

Introduction to Uplift Modeling

Dr. Juan Orduz

PyConDE & PyData Berlin 2022



Motivation

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



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 understanding the *causal effect* $Y_i^1 - Y_i^0$ and the *conditional average treatment effect* $CATE = E[Y_i^1 | X_i] - E[Y_i^0 | X_i]$, where X_i is a feature vector of the i -th person.
- **However, we can not observe them!** 😞

Uplift

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:

$$\mathbf{uplift} = \widehat{CATE} = E[Y_i | X_i, W_i = 1] - E[Y_i | X_i, W_i = 0]$$

Estimating Uplift

- Meta algorithms
- Direct measurements (trees)

Some python implementations

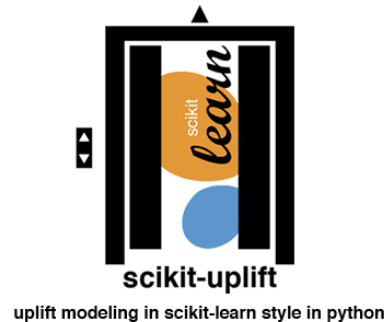
- `causalml`



- `EconML`



- `scikit-uplift`



S-Learner

Step 1: Training

$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} & w_1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \cdots & x_{nk} & w_n \end{pmatrix}}_{X \oplus W} \xrightarrow{\mu} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Step 2: Uplift Prediction

$$\widehat{\text{uplift}} = \hat{\mu} \begin{pmatrix} x_{11} & \cdots & x_{1k} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{m1} & \cdots & x_{mk} & 1 \end{pmatrix} - \hat{\mu} \begin{pmatrix} x_{11} & \cdots & x_{1k} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{m1} & \cdots & x_{mk} & 0 \end{pmatrix}$$

T-Learner

Step 1: Training

$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{n_C 1} & \cdots & x_{n_C k} \end{pmatrix}}_{X|_{\text{control}}} \xrightarrow{\mu_C} \begin{pmatrix} y_1 \\ \vdots \\ y_{n_C} \end{pmatrix}$$
$$\underbrace{\begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{n_T 1} & \cdots & x_{n_T k} \end{pmatrix}}_{X|_{\text{treatment}}} \xrightarrow{\mu_T} \begin{pmatrix} y_1 \\ \vdots \\ y_{n_T} \end{pmatrix}$$

T-Learner

Step 2: Uplift Prediction

$$\widehat{\text{uplift}} = \hat{\mu}_T \begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{mk} \end{pmatrix} - \hat{\mu}_C \begin{pmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{mk} \end{pmatrix}$$

