Learning From Experiments With Causal Machine Learning

A case study using `metalearners`

Francesc Martí Escofet (@fmartiescofet)

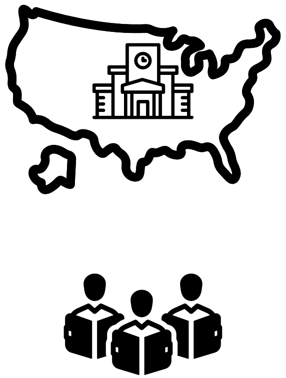
Kevin Klein (@kevkle)


quantco

We live in a budget-constrained world.

Which students should be coached?

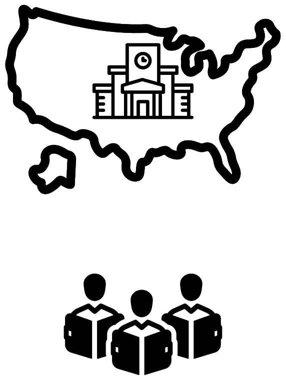
National Study of Learning Mindsets



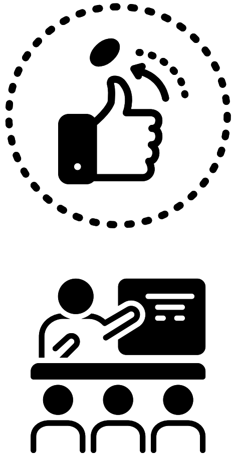
Before data collection

quantco

National Study of Learning Mindsets



Before data collection

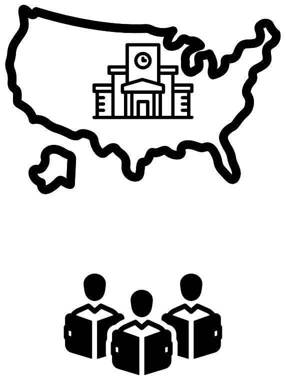


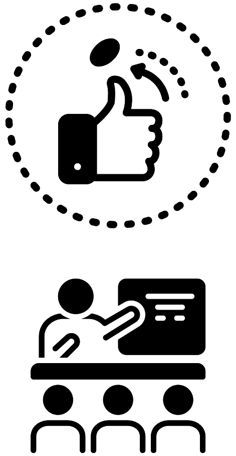
Ninth grade

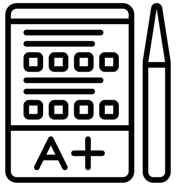
Francesc Martí Escofet @fmartiescofet, Kevin Klein @kevkle

quantco

National Study of Learning Mindsets







Before data collection

Ninth grade

Months later

quantco

The data, more formally

Every datapoint i corresponds to a student.

Name	Symbol	Definition
Covariates	X_i	Properties of the student or the student's school
Treatment assignments	W_i	$\begin{cases} 1 & \text{if received coaching} \\ 0 & \text{if didn't receive coaching} \end{cases}$
Outcome	Y_i	GPA ($\in \mathbb{R}$) after treatment

$$\mathcal{D} = \{(X_i, W_i, Y_i)\}$$

The data, the details

- n = 10,391
  - ~1/3 received coaching
- Originally from National Study of Learning Mindsets
  - Nature, September 2019
- We used an anonymized version from Athey and Wager
  - All continuous features have been transformed to a standard Normal

The data, an excerpt

	schoolid	success_expect	ethnicity	gender	frst_in_family	school_urba
3625	75	5	2	1	1	
3037	5	2	2	1	1	
2574	17	7	4	2	1	
1488	47	6	5	2	1	
3677	74	6	11	2	1	

quantco

Loading the data

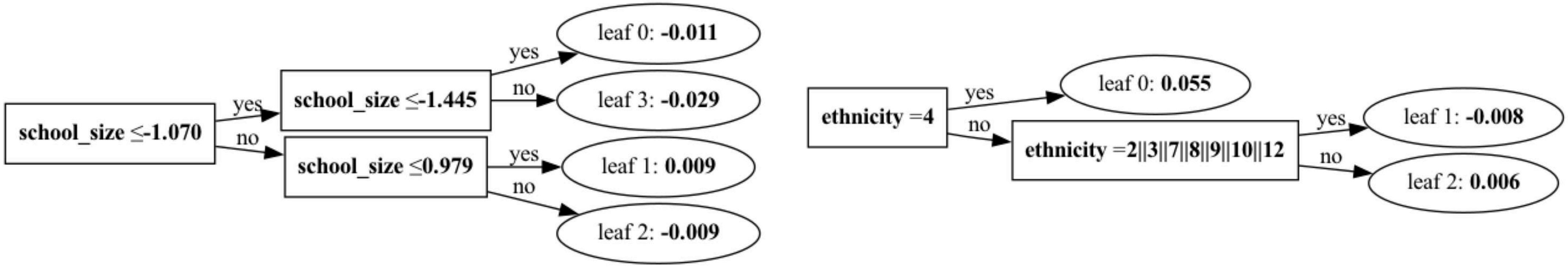
```
import pandas as pd

df = pd.read_csv("learning_mindset.csv")
```

```
categorical_feature_columns = [
    "ethnicity",
    "gender",
    "frst_in_family",
    "school_urbanicity",
    "schoolid",
]
```

```
for column in categorical_feature_columns:
    df[column] = df[column].astype("category")
```

quantco

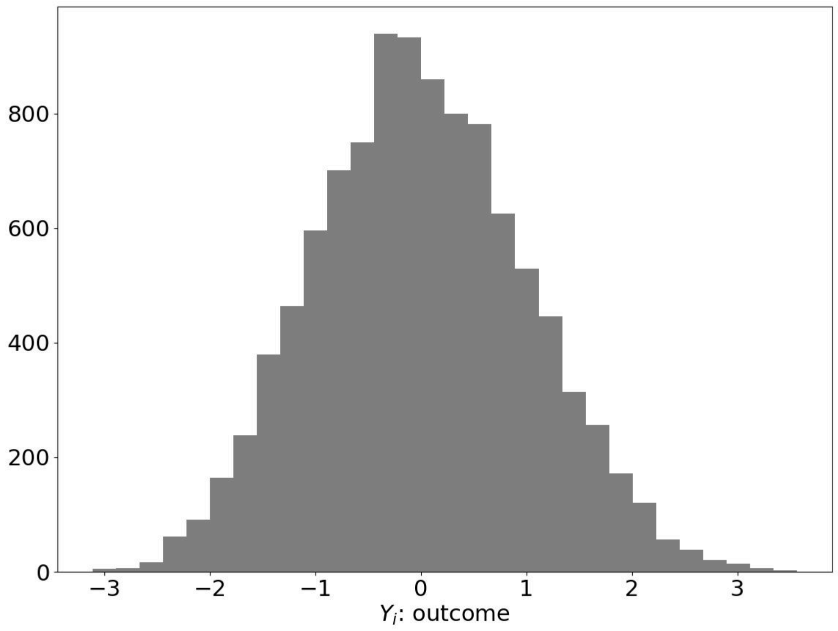
Why categoricals?

