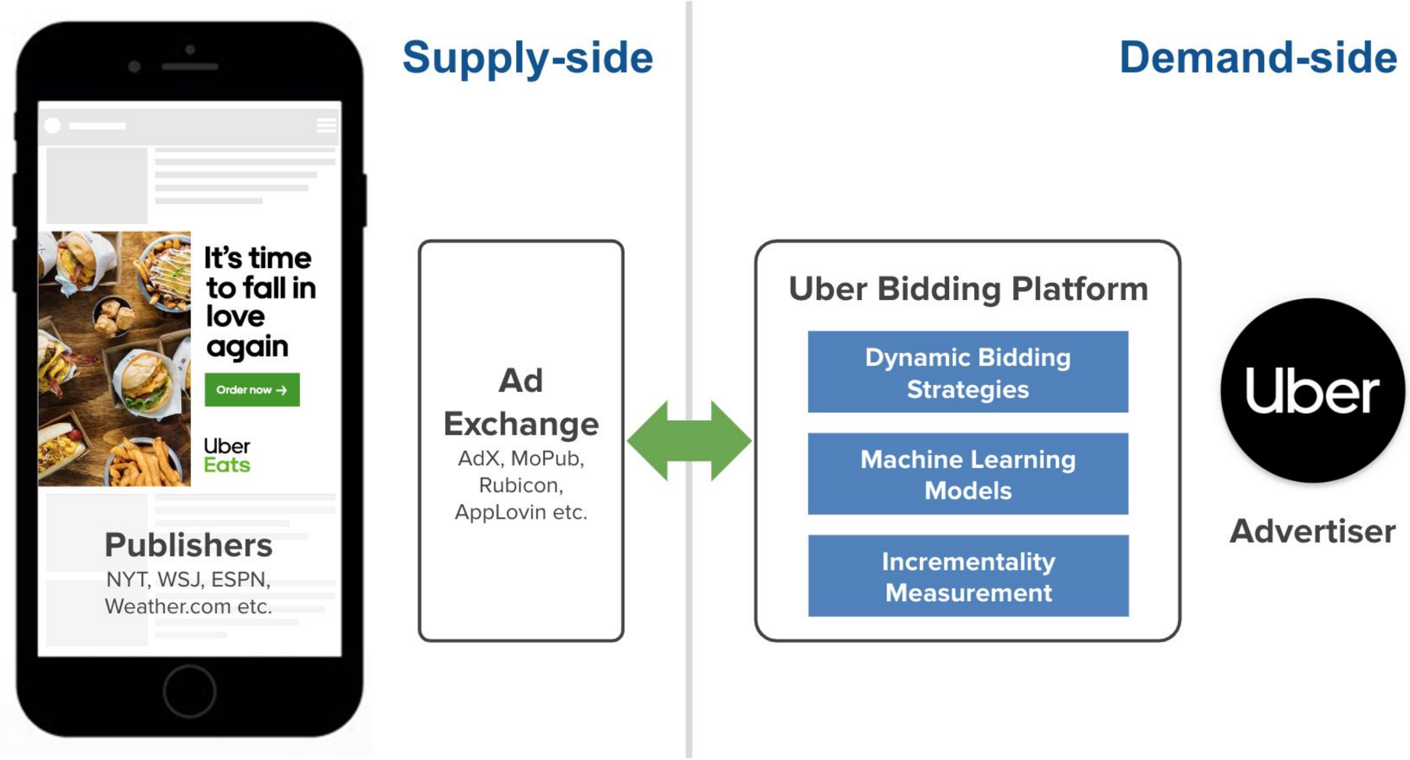
Case Study #2: Targeting Optimization: Bidder at Uber

