27] to eliminate non-informative features. Afterwards, we demonstrate how *Causalteshap* can be applied to a real-world use-case where we try to find the predictive variables of the treatment effect of Noradrenaline on the occurrence of Atrial Fibrillation (AF) in the Intensive Care Unit (ICU) using published risk outcome models [28]. All experiments are performed using a Catboost S-learner using 1000 iterations, categorical feature T, *use_best_model* set to True, a *Causalteshap* split size of 0.3, and a data 80%/20% train-test split for fitting and evaluating *Causalteshap*. In all these experiments the random feature is sampled from a uniform distribution with lower and upper bounds of -1 and 1, respectively. This sampling can be changed according to knowledge of the data distribution. These experiments are evaluated on the precision and recall of finding the true predictive features.

3.2.1 Synthetic benchmarks

The synthetic experiment setup is an expansion of the paper by Hermansson and Svensson [8]. In the original paper, there were 11 cases (M1 to M11), however, we expanded these to include 7 more difficult cases incorporating small and large treatment effects, as well as complex relations with predictive and prognostic features. All cases can be found in Table 1. ϵ represents the Gaussian noise with a mean of 0. Each case covers a specific scenario. Case M4 is a case that does not have any predictive variables while cases M14 and M15 cover the scenarios with very large treatment effects. Other cases such as M2, M6, M8, M9, M11, M12, M14, and M16 are scenarios with features having both a prognostic and predictive component. M9 on the other hand covers a non-linear treatment-effect relationship. In Table 2, different setups for confounding which influences treatment assignment are also presented. The T0 case is for experiments without confounding resulting in balanced treatment groups. T1 has some small confounding where treatment assignment is dependent on features x_1 and x_2 while T2 has stronger confounding and is more influenced by x_1. All variables x_i are drawn from independent normal distributions with mean 0 and standard deviation 1.

All synthetic experiments are performed with 100, 250, 500, 1000, 2500, 5000, and 10, 000 generated samples to investigate the impact of data availability on the

performance. The experiments are repeated 10 times, each with different random seeds to quantify the variance of the method. The standard deviation of the added Gaussian noise is also varied from 0.001, 0.01, 0.1, 0.5, 1.0, 2.5, 5, to 10 to check the robustness against noise. A significance threshold of $\alpha = 0.02$ is chosen as a setting in *Causalteshap* to represent a false positive rate of 2%. *Causalteshap* will also be compared to other models capable of finding predictive variables: the modified covariate method by Tian et al. [20] and the Sparse Additive Models by Park et al. [21]. For the MCM method, we fitted an Ordinary Least Squares regression on the modified features and picked the coefficients that had an $\alpha < 0.05$. For the SAM model we took all nonzero components and determined them as predictive as mentioned in their work [21].

3.2.2 Semi-synthetic benchmarks

Synthetic data is optimal for testing ideal performance, however, it is difficult to model realistic interactions between features, such as unobserved interdependency and confoundness. Therefore, to expand the experiments we also included a DGP evaluation using semi-synthetic data. Here, we use available features in real-world datasets and use these features to model an outcome where we can specify the amount of prognostic and predictive features. In this way, we have more realistic complex data while still having a ground truth for benchmarking. The DGP algorithm is shown in Algorithm 2 and follows the ITErpretability benchmark method presented by Crabbe et al. [5].

Table 2 The different cases for the evaluation on synthetic data for treatment assignment generation

Case	Relationship
T0	*Random*(0, 2, *N*)
T1	$Binomial(1, \frac{exp(0.1+0.5\cdot x_1-0.25\cdot(2+x_2))}{1+exp(0.1+0.5\cdot x_1-0.25\cdot(2+x_2))}, N)$
T2	$Binomial(1, \frac{exp(0.1+1.5\cdot x_1-0.25\cdot(2+x_2))}{1+exp(0.1+1.5\cdot x_1-0.25\cdot(2+x_2))}, N)$

Four datasets were used: the TCGA [29], Twins [30], News [31], and ACIC2016 [32] datasets. The number of samples is set to be equal to available data points in the dataset. For the News and ACIC datasets the 100 most varying features were selected for the DGP as these datasets have large amounts of features. The resulting datasets have 38, 64, 100, and 100 features respectively. The experiments are performed using multiple amounts of prognostic and predictive variables; $10\%, 25\%, 40\%$, and 65% of all features being prognostic and $3\%, 12.5\%, 28\%$, and 50% being predictive to test increasing data complexity. These features can overlap and are selected independently of each other. The experiments are repeated 10 times, each with different random seeds. The strength of the treatment effect w_{pred} is also varied to investigate the sensitivity of the method using four settings: 0.01, 0.1, 0.5, and 1.0. There is no added Gaussian noise for the DGP as the added noise should be adjusted according to the relative size of the output, w_{pred}, the number of features, and every dataset for realistic evaluation. However, as an added experiment, to understand the impact of noise on the semi-synthetic data, the News

Algorithm 2 Semi-Synthetic Data generating process

function DGP($T^n \leftarrow$ Treatment Array, $X^{n\times m} \leftarrow X_1, ..., X_m$, $m_{prog} \leftarrow$ Amount Prognostic Features, $m_{pred} \leftarrow$ Amount Predictive Features, $\beta \leftarrow$ weight threshold, $w_{pred} \leftarrow$ Treatment Weight):
  $\alpha_{prog}^{m\times 1} \leftarrow Uniform(-1, 1, m)$
  $\alpha_{pred}^{m\times 1} \leftarrow Uniform(-1, 1, m)$
  $\alpha_{prog} \leftarrow [\alpha_{i,prog}$ if $|\alpha_{i,prog}| > \beta$ else 0 for $i..m]$
  $\alpha_{pred} \leftarrow [\alpha_{i,pred}$ if $|\alpha_{i,pred}| > \beta$ else 0 for $i..m]$
  $i_{prog}^{m_{prog}\times 1} \leftarrow$ select m_{prog} non-zero weights from α_{prog}
  $i_{pred}^{m_{pred}\times 1} \leftarrow$ select m_{pred} non-zero weights from α_{pred}
  $\mu_{prog}^{n\times 1} = \sum_{i\in i_{prog}}(\alpha_{i,prog} \cdot Xi)$
  $\mu_{pred}^{n\times 1} = \sum_{i\in i_{pred}}(\alpha_{i,pred} \cdot Xi)$
  return $\mu_{prog} + T \cdot w_{pred} \cdot \mu_{pred}$
end function

dataset is also benchmarked using 0.001, 0.01, 0.1, 1, 5, and 10 Gaussian noise standard deviations. The News dataset has lower absolute feature values, making it the most sensitive dataset of the four for the chosen noise standard deviations.

3.3 The effect of noradrenaline on atrial fibrillation

Although benchmarking is not feasible on real-world datasets, the method can be applied to one. To test the method on a real-world use case, *Causalteshap* is applied to a treatment effect analysis of noradrenaline (= the treatment) on the occurrence of Atrial fibrillation (AF) (= the outcome) in the ICU using the AmsterdamUMCdb dataset [33]. AF is a heart rhythm disorder that causes an irregular and often abnormally fast heart rate. It affects between 4.5 to 15% of patients admitted to the intensive care unit (ICU). Several studies have indicated that the occurrence of AF in critically ill patients is associated with poorer outcomes, including prolonged length of stay (LOS) and increased hospital mortality [34, 35]. Although several risk factors for AF are non-modifiable (e.g., age), identifying patients at high risk of developing AF could allow clinicians to preemptively address modifiable risk factors (e.g., electrolyte imbalances or medication). In the study of Verhaeghe et al. [28] noradrenaline was an important predictor for the occurrence of AF in the final model and is, therefore, a candidate for causal analysis of the medication. Model-6 from the same study was used as the outcome model for the S-learner and applied to the AmsterdamUMCdb dataset [33]. This model used features aggregated from routinely collected time series data 18 h to 6 h before the diagnosis of AF or from the same time point as an AF patient for matched non-AF samples. The final features of the S-learner are determined using double feature selection which selects the union of the selected features of the propensity model and the selected features of the outcome model. The feature selection of both the outcome and the propensity model was performed using PowerShap [27]. Both the outcome and the propensity model use the same preprocessing method as described in the original study.