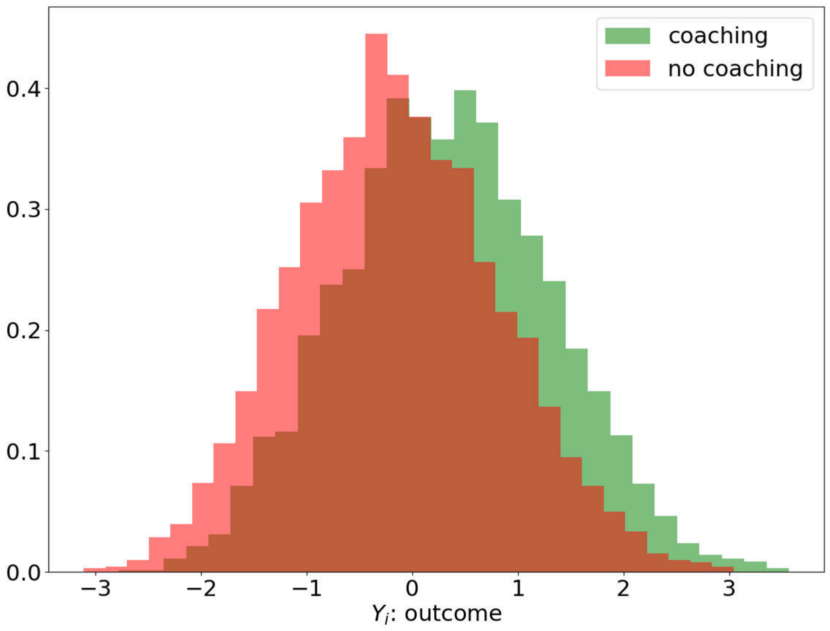


Outcomes

```
ax.hist(df[treatment_column])
```

```
ax.hist(df[W=1][outcome_column], density=True)
ax.hist(df[W=0][outcome_column], density=True)
```





quantco

What do we do with the data now?

- Remember, our original question was

> (A) Which students should be coached?

- We'll reduce said question to the following question

> (B) How much would a student like student i profit from a growth mindset coaching?

What do we do with the data now?

- > (B) How much would a student like student i profit from a growth mindset coaching?

formalism: **Conditional Average Treatment Effect** (CATE)

$$\tau(X_i) = \mathbb{E}[Y(\text{coaching}) - Y(\text{no coaching}) | X = X_i]$$

quantco

What do we do with the data now?

- > (B) How much would a student like student i profit from a growth mindset coaching?

formalism: **Conditional Average Treatment Effect** (CATE)

$$\tau(X_i) = \mathbb{E}[\underbrace{Y(\text{coaching}) - Y(\text{no coaching})}_{\text{treatment effect}}|X = X_i]$$

quantco

What do we do with the data now?

- > (B) How much would a student like student i profit from a growth mindset coaching?

formalism: **Conditional** **Average** **Treatment Effect** (CATE)

$$\tau(X_i) = \underbrace{\mathbb{E}}_{\text{average}} [Y(\text{coaching}) - Y(\text{no coaching}) | X = X_i]$$

quantco

What do we do with the data now?

- > (B) How much would a student like student i profit from a growth mindset coaching?

formalism: Conditional **Average Treatment Effect** (CATE)

$$\tau(X_i) = \mathbb{E}[Y(\text{coaching}) - Y(\text{no coaching})\underbrace{|X = X_i]}_{\text{conditional}}$$

quantco

What do we do with the data now?

- > (B) How much would a student like student i profit from a growth mindset coaching?

  formalism: **Conditional Average Treatment Effect** (CATE)

$$\tau(X_i) = \mathbb{E}[Y(\text{coaching}) - Y(\text{no coaching})|X = X_i]$$

- > (A) Which students should be coached?

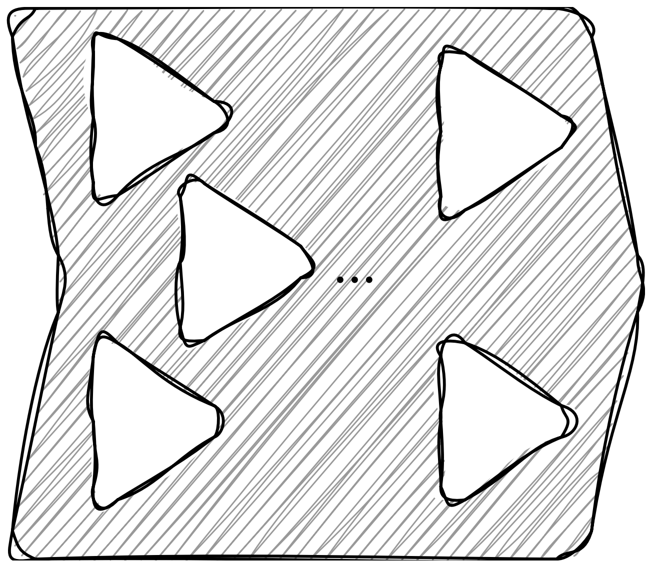
  formalism: **policy**

$$\pi(X_i) = \begin{cases} 1 & \text{if } \hat{\tau}(X_i) \geq c_{budget} \\ 0 & \text{otherwise} \end{cases}$$

quantco

MetaLearners

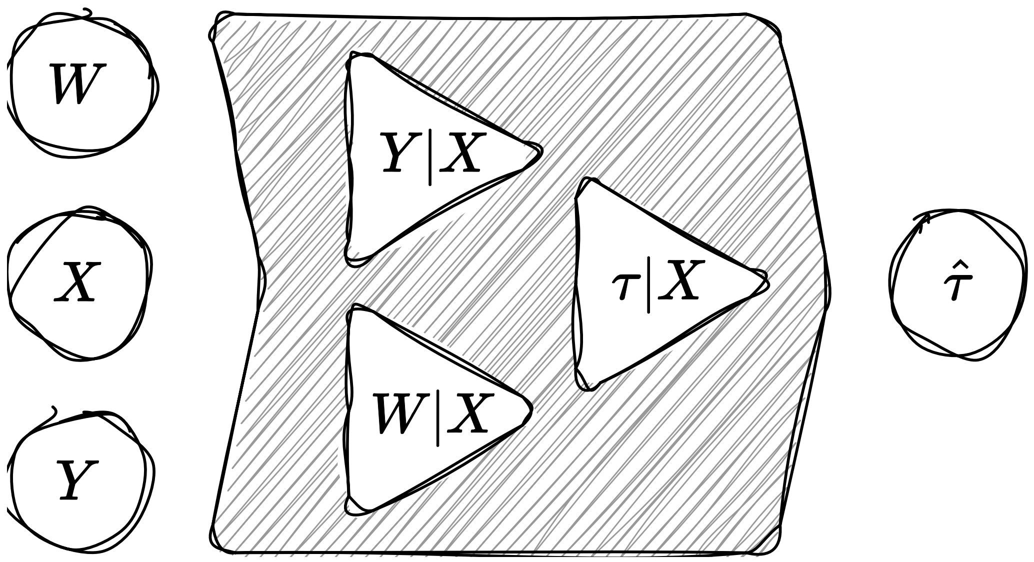
- MetaLearners are **CATE models** which rely on typical, **arbitrary machine learning estimators** (classifiers or regressors) as **components**.
- Some examples include the S-Learner, T-Learner, F-Learner, X-Learner, R-Learner, M-Learner and DR-Learner.



