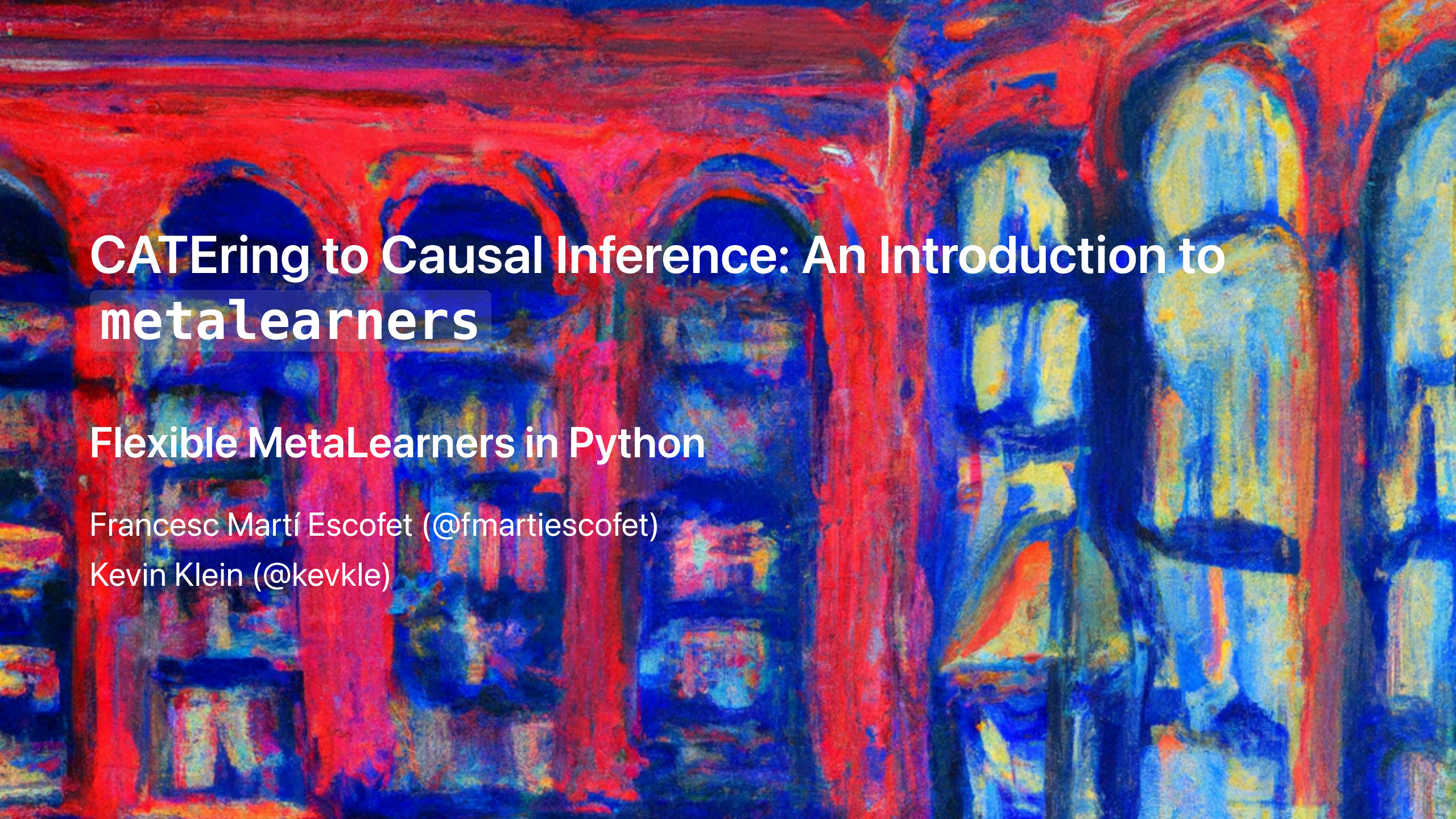


CATEring to Causal Inference: An Introduction to `metalearners`

Flexible MetaLearners in Python

Francesc Martí Escofet (@fmartiescofet)
Kevin Klein (@kevkle)

quantco

Let the data decide!

- We not only want to predict what happens in a world in which we don't have influence on the environment/data generating process.
- Rather, we want to **decide** which **intervention** to choose.
- Intuitively, in order to decide, we'd like to **compare the 'outcomes'** if taking the blue pill compared to the outcome of taking the red pill.

$$Y(\text{blue pill}) - Y(\text{red pill})$$

quantco

The data

- We conducted an empirical experiment to gather data.
- This empirical **data** on individuals, who have
  - had their properties/**covariates** evaluated at the time of the intervention
  - been subject to the **intervention** (taking either of both pills)
  - had the **outcome** of their happiness measured, 5 years after the intervention

quantco

The data, more formally

Our experiment data contains three different kinds of quantities per individual i:

Name	Symbol	Definition
Covariates	X_i	Properties of the time of intervention
Treatment assignments	W_i	$\begin{cases} 1 & \text{if blue pill} \\ 0 & \text{if red pill} \end{cases}$
Outcome	Y_i	Happiness score ($\mathbb{R}$) 5 years after intervention

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

quantco

X: Properties/covariates

	age	#phil books	dominant personality trait	education	atheist	...
Anne	85	14	extraversion	BA	1	...
Bob	17	1	openness	HS	0	...
Charlie	34	0	conscientiousness	PhD	1	...
...	...	...	...	...	...	...

quantco

Treatment effects

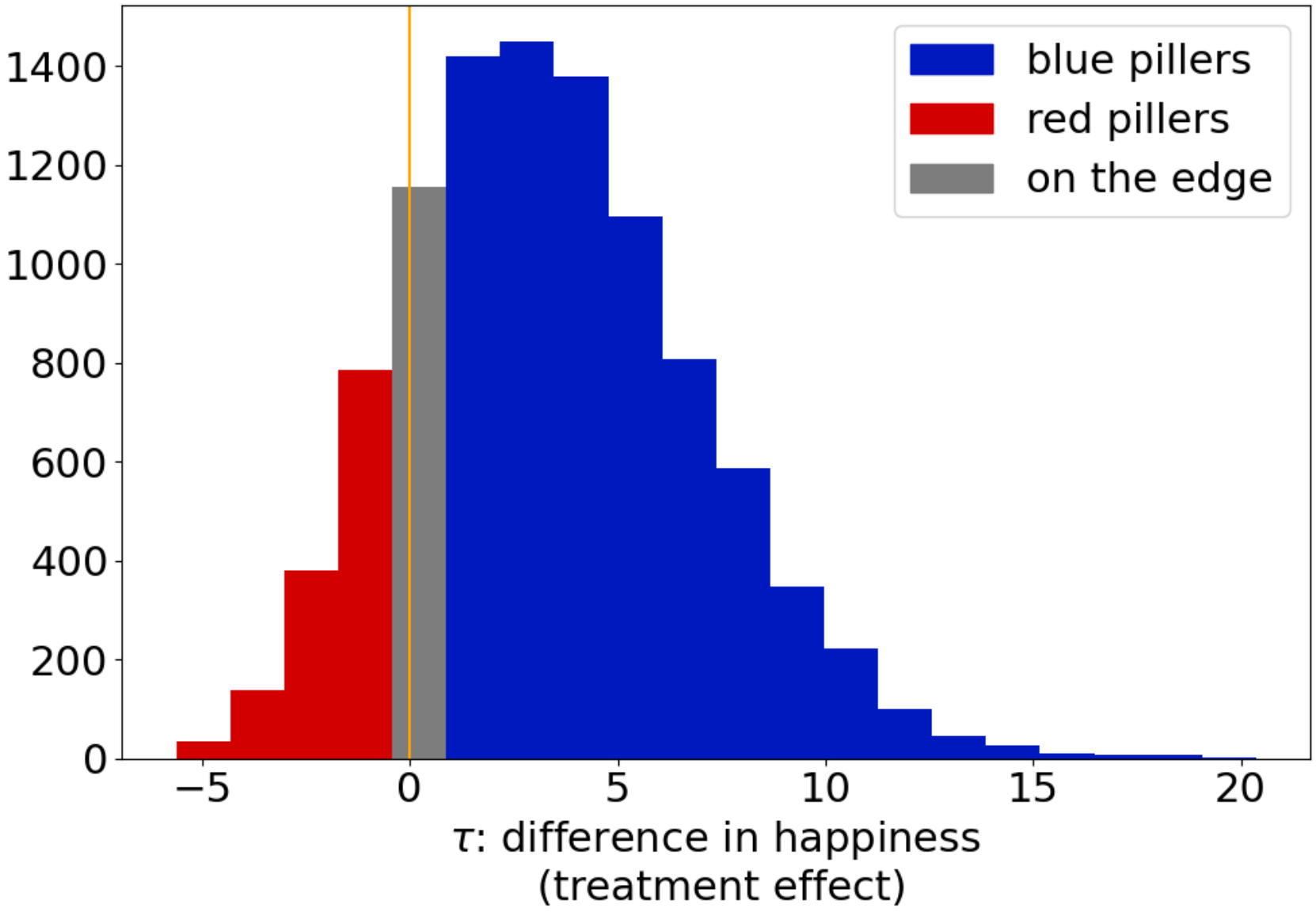
Many Causal Inference techniques allow for the estimation of the **Average Treatment Effect**