4 Results

4.1 Synthetic

The results for the synthetic benchmarking results are shown in Fig. 3. For *Causalteshap*, for all sample sizes, noise levels, and confounding levels, the precision mostly stays above 0.95. However, we see that the recall, or the capability of finding predictive features, is related to the number of available samples. This is a result of using a wrapper-based method that has all the limitations of model-driven approaches, e.g. limited data limiting the overall performance. Only when the number of samples exceeds 2500, does the noise start affecting the recall as the results are less limited by the sample size. For the synthetic benchmarks, given an adequate-sized dataset, *Causalteshap* finds most predictive variables without outputting false positives, even in more high-noise situations. In contrast, for the MCM approach both the precision and recall are much lower for all confounding cases, indicating difficulty in finding all predictive variables while preventing false positives. For the SAM method, interestingly the recall drastically increases with more confounding, however, the method is not robust against false positives. In contrast, the MCM approach exhibits substantially lower precision and recall across all confounding scenarios, suggesting difficulties in identifying predictive variables in these various scenarios. Interestingly, the SAM method demonstrates a significant increase in recall with higher levels of confounding, however, it lacks robustness in controlling false positives.

On the case level for *Causalteshap*, there are difficulties with finding predictive features for cases M10 and M11 even with more than 10,000 samples finding no predictive features. This is mainly attributed to the strength of the prognostic features that drown out the much smaller predictive component of x_0. For cases M2, M9, and M7, only when the number of samples exceeds 2500, does *Causalteshap* also find the predictive features. On the other hand, for cases M13, M14, M15, M3, and M5, *Causalteshap* already finds all predictive features with only 500 samples. The relation between the contribution of the predictive features and the prognostic features determines how difficult finding all predictive features will be. If there is a shared prognostic and predictive component, where the predictive component is relatively small, the chance of detecting the specific predictive component is smaller, such as in case M10 or case M11.

4.2 Semi-synthetic

The results for all semi-synthetic benchmarks are shown in Fig. 4. For all four datasets, the strength of the predictive effect greatly influences whether the predictive features are found or not. This effect is worsened when the number of prognostic features grows, increasing the overall data complexity of the benchmark. In lower dimensionalities, i.e. a low number of prognostic features, the recall reaches levels of 0.8 to 1.0 with a precision above 0.95 showing

that *Causalteshap* finds most predictive features without outputting many false positives. However, when the number of prognostic features grows, the performance reduces as the data complexity increases. Even in those situations recalls above 0.8 are still reached while retaining precisions above 0.8, depending on the dataset. Counterintuitively, if the strength of the treatment effect increases, the false positive rate also increases. This is not the case for the synthetic benchmarks and is attributed to an inherent contradiction in the way semi-synthetic datasets are built. This will be further specified in the discussion section.

The effect of noise on the News semi-synthetic dataset is shown in Fig. 5. Unlike the synthetic benchmarks, the noise here does have a much larger effect. With increased noise, both recall and precision are reduced. However, even with higher noise levels, *Causalteshap* retains precision levels of around 0.7 while having a lower recall. If the treatment effect is masked by noise, it is intrinsically harder to detect. Because the approach is wrapper-based, *Causalteshap* is not immune to this phenomenon. However, in reduced noise situations *Causalteshap* manages to achieve low false positive rates while finding on average 80% of all predictive features.

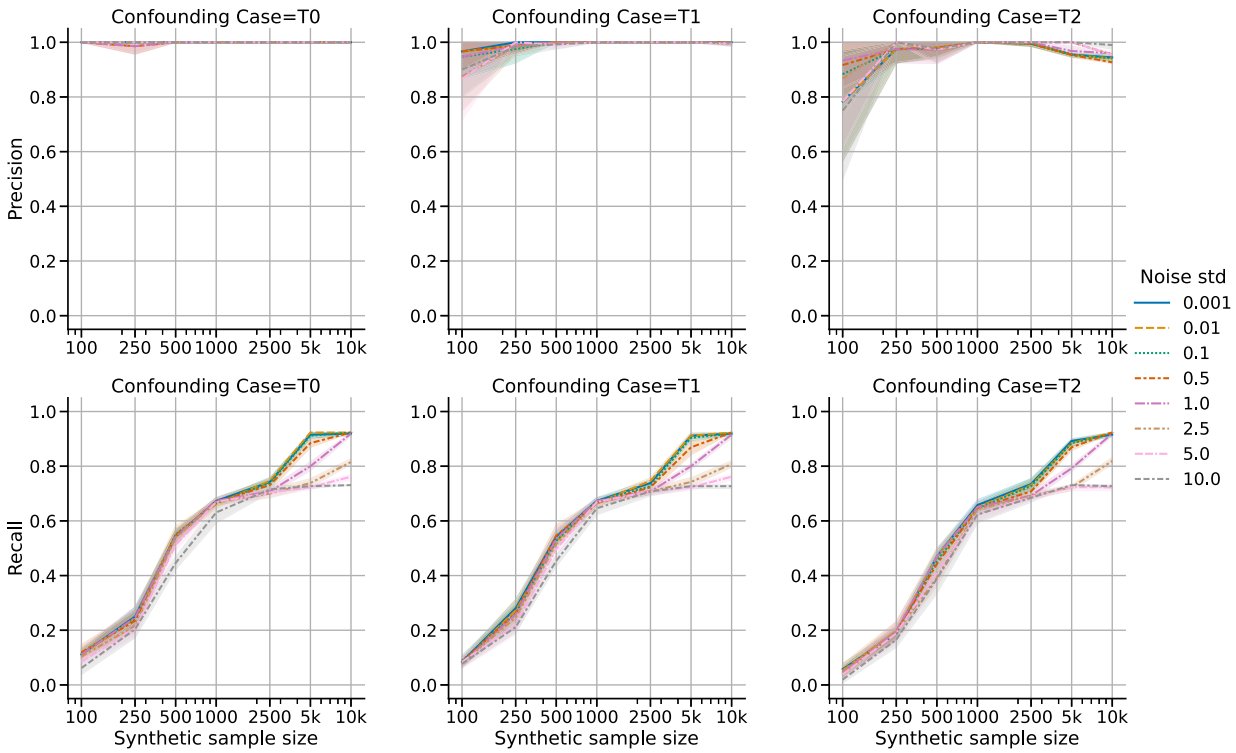
4.3 Noradrenaline

The added value of *Causalteshap* shines through when applied to real-world datasets to facilitate mediator and treatment effect analysis. As mentioned in Sect. 2.2, predictive values are defined as parents of unobserved mediators that influence the treatment, hence these features might provide pointers to the underlying mechanism through which the treatment works, i.e. the mediator(s). Applying *Causalteshap* to the Noradrenaline treatment analysis on AF is shown in Table 3. The Noradrenaline treatment variable (T) is considered predictive, which means that there is a significant treatment effect. The $T^1 - T^1$ and $|T^1| - |T^0|$ columns, which are positive, suggest that Noradrenaline increases the risk for AF. The Central Venous Pressure (CVP) feature is considered predictive, determined by the Fligner test and the KS statistic. The $|T^1| - |T^0|$ column displays that the variance of the CVP Shapley values when $T = 0$ is lower than when $T = 1$. In other words, Noradrenaline increases the effect CVP has on the risk of AF. On the contrary, Noradrenaline decreases the effect Fluid balance and Creatinin have on AF. Lactate and pH have different variances for the T^1 and T^0 distribution, however, their predictive Shapley values are not always larger than the noise feature Shapley values. As a result, they are not considered to be predictive, however, they can still provide hypothetical medical insights. Additionally, the Age feature does have a larger predictive difference than the random feature, but the T^1 and T^0 distributions do not differ significantly. This is explained by the strong prognostic effect of Age on AF, as it is considered a risk factor for AF by the literature [36].