quantco

MetaLearners

- Input
  - W: Treatment assignments
  - X: Covariates/features
  - Y: Outcomes
- Output
  - $\hat{\tau}(X)$: CATE estimates



Creating a first MetaLearner

```python
from metalearners import RLearner
from lightgbm import LGBMRegressor, LGBMClassifier
```

```python
rlearner = RLearner(
    nuisance_model_factory=LGBMRegressor,
    propensity_model_factory=LGBMClassifier,
    treatment_model_factory=LGBMRegressor,
    is_classification=False,
    n_variants=2,
)
```

quantco

Creating a first MetaLearner

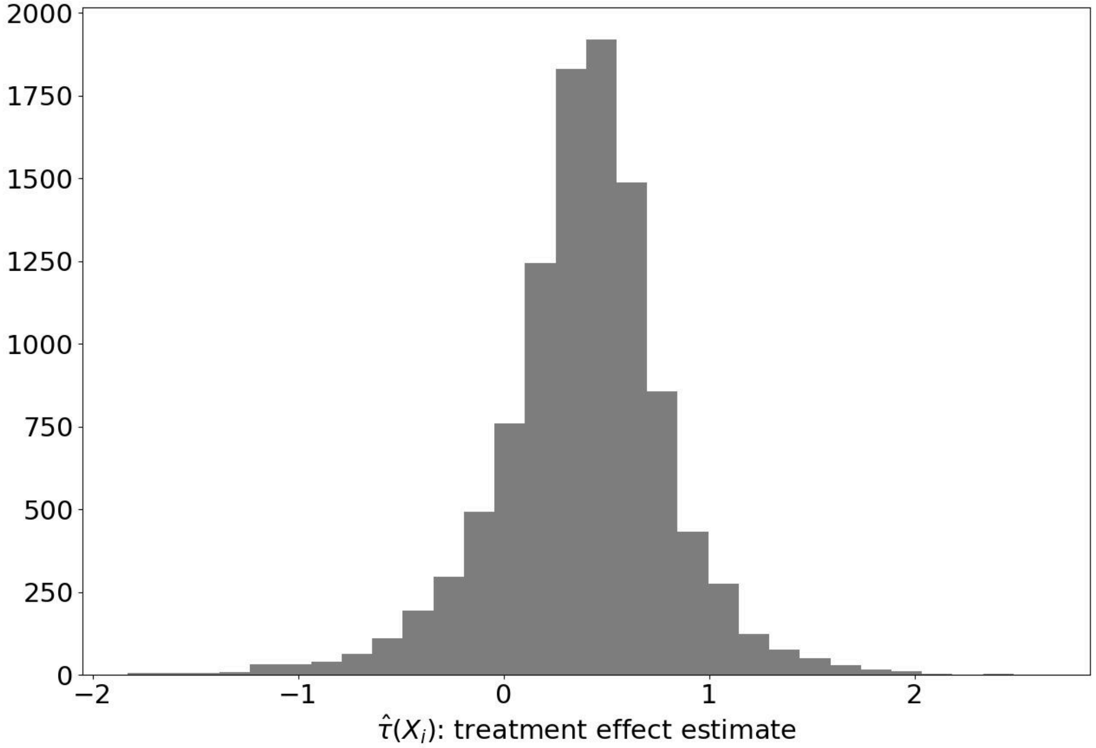
```
from metalearners import RLearner
from lightgbm import LGBMRegressor, LGBMClassifier
```

```
rlearner = RLearner(
    nuisance_model_factory=LGBMRegressor,
    propensity_model_factory=LGBMClassifier,
    treatment_model_factory=LGBMRegressor,
    is_classification=False,
    n_variants=2,
)
```

```
rlearner.fit(
    X=df[feature_columns], y=df[outcome_column], w=df[treatment_column]
)
```

Predicting with a MetaLearner

```python
rlearner.predict(df[feature_columns], is_oos=False)
```