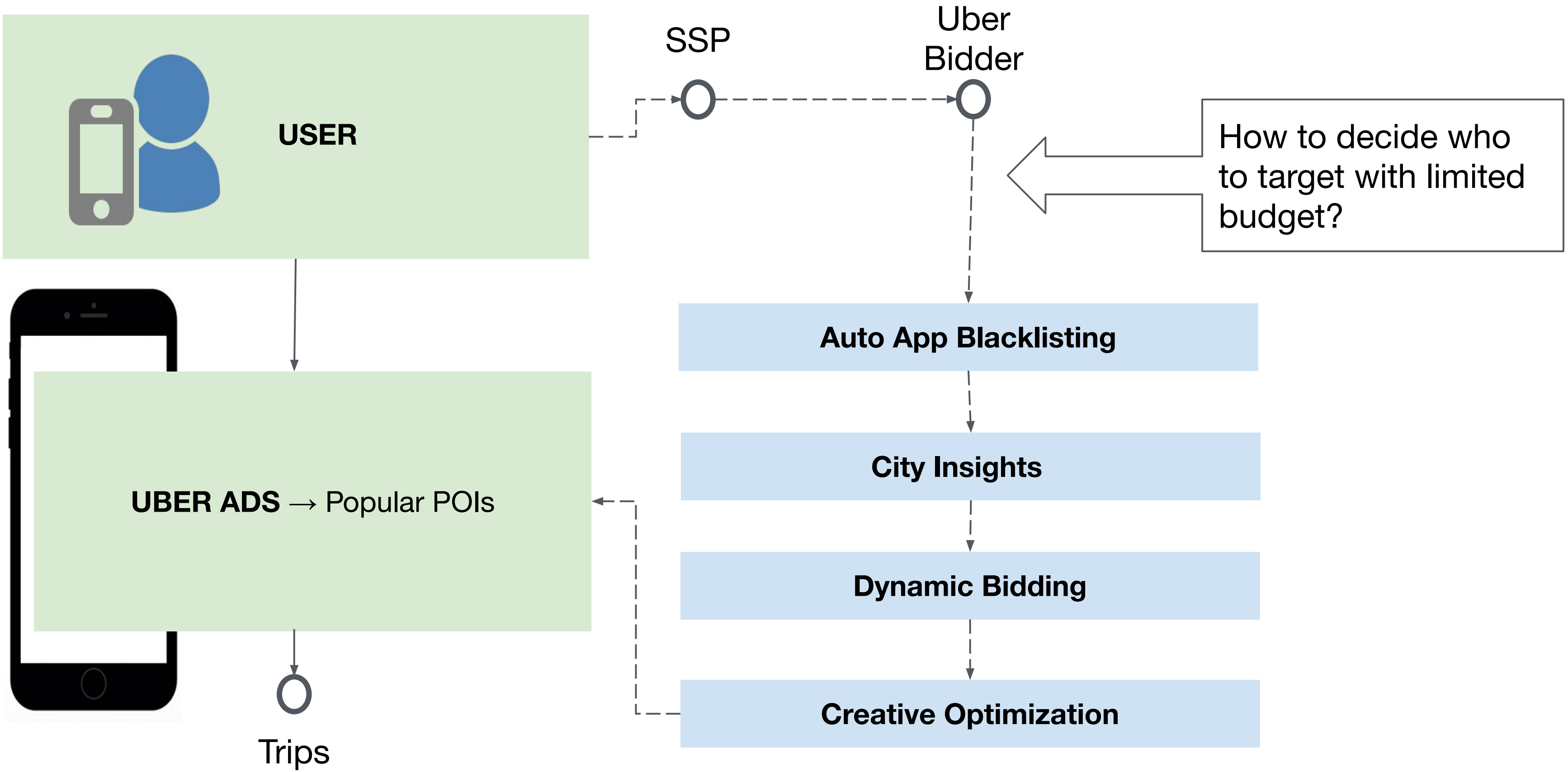
Uber

CausalML

Background | Uber as an Advertiser



Background | Uber Bidder for Media Buying



01 Background

02 Problem Definition

03 Method

04 Evaluation

05 Deployment

Problem Definition

Based on the treatment and the users' response, we split them into four groups.

If treated (**treatment**)	N	Defier	Never-taker
	Y	Always-taker	Persuadable
Convert?		Y	N
		If not treated (**control)**	

As advertisers, in theory, we are interested in "Persuadable User", who only takes more trips/orders when given treatment. Uplift model is designed to identify these users by causal inference and machine learning.