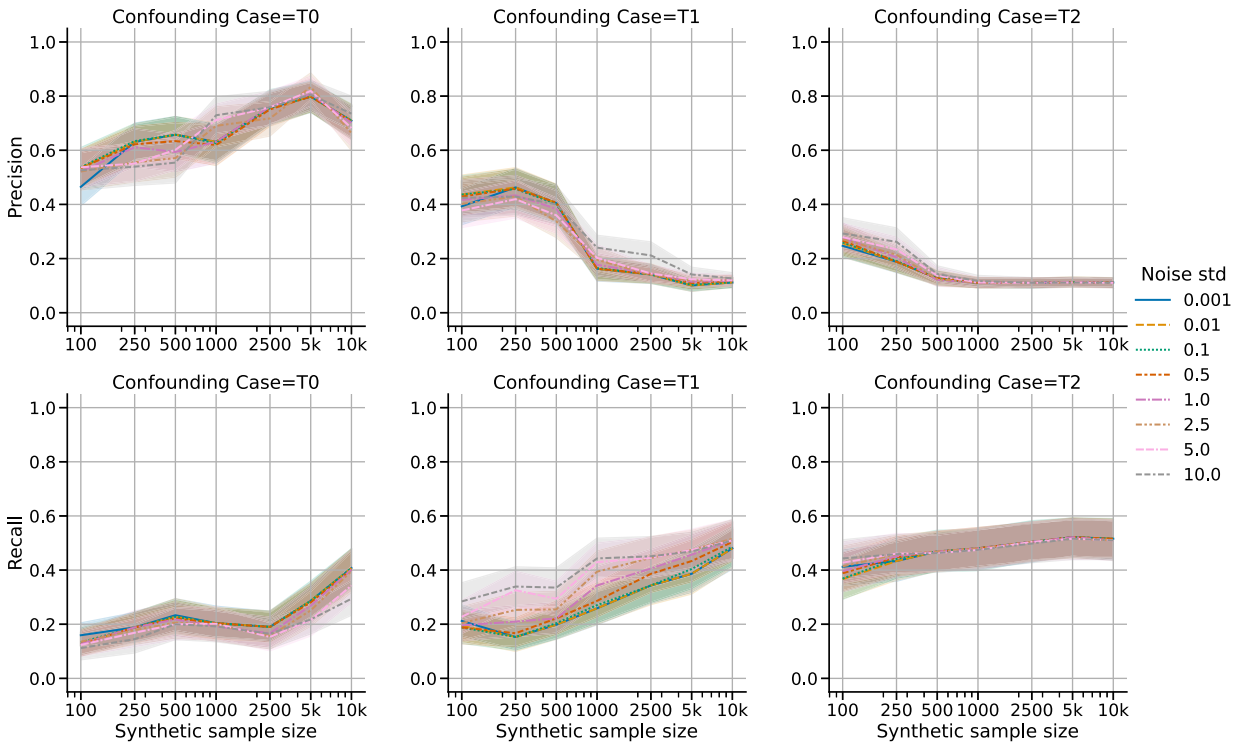
5 Discussion

In this work, we introduced *Causalteshap*, a Shapley-based method for finding predictive features for treatment effect analysis. As presented with the Noradrenaline treatment analysis, *Causalteshap* facilitates testing and finding insights in treatment effect models to further research or understand the treatment. These predictive features can offer valuable insights into potential (unobserved) mediators as they are the parents of these mediators by definition. The method provides explainability built upon previous research for treatment effect analysis using well-known Shapley values in a plug-and-play open-source library to facilitate causal analysis of treatments.

Leveraging the additive properties of Shapley values provides the opportunity for predictive variable analysis and interpretation. However, Shapley values simply look at encountered associations in the model and therefore do not leverage causal structures. Furthermore, ML models try to optimize these associations to optimize their predictions. Therefore, the outputted Shapley values are devoid of causal direction. There are implications for *Causalteshap* because of this. First, suppose we have $T \rightarrow Y \rightarrow Z$ with Z an effect of Y. If Z is included, it will be an important variable to predict Y because of their high association. Furthermore, T will determine Z indirectly. Therefore, the Shapley value distribution of feature Z for $T = 0$ and $T = 1$ can differ and will be considered predictive. Even stronger, Z can even mask the true predictive features as the model might optimize the strongest associations to Y. Second, consider the case with strong confounding and limited predictive features. Suppose $Y \leftarrow C \rightarrow T$ and no true predictive features. Given the following: if $C < -1$ then $T = 0$, if $C > 1$ then $T = 1$, else T is randomly determined. C is uniformly distributed between -10 and 10. If C is included in *Causalteshap*, the Shapley distributions for feature C when $T = 0$ and $T = 1$ can be considered predictive because of its confounding effect and the absence of causal direction in ML models and Shapley. Do note that this effect is less when considering direct meta-learners, such as the X- and R-learner, which are more robust to this confounding effect and can also be integrated in *Causalteshap* as extensions. These are simple examples, however, they can produce false positives. Therefore, when using *Causalteshap* and interpreting its results it is advised to not blindly apply it to data and think about the causal

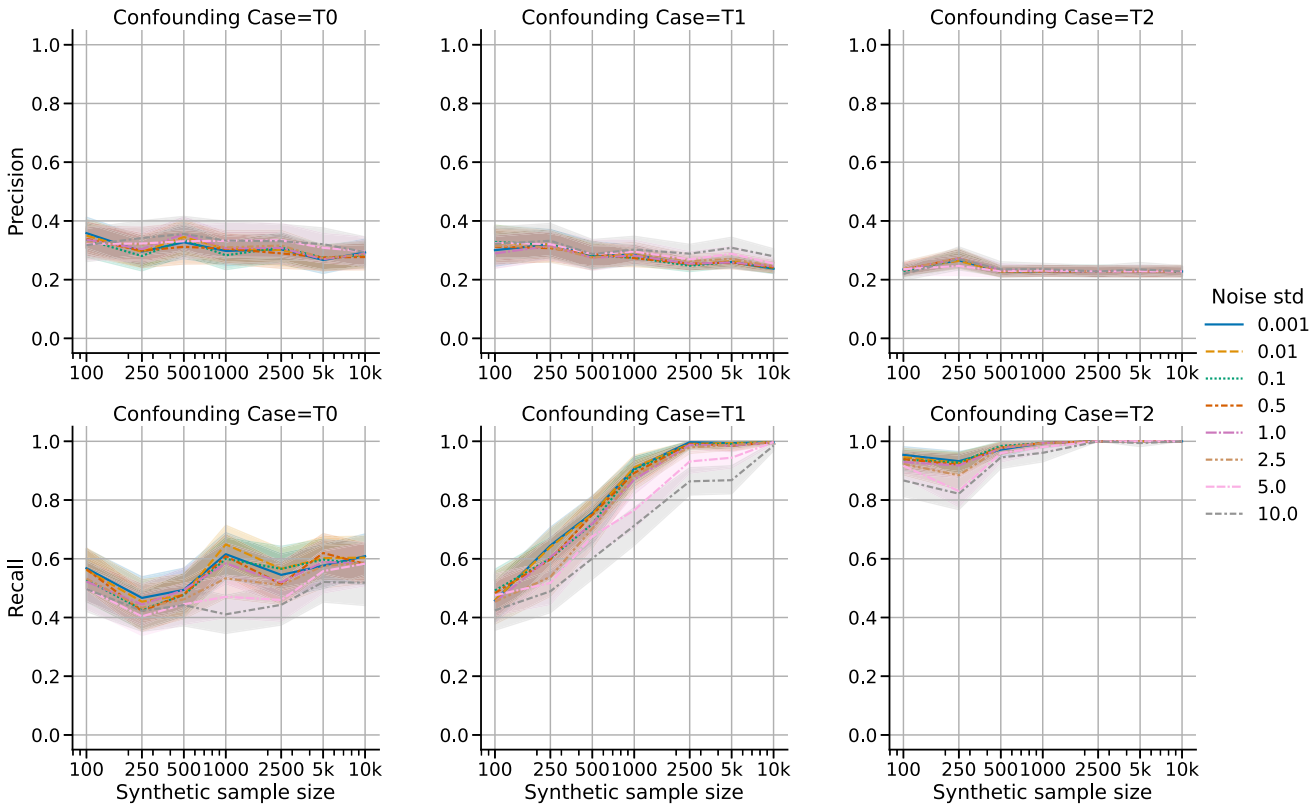


(a) Causalteshap



(b) Modified Covariate Method (MCM) by Tian et al. [20]

