Introduction to Uplift Modeling

Dr. Juan Orduz

Mathematician & Data scientist

PyConDE & PyData Berlin 2022

Motivation

How can we optimally select customers to be treated by marketing incentives?

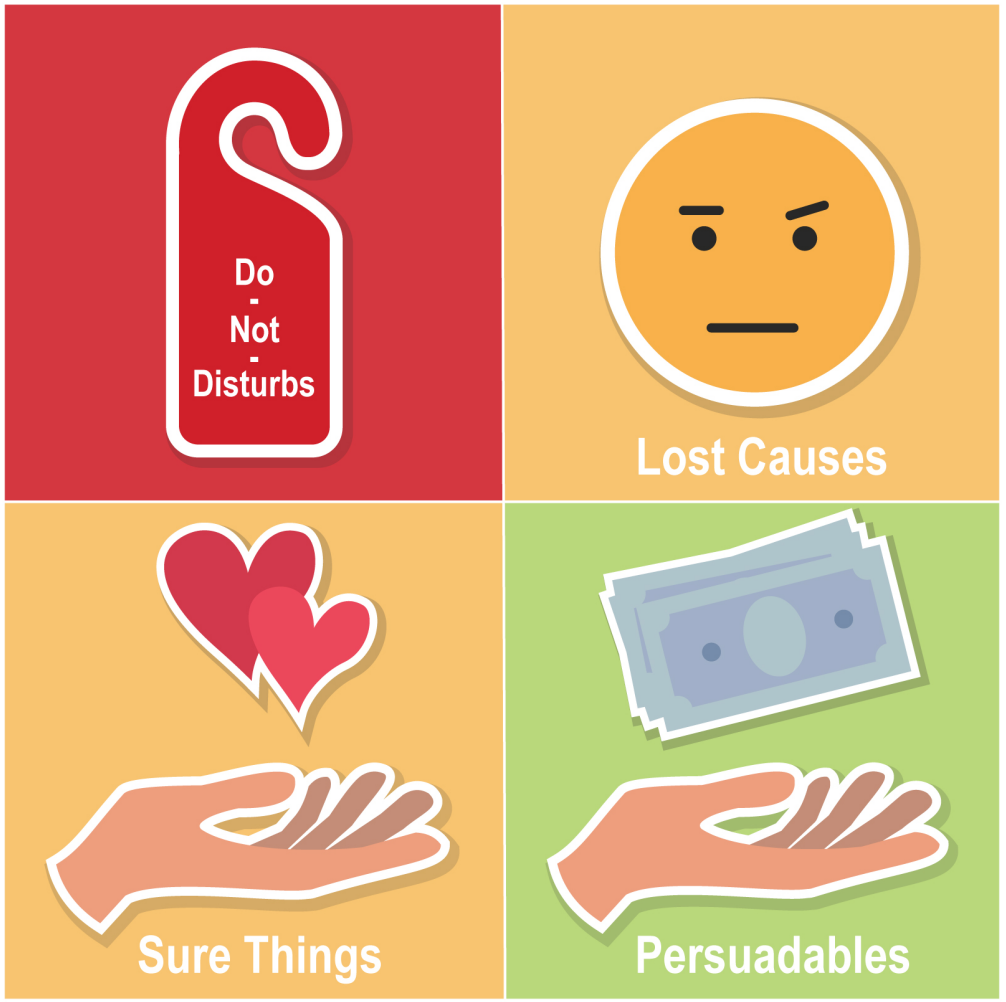


Image taken from https://www.uplift-modeling.com/en/latest/user_guide/introduction/clients.html

We can not send and not send incentives to the same customers at the same time

What is Uplift Modeling?

From Gutierrez, P., & Gérardy, J. Y. (2017). "Causal Inference and Uplift Modelling: A Review of the Literature"

- **Uplift modeling refers to the set of techniques used to model the incremental impact of an action or treatment on a customer outcome.**
- **Uplift modeling is therefore both a Causal Inference problem and a Machine Learning one.**

Conditional Average Treatment Effect

- Let Y_i^1 denote person i's outcome when it receives the treatment and Y_i^0 when it does not receive the treatment.
- We are interested in:
  - The *causal effect* $\tau_i := Y_i^1 - Y_i^0$
  - Given a feature vector X_i of the i-th person, we would like to estimate the *conditional average treatment effect*

$$CATE : \tau(X_i) := E[Y_i^1 | X_i] - E[Y_i^0 | X_i]$$

- **However, we can not observe them!** 😕

Taken from Gutierrez, P., & Gérardy, J. Y. (2017). *"Causal Inference and Uplift Modelling: A Review of the Literature"*

CATE Estimation

Let W_i is a binary variable indicating whether person i received the treatment, so that

$$Y_i^{obs} = Y_i^1 W_i + (1 - W_i) Y_i^0$$

Unconfoundedness Assumption

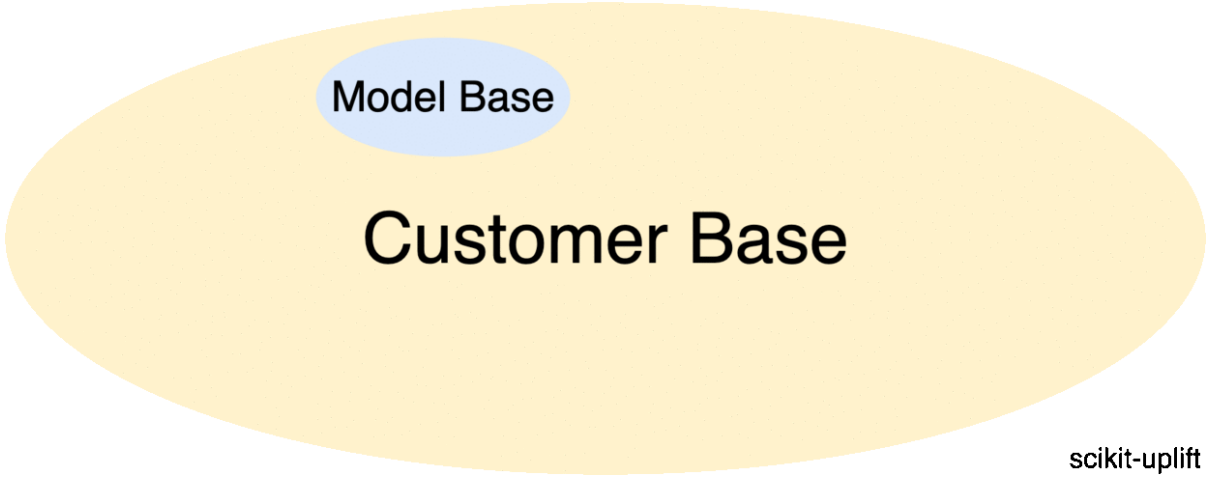
If we **assume** that the treatment assignment W_i is independent of Y_i^1 and Y_i^0 conditional on X_i, then we can estimate the $CATE$ from observational data by computing the empirical counterpart:

$$\textbf{uplift} = \widehat{\tau}(X_i) = E[Y_i^{obs} | X_i, W_i = 1] - E[Y_i^{obs} | X_i, W_i = 0]$$

Taken from Gutierrez, P., & Gérardy, J. Y. (2017). *"Causal Inference and Uplift Modelling: A Review of the Literature"*

🤔 Data Collection

1. Randomly subset the customer base to create a representative model base.



scikit-uplift

Taken from https://www.uplift-modeling.com/en/latest/user_guide/introduction/data_collection.html

Estimating Uplift

- Meta Algorithms ⟵ Today
- The Class Transformation Method
- Direct measurements (e.g. trees)

See Gutierrez, P., & Gérardy, J. Y. (2017). *"Causal Inference and Uplift Modelling: A Review of the Literature"*

S-Learner

Step 1: Training

$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} & w_1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{11} & \cdots & x_{nk} & w_n \end{pmatrix}}_{X \bigoplus W} \xrightarrow{\mu} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Step 2: Uplift Prediction

$$\widehat{\tau}(X) = \hat{\mu} \begin{pmatrix} x_{11} & \cdots & x_{1k} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{11} & \cdots & x_{mk} & 1 \end{pmatrix} - \hat{\mu} \begin{pmatrix} x_{11} & \cdots & x_{1k} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{11} & \cdots & x_{mk} & 0 \end{pmatrix}$$

Taken from https://www.uplift-modeling.com/en/latest/user_guide/models/index.html

T-Learner

Step 1: Training

$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{n_C k} \end{pmatrix}}_{X^C := X|_{\text{control}}} \xrightarrow{\mu_C} \begin{pmatrix} y_1 \\ \vdots \\ y_{n_C} \end{pmatrix}$$