X-Learner

Step 1: Training: Same as T-Learner

Step 2: Compute imputed treatment effects

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

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

X-Learner

Step 3: Train with different targets

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

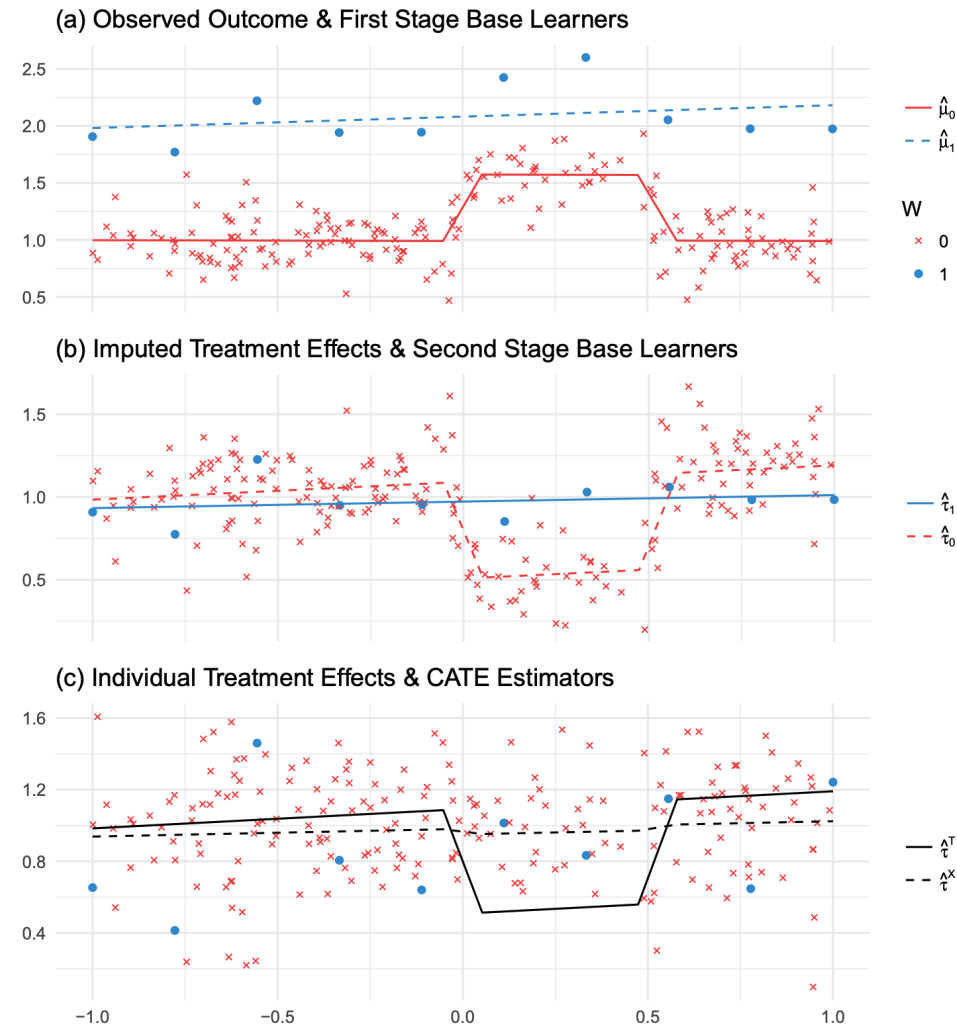
X-Learner

Step 4: Uplift Prediction

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

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

Intuition behind the X-Learner



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_learner.predict(X=x)