Fig. 3 Benchmarking results synthetic datasets



(c) Sparse Additive Models (SAM) by Park et al. [21]

Fig. 3 continued

data-generating process, before and after applying the method. Further, when applying *Causalteshap*, verify the dataset to the original Causal diagram of *Causalteshap*, shown in Fig. 2. If it violates the diagram, *Causalteshap* can still be applied, although with caution.

The synthetic and semi-synthetic datasets do provide a closed environment in which the method can be developed, evaluated, and tested. These datasets try to mimic real-world datasets, while still having access to the ground truth behind these datasets. However, there is a large issue with the DGP of the semi-synthetic datasets for predictive and prognostic features. The predictive and prognostic features are chosen at random to determine the outcome. However, the values of the features themselves are not necessarily independent. Suppose we have two features X and Z, where in its true DGP $X \rightarrow Z$. For the semi-synthetic dataset, X is selected to be solemnly prognostic, while Z is selected to be predictive. X will then be the cause of the predictive feature. Therefore, X could then also be interpreted as a predictive feature, which is a contradiction to its label. Subsequently, giving the label *predictive* to X is intrinsically not wrong, but will be considered wrong in the semi-synthetic benchmark. This biases the precision, making the true precision higher than it seems. This factor has to be taken into account when working with semi-synthetic datasets as the semi-synthetic benchmark results can be worse than their true results.

Causalteshap is developed on the assumption that all included features for analysis are informative features relevant to either the treatment or the outcome. Incorporating numerous non-informative features, particularly in high-dimensional settings, can increase the complexity of the data analysis, potentially degrade the performance of *Causalteshap*, and consequently elevate the risk of false positives. Therefore, before applying *Causalteshap*, make sure to do proper feature selection beforehand, e.g. using PowerShap [27], or test the relevance of every feature to the hypothesis that you want to evaluate. For feature selection methods in machine learning, we refer to the survey of Dhal et al [37].

The presented method and code library are currently limited to an S-learner for CATE estimation. Future work can expand the method to other meta-learners, such as T-, X-, or R-learners. For the T-learner, the Shapley baseline

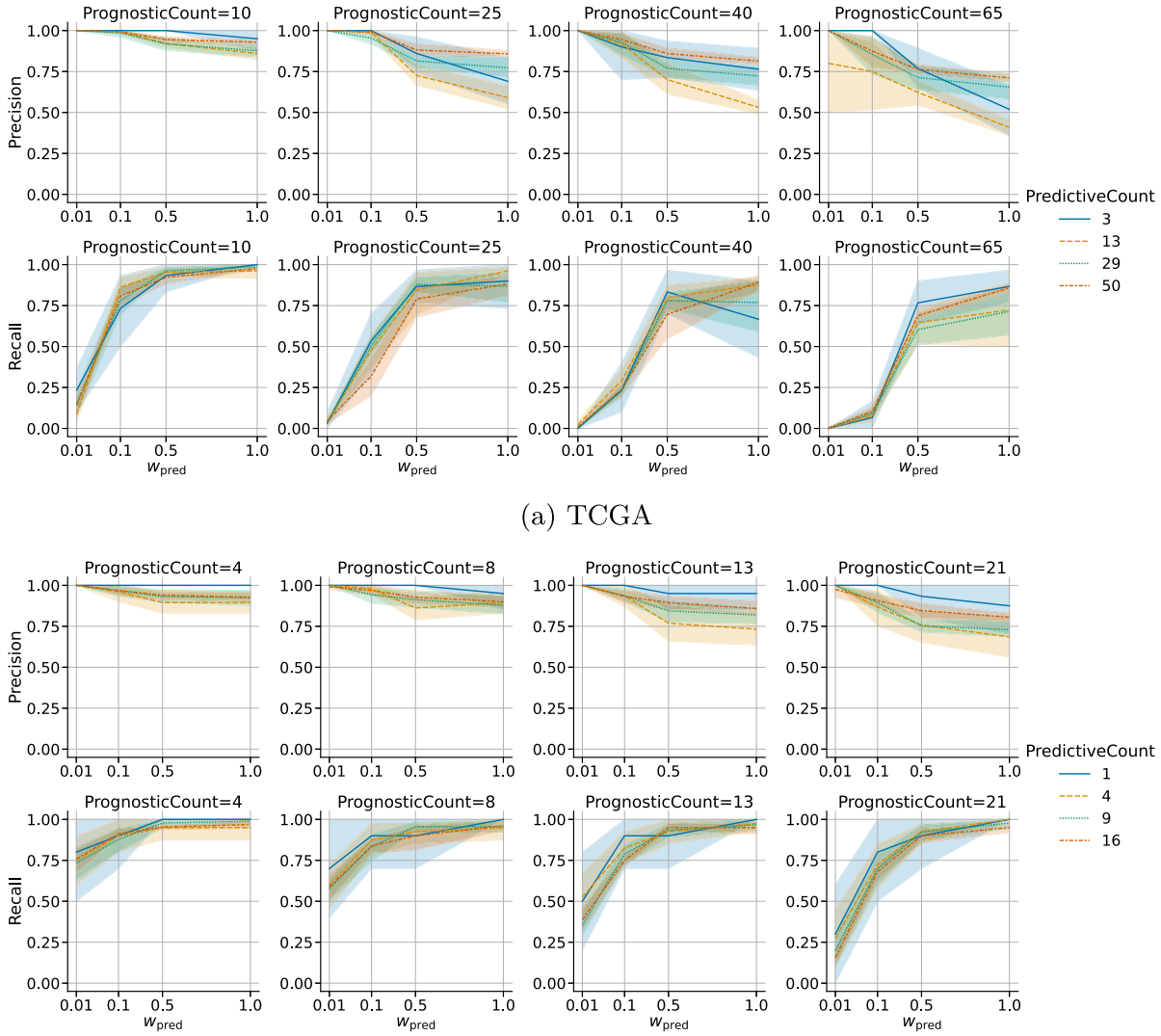
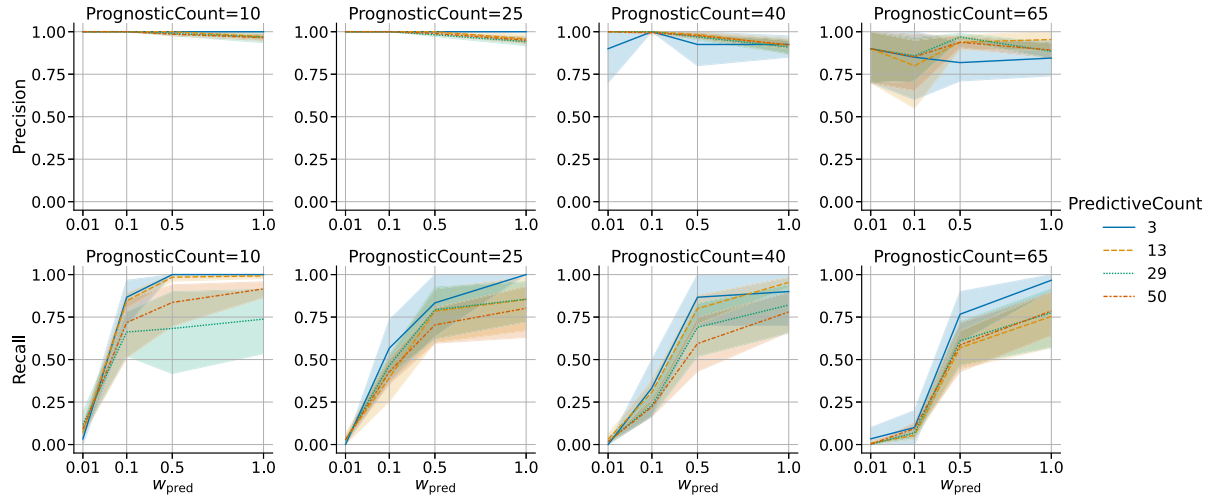