01 Background
02 Problem Definition
03 Method
04 Evaluation
05 Deployment

Method | Overview

Experiment Setup

- Understand how to use the modeling results
- Define business metrics
- Define user
- Define treatment

Data Collection

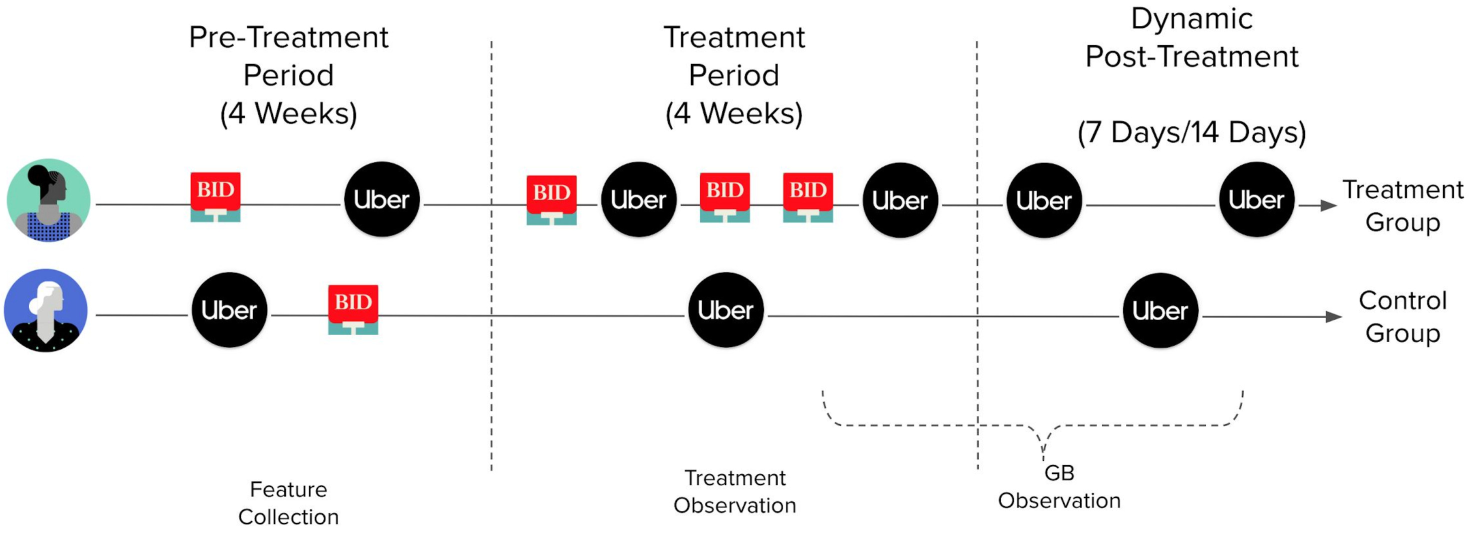
- Randomized Experiment

Modeling

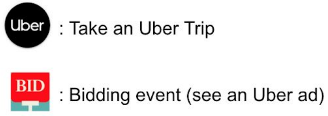
- Evaluate accuracy and robustness
- Insights into causality

Method | Experiment Setup

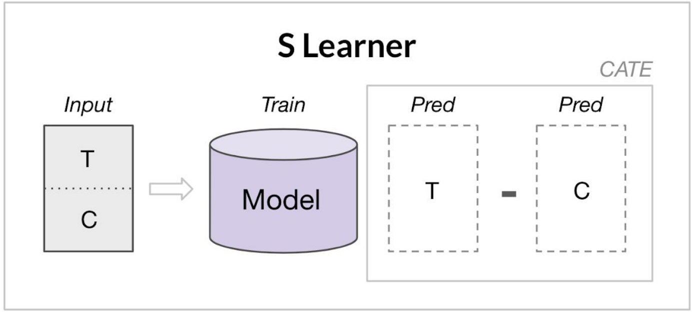
Experiment Setup → **Data Collection** → **Modeling**

