25th International Conference on Artificial Intelligence and Statistics
AISTATS 2022

Counterfactual Explanation Trees: Transparent and Consistent Actionable Recourse with Decision Trees

Kentaro Kanamori (Hokkaido University)
Takuya Takagi (Fujitsu Ltd.)
Ken Kobayashi (Fujitsu Ltd. / Tokyo Institute of Technology)
Yuichi Ike (The University of Tokyo)

Background: Counterfactual Explanation (CE)

Explain an “action” for obtaining the desired prediction result

- Post-hoc methods for extracting “**local explanations**” from complex ML models have been massively studied.
- **Counterfactual Explanation (CE)** [Wachter+ 18]
  - As a local explanation for an instance $x \in \mathcal{X}$, CE provides an **action a^*** for obtaining the desired prediction result $y^* \in \mathcal{Y}$ from a model $f\colon \mathcal{X} \to \mathcal{Y}$.

$$a^* = \arg\min_{a \in \mathcal{A}} c(a \mid x) \;\; \text{subject to} \;\; f(x+a) = y^*$$

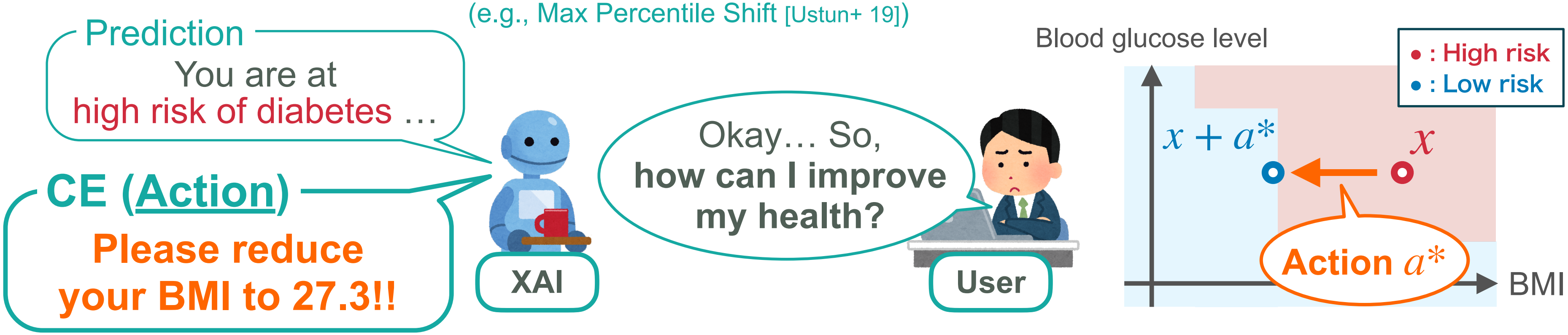
Background: Counterfactual Explanation (CE)

Explain an "action" for obtaining the desired prediction result

- Post-hoc methods for extracting "**local explanations**" from complex ML models have been massively studied.
- **Counterfactual Explanation (CE)** [Wachter+ 18]
  - As a local explanation for an instance $x \in \mathcal{X}$, CE provides an **action a^*** for obtaining the desired prediction result $y^* \in \mathcal{Y}$ from a model $f \colon \mathcal{X} \to \mathcal{Y}$.

$$a^* = \arg\min_{a \in \mathcal{A}} \boxed{c(a \mid x)} \text{ subject to } f(x+a) = y^*$$

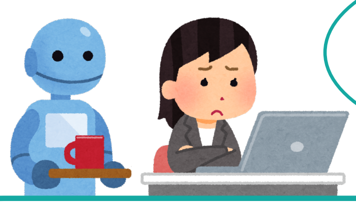
cost function
(e.g., Max Percentile Shift [Ustun+ 19])



Motivation: CE for "Multiple" Instances

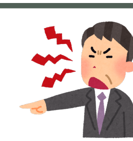
Assign actions to multiple instances $X \subset \mathcal{X}$ simultaneously

- Actions a optimized for individuals x are <u>not necessarily executed by the individuals themselves</u> [Karimi+ 20].
  - Ex.) **Attrition risk prediction** (e.g. *IBM HR Analytics Employee Attrition*[*1]):
    A company assigns actions to the employees to reduce their attrition risk.
- An action a for an individual x (e.g., increasing salary) may <u>affect other individuals</u> (e.g. changing payroll systems in the company).
  - In such a case, <u>optimizing an action for each of the individuals is insufficient</u>.