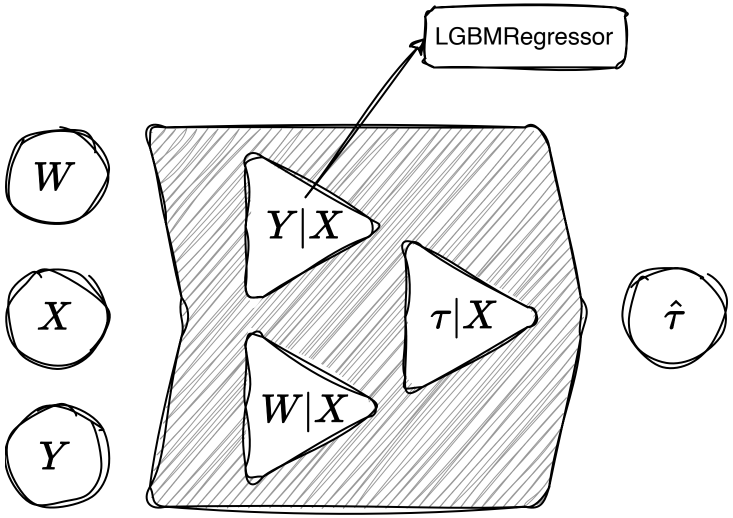


quantco

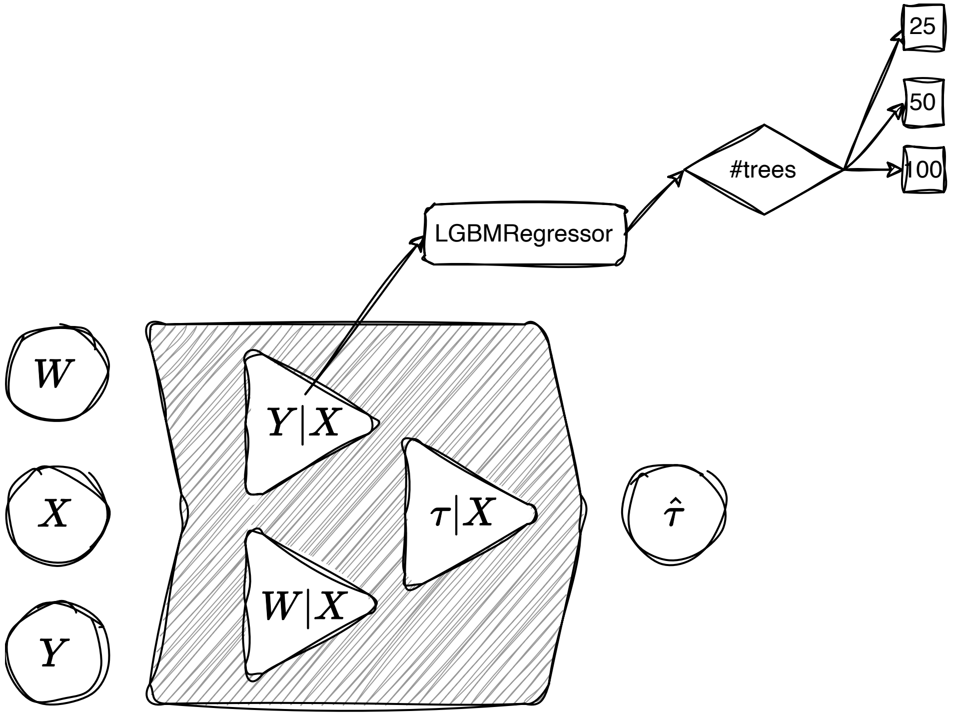
Hyperparameter optimization

- HPO can have massive impacts on the prediction quality in regular Machine Learning
- According to Machlanski et. al (2023) this also happens in MetaLearners
- Three levels to optimize for:
  - The MetaLearner architecture
  - The model to choose per base estimator
  - The model hyperparameters per base model