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

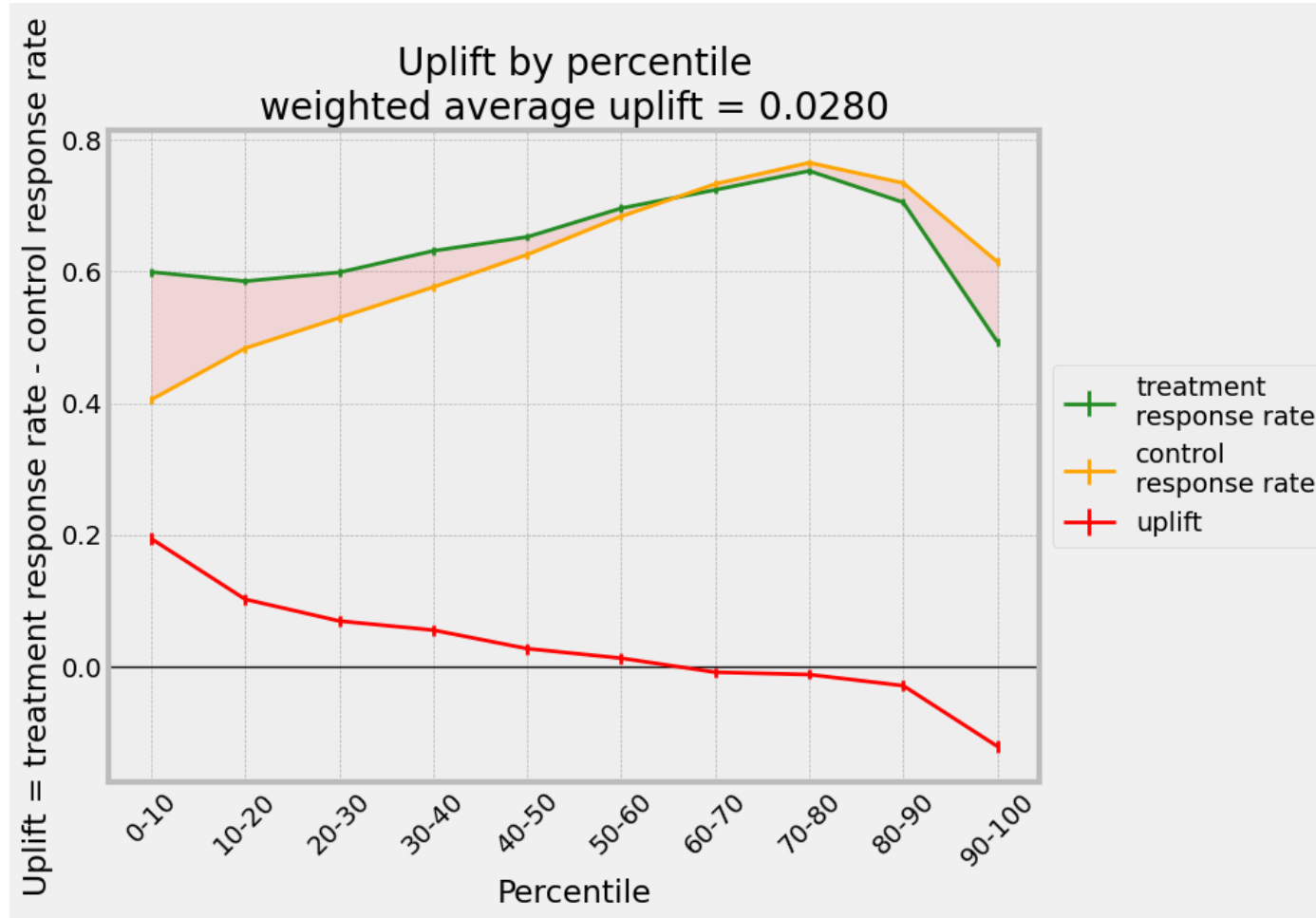

Uplift Metrics: Cumulative gain chart

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.

Uplift Metrics: Cumulative gain chart



Plot function from `scikit-uplift`. The weighted average is computed as `response_rate_treatment - response_rate_control`.