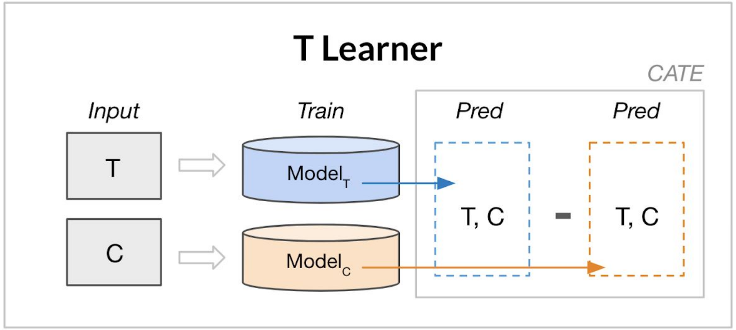


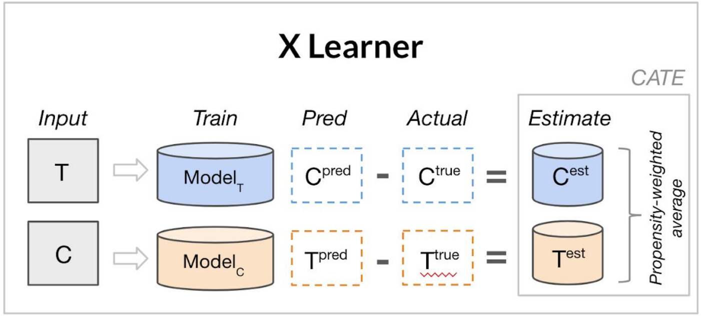
- Treatment Group: Riders who saw at least one ad within treatment period
- Control Group: Riders who never saw any ad until the end of post-treatment period

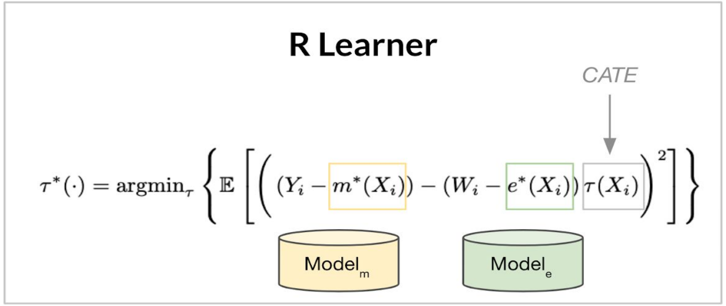


Method | Models Recap









Künzel, Sören R., et al. "Metalearners for estimating heterogeneous treatment effects using machine learning." Proceedings of the national academy of sciences 116.10 (2019): 4156-4165.

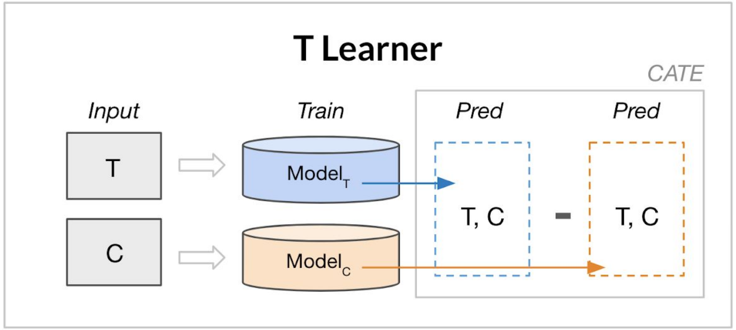
Nie, Xinkun, and Stefan Wager. "Quasi-oracle estimation of heterogeneous treatment effects." arXiv preprint arXiv:1712.04912 (2017).

Method | CATE for User Targeting


```
    def predict(self, X, treatment=None, y=None, p=None, return_components=False,
verbose=True):
        """Predict treatment effects.
        Args:
            X (np.matrix or np.array or pd.Dataframe): a feature matrix
            treatment (np.array or pd.Series, optional): a treatment vector
            y (np.array or pd.Series, optional): an outcome vector
            return_components (bool, optional): whether to return outcome for
treatment and control separately
            verbose (bool, optional): whether to output progress logs
        Returns:
            (numpy.ndarray): Predictions of treatment effects.
        """
```

CausalML provides easy-to-use interface to calculate Conditional Average Treatment Effect (CATE) for each meta learner.