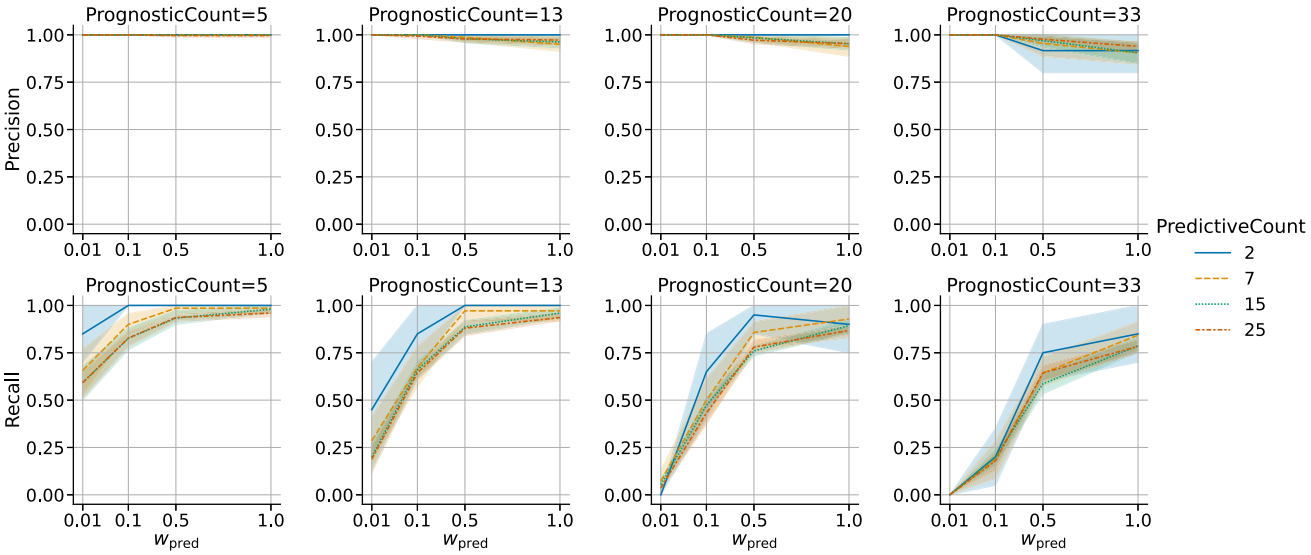
Fig. 4 Benchmarking results on semi-synthetic datasets. PrognosticCount represents the number of prognostic features, likewise for PredictiveCount. w_pred represents the strength of the predictive features

values can be different for each estimator. Therefore, comparing them using the current algorithm solution will not yield correct results, which requires changing the main algorithm to solve the problem. For direct meta-learners (such as an R-learner or X-learner), the direct CATE estimator in the meta-learner can be used to calculate the predictive Shapley values, however, the same problem as with the T-learner and incompatible Shapley values exists as well. A more detailed analysis can be found in the appendix C. As *Causalteshap* relies on meta-learners, it inherits their limitations. Violations of the unconfoundedness assumption (i.e., the presence of hidden confounders) can bias the estimated effects in unpredictable ways, as the underlying meta-learner would produce unreliable treatment effect estimates. Similarly, when the overlap (positivity) assumption is violated, i.e. meaning there is limited or no overlap in covariate distributions between treated and control groups, the model may generalize poorly. This can lead to exaggerated or underestimated Shapley values for predictive features, depending on the extent of distributional mismatch between $T = 0$ and $T = 1$ groups.

All benchmarks were currently performed using CatBoost and can differ when using different models [38]. Using another feature attribution method besides SHAP, that solves the causal direction problem, will also improve the results. *Causalteshap* can also be extended to multi-output regression or classification, i.e., scenarios with



(c) NEWS



(d) ACIC2016

Fig. 4 continued

multivariate outcomes, as SHAP inherently supports such settings by providing Shapley values for each outcome with respect to every variable. By aggregating these Shapley values across outcomes for each feature, e.g. through summation, the standard *Causalteshap* framework remains applicable, as the underlying hypotheses hold for the aggregated values. Finally, *Causalteshap* is currently developed for binary treatments or situations that can be reduced to binary treatments. Further work can include finding predictive features for continuous treatments to widen the application potential of the method.

6 Conclusion

We aimed to create an open-source plug-and-play sklearn-compatible method, named *Causalteshap*, for finding predictive features in treatment effect estimators by leveraging Shapley values and statistical tests. The benchmark results reveal the high precision of *Causalteshap*, while still achieving high recall rates, making it a tool for better treatment effect analysis, research, and interpretability. With *Causalteshap*, the step to truly discern predictive from prognostic features hereby comes closer.

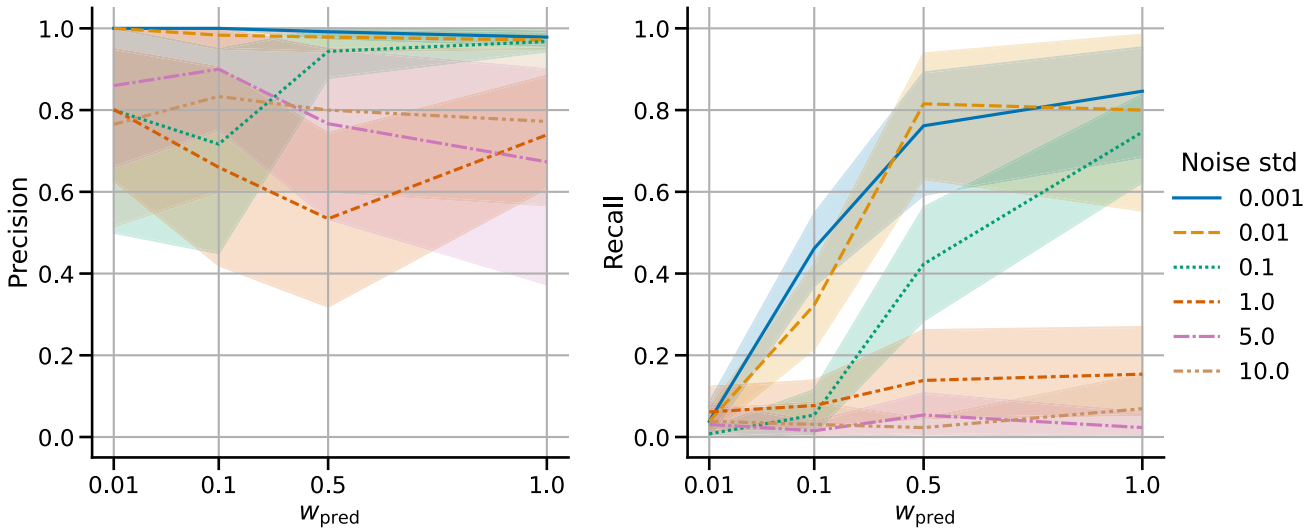


Fig. 5 Varying noise benchmarking results on the News semi-synthetic dataset with *N_pred* = 25 and *N_prog* = 13