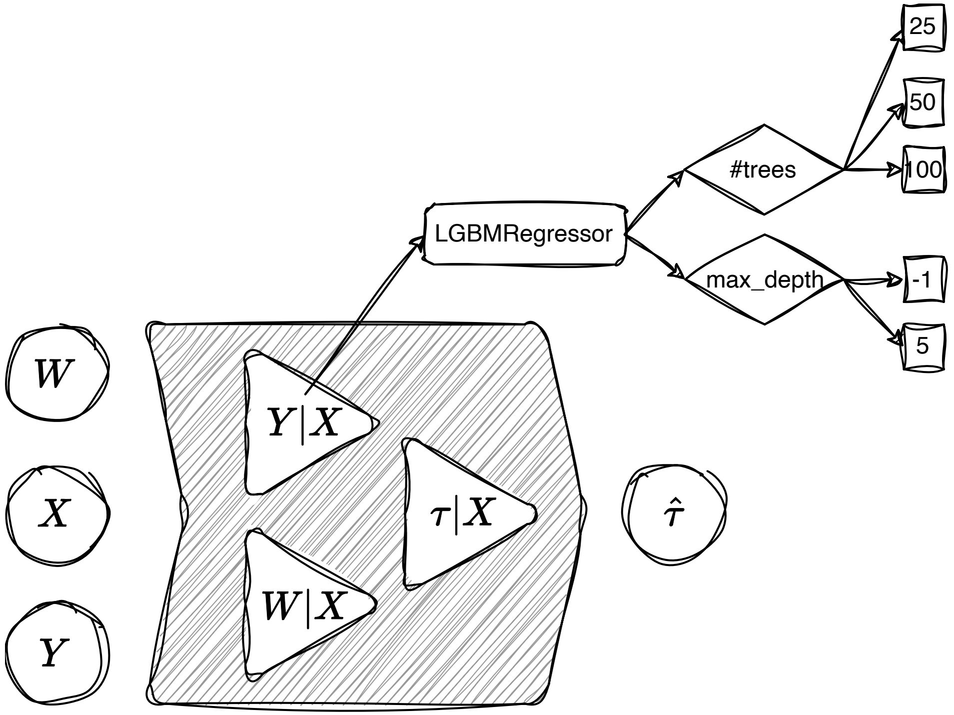
Performing a grid search



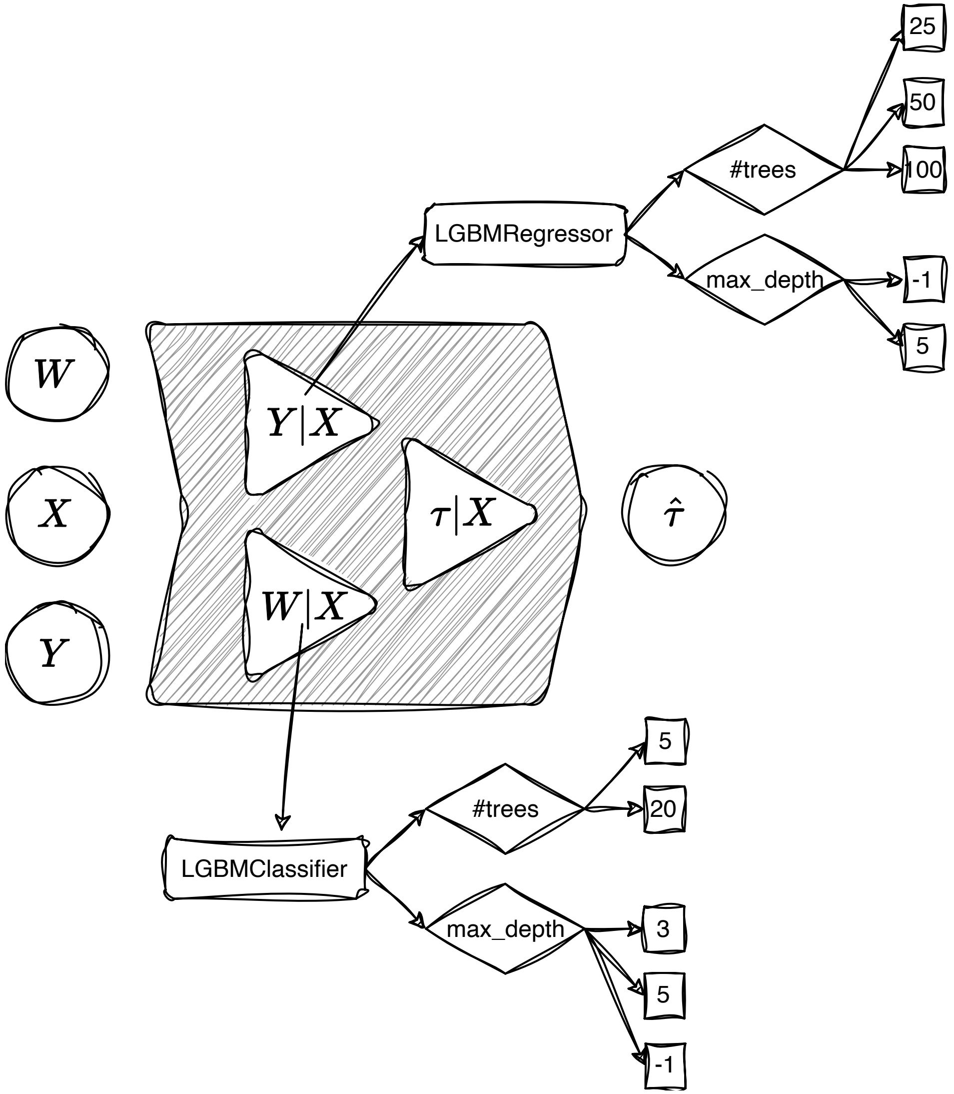
Performing a grid search



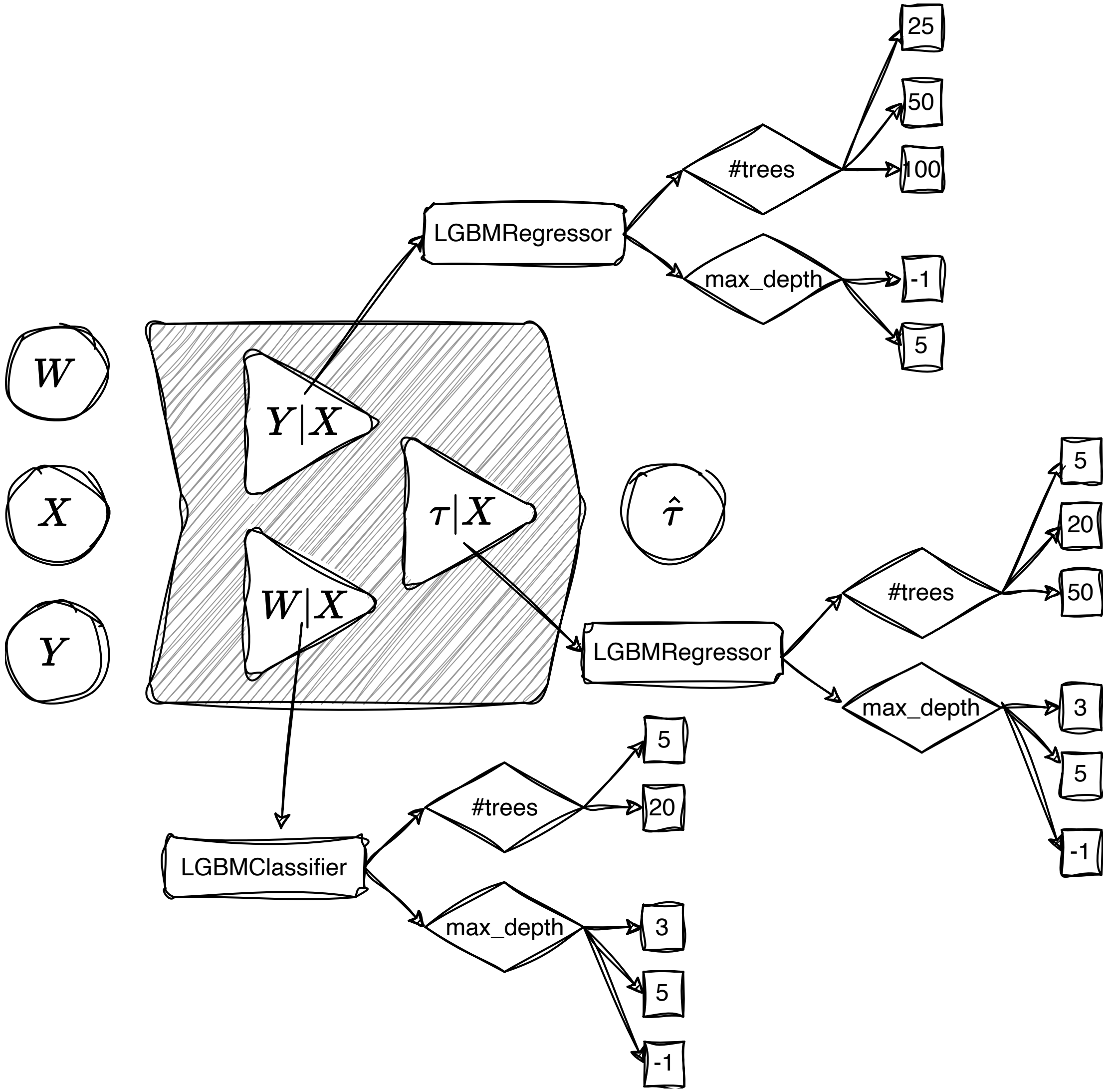
Performing a grid search



Performing a grid search



Performing a grid search



quantco

Performing a grid search

```
base_learner_grid = {
    "outcome_model": [LGBMRegressor],
    "propensity_model": [LGBMClassifier],
    "treatment_model": [LGBMRegressor],
}
```

```
param_grid = {
    "outcome_model": {
        "LGBMRegressor": {"n_estimators": [25, 50, 100], "max_depth": [-1, 5]}
    },
    "treatment_model": {
        "LGBMRegressor": {"n_estimators": [5, 20, 50], "max_depth": [-1, 3, 5]}
    },
    "propensity_model": {
        "LGBMClassifier": {"n_estimators": [5, 50], "max_depth": [-1, 3, 5]}
    },
}
```

quantco

Performing a grid search

```
gs = MetaLearnerGridSearch(
    metalearner_factory=RLearner,
    metalearner_params={"is_classification": False, "n_variants": 2},
    base_learner_grid=base_learner_grid,
    param_grid=param_grid,
)
```

```
gs.fit(X_train, y_train, w_train, X_validation, y_validation, w_validation)
```

```
gs.results_
```

hyperparameters	time fit	time score	train propensity	train outcome	train r_loss	trea
-1, 25, -1, 5, -1, 5	0.899935	0.304743	-0.631725	-0.817461	0.795676	-1.[illegible]
-1, 25, -1, 5, -1, 20	0.965532	0.312325	-0.631725	-0.817461	0.798854	-1.[illegible]
-1, 25, -1, 5, -1, 50	1.19587	0.365287	-0.631725	-0.817461	0.804784	-1[illegible]
...	...	...	...	...	...	
5, 25, 3, 5, 3, 20	1.79398	0.108887	-0.630564	-0.818076	0.796231	-1.[illegible]
...	...	...	...	...	...	
5, 100, 5, 50, 5, [illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