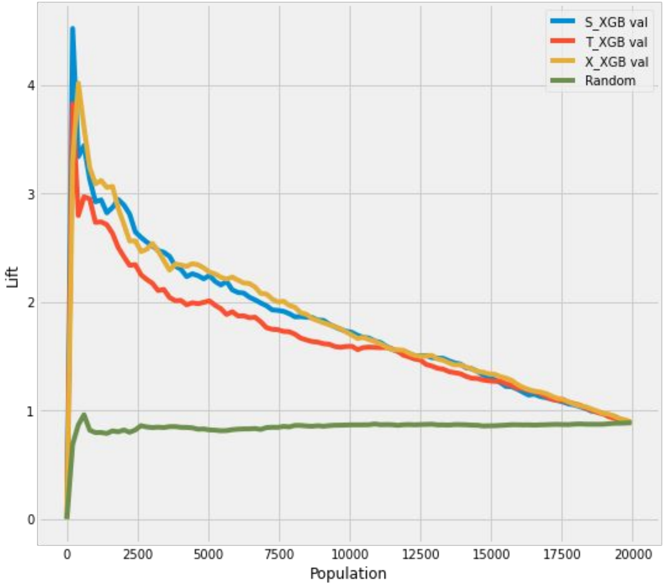
Evaluation | Base Learners




```
# Treatment
# base learner prediction
# models_t[1] holds the model instance
scored_val_tr[MU_PRED] = learner.models_t[1].predict(X_tr)

[Example Output]
T_XGB
INFO:NBOE:Validating T_XGB base model
INFO:NBOE:Validation result on hold out data
INFO:NBOE:       model    segment    rmse   smape   rsquare learner        type
0    mu_model    overall  5.0344 0.1952    0.0584    T_XGB  validation
1    mu_model  treatment  5.0586 0.1927    0.0748    T_XGB  validation
2    mu_model    control  5.0098 0.1976    0.0263    T_XGB  validation
```

Besides final CATE prediction, base learner prediction can also be accessed through learner instance.

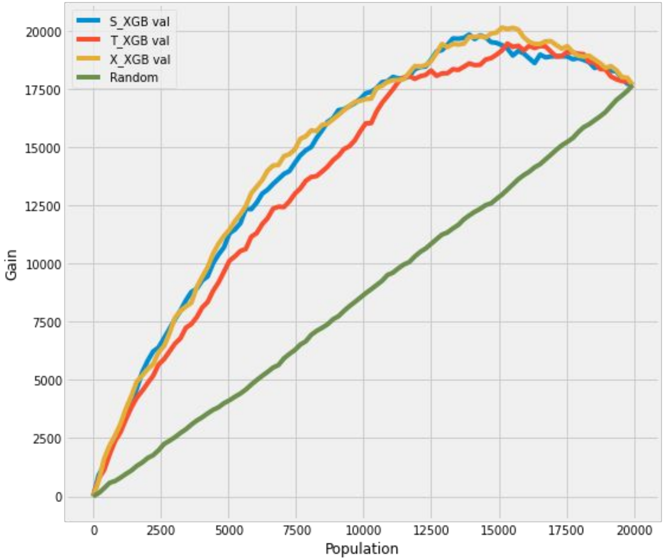
Evaluation | Lift Curve



Lift curve is intuitively easy to understand and hard to interpret.

```
# Plot the lift chart of model estimates in cumulative population.
plot_lift(pred_df, outcome_col=y_col, treatment_col=treatment_col, figsize=(8, 8))
```

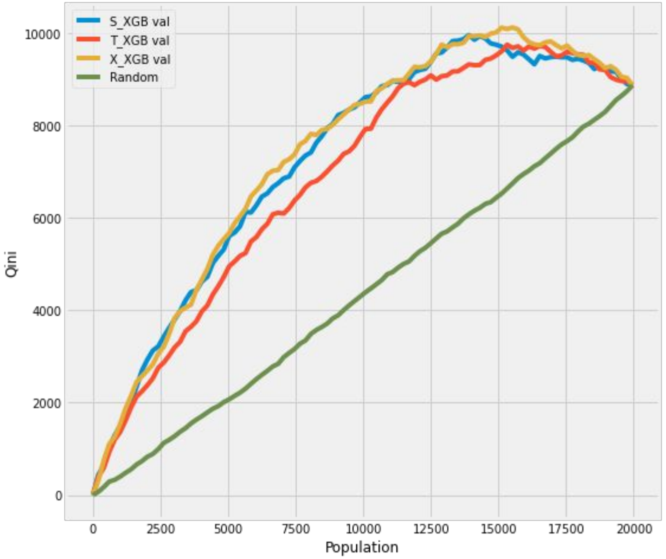
Evaluation | Gain Chart




```
# Plot the cumulative gain chart (or uplift curve) of model estimates.
plot_gain(pred_df, outcome_col=y_col, treatment_col=treatment_col, figsize=(8,
8))
```

Gain charts are built by sorting the main population from the best to the worst lift performance and partitioning in segments. The y-axis represents the cumulative incremental gains and the x-axis the proportion of the population targeted.