Table 3 Causalteshap results for the Noradrenaline treatment analysis

Feature	KS statistic	ttest	Fligner	$T^1 - T^0$	$\lvert T^1\rvert - \lvert T^0\rvert$	Predictive
CVP	**0.03**	0.64	**0.00**	− 0.01	0.05	**1**
Fluid balance	**0.03**	0.87	**0.00**	0.00	− 0.07	**1**
Creatinin (blood)	**0.09**	0.98	**0.00**	0.00	− 0.03	**1**
T	**0.00**	**0.00**	**0.00**	0.62	0.43	**1**
Lactate (blood)	2.10	0.44	**0.02**	0.00	0.01	0
pH (blood)	0.21	0.95	**0.02**	0.00	0.02	0
Age	**0.00**	0.69	0.82	0.00	0.00	0
Heart frequency	9.44	0.91	0.76	0.00	0.00	0
O2 concentration (Set)	1.47	0.34	0.72	− 0.01	− 0.01	0
urgency	13.61	0.99	0.58	0.00	− 0.01	0
UrineCAD	0.21	0.90	0.26	0.00	− 0.01	0
Furosemide (Lasix)	7.01	0.96	0.64	0.00	0.01	0
ABP mean	6.26	0.88	0.62	0.00	− 0.01	0
O2-Saturation (blood)	5.19	0.99	0.04	0.00	0.02	0
Fosfate (blood)	19.87	0.96	0.91	0.00	0.00	0
ABP systolic	2.28	0.91	0.12	0.00	− 0.01	0
PEEP (Set)	3.57	0.92	0.52	0.00	− 0.01	0
Fluid in	3.66	0.55	0.11	0.00	− 0.01	0
Active HCO3 (blood)	8.15	0.50	0.97	0.00	0.00	0
Anion-Gap (blood)	26.11	0.98	0.60	0.00	0.00	0
Enoximon (Perfan)	20.44	0.90	0.85	0.00	0.00	0
Fentanyl	8.15	0.97	0.47	0.00	− 0.01	0
Glucose (blood)	10.76	0.68	0.18	0.00	− 0.01	0
Hydrocortisone	19.54	0.72	0.78	0.00	0.00	0
Magnesiumsulfate	18.97	0.76	0.96	0.00	0.00	0
Midazolam (Dormicum)	27.13	0.93	0.91	0.00	0.00	0
Propofol (Diprivan)	15.05	0.94	0.38	0.00	0.01	0

Bold indicates significance with $\alpha = 0.02$

CVP central venous pressure; *ABP* arterial blood pressure; KS statistic condition is 0.07 for $\alpha = 0.02$

Detailed formulations of statistical tests

Welch's t-test

Welch's t-test is calculated using the following formulae, with μ being the mean, X, A, and B arrays, and N the length of the respective array [39]:

$$s_X = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - \mu(X))^2} \tag{8}$$

$$t(A,B) = \frac{\mu(A) - \mu(B)}{\sqrt{\left(\frac{s_A}{\sqrt{N_A}}\right)^2 + \left(\frac{s_B}{\sqrt{N_B}}\right)^2}} \tag{9}$$

The degrees of freedom v for the test is then calculated using the Welch–Satterthwaite equation as:

$$v(A,B) = \frac{s_A^4 + s_B^4}{v_A^{-1}s_A^4 + v_B^{-1}s_B^4} \tag{10}$$

The equality only holds if the sample sizes of the two distributions are the same, i.e. $N_A = N_B$, which is always the case in this work.

With $v_X = N_X - 1$ the degrees of freedom for X. Then, given v, we can define the Cumulative Distribution Function (CDF) of the t-distribution $T(t)$ and estimate the two-tailed p-value using:

$$WelchsTTest(A,B) = 2(1 - T(|t(A,B)|, v(A,B))) \tag{11}$$

The Fligner test

The test is calculated as follows given two arrays, A and B, each with a length N_A and N_B, medians η_A and η_B, and $N_{tot} = N_A + N_B$ the total length of A and B together.

First, define $V = |A - \eta_A|$ and $W = |B - \eta_B|$. Then, calculate the rank of each element of the combined array $[V, W]$ as $R_V = rank(V, [V, W])$ and $R_W = rank(W, [V, W])$. Given these ranks, the standard deviations away from the mean of the total array $[V, W]$ are calculated for every element using the percentile point function (i.e. the inverse of the CDF) of the Normal distribution *NormPercentile* as follows:

$$S_A = NormPercentile\left(\frac{R_V}{2(N_{tot}+1)} + 0.5\right) \tag{12}$$

$$S_B = NormPercentile\left(\frac{R_B}{2(N_{tot}+1)} + 0.5\right) \tag{13}$$

With S being S_A and S_B appended. This now allows us to calculate the p-value of the Fligner test which is calculated using the Chi-squared CDF function $Chi(\chi)$ with estimated chi-square statistic over S (i.e. Fligner statistic) χ_{est}:

$$\sigma(S) = \frac{\sum_i (S_i - mean(S))^2}{len(S) - 1} \tag{14}$$

$$\chi_{est} = \frac{N_A(\sum_i(S_{A,i}) - mean(S))^2 + N_B(\sum_i(S_{B,i}) - mean(S))^2}{\sigma(S)} \tag{15}$$

$$Fligner(A,B) = 1 - Chi(\chi_{est}, 1) \tag{16}$$

This results in the Fligner–Killeen p-value of A and B.

Kolmogorov–Smirnov test

To calculate the KS-test we first need to calculate the two CDFs F_i and F_R:

$$F_i = CDF(|S_{pred}(X_i[1..N])|) \tag{17}$$

$$F_R = CDF(|S_{pred}(X_r[1..N])|) \tag{18}$$

Given these CDFs, the null hypothesis for the KS statistic is defined as:

$$\forall x : F_i(x) <= F_R(x) \tag{19}$$

The KS statistic for this case is then calculated as

$$D^+ = \sqrt{n} \cdot sup_x(F_i(x) - F_R(x)) \tag{20}$$

To verify whether the KS statistic rejects the null hypothesis or not we need to define a threshold. Suppose α represents the probability of the value being lower than the random variable K (Kolmogorov distribution). Then, given the approximate CDF of this function (which holds for $N > 30$) [40] we get:

$$\alpha = F_K(x) = Pr[K \leq x] = 1 - e^{-2x^2} \tag{21}$$

We can calculate the quantile function by inversing $F_K(x)$:

$$Q_K(\alpha) = \sqrt{-0.5 * ln(1 - \alpha)} \tag{22}$$

Then given a predefined α, we can reject the hypothesis if $D^+ > Q_K(\alpha)$.

Multiple testing corrections and bounds for Causalteshap

Causalteshap performs multiple statistical testing for every single feature. Therefore, we need to discuss the multiple testing problems for a single feature. For Part 1, we have a hypothesis that is the union of the Fligner test T_f and of Welch's t-test T_w. For clarity, we will call the first part, part A or test T_A. Part 2 consists of a single Kolmogorov–Smirnov test, T_{KS}. We will denote the second part as part B or test $T_B = T_{KS}$. Part A and Part B must both be true for *Causalteshap* to flag a feature as predictive. We will denote

this composite test as T_p, which always applies to a single feature.

$$T_A = T_f \vee T_w \tag{23}$$

$$T_p = T_A \wedge T_B = (T_f \vee T_w) \wedge T_B \tag{24}$$

We will assume a single α for T_f, T_w, and T_{KS}. So T_A is positive if either T_f or T_w is positive. Supposing all tests are independent we then have the following α_A for part A:

$$\alpha_A = 1 - (1 - \alpha)^2 = 2\alpha - \alpha^2 \tag{25}$$

For part B we have $\alpha_B = \alpha$ because we are only performing a single test. Given independence between part A and part B, the total α_p is:

$$\alpha_p = \alpha_A \alpha_B = 2\alpha^2 - \alpha^3 \tag{26}$$

However, independence between all these tests is hard to guarantee. Therefore, given the setup, a positive dependency is highly likely. Thus, we also will assume positive dependency to assess the robustness of the composite test. Given positive dependency between T_f and T_w the lower and upper bound of T_A becomes:

$$\alpha \leq \alpha_A \leq 2\alpha - \alpha^2 \tag{27}$$

Likewise, if there is positive dependency between T_A and T_B the upper and lower bound of T_p becomes:

$$\alpha_A \alpha_B \leq \alpha_p \leq min(\alpha_A, \alpha_B) \tag{28}$$

Combining these bounds, we get the following lower and upper bounds on α_p:

$$\alpha^2 \leq \alpha_p \leq \alpha \tag{29}$$

Therefore, given even positive dependency between any of the tests controlling for false positives or multiple testing for T_p is not necessary as the actual false positive rate will be most likely lower than the specified α. Therefore, by setting the significance threshold for all base tests (T_f, T_w, T_{KS}) to α guarantees at least a significance threshold for T_p of α as well.