$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{n_T k} \end{pmatrix}}_{X^T := X|_{\text{treatment}}} \xrightarrow{\mu_T} \begin{pmatrix} y_1 \\ \vdots \\ y_{n_T} \end{pmatrix}$$

Taken from https://causalml.readthedocs.io/en/latest/methodology.html#meta-learner-algorithms

T-Learner

Step 2: Uplift Prediction

$$\widehat{\tau}(X) = \hat{\mu}_T \begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{mk} \end{pmatrix} - \hat{\mu}_C \begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{mk} \end{pmatrix}$$

Taken from https://causalml.readthedocs.io/en/latest/methodology.html#meta-learner-algorithms

X-Learner

Step 1: Training: Same as T-Learner

Step 2: Compute imputed treatment effects

$$\tilde{D}^T := \begin{pmatrix} y_1 \\ \vdots \\ y_{n_T} \end{pmatrix} - \hat{\mu}_C \begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{n_T k} \end{pmatrix}$$

$$\tilde{D}^C := \hat{\mu}_T \begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{n_C k} \end{pmatrix} - \begin{pmatrix} y_1 \\ \vdots \\ y_{n_C} \end{pmatrix}$$

Taken from https://causalml.readthedocs.io/en/latest/methodology.html#meta-learner-algorithms

X-Learner

Step 3: Train with different targets

$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{n_C k} \end{pmatrix}}_{X|_{\text{control}}} \xrightarrow{\tau_C} \begin{pmatrix} \tilde{D}_1^C \\ \vdots \\ \tilde{D}_{n_T}^C \end{pmatrix}$$

$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{n_C k} \end{pmatrix}}_{X|_{\text{treatment}}} \xrightarrow{\tau_T} \begin{pmatrix} \tilde{D}_1^T \\ \vdots \\ \tilde{D}_{n_T}^T \end{pmatrix}$$

Taken from https://causalml.readthedocs.io/en/latest/methodology.html#meta-learner-algorithms

X-Learner

Step 4: Uplift Prediction

$$\widehat{\tau}(X) = g(X)\hat{\tau}_C(X) + (1 - g(X))\hat{\tau}_T(X)$$

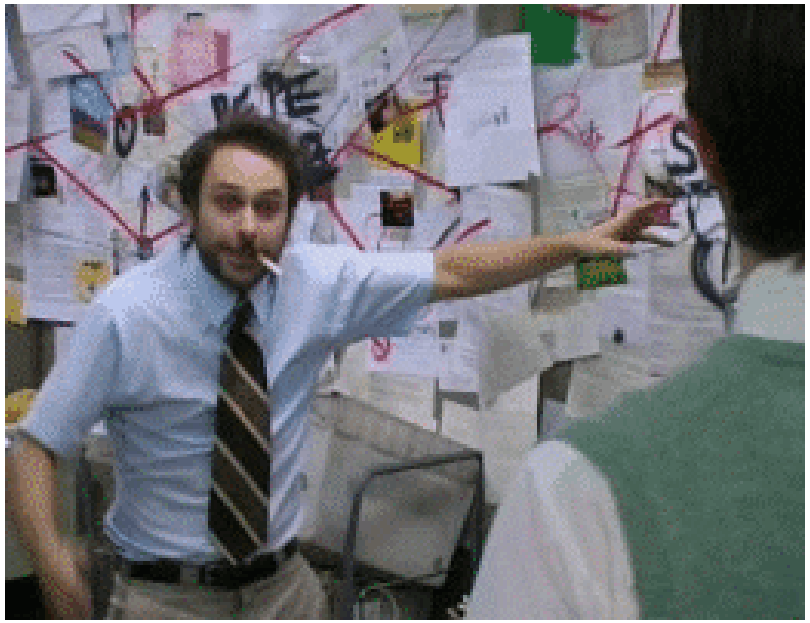
where $g(X) \in [0, 1]$ is a weight function.

Remark: A common choice for $g(X)$ is an estimator of the **propensity score**, which is defined as the probability of treatment given the covariates X, i.e. $p(W_i = 1|X_i)$.

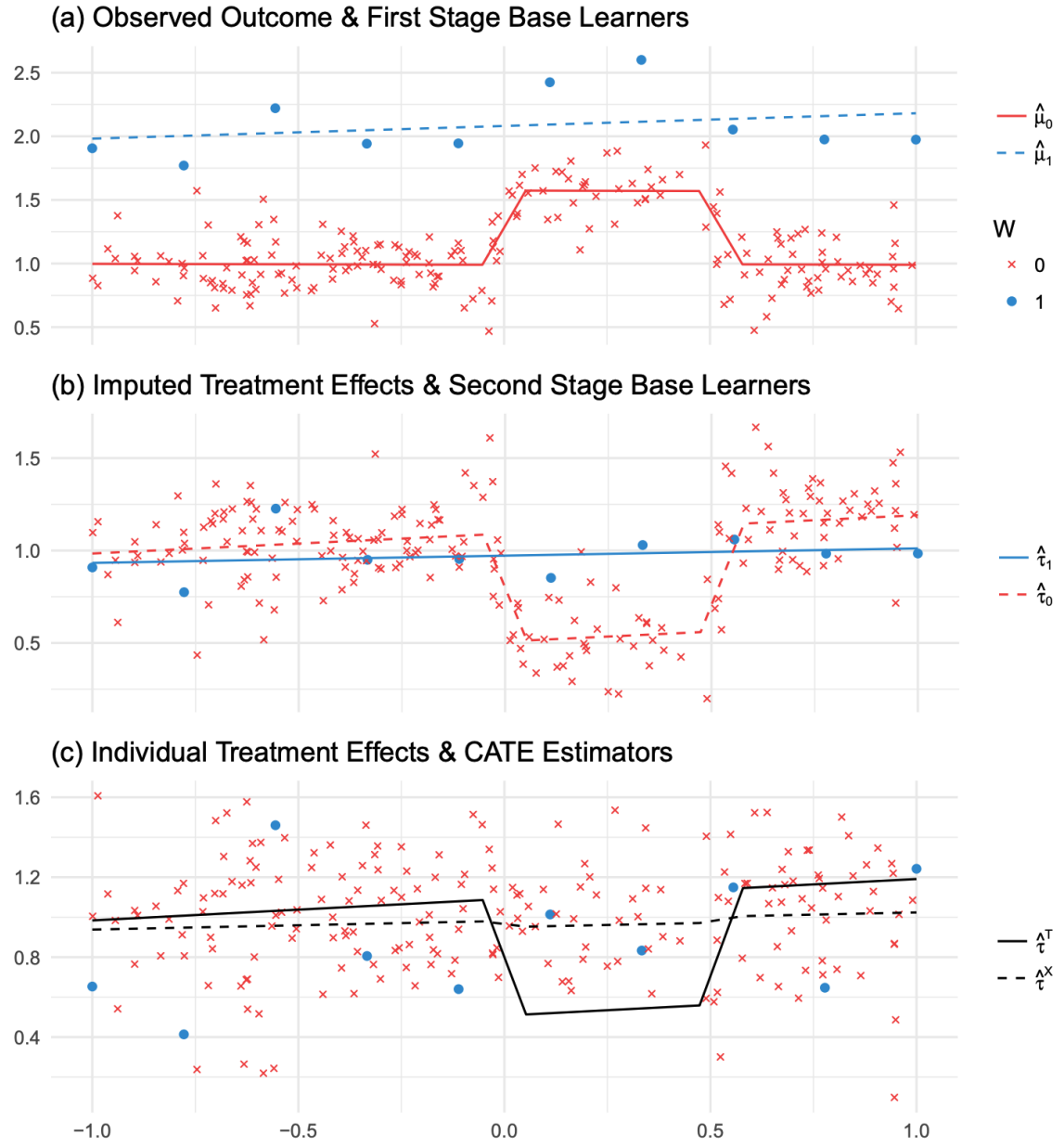
Taken from https://causalml.readthedocs.io/en/latest/methodology.html#meta-learner-algorithms

Intuition behind the X-Learner

We study an simulated example where we know the uplift is exactly $\tau = 1$.