$$\mathbb{E}[Y(\text{blue pill}) - Y(\text{red pill})]$$

quantco


1400
1200
1000
800
600
400
200
0
−5
0
5
10
15
20
τ: difference in happiness
(treatment effect)

quantco


blue pillers
red pillers
on the edge
1400
1200
1000
800
600
400
200
0
−5
0
5
10
15
20
τ: difference in happiness
(treatment effect)

quantco

Capturing heterogeneity

Instead of estimating

$$\mathbb{E}[Y(\text{blue pill}) - Y(\text{red pill})]$$

we'd like to estimate

$$Y_i(\text{blue pill}) - Y_i(\text{red pill})$$

quantco

But how?

In an ideal world:

	age	#phil books	...	$Y(\text{blue pill})$	$Y(\text{red pill})$	τ
Anne	85	14	...	15.8	13.2	2.6
Bob	17	1	...	32.3	32.4	-0.1
Charlie	34	0	...	7.0	9.2	-2.2
...	...	...	...	...	...	...

quantco

But how?

In reality:

	age	#phil books	...	$Y(\text{blue pill})$	$Y(\text{red pill})$	τ
Anne	85	14	...	15.8	?	?
Bob	17	1	...	?	32.4	?
Charlie	34	0	...	?	9.2	?
...	...	...	...	...	...	...

The **fundamental problem of Causal Inference** states that we can never observe $Y(\text{blue pill})$ and $Y(\text{red pill})$ simultaneously for the same 'unit'.

quantco

What now?

- We can't know the Individual Treatment Effect (ITE)

$$Y_i(\text{blue pill}) - Y_i(\text{red pill})$$

- Yet, we **can estimate** the **Conditional Average Treatment Effect (CATE)**

$$\tau(X) := \mathbb{E}[Y(\text{blue pill}) - Y(\text{red pill})|X]$$

  - Note the difference from the Average Treatment Effect

$$\mathbb{E}[Y(\text{blue pill}) - Y(\text{red pill})]$$

quantco

Learning a policy

In order to get to a **decision** as to give what pill to whom, we can now follow the following process:

1. Estimate CATEs $\hat{\tau}(X) = \mathbb{E}[Y(\text{blue pill}) - Y(\text{red pill})|X]$
2. Distribute pills according to this policy:

$$\pi(X) := \begin{cases} \text{blue pill} & \text{if } \hat{\tau}(X) \geq 0 \\ \text{red pill} & \text{if } \hat{\tau}(X) < 0 \end{cases}$$

quantco

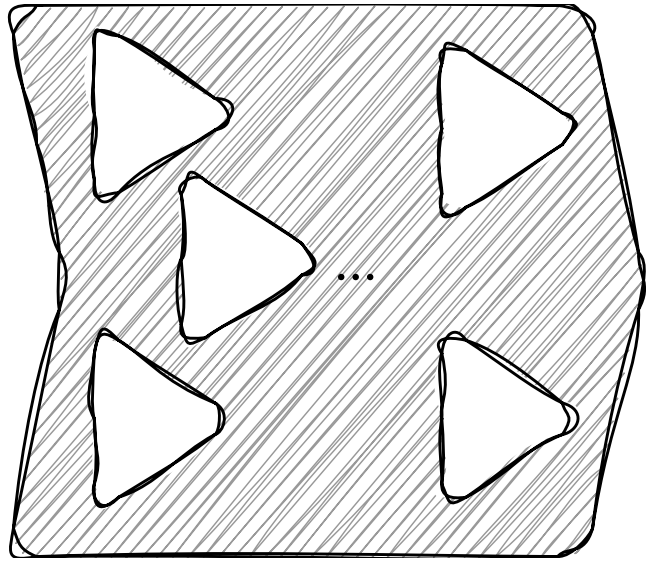
Estimating CATEs

- There are various approaches for estimating CATEs.
- MetaLearners are a family of approaches for estimating CATEs.
  - Work by Chernozhukov(2016), Nie(2017), Kunzel(2017), Kennedy(2020) and more.

quantco

MetaLearners

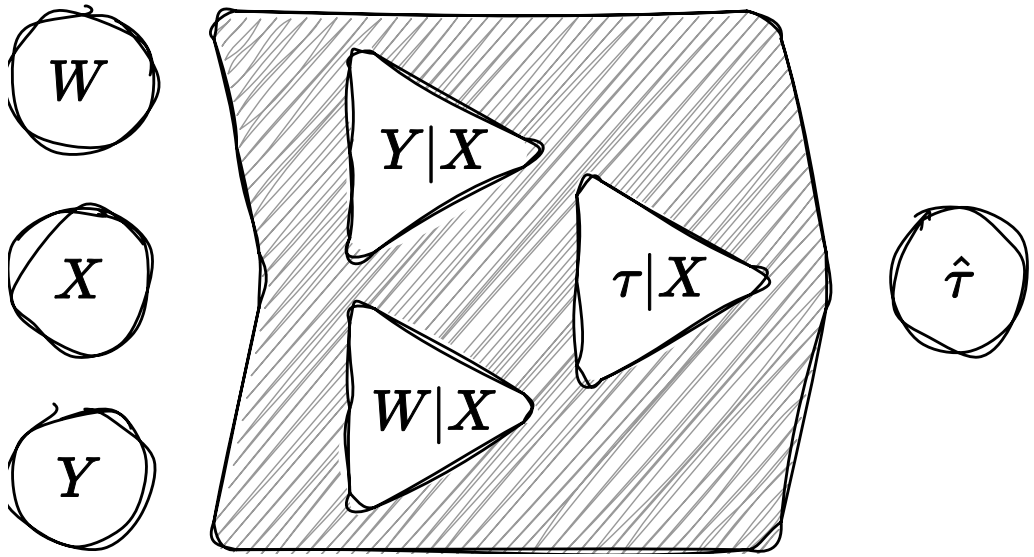
- MetaLearners are **CATE models** which rely on typical, **arbitrary machine learning estimators** (classifiers or regressors) as **components**.
- Some examples include the S-Learner, T-Learner, F-Learner, X-Learner, R-Learner, M-Learner and DR-Learner.