As the hypotheses of *Causalteshap* are formulated for a single feature we are not making any conclusions over a set of features. Therefore, there is no need to control for multiple comparisons across these features. However, if there are conclusions that are being made on the model, such as a set of features being predictive, then of course a multiple comparison adjustment must incorporated.

Extension of Causalteshap to other meta-learners

T-learner

A T- or Twin-learner is a meta-learner that fits two estimators: One for $T = 1$ and one for $T = 0$:

$$\hat{\mu}_0(X) = \hat{E}[Y|X, T = 0] \tag{30}$$

$$\hat{\mu}_1(X) = \hat{E}[Y|X, T = 1] \tag{31}$$

The extension to *Causalteshap* is then trivial with S_{μ_0} simply the Shapley values of $\hat{\mu}_0(X)$ and vice versa. However, the main issue here lies with the baseline values of Shapley that are different which makes comparing these Shapley values with each other more complex and more prone to false positives.

X-learner

An X-learner is considered a direct or A-learner and consists of three estimators [41]: the treatment outcome estimator μ_1, the control outcome estimator μ_0, and the CATE model τ. The first estimators are trained as follows, analogous to a T-learner:

$$\hat{\mu}_0(X) = \hat{E}[Y|X, T = 0] \tag{32}$$

$$\hat{\mu}_1(X) = \hat{E}[Y|X, T = 1] \tag{33}$$

We then define the pseudo-outcomes CATE estimators τ_0 and τ_1:

$$\hat{\tau}_0(X) = \hat{E}[\hat{\mu}_1(X) - Y|X, T = 0] \tag{34}$$

$$\hat{\tau}_1(X) := \hat{E}[Y - \hat{\mu}_0(X)|X, T = 1] \tag{35}$$

We then define a new function $g(x)$ that weights τ_0 and τ_1 according to a function, often also a propensity estimator. This results in the final CATE estimator:

$$\hat{\tau}(X) = g(x)\hat{\tau}_0(X) + (1 - g(x))\hat{\tau}_1(x) \tag{36}$$

The first testing procedure of *Causalteshap* can be applied to the T-learner part $\hat{\mu}_0(X)$ and $\hat{\mu}_1(X)$, but then we have the same issue of having different baseline values. The second part of *Causalteshap* is easy to apply to an X-learner, simply calculate the Shapley values of $\hat{\tau}(X)$ with an added random variable and perform the KS-test. An alternative would be to perform the KS-test on both $\hat{\tau}_0(X)$ and $\hat{\tau}_1(X)$ and do a Bonferroni correction and take the features that are predictive in both.

R-learner

An R-learner is considered a direct or A-learner and consists of three base learners: a nuisance estimator $\hat{\mu}$, propensity estimator $\hat{\pi}$, and pseudo-outcome estimator $\hat{\tau}$. Given the features X, treatment T, and outcome Y, the learners are trained as follows [42]:

$$\hat{\mu} = \hat{E}[Y|X] \tag{37}$$

$$\hat{\pi} = \hat{E}[T|X] \tag{38}$$

If you are training a gradient-boosting pseudo-outcome estimator you fit the following function:

$$\hat{\tau} = \hat{E}\left[\frac{Y - \hat{\mu}(X)}{T - \hat{\pi}(X)}|X\right] \tag{39}$$

You do add weighted learning where every sample is weighted with weights $w = (T - \hat{\pi}(X))^2$. On the contrary, if you train a linear model such as lasso you regress $Y - \hat{\mu}\ T - \pi(\hat{X})$.

Given these models you can define the predicted potential outcome under control $\hat{\mu}_0$ and outcome under treatment $\hat{\mu}_1$ as follows:

$$\hat{\mu}_0(X) = \hat{\mu}(X) - \hat{\pi}(X)\hat{\tau}(X) \tag{40}$$

$$\hat{\mu}_1(X) = \hat{\mu}(X) + (1 - \hat{\pi}(X))\hat{\tau}(X) \tag{41}$$

Consequently, we can define the Shapley values for each model and the potential outcomes. $S_\mu(X)$ are the shapley values of model $\hat{\mu}$ on features X, and likewise for all other models.

$$S_{\mu_0}(X) = S_\mu(X) - S_\pi(X)S_\tau(X) \tag{42}$$

$$S_{\mu_1}(X) = S_\mu(X) + (1 - S_\pi(X))S_\tau(X) \tag{43}$$

In theory, all variables included in the pseudo-outcome model $\hat{\tau}$ should be predictive as only these will explain the treatment outcome. However, in practice, it could be that other variables are included in $\hat{\tau}$ as this model trains on the whole covariate set, especially if trained on a high-dimensional covariate set or with a complex treatment effect. If the treatment effect is simple, an easily interpretable model such as LASSO regression will suffice, however, if the treatment effect is more complex a stronger non-linear model might be required where the predictive variables might not be easily interpreted. To directly apply *Causalteshap* to an R-learner, the first part (Fligner and Welch's test) will be applied to $S_{\mu_0}(X)$ for $T = 0$ and $S_{\mu_1}(X)$ for T=1. Likewise, the second part (KS-test) can be applied on S_τ which must be fitted on $[X, X_r]$ with X_r a random feature.

However, $S_{\mu_1}(X) - S_{\mu_0}(X) = S_\tau(X)$. Therefore, the Shapley values of $\hat{\tau}$ are the only difference between $T = 0$ and $T = 1$. Compared to an S-learner, where you compare the model to a slightly different version of itself, this R-learner will be much less robust to random noise in Shapley values. If we have considerable Shapley attribution to noise in $\hat{\tau}$, the Fligner test or Welch's t-test can generate more false positives due to this noise attribution. A perfectly viable alternative would be to perform feature selection on the $\hat{\tau}$ estimator.

Acknowledgements Jarne Verhaeghe is funded by the Research Foundation Flanders (FWO, Ref. 1S59522N). Special thanks go to Thomas De Corte for a medical analysis of the predictive features of Noradrenaline for AF. This research was funded by the FWO Junior Research project HEROI2C which investigates hybrid machine learning for improved infection management in critically ill patients (Ref. G085920N).

Author Contributions Jarne Verhaeghe conceived the study design, wrote the manuscript, designed the library, and created the experiments. Femke Ongenae and Sofie Van Hoecke supervised the study and reviewed the code and manuscript.

Funding Jarne Verhaeghe is funded by the Research Foundation Flanders (FWO, Ref. 1S59522N). The other authors declare no conflict of interest. Part of the research was funded by the FWO Junior Research project HEROI2C which investigates hybrid machine learning for improved infection management in critically ill patients (Ref. G085920N) and by the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

Data availability The analysed and used datasets and code, except for the AF dataset are available on GitHub using the following link: https://github.com/predict-idlab/causalteshap. The AF dataset is based on the AmsterdamUMCdb dataset, which requires approval. More information can be found here: https://github.com/AmsterdamUMC/AmsterdamUMCdb.