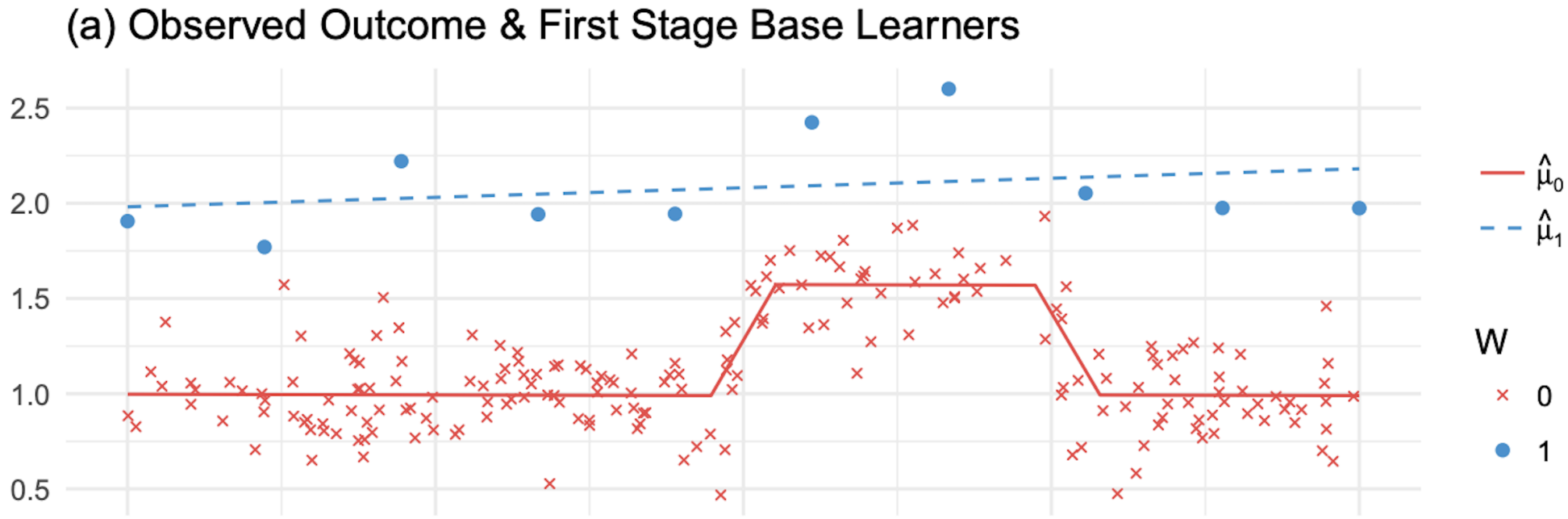


Taken from Sören, R, et.al. (2019) *"Meta-learners for Estimating Heterogeneous Treatment Effects using Machine Learning"*



X-Learner Step 1 (same as T-Learner):

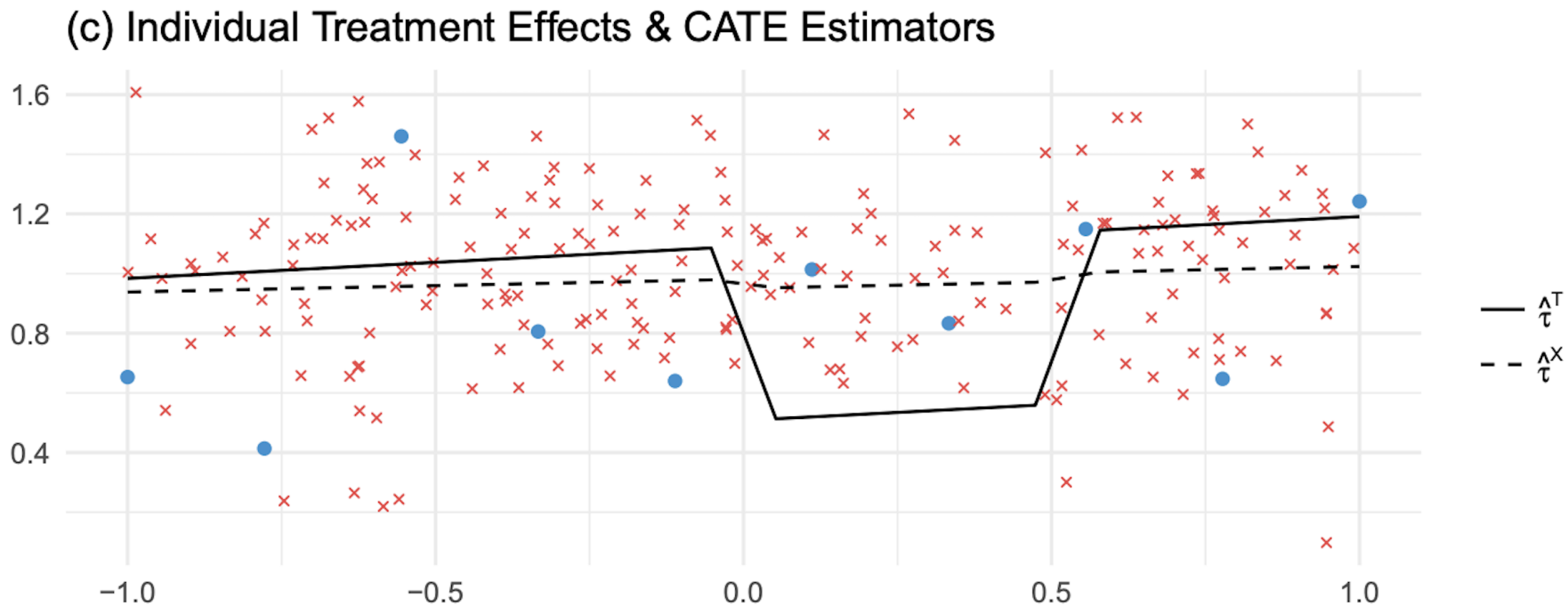
Model fit for control (red) and treatment (blue) groups.



Taken from Sören, R, et.al. (2019) *"Meta-learners for Estimating Heterogeneous Treatment Effects using Machine Learning"*

T-Learner Estimation:

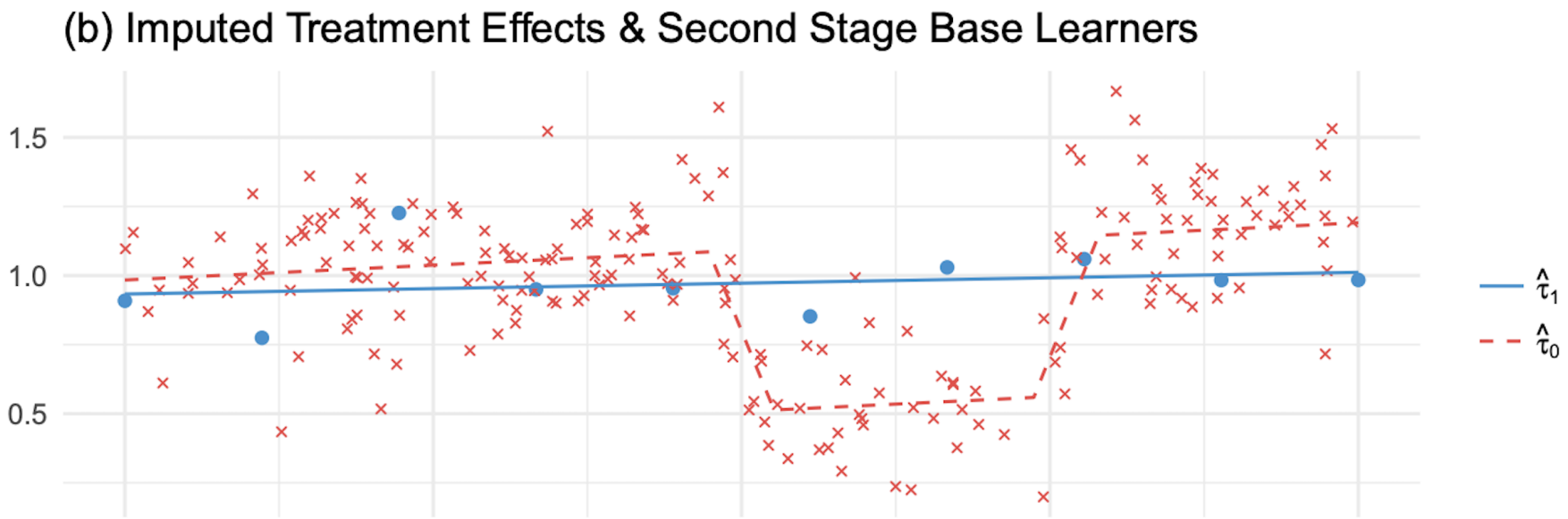
- The solid line represents the difference between the model fit for the control group and the treatment groups.
- The estimation is not good as the treatment group is very small.