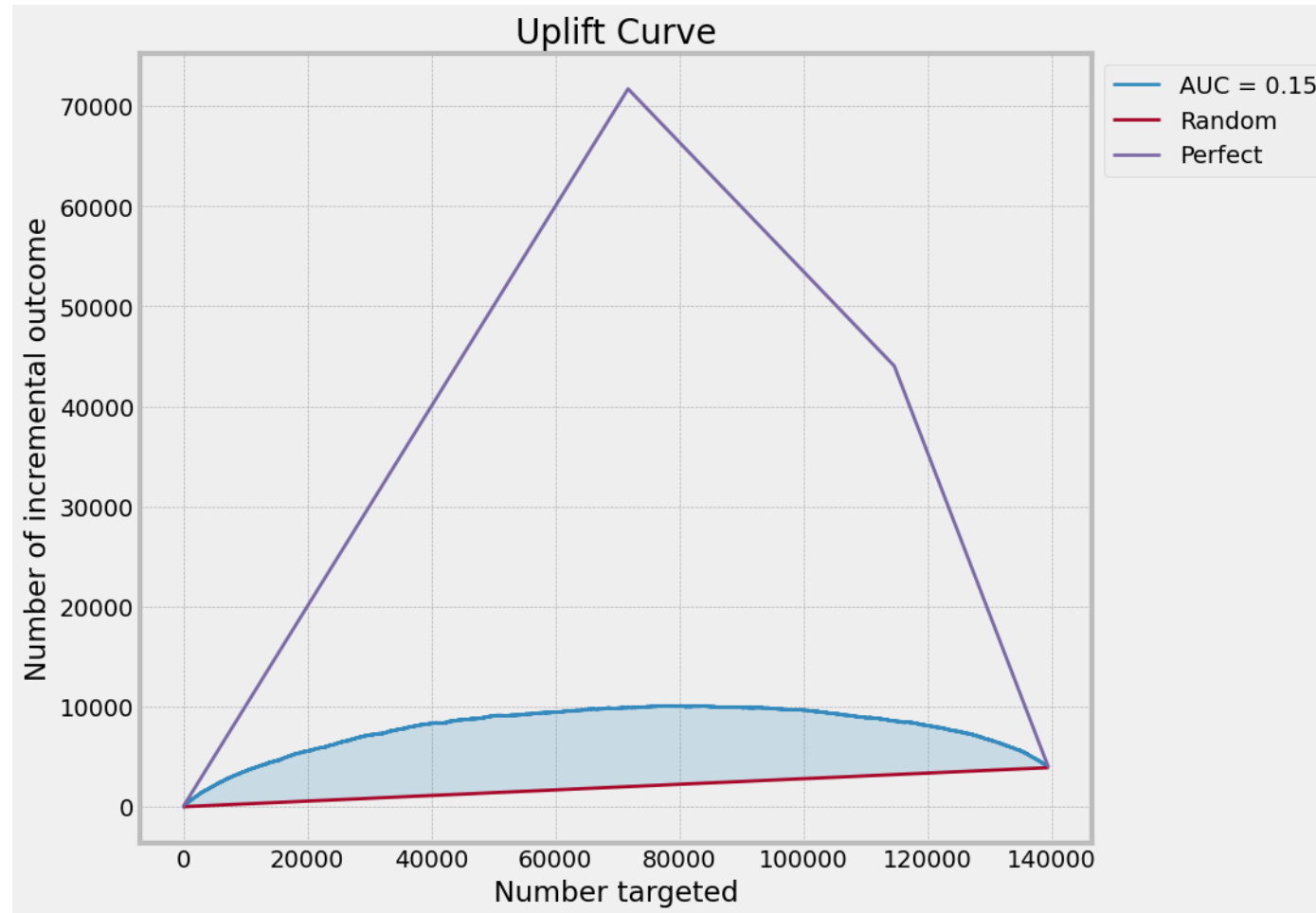
Uplift Metrics: Uplift Curve

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.