quantco

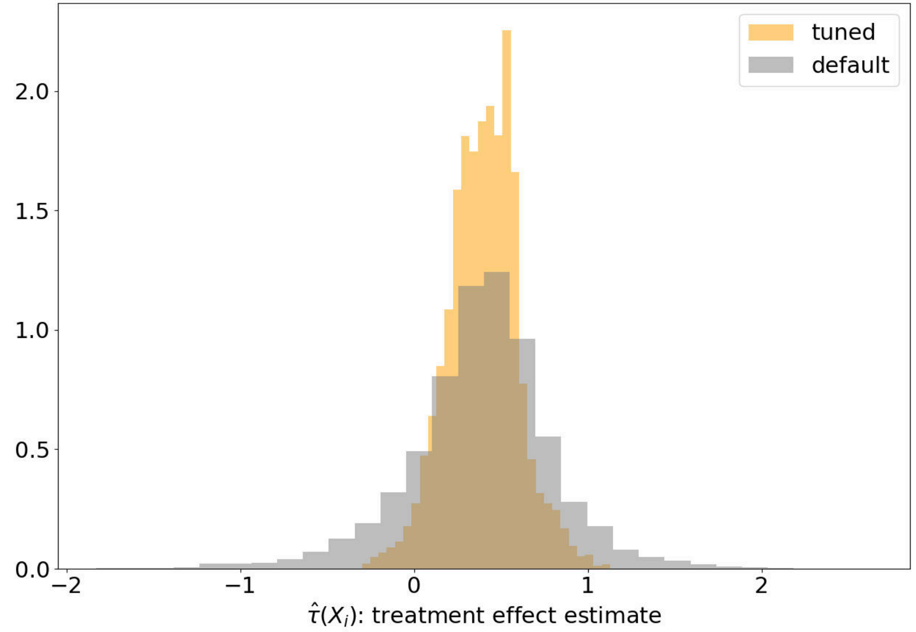
```
gs.results_
```

hyperparameters	time fit	time score	train propensity	train outcome	train r_loss	trea
-1, 25, -1, 5, -1, 5	0.899935	0.304743	-0.631725	-0.817461	0.795676	-1.
-1, 25, -1, 5, -1, 20	0.965532	0.312325	-0.631725	-0.817461	0.798854	-1.
-1, 25, -1, 5, -1, 50	1.19587	0.365287	-0.631725	-0.817461	0.804784	-1
...	...	...	...	...	...	
5, 25, 3, 5, 3, 20	1.79398	0.108887	-0.630564	-0.818076	0.796231	-1.
...	...	...	...	...	...	
5, 100, 5, 50, 5,	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

quantco

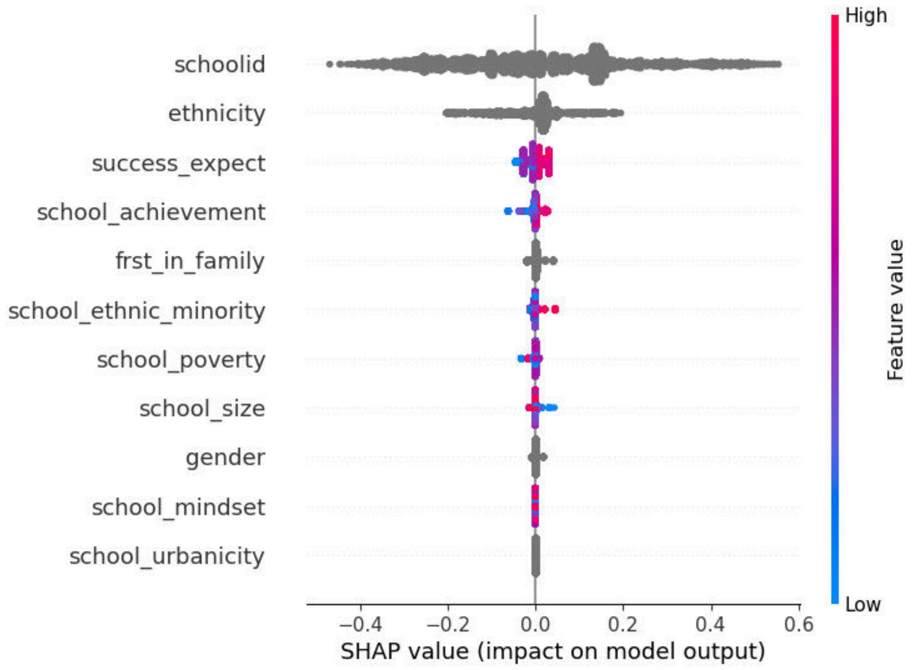
Predicting with a tuned MetaLearner

```
tuned_rlearner.predict(df[feature_columns], is_oos=False)
```



SHAP values

```
from shap import TreeExplainer, summary_plot
explainer = learner.explainer()
shap_values = explainer.shap_values(df[feature_columns], TreeExplainer)
summary_plot(shap_values[0], features=df[feature_columns])
```