Taken from Sören, R, et.al. (2019) *"Meta-learners for Estimating Heterogeneous Treatment Effects using Machine Learning"*

Imputed Treatment Effects:

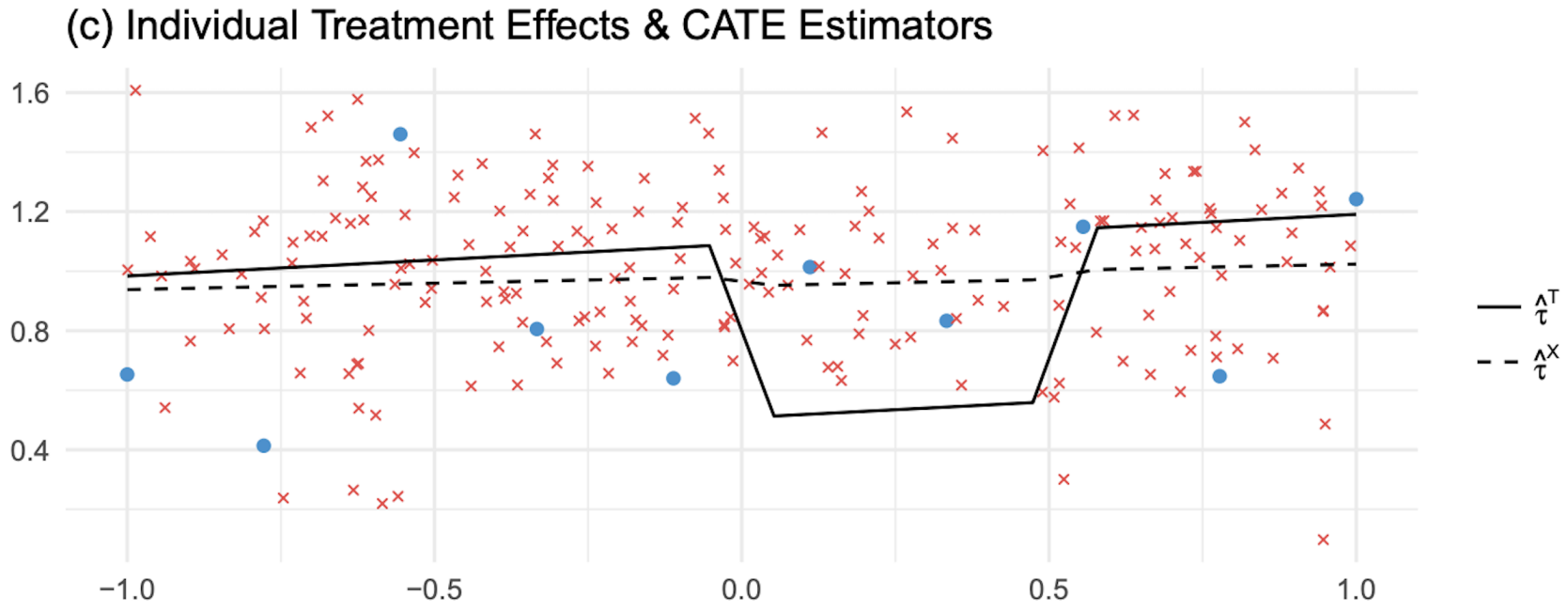
$$\tilde{D}^T = Y^T - \hat{\mu}_C(X^T)$$
$$\tilde{D}^C = \hat{\mu}_T(X^C) - Y^C$$



Taken from Sören, R, et.al. (2019) *"Meta-learners for Estimating Heterogeneous Treatment Effects using Machine Learning"*

X-Learner Estimation:

- The dashed line represents the X-Learner estimation.
- It combines the fit from the imputed effects by using and estimator of the *propensity score*, i.e. $g(x) = \hat{e}(x)$. In this example $\hat{e}(x)$ will be small as we have much more observations in the control group. Hence the estimated uplift will be close to $\hat{\tau}^T$.



Taken from Sören, R, et.al. (2019) *"Meta-learners for Estimating Heterogeneous Treatment Effects using Machine Learning"*