quantco

MetaLearners

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



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,
)
```

quantco

Creating a first MetaLearner

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

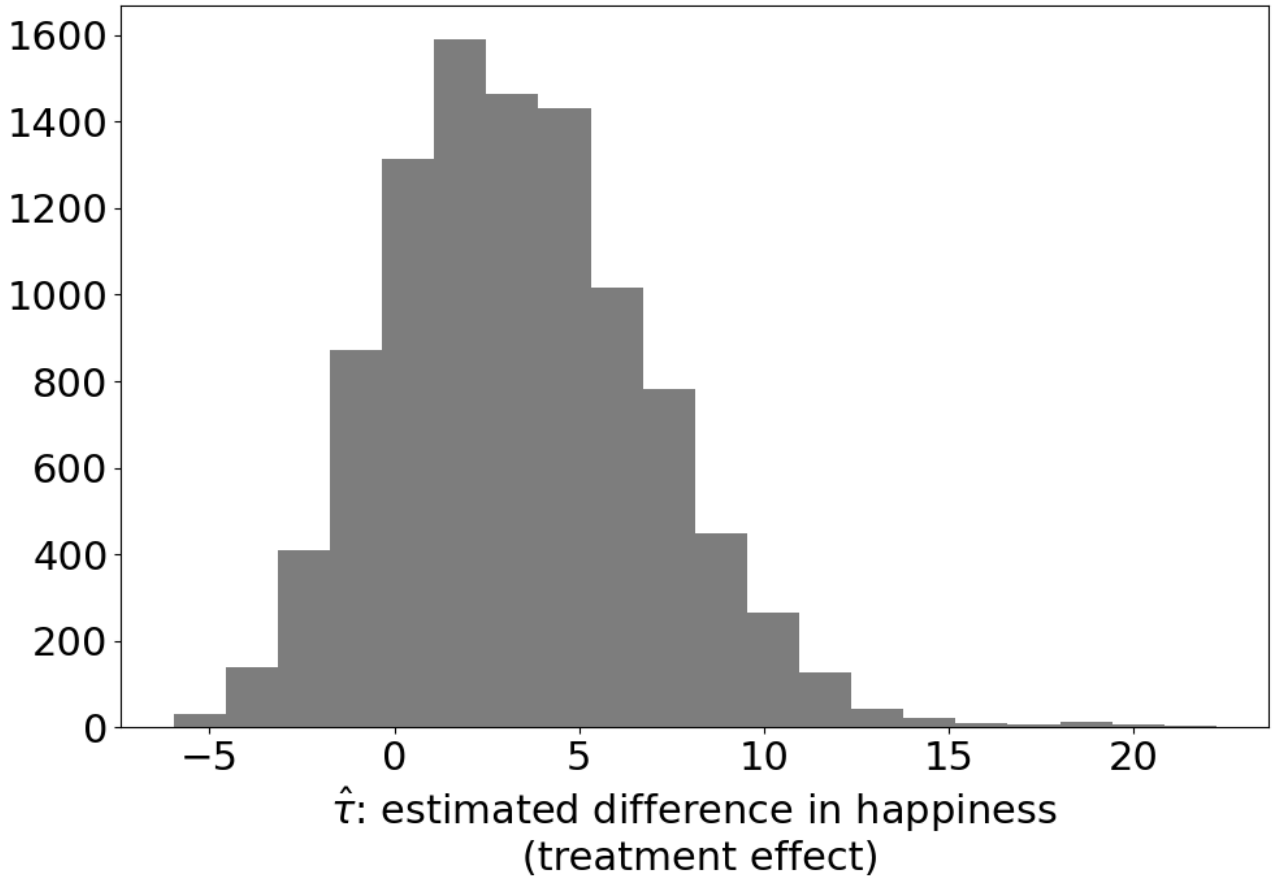
```
rlearner = RLearner(
    nuisance_model_factory=LGBMRegressor,
    propensity_model_factory=LGBMClassifier,
    treatment_model_factory=LGBMRegressor,
    is_classification=False,
    n_variants=2,
)