Declarations

Ethics approval The AF dataset is based upon the AmsterdamUMCdb, which is an open-source dataset and therefore requires no ethical approval.

References

1. Pearl J, Mackenzie D (2018) The book of why. Basic Books, New York

2. Forney A, Mueller S (2022) Causal inference in AI education: A primer. J Causal Inference 10:141–173. https://doi.org/10.1515/jci-2021-0048/html
3. Feuerriegel S et al (2024) Causal machine learning for predicting treatment outcomes. Nat Med 30:958–968. https://doi.org/10.1038/s41591-024-02902-1
4. Ling Y, Upadhyaya P, Chen L, Jiang X, Kim Y (2023) Emulate randomized clinical trials using heterogeneous treatment effect estimation for personalized treatments: methodology review and benchmark. J Biomed Inf 137:104256
5. Crabbé J, Curth A, Bica I, van der Schaar M (2022) Benchmarking heterogeneous treatment effect models through the lens of interpretability. arXiv:2206.08363
6. Rubin DB (2005) Causal inference using potential outcomes. J Am Stat Assoc 100:322–331. https://doi.org/10.1198/016214504000001880
7. Künzel SR, Sekhon JS, Bickel PJ, Yu B (2019) Metalearners for estimating heterogeneous treatment effects using machine learning. Proc Natl Acad Sci 116:4156–4165. https://doi.org/10.1073/pnas.1804597116. (**Publisher: Proceedings of the National Academy of Sciences**)
8. Hermansson E, Svensson D, Gervasi O et al (2021) (eds) On discovering treatment-effect modifiers using virtual twins and causal forest ml in the presence of prognostic biomarkers. In: Gervasi O et al. (eds) Computational science and its applications—ICCSA 2021, Springer International Publishing, Cham, pp 624–640
9. Lundberg SM, Lee S-I (2017) A unified approach to interpreting model predictions. In: Guyon I et al (eds) Advances in Neural Information Processing Systems, vol 30, Curran Associates, Inc., pp 4765–4774. http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf
10. Linardatos P, Papastefanopoulos V, Kotsiantis S (2020) Explainable AI: a review of machine learning interpretability methods. Entropy 23:18
11. Zhang Z, Seibold H, Vettore MV, Song W-J, François V (2018) Subgroup identification in clinical trials: an overview of available methods and their implementations with r. Ann Transl Med 6. https://atm.amegroups.com/article/view/19049
12. Liu Y et al (2019) Look before you leap: systematic evaluation of tree-based statistical methods in subgroup identification. J Biopharm Stat 29:1082–1102
13. Alemayehu D, Chen Y, Markatou M (2018) A comparative study of subgroup identification methods for differential treatment effect: Performance metrics and recommendations. Stat Methods Med Res 27:3658–3678
14. Bonetti M, Gelber RD (2004) Patterns of treatment effects in subsets of patients in clinical trials. Biostatistics (Oxford, England) 5:465–481
15. Lipkovich I, Dmitrienko A, Denne J, Enas G (2011) Subgroup identification based on differential effect search—a recursive partitioning method for establishing response to treatment in patient subpopulations. Stat Med 30:2601–2621
16. Foster JC, Taylor JM, Ruberg SJ (2011) Subgroup identification from randomized clinical trial data. Stat Med 30. https://doi.org/10.1002/sim.4322
17. Cai T, Tian L, Wong PH, Wei LJ (2011) Analysis of randomized comparative clinical trial data for personalized treatment selections. Biostatistics 12:270–282
18. Doove LL, Van Deun K, Dusseldorp E, Van Mechelen I (2016) QUINT: A tool to detect qualitative treatment-subgroup interactions in randomized controlled trials. Psychother Res 26:612–622
19. Guo X et al (2023) Assessing the most vulnerable subgroup to type II diabetes associated with statin usage: evidence from electronic health record data. J Am Stat Assoc 118:1488–1499
20. Tian L, Alizadeh AA, Gentles AJ, Tibshirani R (2014) A simple method for estimating interactions between a treatment and a large number of covariates. J Am Stat Assoc 109:1517–1532
21. Park H, Petkova E, Tarpey T, Ogden RT (2022) A sparse additive model for treatment effect-modifier selection. Biostatistics 23:412–429
22. De Luna X, Waernbaum I, Richardson TS (2011) Covariate selection for the nonparametric estimation of an average treatment effect. Biometrika 98:861–875
23. Cheng D et al (2023) Local search for efficient causal effect estimation. IEEE Trans Knowl Data Eng 35:8823–8837
24. Lumley T, Diehr P, Emerson S, Chen L (2002) The importance of the normality assumption in large public health data sets. Annu Rev Public Health 23:151–169. https://doi.org/10.1146/annurev.publhealth.23.100901.140546
25. Fligner MA, Killeen TJ (1976) Distribution-free two-sample tests for scale. J Am Stat Assoc 71:210–213. https://doi.org/10.1080/01621459.1976.10481517
26. Pedregosa F, Varoquaux G, Gramfort A et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
27. Verhaeghe J, Van Der Donckt J, Ongenae F, Van Hoecke Sm Amini M-R et al (2023) (eds) Powershap: a power-full shapley feature selection method. In: Amini M-R et al (eds) Machine learning and knowledge discovery in databases, Springer International Publishing, pp 71–87
28. Verhaeghe J et al (2023) Generalizable calibrated machine learning models for real-time atrial fibrillation risk prediction in ICU patients. Int J Med Inform 175:105086
29. Weinstein JN et al (2013) The cancer genome atlas pan-cancer analysis project. Nat Genet 45:1113–1120
30. Almond D, Chay KY, Lee DS (2005) The costs of low birth weight. Q J Econ
31. Newman D (2008) Bag of Words. https://archive.ics.uci.edu/dataset/164
32. Dorie V, Hill J, Shalit U, Scott M, Cervone D (2019) Automated versus do-it-yourself methods for causal inference: lessons learned from a data analysis competition. Stat Sci 34. https://projecteuclid.org/journals/statistical-science/volume-34/issue-1/Automated-versus-Do-It-Yourself-Methods-for-Causal-Inference/10.1214/18-STS667.full
33. Thoral PJ et al (2021) Sharing ICU patient data responsibly under the society of critical care medicine/European society of intensive care medicine joint data science collaboration: the Amsterdam University Medical Centers Database (AmsterdamUMCdb) Example*. Crit Care Med 49:e563
34. Yoshida T, Fujii T, Uchino S, Takinami M (2015) Epidemiology, prevention, and treatment of new-onset atrial fibrillation in critically ill: a systematic review. J Intens Care 3:0000062
35. Moss TJ et al (2017) New-onset atrial fibrillation in the critically ill. Crit Care Med 45(790–797):0000096
36. Wasmer K, Eckardt L, Breithardt G (2017) Predisposing factors for atrial fibrillation in the elderly. J Geriat Cardiol 14:179–184
37. Dhal P, Azad C (2021) A comprehensive survey on feature selection in the various fields of machine learning. Appl Intell 52:4543–4581. https://doi.org/10.1007/s10489-021-02550-9
38. Prokhorenkova L, Gusev G et al (2019) CatBoost: unbiased boosting with categorical features. arXiv:1706.09516
39. Welch BL (1947) The generalization of 'student's' problem when several different population varlances are involved. Biometrika 34:28–35. https://doi.org/10.1093/biomet/34.1-2.28
40. Knuth DE (1997) The art of computer programming, Ch. 3.3.1, 52. Addison-Wesley, Reading, Mass, 3rd edn
41. Künzel SR, Sekhon JS, Bickel PJ, Yu B (2019) Metalearners for estimating heterogeneous treatment effects using machine learning. Proc Natl Acad Sci 116:4156–4165

42. Nie X, Wager S (2020) Quasi-oracle estimation of heterogeneous treatment effects. Biometrika 108:299–319. https://doi.org/10.1093/biomet/asaa076