Some Python Implementations

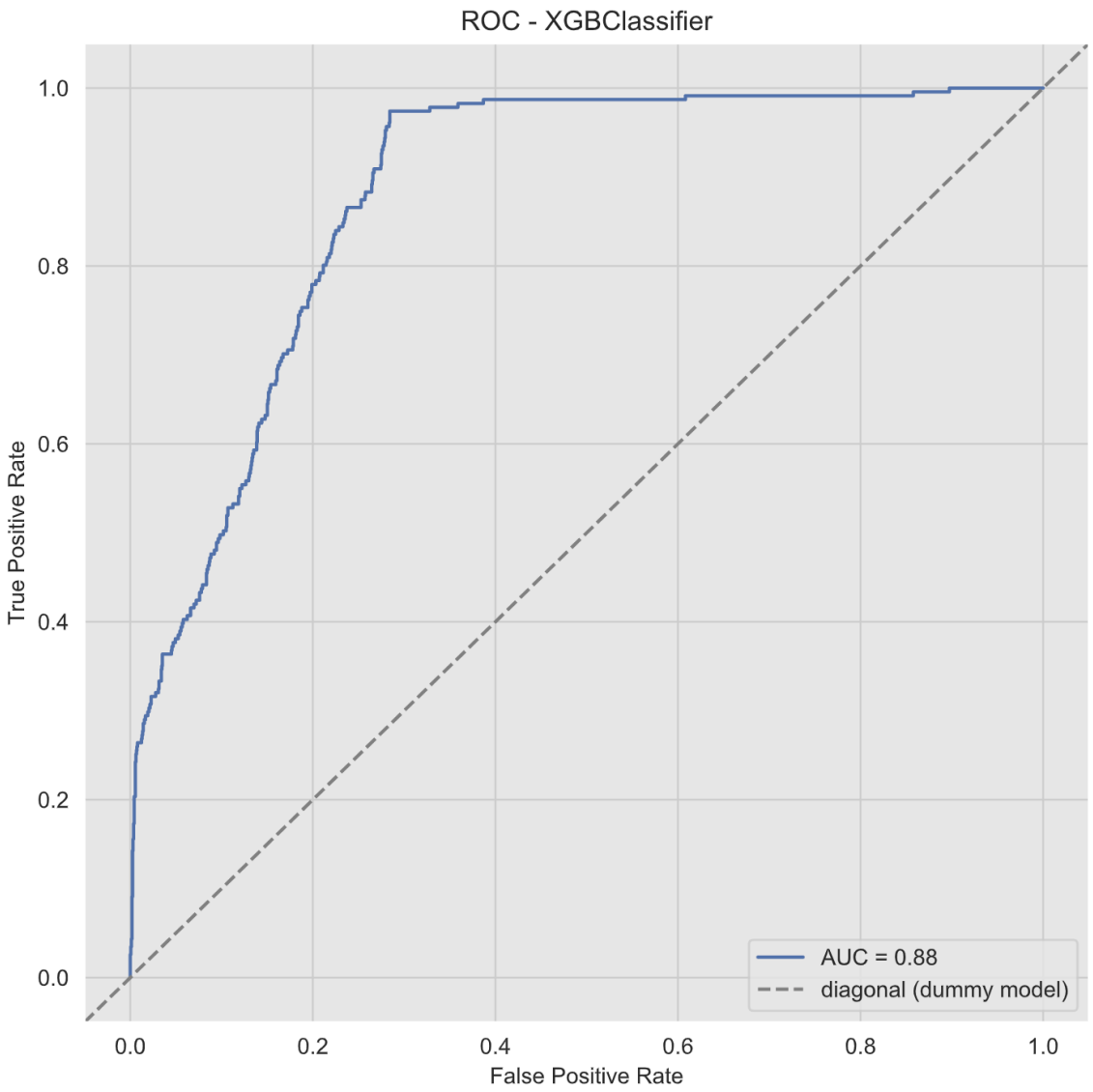
- `causalml`

- `EconML`

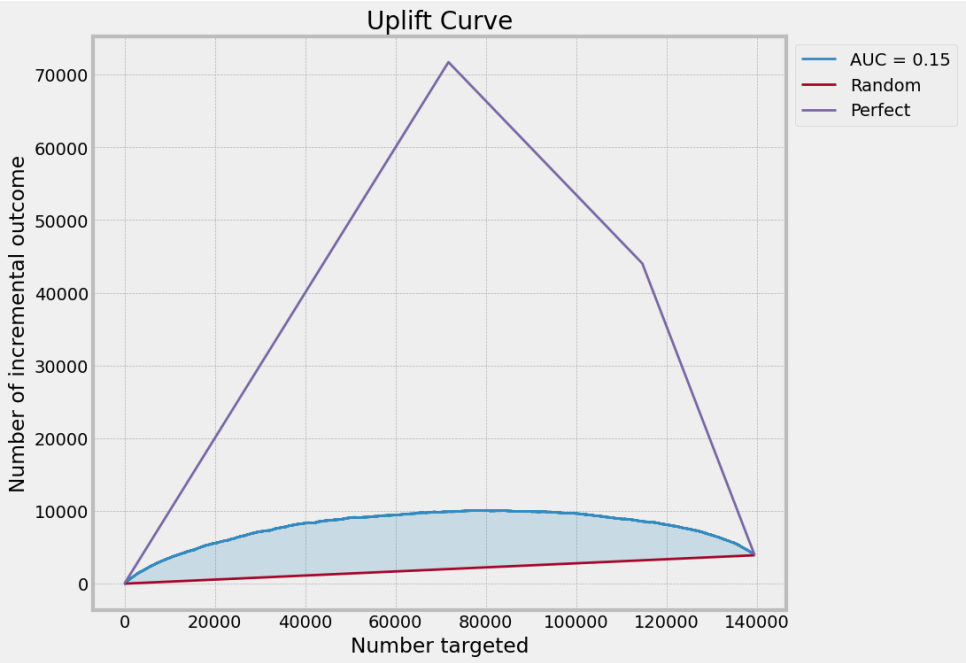
- `scikit-uplift`

Python Code: Example

```
from causalml.inference.meta import BaseTClassifier
from sklearn.ensemble import HistGradientBoostingClassifier

# define ml model
learner = HistGradientBoostingClassifier()

# set meta-model
t_learner = BaseTClassifier(learner=learner)

# compute ate
t_ate_lwr, t_ate, t_ate_upr = t_learner.estimate_ate(X=x, treatment=w, y=y)

# predict treatment effects
t_learnet.predict(X=x)

# access ml models
t_learner.models_c[1]
t_learner.models_t[1]
```



Uplift Model Evaluation?



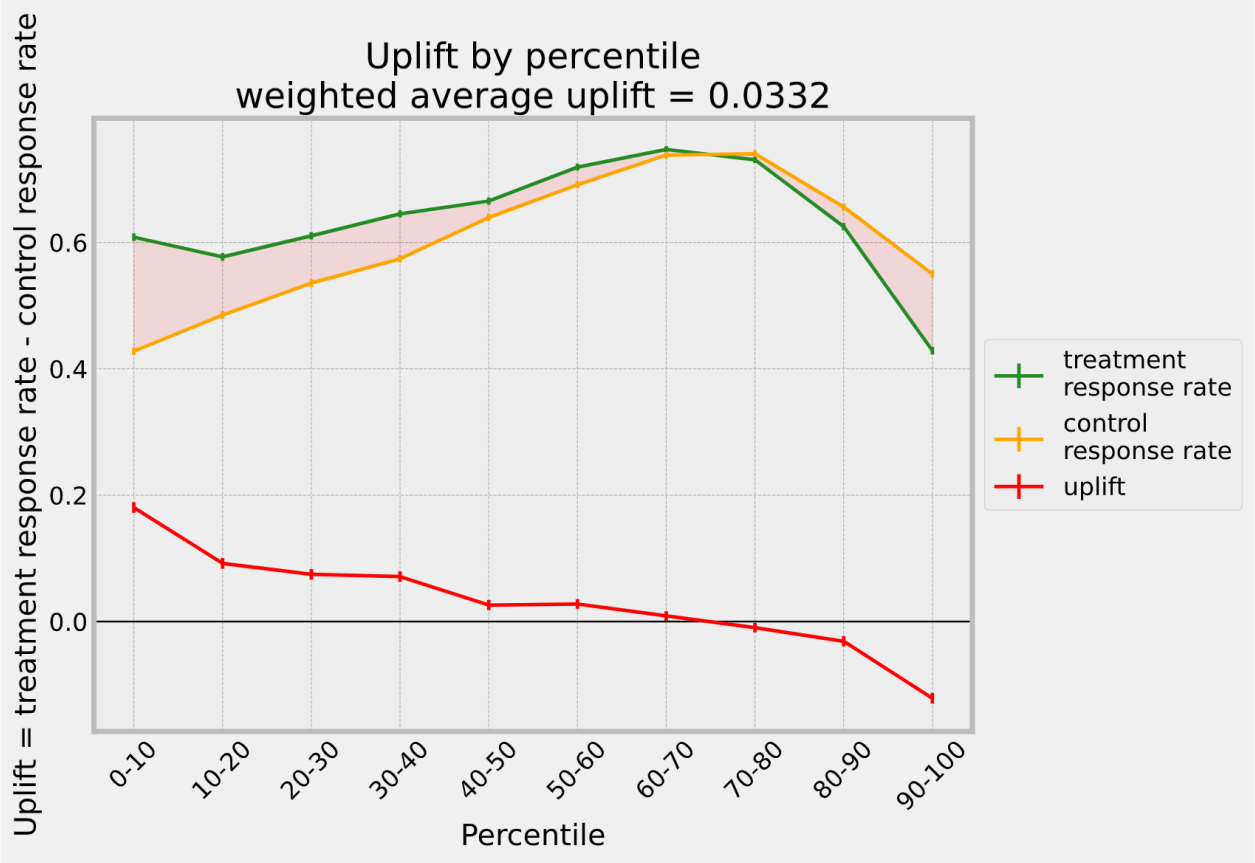
Uplift Evaluation: Uplift by Percentile

1. Sort uplift predictions by decreasing order.
2. Compute percentiles.
3. Predict uplift for both treated and control observations per percentile.
4. The difference between those averages is taken for each percentile.

percentile	n_treatment	n_control	response_rate_treatment	response_rate_control	uplift
0-10	6870	7082	0.608297	0.427845	0.180452
10-20	6974	6978	0.577144	0.485096	0.092048
20-30	7082	6870	0.610421	0.535662	0.074758
30-40	7200	6751	0.645278	0.574137	0.071141
40-50	7086	6865	0.665538	0.639476	0.026062
50-60	6959	6992	0.719213	0.691362	0.027851
60-70	6955	6996	0.747088	0.738136	0.008952
70-80	7002	6949	0.730934	0.740538	-0.009604
80-90	6927	7024	0.624946	0.656179	-0.031233
90-100	6667	7284	0.428679	0.550110	-0.121431
total	69722	69791	0.636743	0.603531	0.033213

Uplift Evaluation: Uplift by Percentile

A well performing model would have large values in the first percentiles and decreasing values for larger ones



Plot function `plot_uplift_by_percentile` from `scikit-uplift`.

Uplift Evaluation: Cumulative Gain Chart

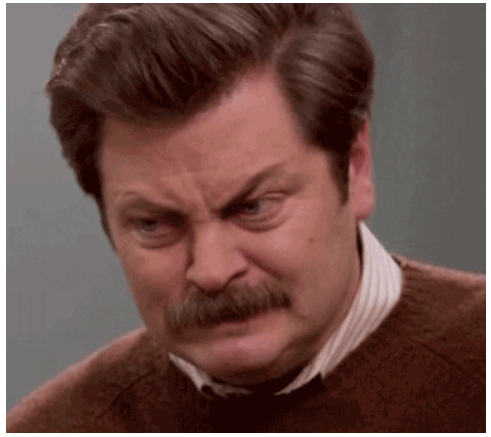
> Predict uplift for both treated and control observations and compute the average prediction per decile (bins) in both groups. Then, the difference between those averages is taken for each decile.

$$\left(\frac{Y^T}{N^T} - \frac{Y^C}{N^C}\right)(N^T + N^C)$$

- Y^T / Y^C: sum of the treated / control individual outcomes in the bin.
- N^T / N^C: number of treated / control observations in the bin.