```

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

quantco

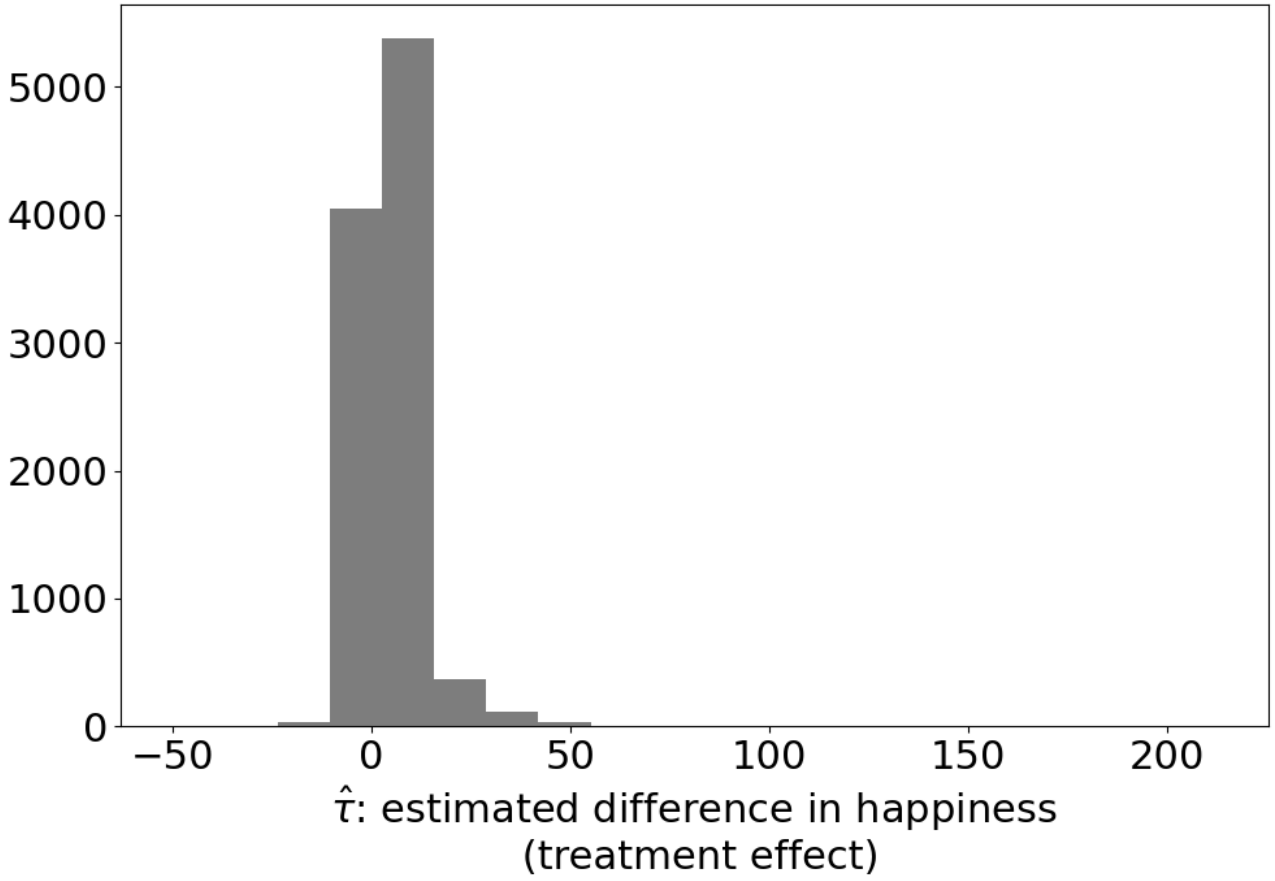
Predicting with a MetaLearner

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



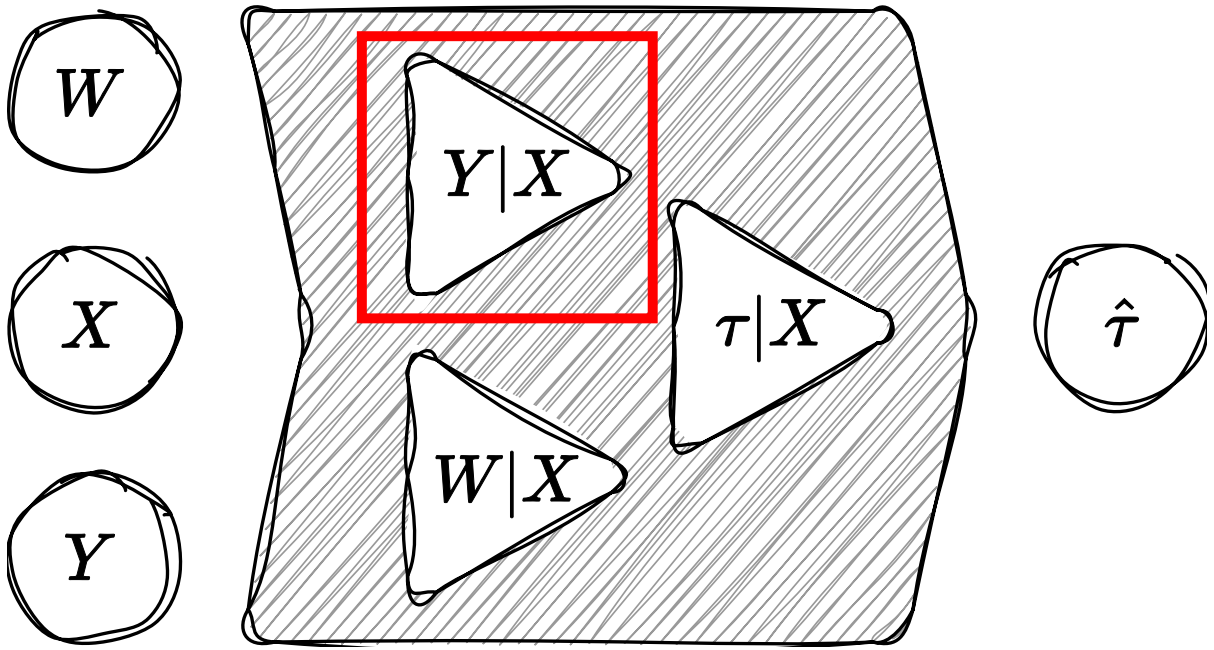
quantco

But what if?



Accessing base models

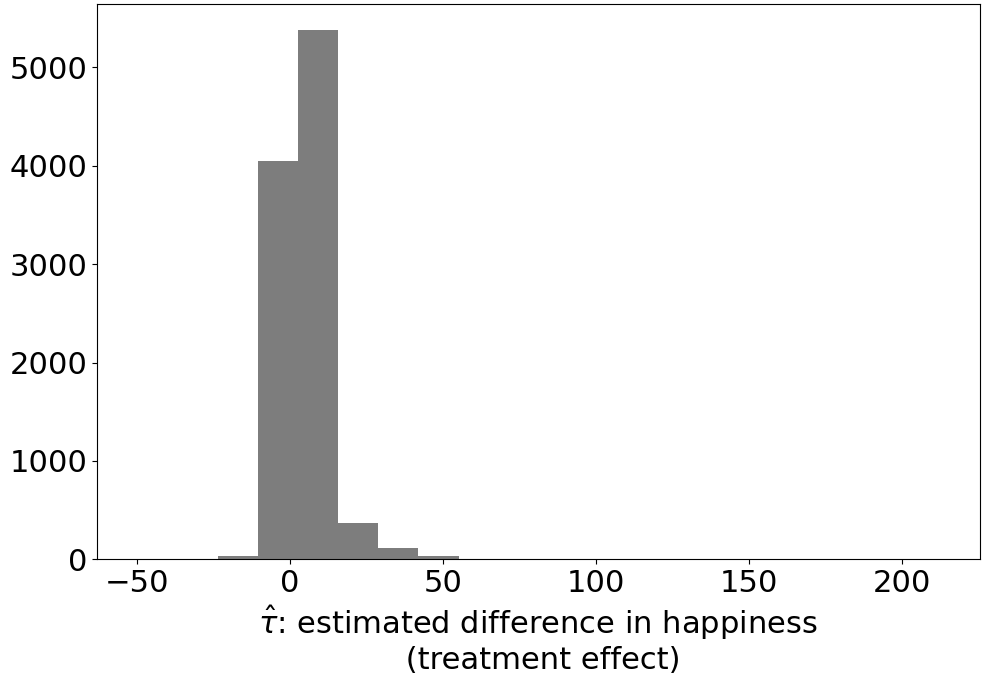
Thanks to the modular design of `metalearners`, we can conveniently isolate the base models to evaluate them:

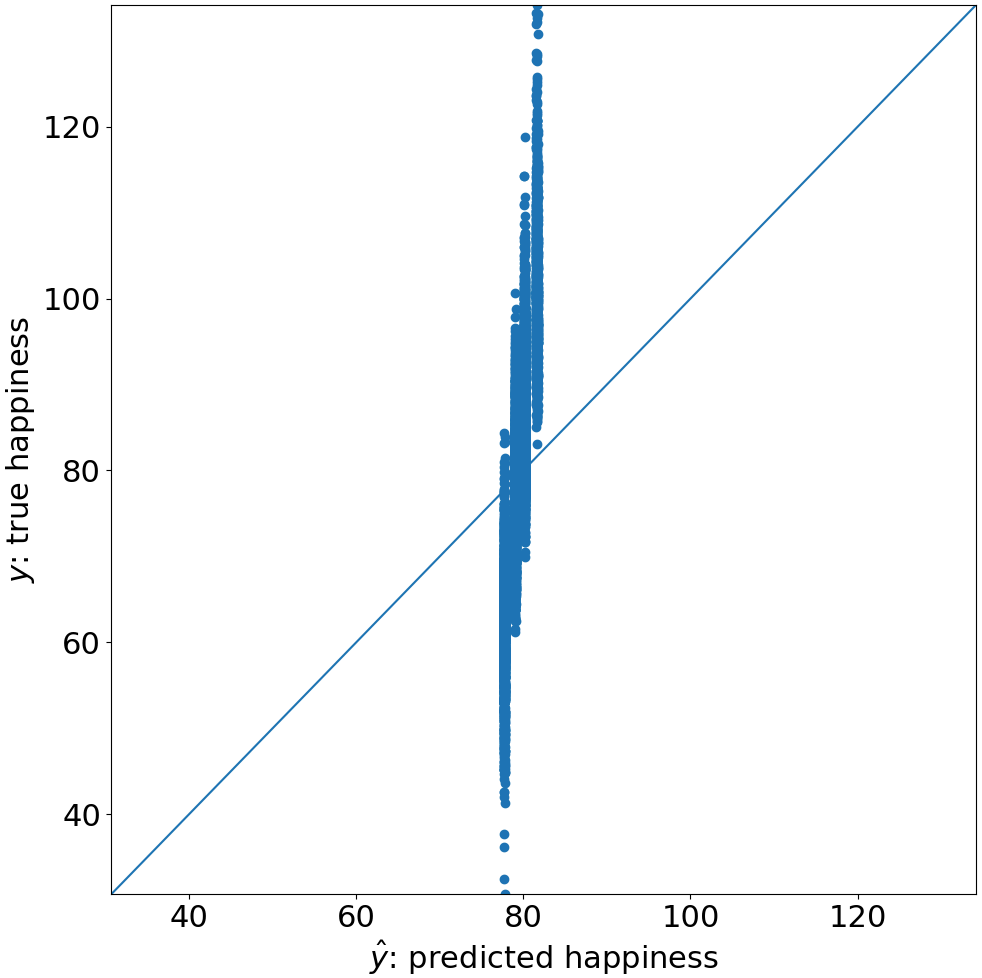