Uplift Metrics: Uplift Curve



Perfect Uplift Curve

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

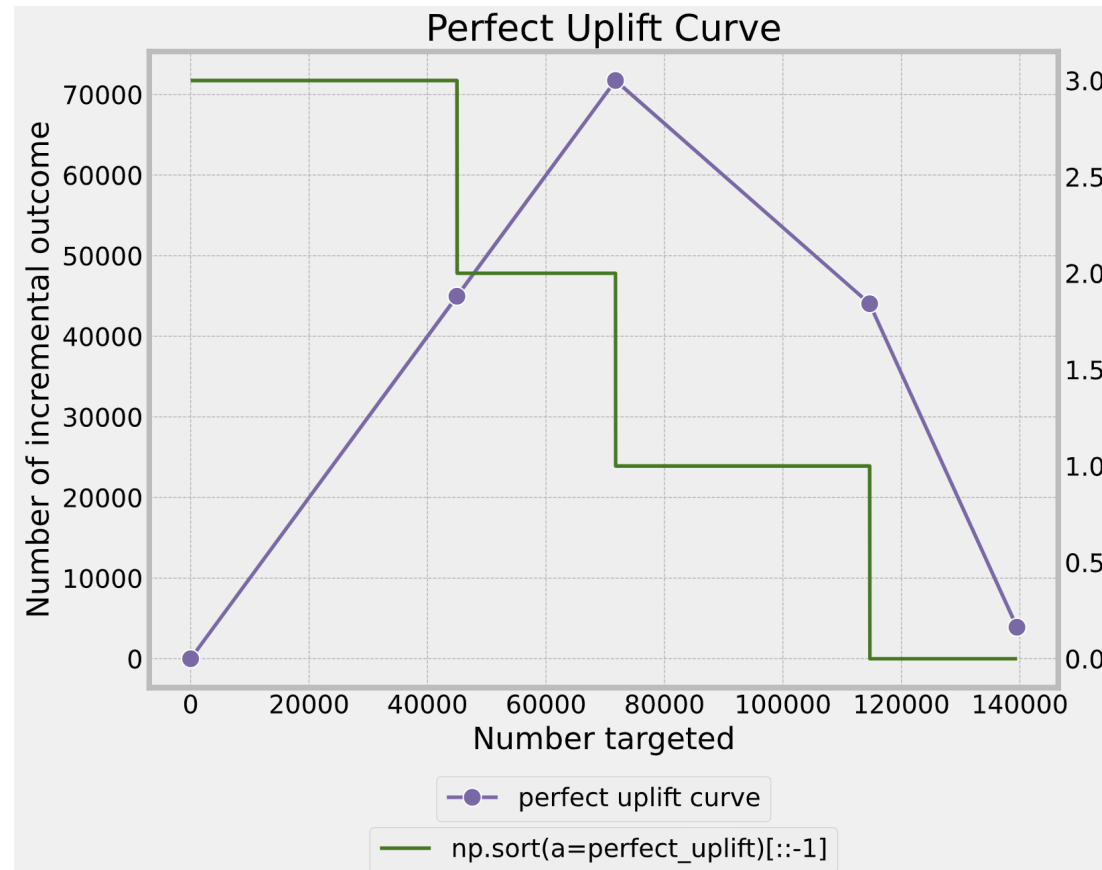
```
cr_num = np.sum((y_true == 1) & (treatment == 0)) # Control Responders
tn_num = np.sum((y_true == 0) & (treatment == 1)) # Treated Non-Responders

summand = y_true if cr_num > tn_num else treatment
perfect_uplift = 2 * (y_true == treatment) + summand
```

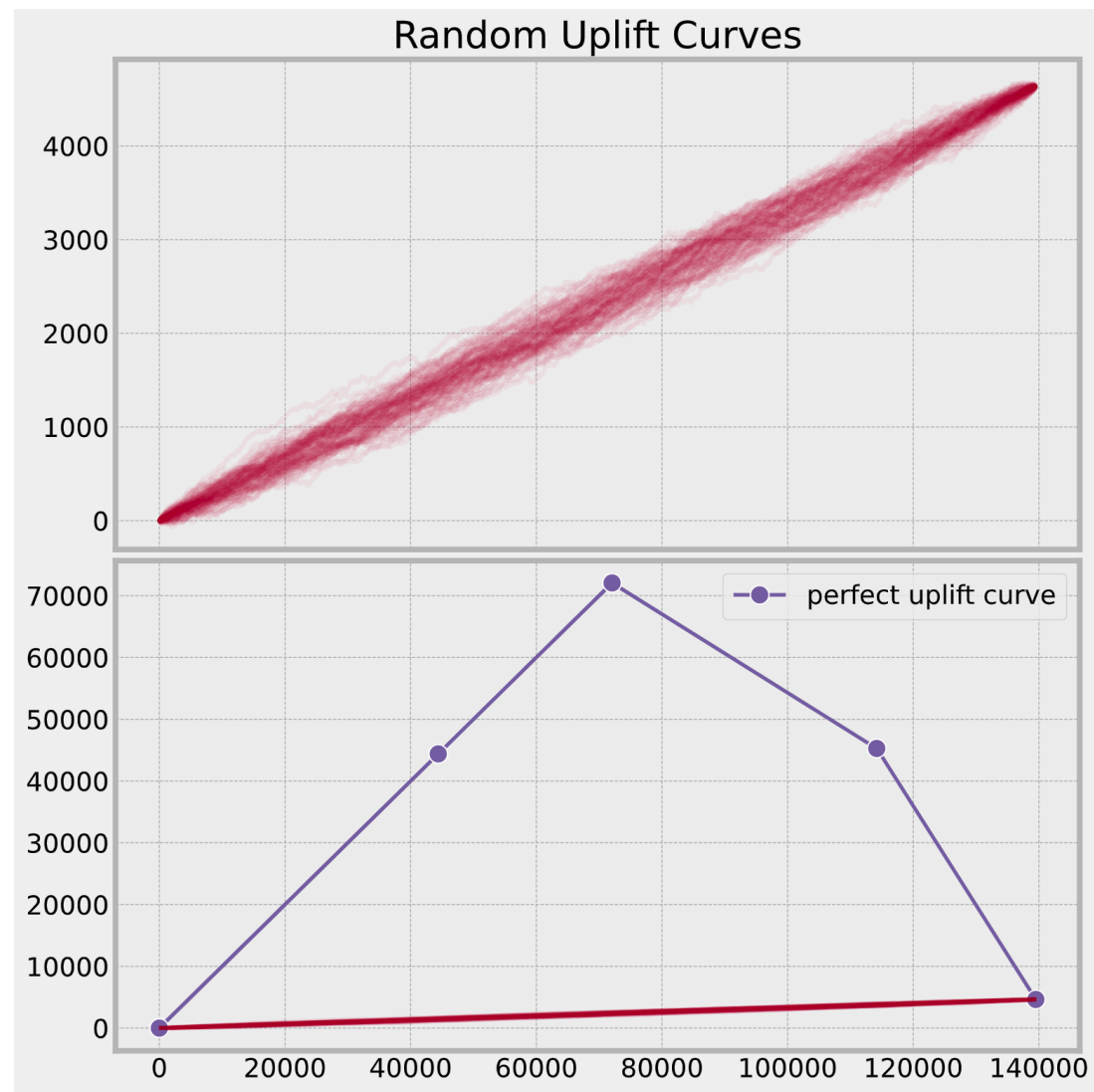
Perfect Uplift Curve

```
from sklift.metrics import uplift_curve
```