XAI & Decision-maker

Assigning actions to multiple employees simultaneously

Employees			…	
Actions	**Salary**: +12K$	**Overtime**: True→False	…	**Dept.**: Sales→HR

*1. https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset

Desideratum and Our Approach

Learn a transparent and consistent model assigning actions

- Desideratum of CE for multiple instances:
  - **Transparency** [Rawal+ 20]:
    We should explain how actions are determined for entire individuals.
  - **Consistency** [Rudin+ 19]:
    We should provide reasons of actions without conflicts between individuals.
    - Ex.) A reason (rule) "Age>35 & Dept.=Sales" conflicts between two employees because both of them satisfy the rule.

Why am I transferred?

Employees		
Features	**Age**: 37 **Dept.**: Sales **Overtime**: False **Performance**: A …	**Age**: 42 **Dept.**: Sales **Overtime**: False **Performance**: B …
Actions	**Salary: +12K$**	**Dept.: Sales → HR**

- **Our approach**:

Idea 1. Design a model that assigns effective actions over the entire input space $\mathcal{X}$ in a transparent and consistent way.

Idea 2. Design an algorithm for learning such a model from given instances $X \subset \mathcal{X}$.

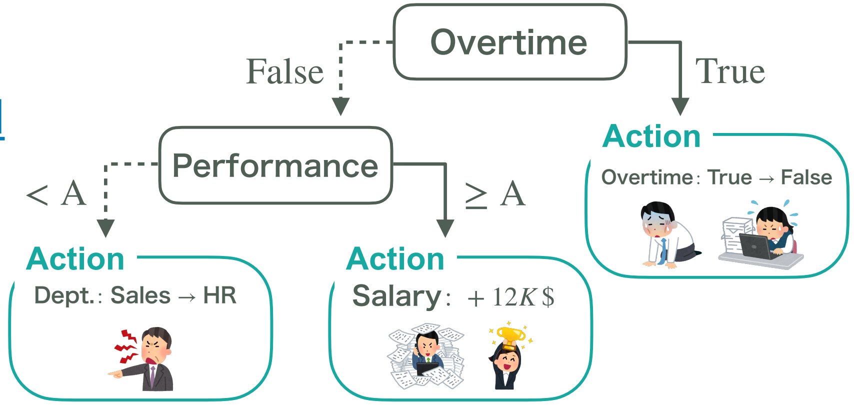
Our Framework: Counterfactual Explanation Tree

Decision tree for assigning effective actions over input space

Def. Counterfactual Explanation Tree (CET)

For a set of feasible actions $\mathcal{A}$, **Counterfactual Explanation Tree (CET)** is a **decision tree $h: \mathcal{X} \to \mathcal{A}$** assigning an action for an input instance $x \in \mathcal{X}$.

- Advantages of a decision tree:
  - It can provide a reason of each assigned action as a form of rules (transparency).
  - It guarantees to assign a unique pair of an action and its rule (consistency).



- Learning a CET h from $X \subset \mathcal{X}$ based on **Invalidity** score $i_\gamma(a \mid x) := \underbrace{c(a \mid x)}_{\text{Cost}} + \gamma \cdot \underbrace{l(f(x+a), y^*)}_{\text{Loss}}$.
  - Whether the action $a = h(x)$ assigned by h is effective for each instance $x \in X$.

(w.r.t. constraint $f(x+a) = y^*$)

Our Framework: Learning Algorithm for CET

Learn a CET from given instances by stochastic local search

Prob. Learning CET

Given instances $X \subseteq \mathcal{X}$ and parameters $\gamma, \lambda > 0$, find a CET h^* such that:

$$h^* = \arg\min_{h \in \mathcal{H}} \underbrace{\frac{1}{|X|} \sum_{x \in X} i_\gamma(h(x) \mid x)}_{\text{Average invalidity of actions}} + \lambda \cdot \underbrace{|\mathcal{L}(h)|}_{\text{\# Leaves (= \# Actions)}},$$

Theorem 1

$$|\mathcal{L}(h^*)| \leq \frac{\gamma + \lambda}{\lambda}$$

where, $\mathcal{H}$ is a set of CETs $h\colon \mathcal{X} \to \mathcal{A}$, and $\mathcal{L}(h)$ is the set of leaves in h.

- Adjust trade-off between effectiveness of actions by h and interpretability of h.

- Algorithm: **stochastic local search** (cf. [Wang 19] [Pan+ 20])
  - Branching rules in the internal nodes of the current CET $h^{(t)}$ are randomly updated by some edit operations (e.g., *insert*, *delete*, and *replace*).
  - As its subroutine, an action assigned to instances in each leaf is optimized by **extended MILO** (cf. [Ustun+ 19] [Kanamori+ 20])

Experiments (IBM Attrition dataset)

CET could assign effective actions in an interpretable way

- Comparison with "**AReS** [Rawal+ 20]" based on a rule set.
  - **Quantitative** comparison: effectiveness of actions assigned by each method.
  - **Qualitative** comparison: human-interpretability of each method by user study.

