Supercharging your A/B testing with automated causal inference

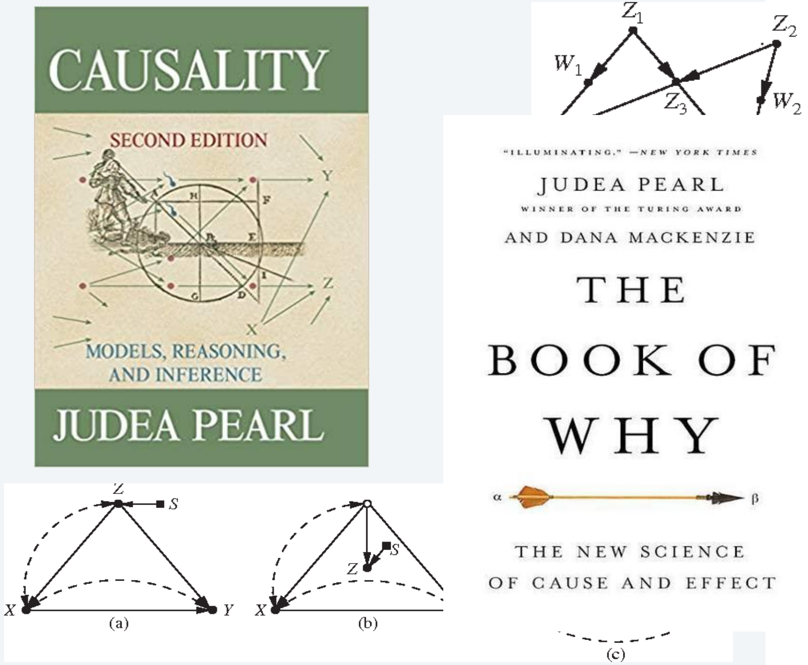
Community Workshop on Microsoft's Causal Tools
May 3, 2022

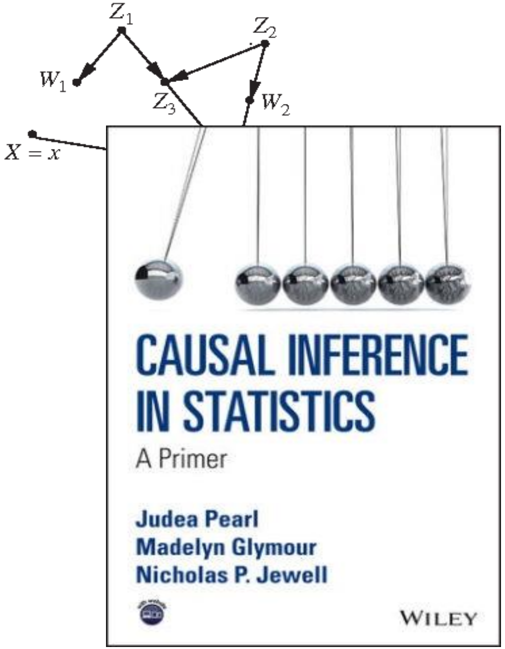
Egor Kraev
Head of AI, Wise Plc
egor.kraev@wise.com

What is causal inference anyway?

- **Causal inference** tries to estimate impacts of **actions**, ideally at a single-observation level
- The fundamental problem when doing that is you can **never directly verify** such estimates at individual level – you can't send and not send the same email to the same customer, to observe the impact!
- **Randomization** when gathering data is helpful for causal inference, but not strictly necessary

Causal inference is a fascinating, deep domain

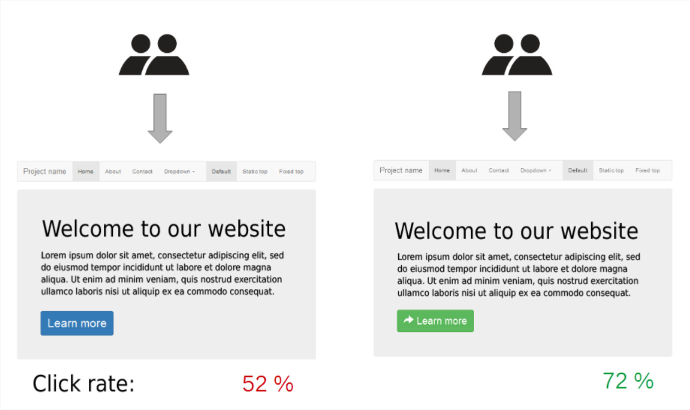




Here, we'll apply it to the simplest case possible

Strengths and weaknesses of classic A/B testing.