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

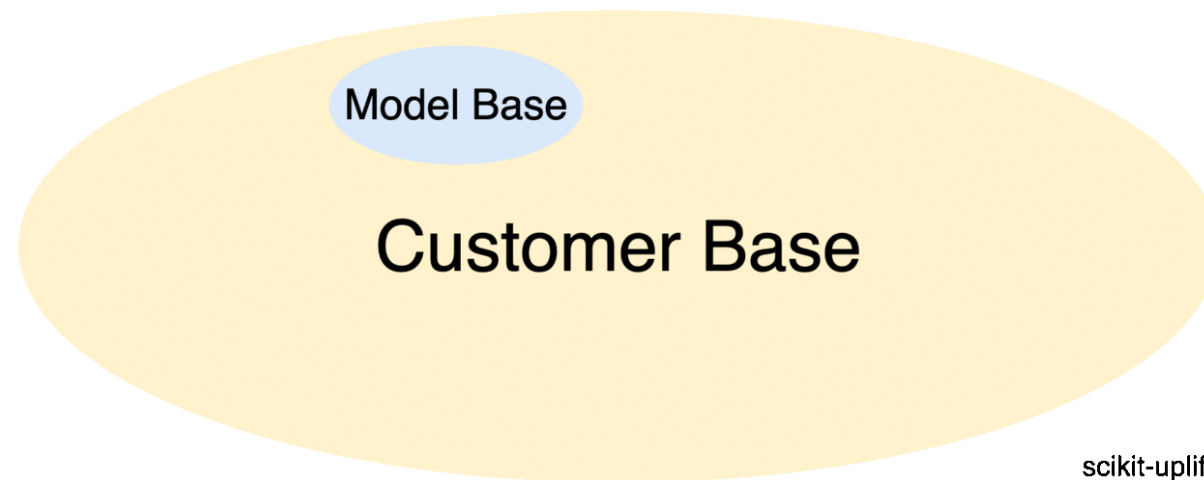


Random Uplift Curves



Data Collection

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



scikit-uptift

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