```
outcome_model = rlearner._nuisance_models["outcome_model"]
outcome_estimates = outcome_model.predict(X, is_oos=False)
```

quantco

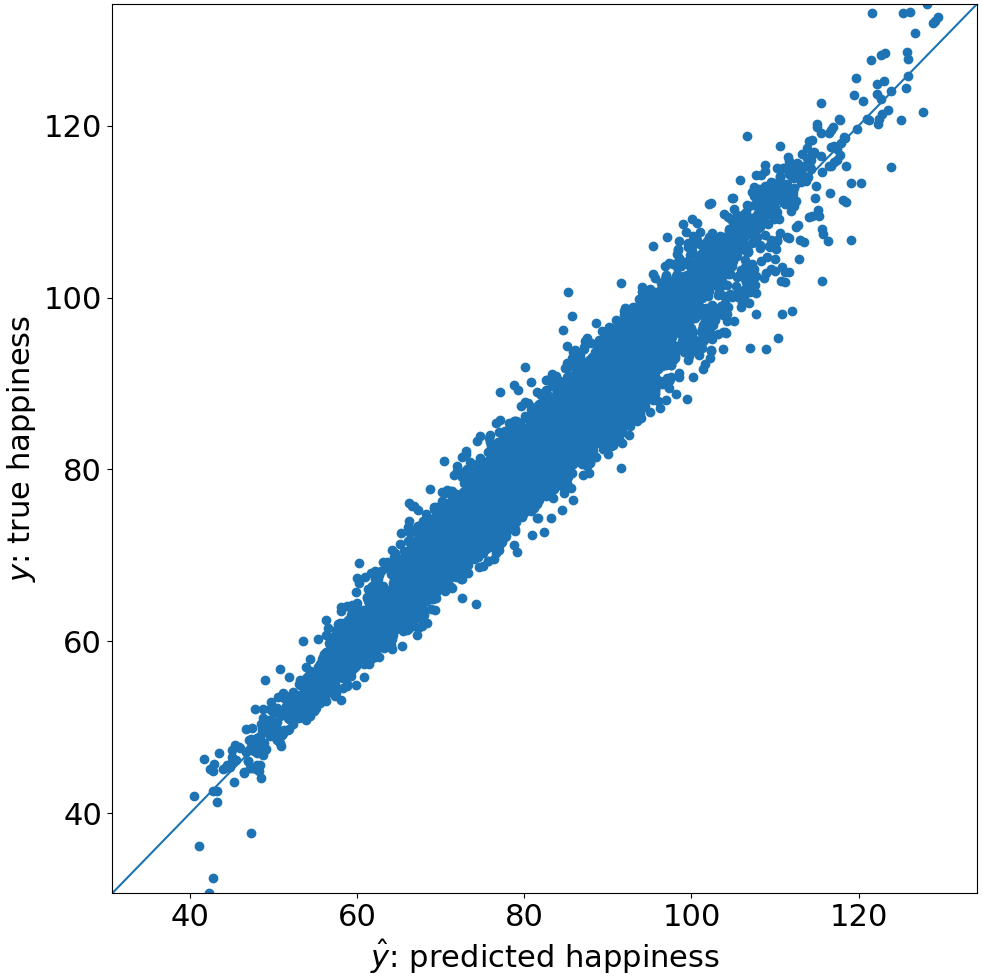
Problem: From CATE model to base model

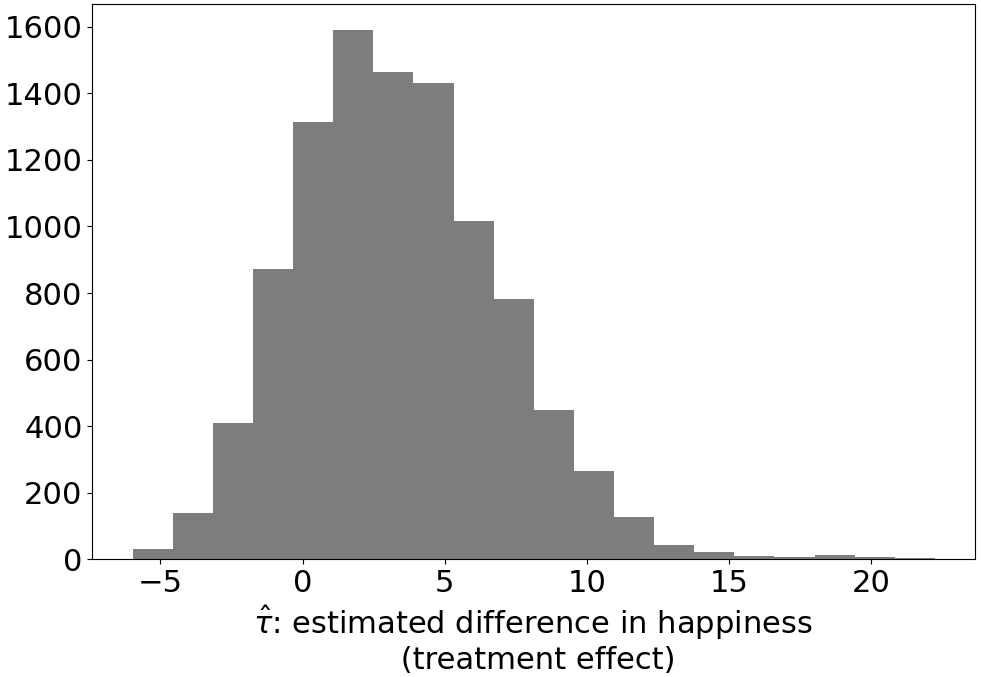




quantco

Solution: From base model to CATE model





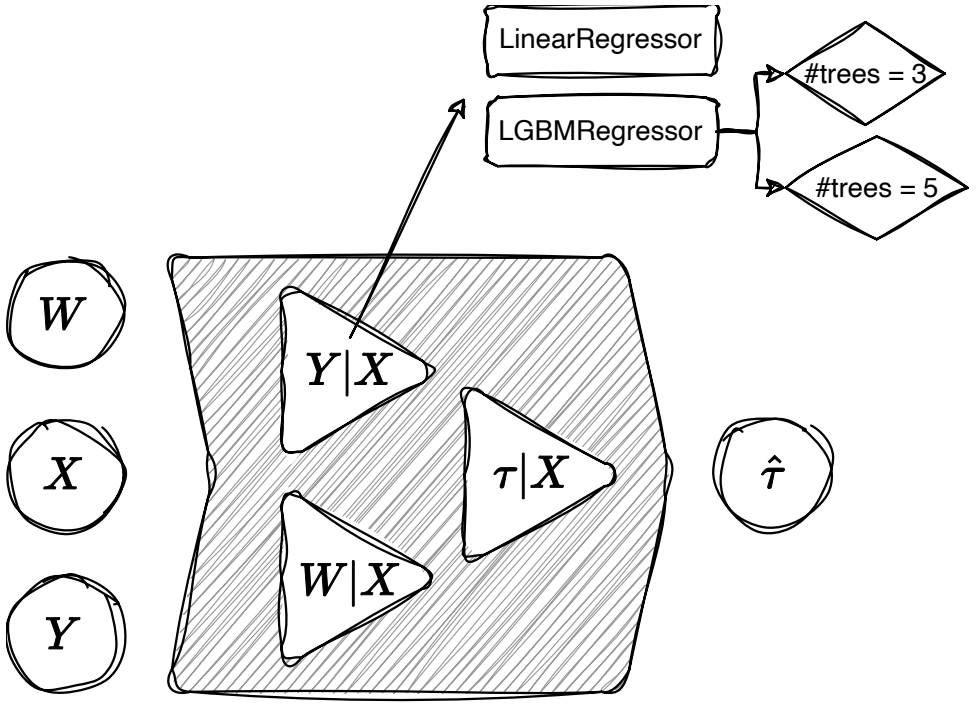
quantco