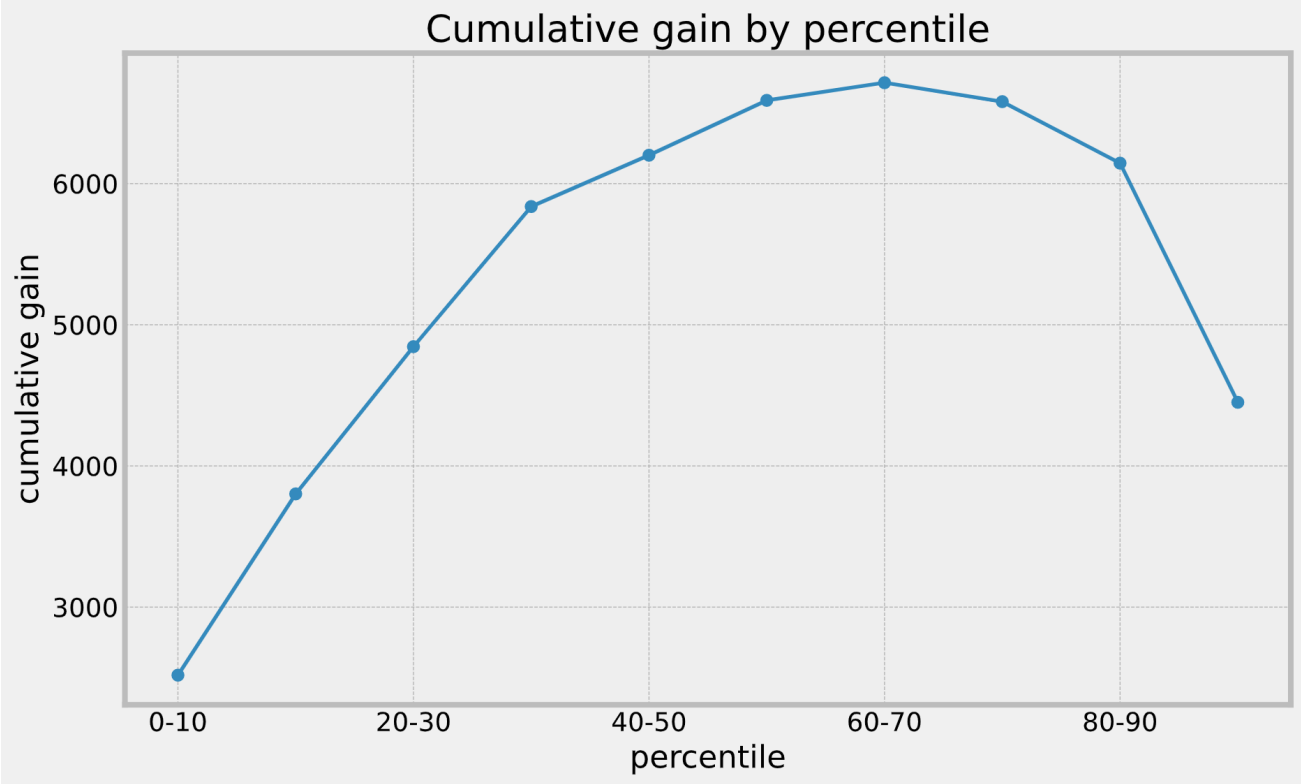
Taken from Gutierrez, P., & Gérardy, J. Y. (2017). *"Causal Inference and Uplift Modelling: A Review of the Literature"*

How to compute it ? 🫠



```
df = uplift_by_percentile_df

# compute cumulative response rates
df["responses_treatment"] = df["n_treatment"] * x["response_rate_treatment"]
df["responses_control"] = df["n_control"] * x["response_rate_control"]
df["n_treatment_cumsum"] = df["n_treatment"].cumsum()
df["n_control_cumsum"] = df["n_control"].cumsum()
df["responses_treatment_cumsum"] = df["responses_treatment"].cumsum()
df["responses_control_cumsum"] = df["responses_control"].cumsum()
df["response_rate_treatment_cumsum"] = df["responses_treatment_cumsum"] / x["n_treatment_cumsum"]
df["response_rate_control_cumsum"] = df["responses_control_cumsum"] / x["n_control_cumsum"]

# compute uplifts (at cumulative level)
df["uplift_cumsum"] = df["response_rate_treatment_cumsum"] - df["response_rate_control_cumsum"]
# compute cumulative gains
df["cum_gain"] = df["uplift_cumsum"] * (df["n_treatment_cumsum"] + df["n_control_cumsum"])
```

Uplift Evaluation: Cumulative Gain Chart



- We can assess whether the treatment has a global positive or negative effect and if one can expect a better gain by targeting part of the population.
- We can thus choose the decile that maximizes the gain as the limit of the population to be targeted.

Uplift Metrics: Uplift Curve

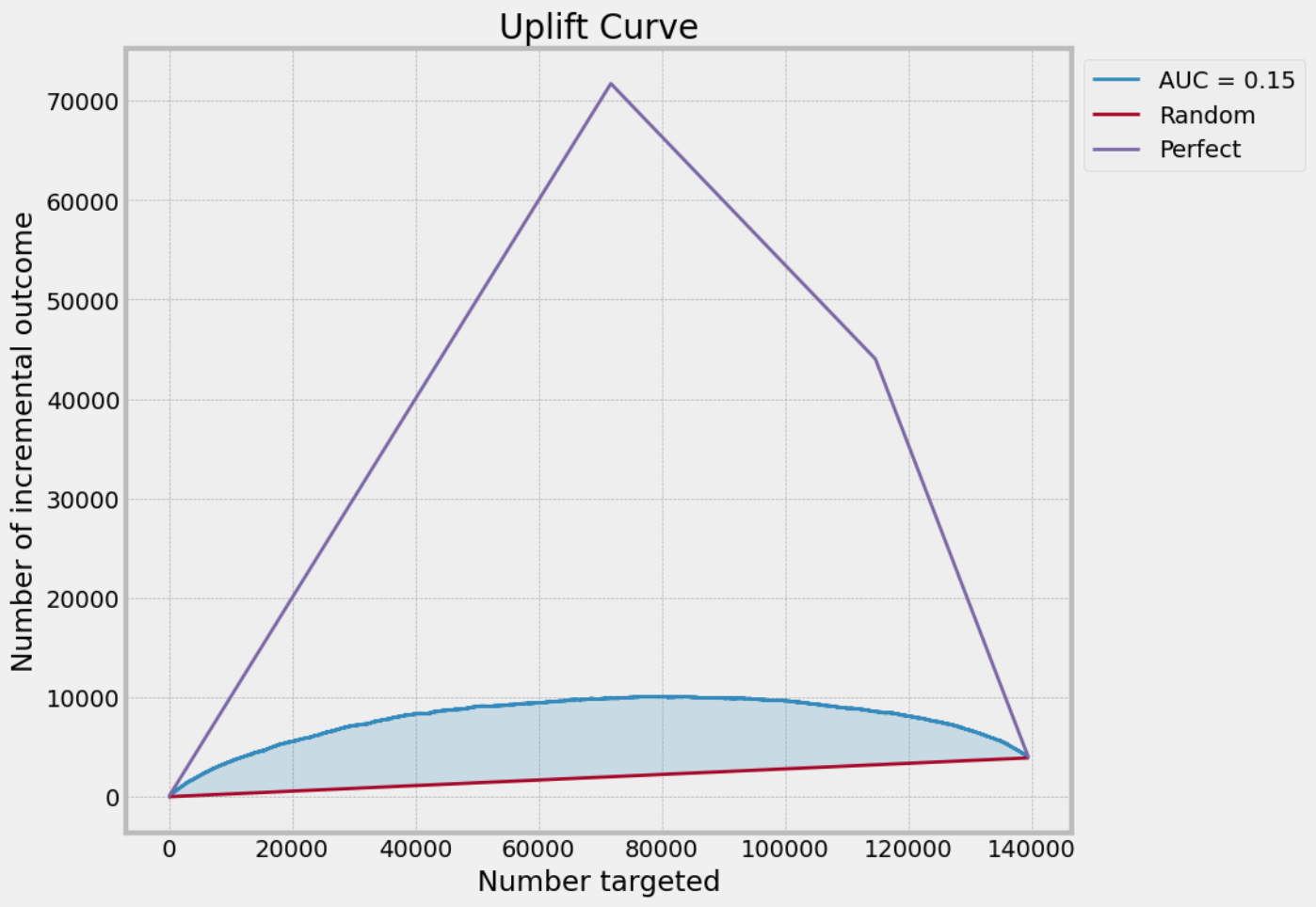
We can generalize the cumulative gain chart for each observation of the test set:

$$f(t) = \left(\frac{Y_t^T}{N_t^T} - \frac{Y_t^C}{N_t^C} \right) (N_t^T + N_t^C)$$

where the t subscript indicates that the quantity is calculated for the first t observations, sorted by inferred uplift value.