Results

- Our CET could assign more effective actions in terms of cost, loss, and invalidity than AReS, while CET ensured transparency and consistency.
- The behavior of our CET was easily understood by human-users.

▸ Our CET succeeded to assign effective actions in an interpretable way!

Dataset	Method	Cost	Loss	Invalidity
Train	AReS	0.436 ± 0.06	0.435 ± 0.07	0.871 ± 0.04
	CET	$\mathbf{0.349 \pm 0.1}$	$\mathbf{0.4 \pm 0.11}$	$\mathbf{0.749 \pm 0.05}$
Test	AReS	0.45 ± 0.08	$\mathbf{0.298 \pm 0.09}$	0.748 ± 0.09
	CET	$\mathbf{0.383 \pm 0.12}$	0.318 ± 0.19	$\mathbf{0.701 \pm 0.12}$

Method	User Acc.	Time [s]
AReS	95.12%	784.8 ± 202
CET	**100.0%**	$\mathbf{674.0 \pm 392}$

Experiments (IBM Attrition dataset)

CE… an interpretable way

• Con… as…
  • Q… …ethod.
  • Q… o… study.

Resu…

• Our … in terms of cost, loss, and inval… …sparency and consistency.

• The behavior of our CET was easily understood by huma…

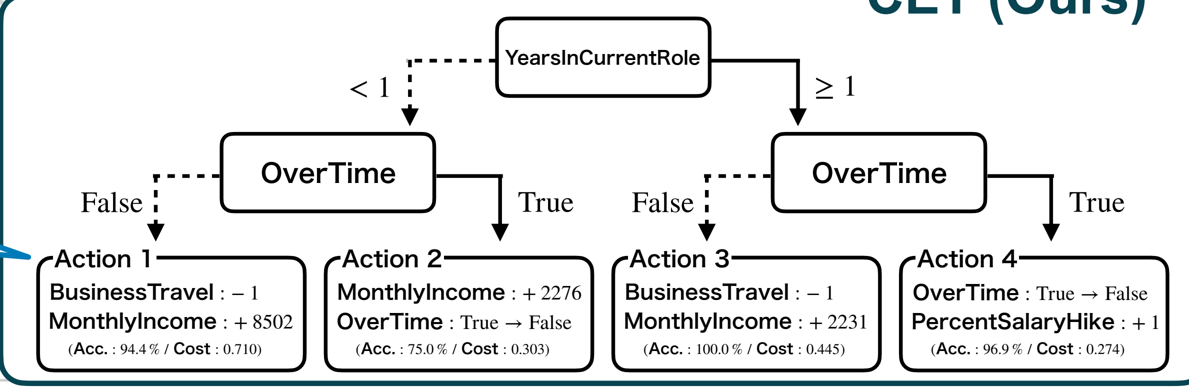
AReS [Rawal+ 20]

	Rule	Action
Recourse rule 1	If **'OverTime=True'** AND **'OutstandingPerformanceRating=False'** (Probability: 58.2%)	OverTime=False
Recourse rule 2	If **'BusinessTravel>=1'** AND **'OverTime=False'** (Probability: 13.9%)	BusinessTravel<1 AND OverTime=False
Recourse rule 3	If **'JobLevel<2'** AND **'MonthlyIncome<2275'** AND **'OverTime=False'** (Probability: 12.7%)	MonthlyIncome>=15170 AND OverTime=False
Recourse rule 4	If **'OverTime=True'** AND **'2<=YearsInCurrentRole<3'** (Probability: 24.1%)	OverTime=False AND 2<=YearsInCurrentRole<3
Default rule	**Else**	MonthlyIncome>=15170 AND OverTime=False

Cost: 47.0%
Loss: 8.7%

User Acc.: 95.1%
Time: 784.8 sec.

CET (Ours)



Cost: 41.0%
Loss: 4.3%

User Acc.: 100%
Time: 674.0 sec.

Summary of Our Contributions

A new framework of CE assigning actions to multiple instances

- We introduce **Counterfactual Explanation Tree (CET)**, that assigns effective actions to input instances with a **decision tree**.
  - **Transparency**: explain how actions are determined over the entire input space.
  - **Consistency**: explain reasons of assigned actions without conflicts between instances.
- We propose an **efficient algorithm for learning a CET** from given instances based on stochastic local search and MILO.
- By experiments and user studies, we confirmed **the efficacy and interpretability of our CET**.
- Future Work:
  - Scalability of our learning algorithm
  - Modeling interactions between instances