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
- It is not clear how to evaluate the performance of a CATE estimator

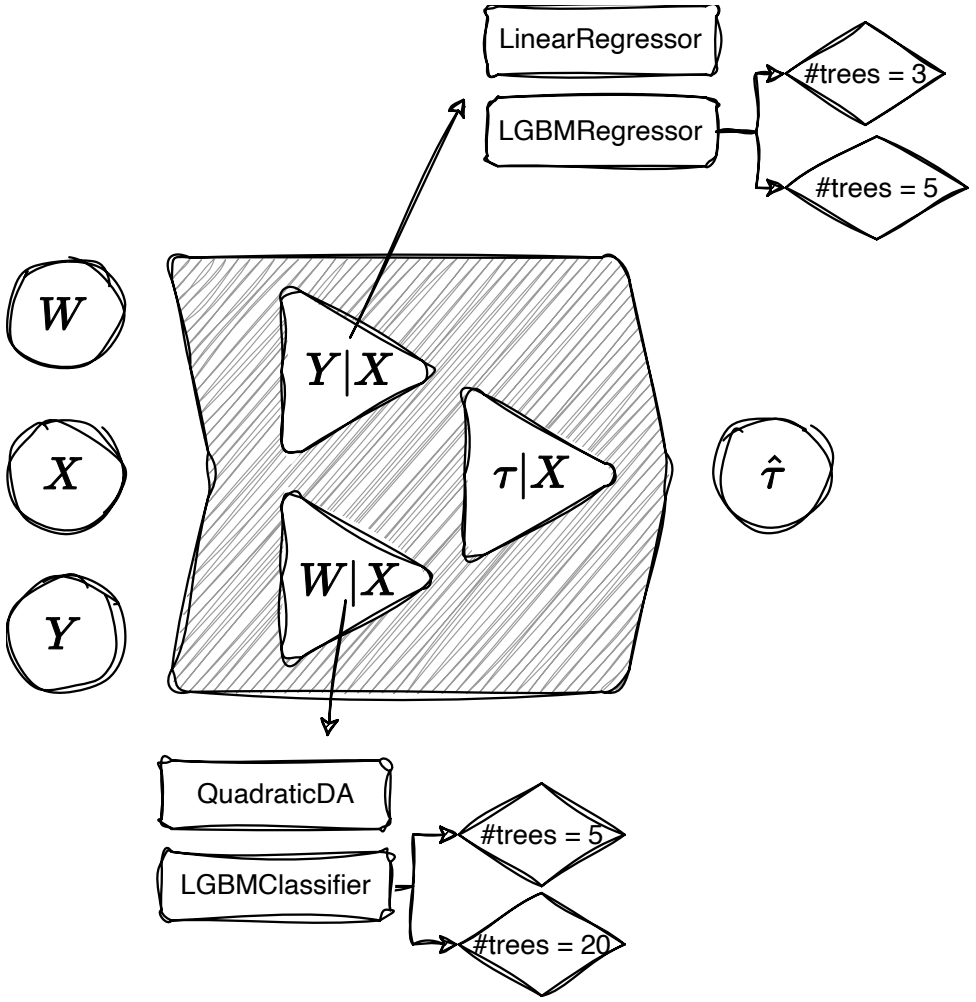
quantco

Performing a grid search



quantco

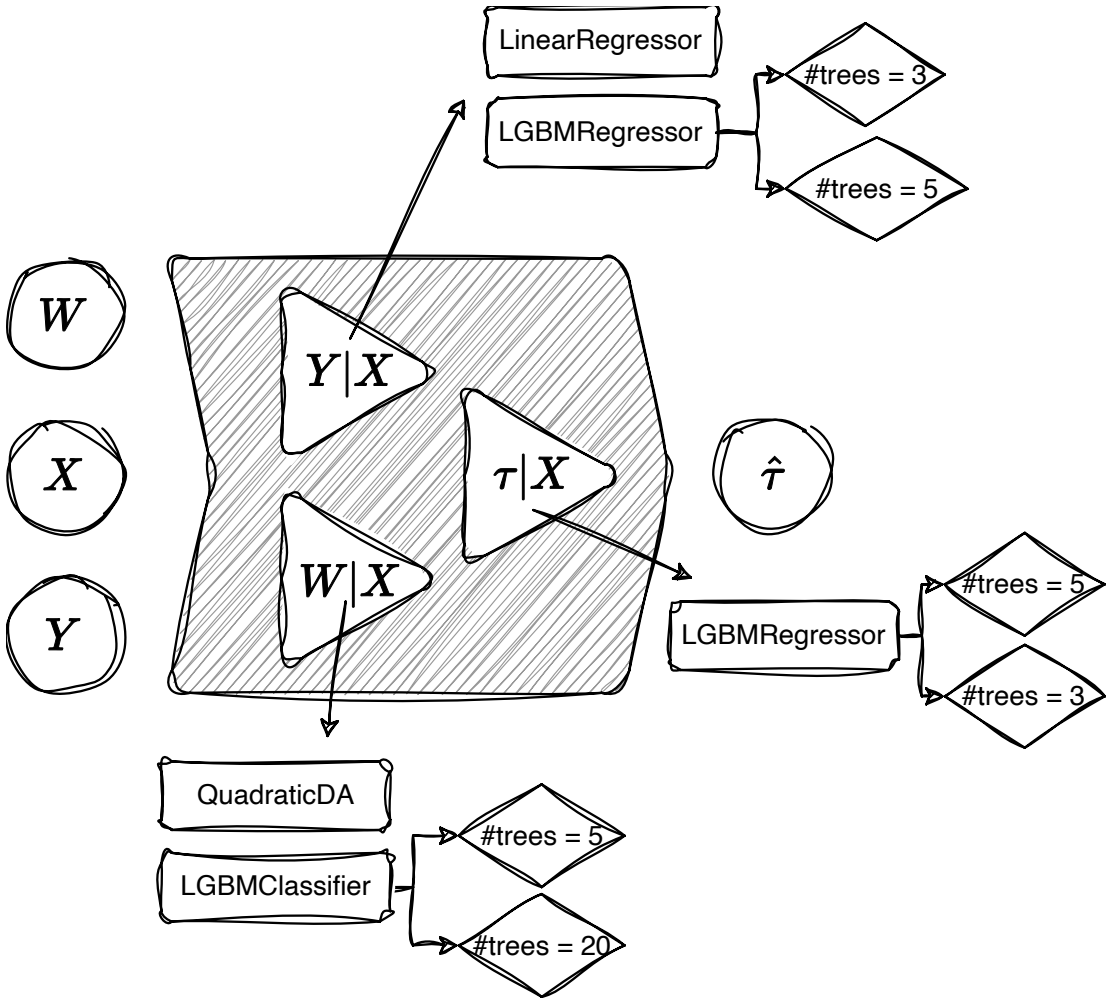
Performing a grid search



Francesc Martí Escofet @fmartiescofet, Kevin Klein @kevkle

quantco

Performing a grid search



quantco

Performing a grid search