Evaluation | Qini Curve



The Qini-Coefficient is the difference between the area under the Uplift Curve and the random curve. A Qini-Coefficient near one represents a good performance of the Uplift Model and a Qini-Coefficient near zero a worse one.

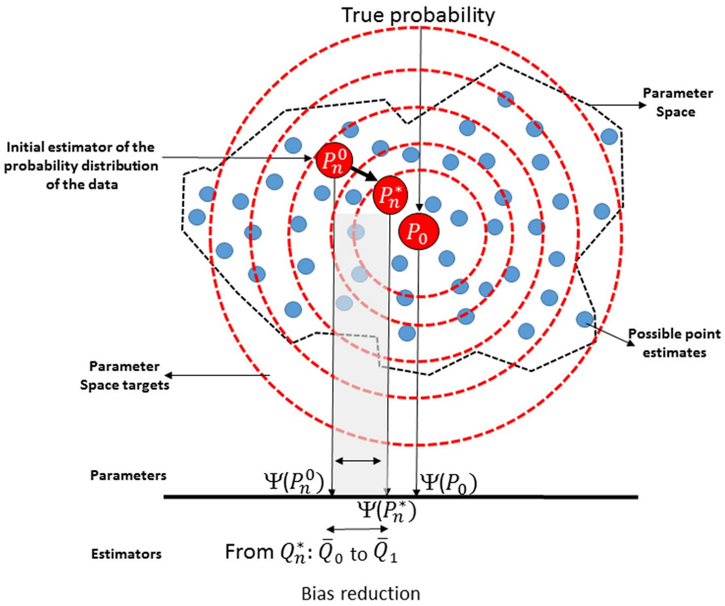
```
# Plot the Qini chart (or uplift curve) of model estimates.
plot_qini(pred_df, outcome_col=y_col, treatment_col=treatment_col, figsize=(8,
8))
```

Evaluation | Area Under Uplift Curve

```
auuc_df = pd.DataFrame(auuc_score(pred_df, outcome_col=y_col,
     treatment_col=treatment_col, normalize=False)).reset_index()
auuc_df.columns = ['Learner', 'auuc']
auuc_df['Lift'] = (auuc_df['auuc'] /
                   auuc_df[auuc_df.Learner == 'Random'].auuc.values) - 1
auuc_df = auuc_df.sort_values('auuc', ascending=False

[Example Output]
INFO:NBOE:     Learner       auuc   Lift
2  X_XGB val 14718.1328 0.7126
0  S_XGB val 14511.6713 0.6886
1  T_XGB val 13783.1099 0.6038
3     Random  8593.9235 0.0000


qini_df = pd.DataFrame(qini_score(pred_df, outcome_col=y_col,
           treatment_col=treatment_col,normalize=False)).reset_index()
qini_df.columns = ['Learner', 'qini']
qini_df = qini_df.sort_values('qini', ascending=False)

[Example Output]
INFO:NBOE:     Learner      qini
2  X_XGB val 3038.7158
0  S_XGB val 2926.5192
1  T_XGB val 2535.8555
3     Random    0.0000
```

- AUUC score and Qini score calculate the area under the two different uplift curves respectively.
- They are quantitative metrics to compare model performance.

Evaluation | TMLE for Robust Eval.

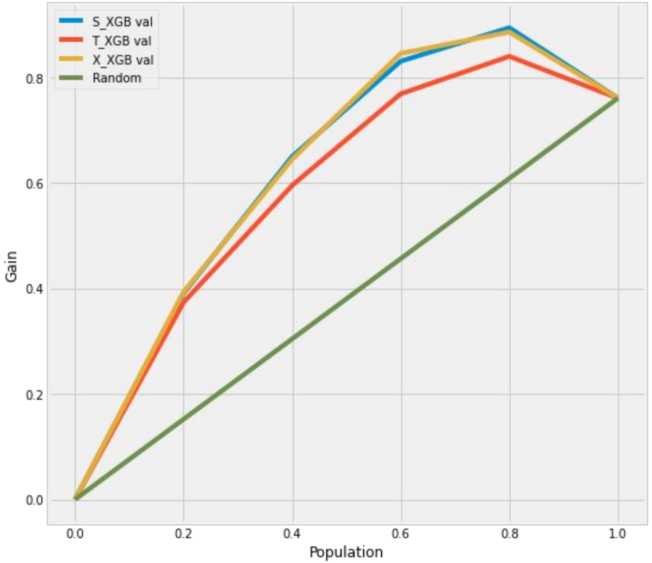


Luque‑Fernandez (2018)