quantco

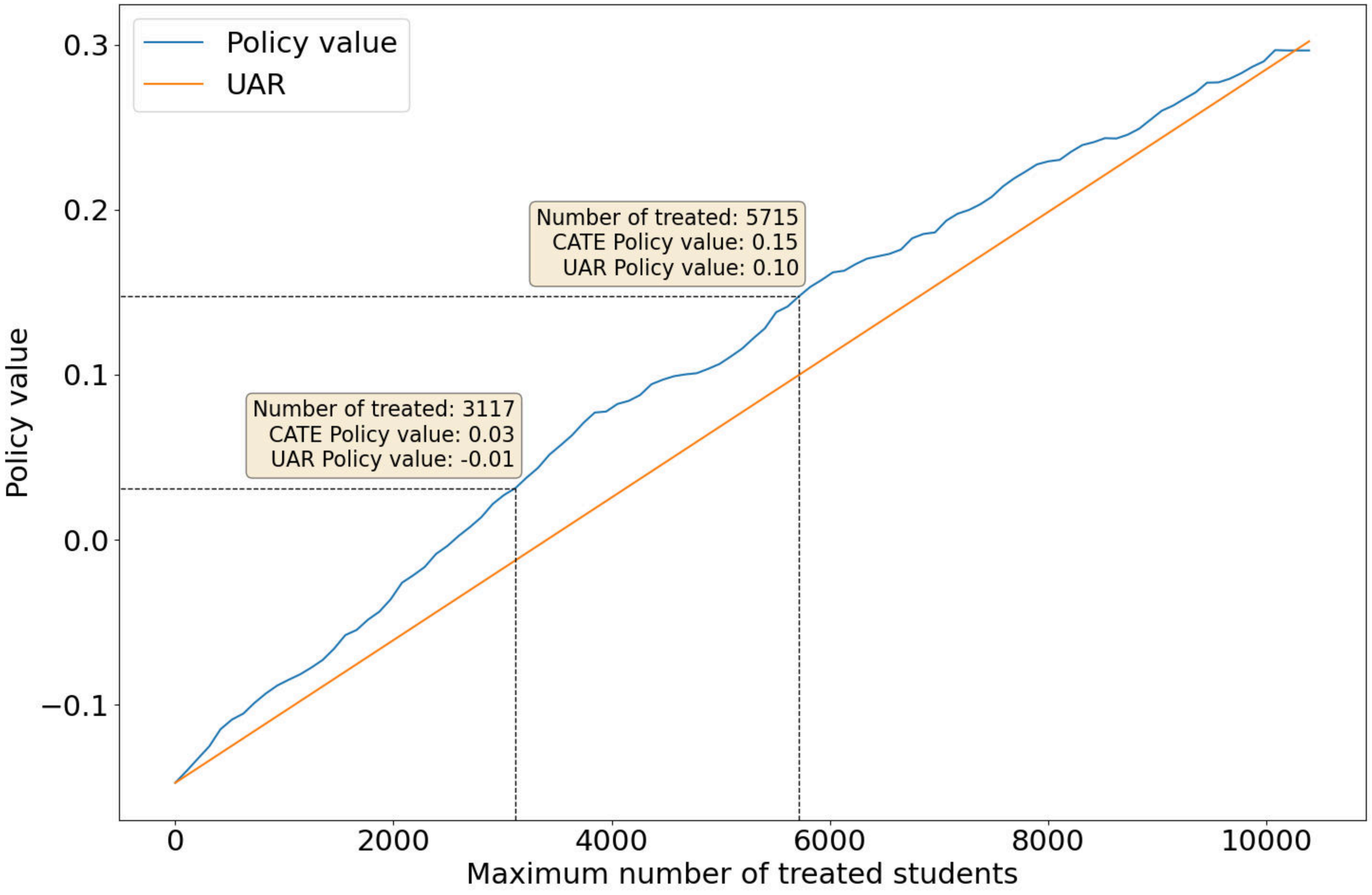
But now, how are we actually doing?

We can define the policy value as:

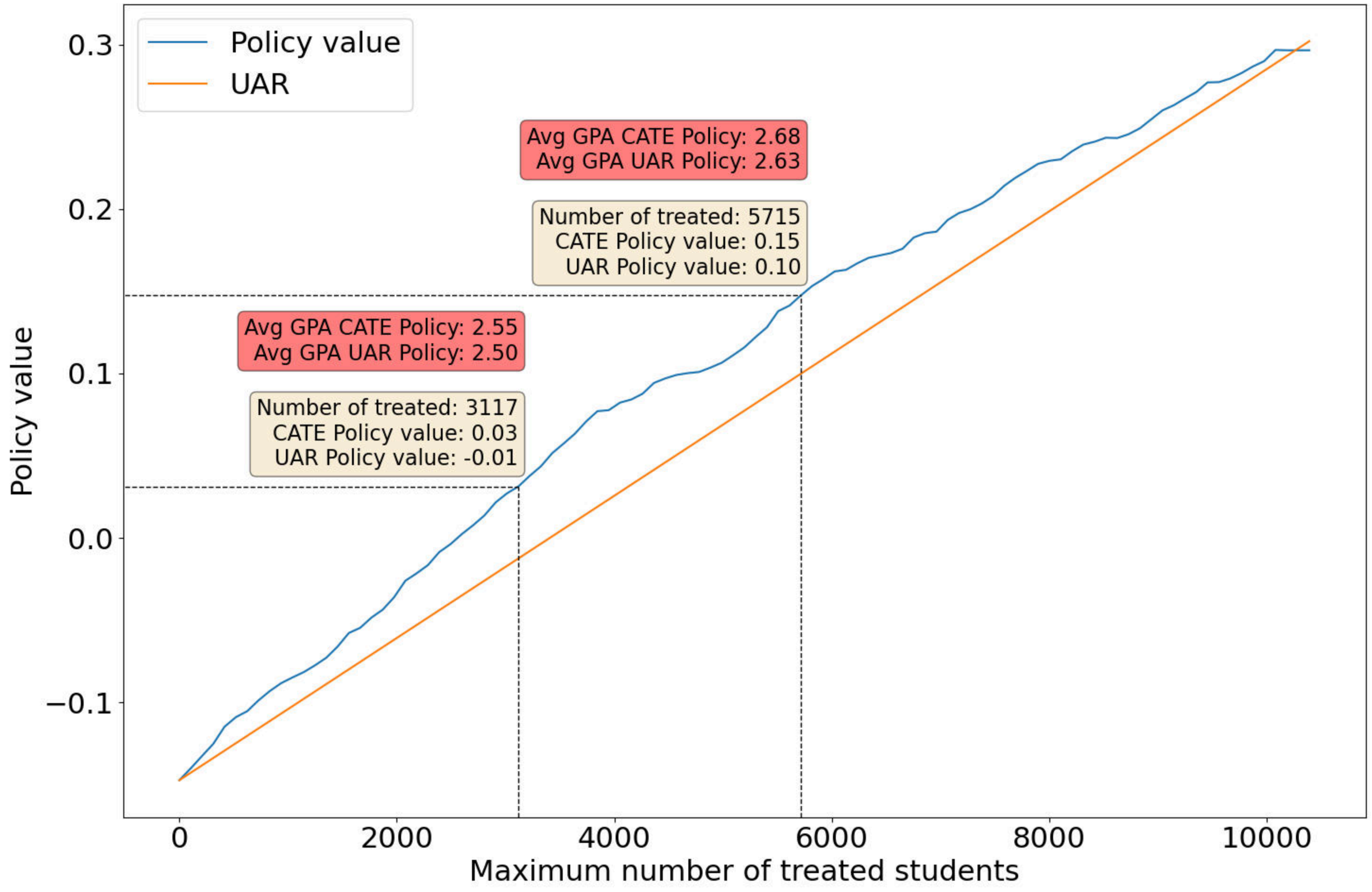
$$V(\pi) = \mathbb{E}[Y_i(\pi(X_i))]$$

Using our CATE estimates, we can define a policy that targets the most promising students, specifically, those with the highest CATE estimates.

But now, how are we actually doing?



Making it tangible



quantco

Would you like to work on such topics, too?