```
gs = MetaLearnerGridSearch(
    metalearner_factory=RLearner,
    metalearner_params={"is_classification": False, "n_variants": 2},
    base_learner_grid={
        "outcome_model": [LinearRegression, LGBMRegressor],
        "propensity_model": [LGBMClassifier, QuadraticDiscriminantAnalysis],
        "treatment_model": [LGBMRegressor],
    },
    param_grid={
        "variant_outcome_model": {"LGBMRegressor": {"n_estimators": [3, 5]}},
        "treatment_model": {"LGBMRegressor": {"n_estimators": [3, 5]}},
        "propensity_model": {"LGBMClassifier": {"n_estimators": [5, 20]}},
    },
)
gs.fit(X_train, y_train, w_train, X_validation, y_validation, w_validation)
```

quantco

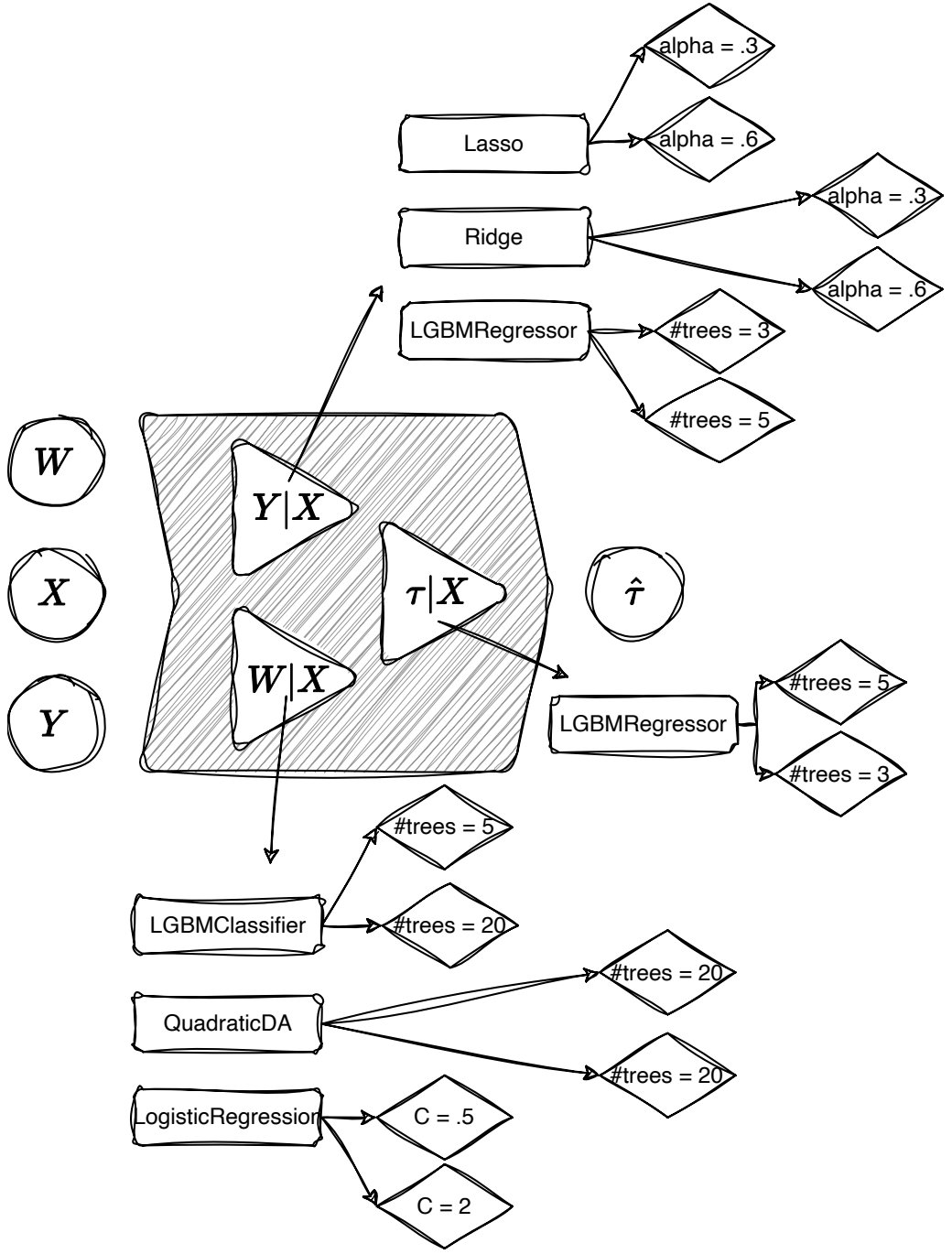
(Naïve) Exploration is expensive

Reuse of **mutually independent base models** could get the number of these base model `fit` calls from

$$\Theta(\#leaves(top) \cdot \#leaves(bottom))$$

down to

$$\Theta(\#leaves(top) + \#leaves(bottom))$$



quantco

Reusing base models