Taken from Gutierrez, P., & Gérardy, J. Y. (2017). *"Causal Inference and Uplift Modelling: A Review of the Literature"*

Uplift Metrics: Uplift Curve & AUC

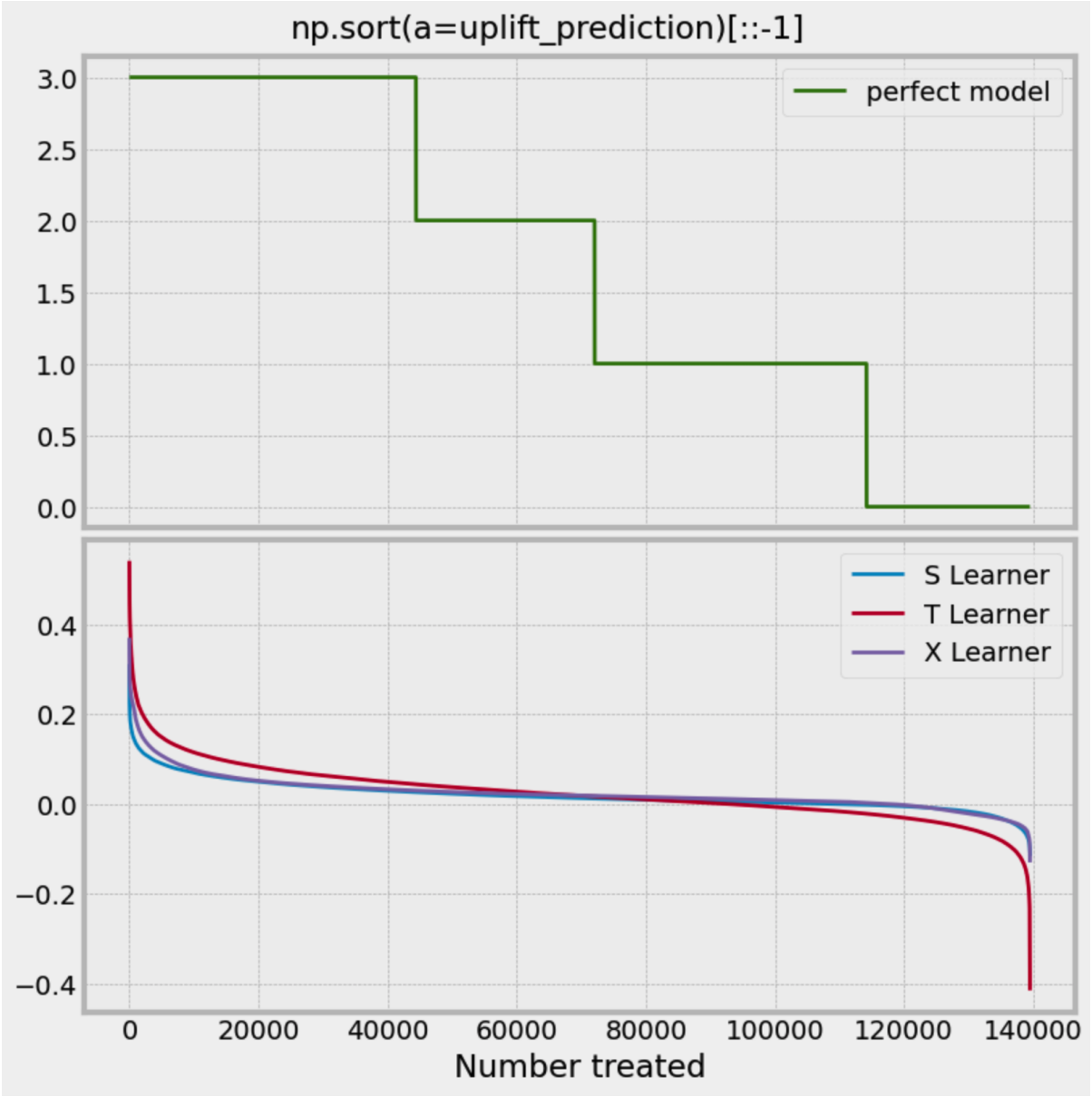


Plot function `plot_uplift_curve` from `scikit-uplift`

Best uplift model?

> A **perfect model** assigns higher scores to all treated individuals with positive outcomes than any individuals with negative outcomes.

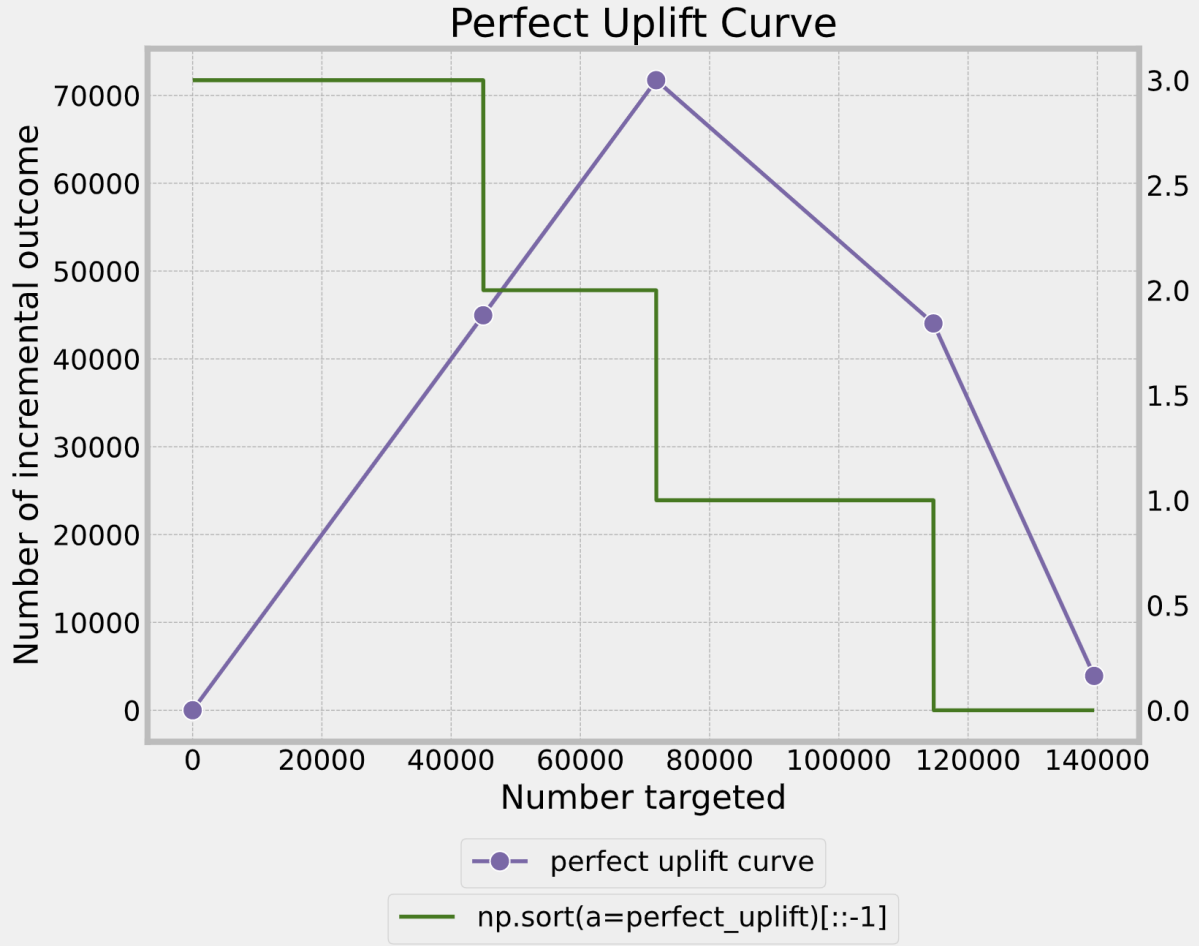
```
# Control Responders
cr_num = np.sum((y_true == 1) & (treatment == 0))
# Treated Non-Responders
tn_num = np.sum((y_true == 0) & (treatment == 1))

summand = y_true if cr_num > tn_num else treatment

perfect_uplift = 2 * (y_true == treatment) + summand
```



Taken from Diemert, Eustache, et.al. (2020) *"A Large Scale Benchmark for Uplift Modeling"*

Perfect Uplift Curve

```
from sklift.metrics import uplift_curve

a, b = uplift_curve(y_true=y_true, uplift=perfect_uplift, treatment=treatment)
```