Join us!

quantco.com

DEEP LEARNING

Deep Learning Engineer
APPLY
HYBRID — FULL-TIME EUROPE

Research Scientist - Virdx
APPLY
HYBRID — FULL-TIME LONDON, ENGLAND / ZURICH, SWITZERLAND

ENGINEERING

Senior Software Engineer
APPLY
HYBRID — FULL-TIME EUROPE

Software Engineer
APPLY
HYBRID — FULL-TIME KARLSRUHE, BADEN-WÜRTTEMBERG

Software Engineer
APPLY
HYBRID — FULL-TIME ZURICH, SWITZERLAND

Software Engineer
APPLY
HYBRID — FULL-TIME BERLIN, BERLIN

Software Engineer
APPLY
HYBRID — FULL-TIME MUNICH, BAVARIA

Software Engineering Intern
APPLY
ON-SITE — INTERN EUROPE

DATA SCIENCE

Data Science Intern
APPLY
ON-SITE — INTERN EUROPE

Data Scientist
APPLY
HYBRID — FULL-TIME ZURICH, SWITZERLAND

Data Scientist
APPLY
HYBRID — FULL-TIME MUNICH, BAVARIA

Data Scientist
APPLY
HYBRID — FULL-TIME BERLIN, BERLIN

Data Scientist
APPLY
HYBRID — FULL-TIME LONDON, ENGLAND

Medical Physicist - Virdx
APPLY
HYBRID — FULL-TIME BOSTON, MASSACHUSETTS

Quantitative Researcher
APPLY
HYBRID — FULL-TIME EUROPE

Quantitative Researcher
APPLY
HYBRID — FULL-TIME USA

Please leave feedback on GitHub! :)

github.com/QuantCo/metalearners

github.com/kklein/sps24-metalearners

Backup

quantco

Data dictionary

Name	Type	Meaning
ethnicity	categorical	student race/ethnicity
gender	categorical	student-identified gender
success_expect	discrete	self-reported expectations for success in the future
frst_in_family	boolean	first in family to go to college
schoolid	categorical	identifier for each of 76 high schools
school_urbanicity	categorical	school's urbanicity (urban, rural, etc.)
school_mindset	numerical	school's mean mindset

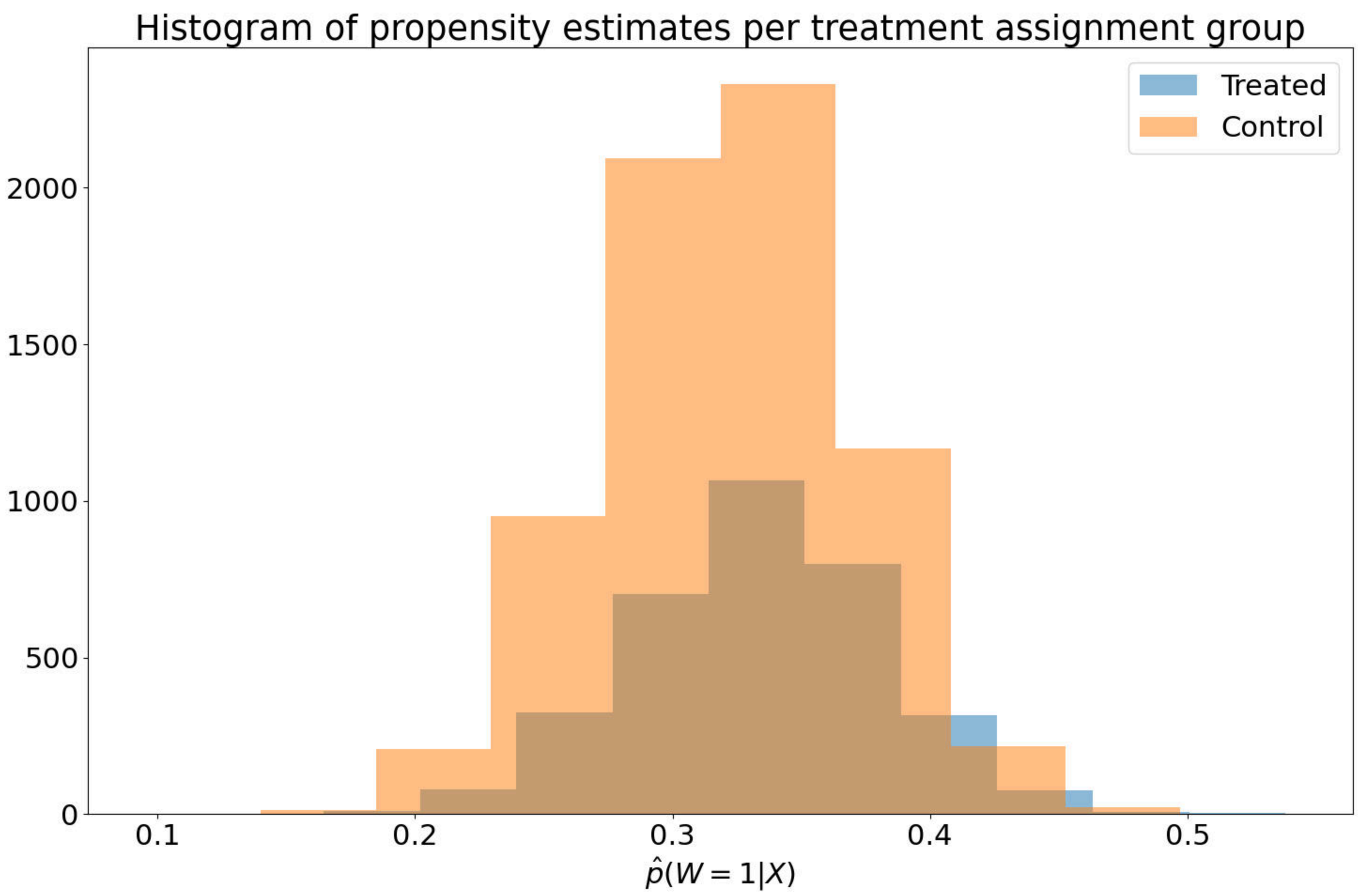
quantco

Conventional assumptions for estimating CATEs

- Positivity/overlap
- Conditional ignorability/unconfoundedness
- Stable Unit Treatment Value (SUTVA)

A randomized control trial usually gives us the first two for free.

For more information see e.g. Athey and Imbens, 2016.



Histogram of propensity estimates per treatment assignment group
Treated
Control
2000
1500
1000
500
0
0.1
0.2
0.3
0.4
0.5
$\hat{p}(W = 1|X)$

Policy value estimation

We estimated policy values via the 'Inverse-Propensity Weighting' estimator:

$$\hat{V}_{IPW}(\pi) = \frac{1}{n} \sum_{i=1}^{n} \frac{Y_i \mathbb{I}[W_i = \pi(X_i)]}{\Pr[W_i = \pi(X_i) | X_i]}$$

For more details, please see Stefan Wager's lecture notes.

Python implementations of MetaLearners

	`metalearners`	`causalml`	`econml`
MetaLearner implementations	✓	✓	✓
Support* for `pandas`, `scipy`, `polars`	✓	✗	✗
HPO integration	✓	✗	✗
Concurrency across base models	✓	✗	✗
>2 treatment variants	✓	✓	✗
Classification*	✓	✗	✓
Other Causal Inference methods	✗	✓	✓