```
from sklearn.linear_model import LinearRegression, LogisticRegression
```

```
rlearner_new = RLearner(
    propensity_model_factory=LogisticRegression,
    treatment_model_factory=LinearRegression,
    fitted_nuisance_models={"outcome_model": outcome_model},
    is_classification=False,
    propensity_model_params={"max_iter": 500},
    n_variants=2,
)
```

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

quantco

Reusing base models across different MetaLearners

```
from metalearners import DRLearner

trained_propensity_model = rlearner._nuisance_models["propensity_model"][0]
```

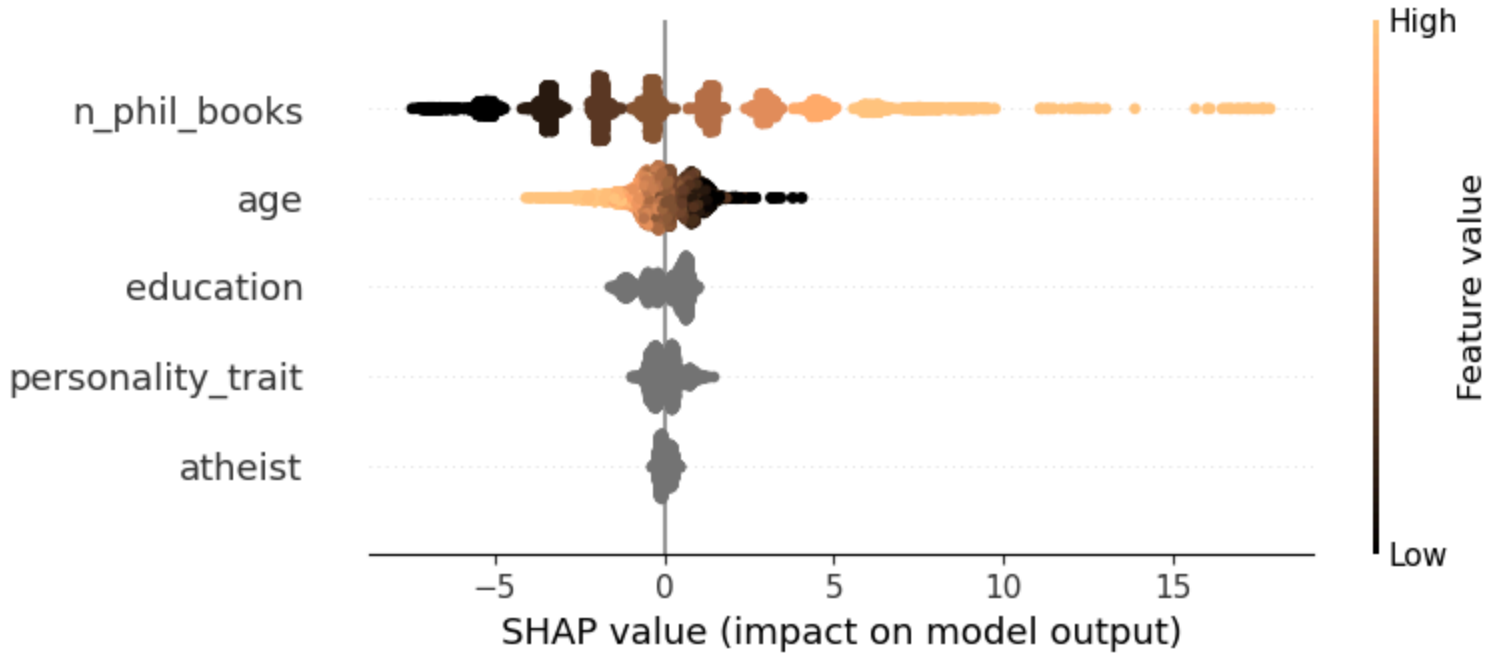
```
drlearner = DRLearner(
    nuisance_model_factory=LGBMRegressor,
    treatment_model_factory=LGBMRegressor,
    fitted_propensity_model=trained_propensity_model,
    is_classification=False,
    n_variants=2,
)
```

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

quantco

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

And much more...

- Integrated with `optuna`, `lime`, `onnx`
- Supports `pandas`, `numpy`, `scipy.sparse`
  - `polars` in the next release

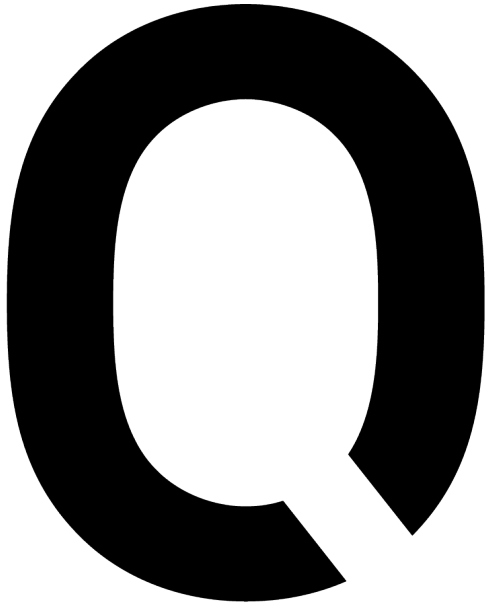
quantco

The Matrix	Blue Pill	Red Pill
Marketing	Gets a voucher	Doesn't get a voucher
Education	Teaching w/ tablet	Teaching /wo tablet
Medicine	Pfizer	Moderna
Fraud detection	Human processing	Automated processing

Please leave feedback on GitHub! :)

github.com/QuantCo/metalearners

github.com/kklein/pdp24-metalearners



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

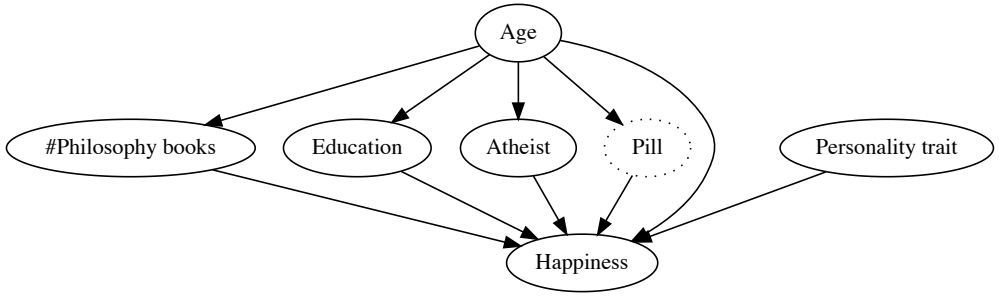
quantco

Backup

quantco

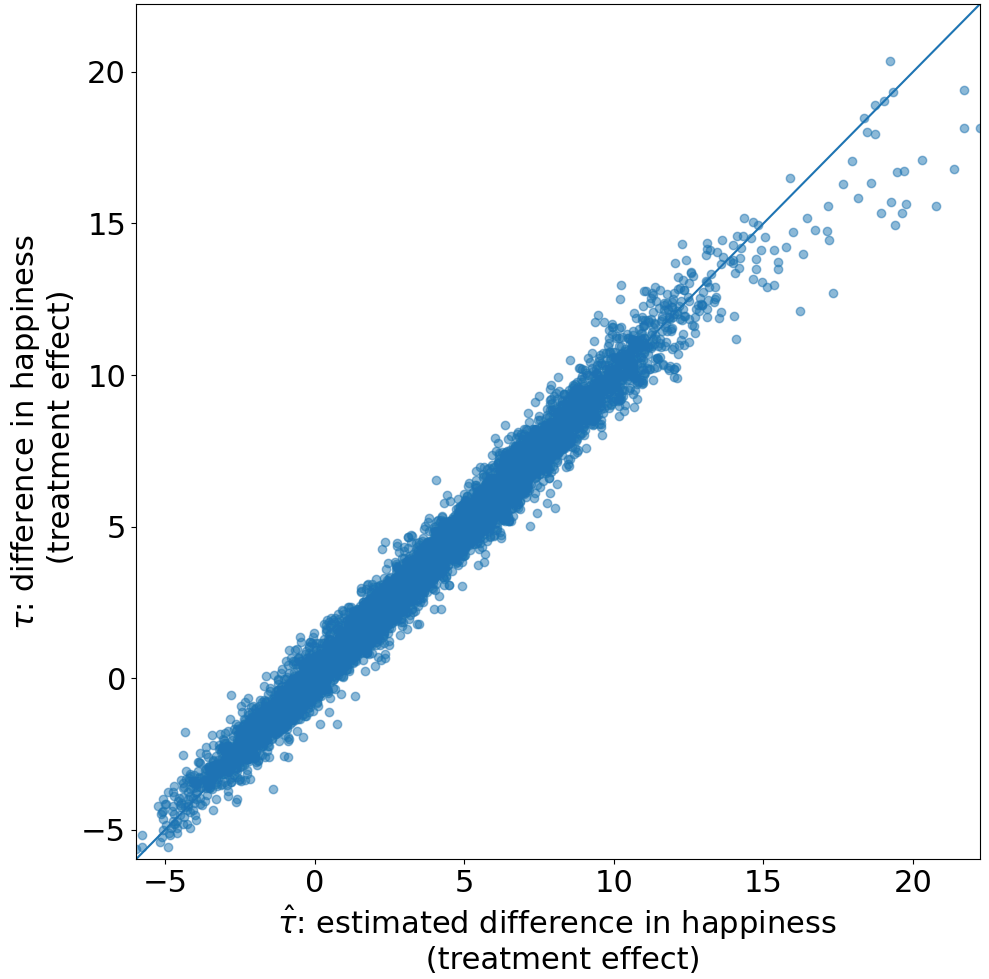
Dataset

- #Observations: 10,000
- Share of blue pill: 49%
- $\rho_{\text{age},p(W|X)}$: 1
- DGP:



quantco

Estimation quality



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.

quantco

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	✗	✓	✓