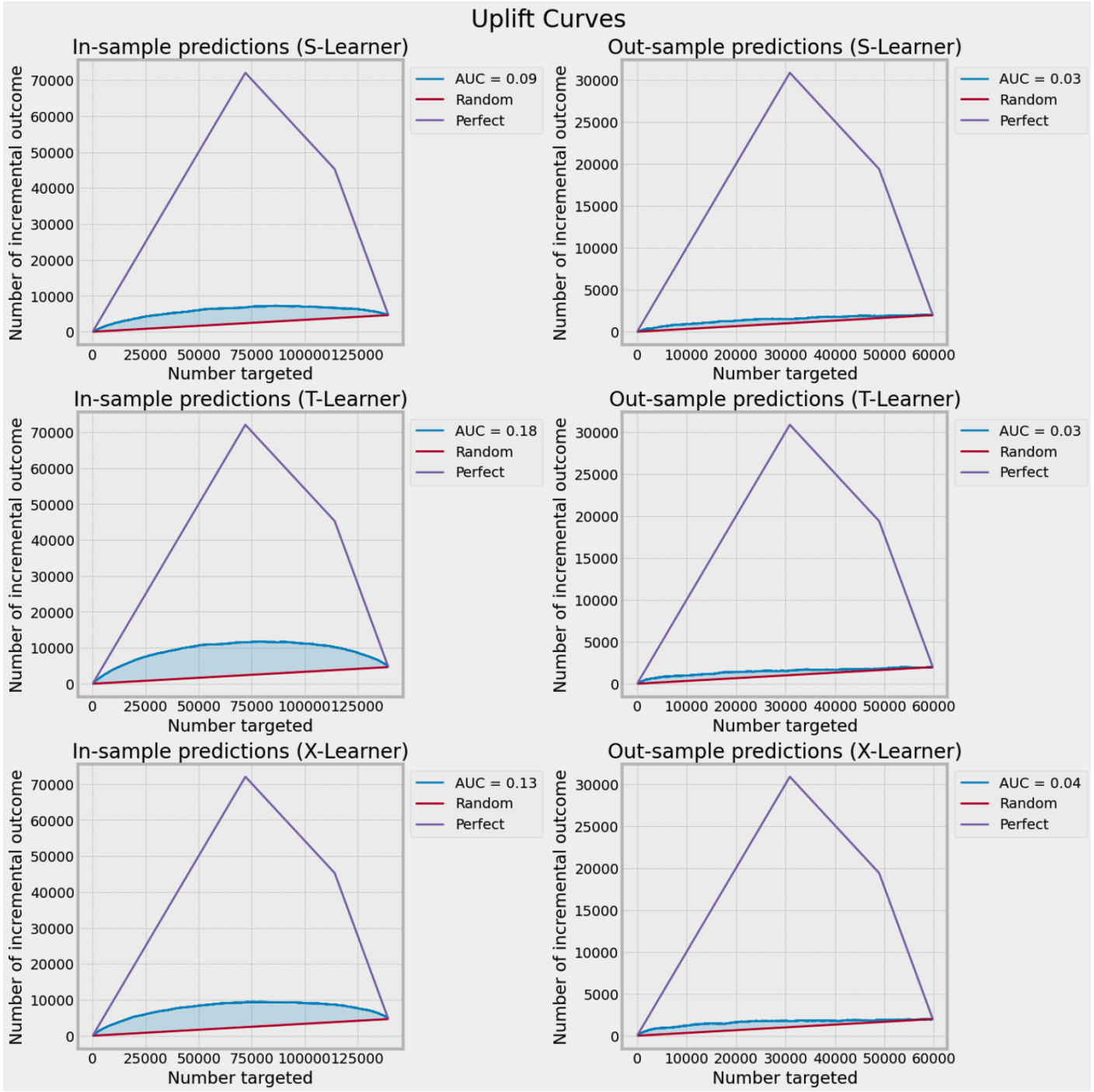


Random Uplift Curves

```
# For example:

np.random.uniform(
    low=-1,
    high=1,
    size=(n, n_samples),
)
```

Model Comparison

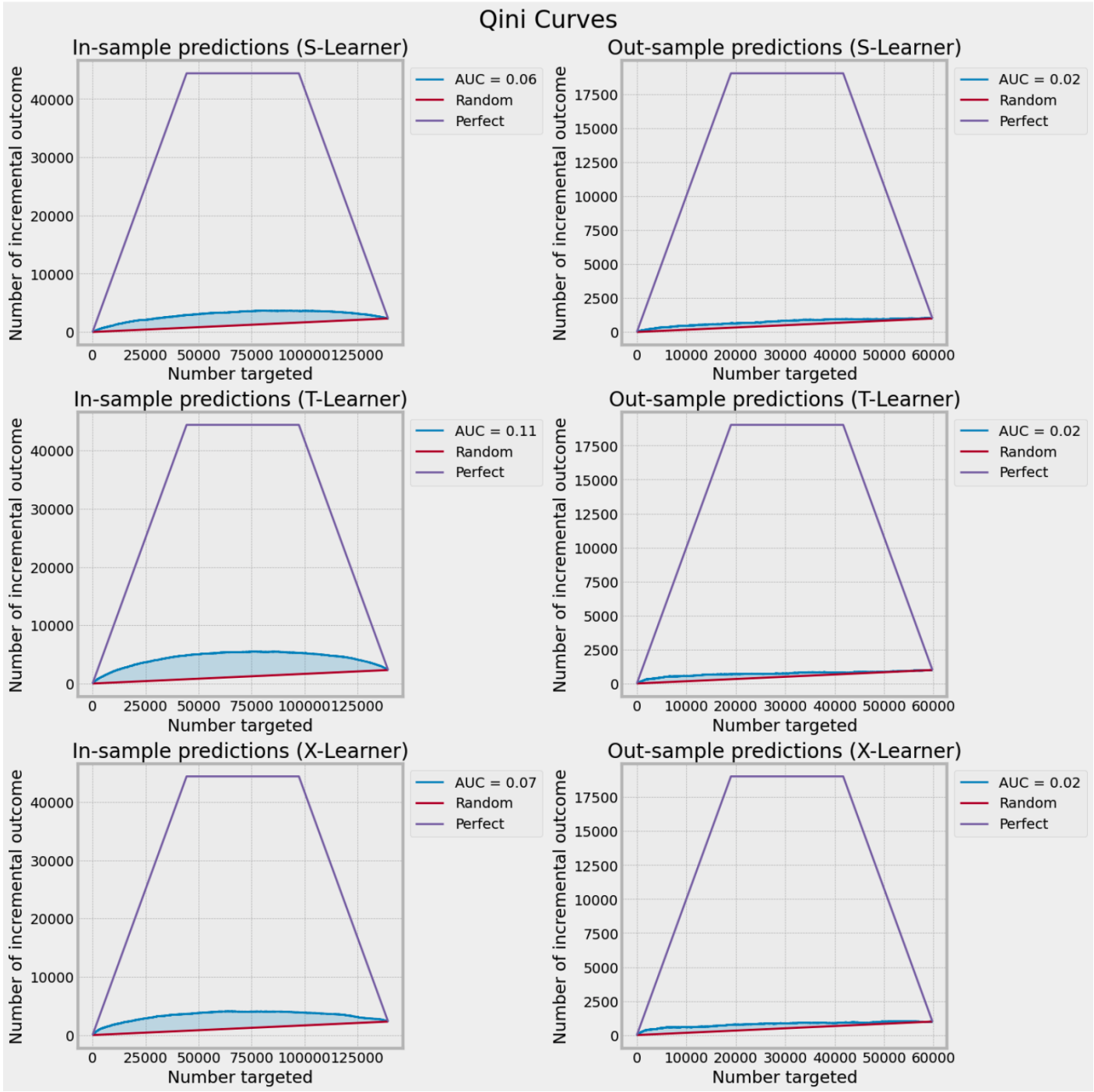
Compute AUC on a test set.



Other metrics: Qini Curve

$$g(t) = Y_t^T - Y_t^C \left(\frac{N_t^T}{N_t^C} \right)$$

> Corrects uplifts of selected individuals with respect to the number of individuals in treatment/control using the N_t^T / N_t^C factor.



Demo 💻

Notebook Link

See https://juanitorduz.github.io/uplift/

References 📚

- Diemert, Eustache, et.al. (2020) *"A Large Scale Benchmark for Uplift Modeling"*
- Gutierrez, P., & Gérardy, J. Y. (2017). *"Causal Inference and Uplift Modelling: A Review of the Literature"*
- Karlsson, H. (2019) *"Uplift Modeling: Identifying Optimal Treatment Group Allocation and Whom to Contact to Maximize Return on Investment"*
- Sören, R, et.al. (2019) *"Meta-learners for Estimating Heterogeneous Treatment Effects using Machine Learning"*

Thank you!

More Info: juanitorduz.github.io