TMLE (Targeted Maximum Likelihood Estimator)[Mark van der Laan] is a general algorithm for the construction of double-robust, semiparametric, efficient substitution estimators.

TMLE allows for data-adaptive estimation while obtaining valid statistical inference.

Evaluation | TMLE Result

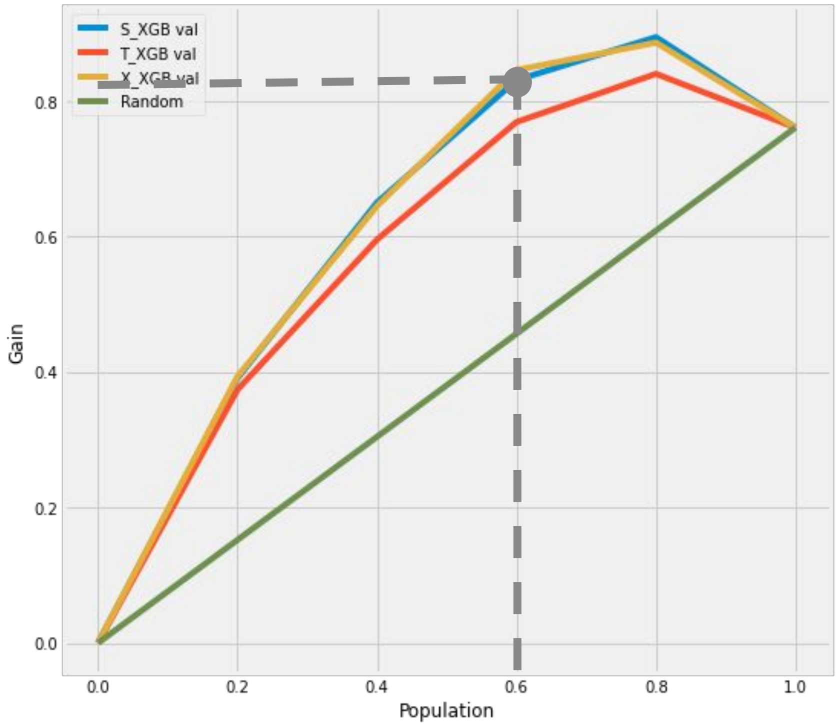



```
# Plot the lift chart based of TMLE estimation
plot_tmlegain(pred_df, inference_col, outcome_col=y_col,
                 treatment_col=treatment_col, p_col=p_col)
```

TMLE (Targeted Maximum Likelihood Estimator)[Mark van der Laan] is a general algorithm for the construction of double-robust, semiparametric, efficient substitution estimators.

TMLE allows for data-adaptive estimation while obtaining valid statistical inference.

Deployment | Targeting Strategy



Targeting users with top 50th percentile uplift score will generate nearly all the Average Treatment Effect (ATE)

Deployment | Explore/Exploit Setup

Audience Group	Phase 1	Phase 2	Phase 3
Holdout (10%)	No Ads	No Ads	No Ads
Exploit (80%)	Persuadables by Model 0	Persuadables by Model 1 P	Persuadables by Model 2
Explore_Ads (5%)	All Users	All Users	All Users
Explore_NoAds (5%)	All Users No Bidding	All Users No Bidding	All Users No Bidding

Model 1 **Model 2** **Model 3**

What’s next?

Reference

[1] Luque‐Fernandez MA, Schomaker M, Rachet B, Schnitzer ME. Targeted maximum likelihood estimation for a binary treatment: A tutorial. *Statistics in Medicine*. 2018;37:2530–2546. https://doi.org/10.1002/sim.7628 [PMC free article] [PubMed] [Google Scholar]

[2] Van Der Laan, Mark J., and Daniel Rubin. "Targeted maximum likelihood learning." The international journal of biostatistics 2.1 (2006).