Traditional A/B testing



"A/B testing is the golden standard for learning cause and effect"

- **Randomly** split your audience into **test** and **control** groups
- Subject the test group to a '**treatment**', such as sending a marketing email or enabling a new product feature
- Measure the average difference in some '**target**' variable, such as post-treatment revenue, between the two groups
  - Choose sample sizes **large enough** for the difference to be statistically significant

Downsides of traditional A/B testing

- **Treats differences between customers as noise**: if it's harmful to some customers but beneficial to others, we'll just see the zero average impact

- **Wasteful**: collect and process 100Ks of data points just to learn a boolean, or at most **a single number** (average impact)

- **Can be a hard sell to product teams**: if we take the effort to build a feature, we kind of assume it'll be useful – and in any case it’s already built, so what’s the value of the test?

How can causal inference help?

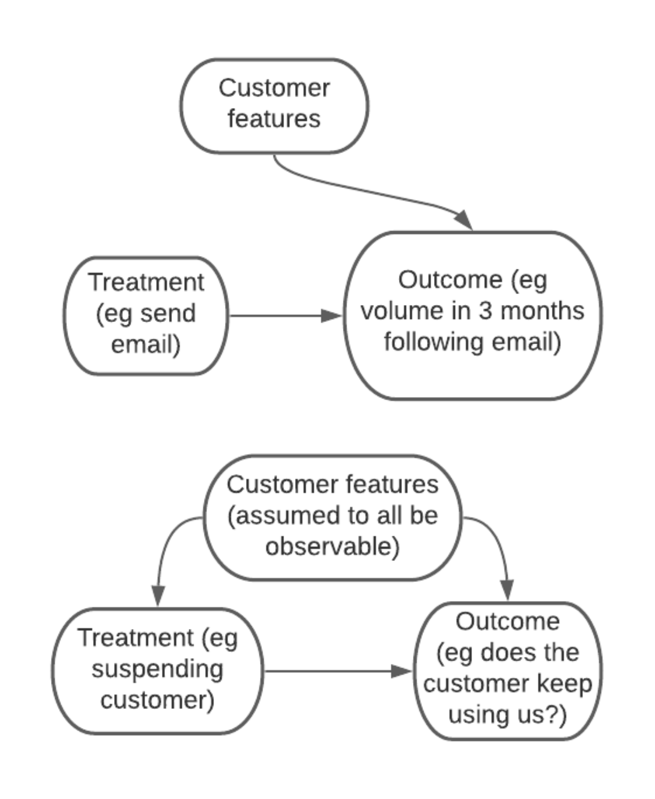
How can causal inference help in A/B testing?

- Causal inference models will estimate impact **per customer** as a function of their features (also known as <u>Conditional</u> Average Treatment Effect, or CATE)
    - Most models also supply **confidence intervals** for those estimates.

- This means you can take the same dataset you collected during A/B testing, enriched with customer features, and get **customer segmentation by impact**

- Thus, **customer heterogeneity** becomes a **valuable information source**, rather than noise to be averaged over
    - This promises smaller sample sizes required for significance

Our scope: “No unobserved confounders”

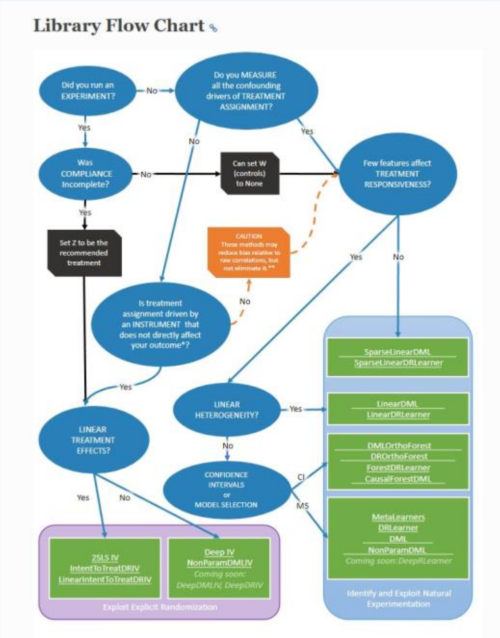


- **Most important example:** we run an A/B/N test and want to better understand its results
- Second option: treatment assignment is **random but biased** based on customer features
- Final class of examples: we want to draw causal conclusions about a situation where we **can’t make an experiment** (eg impact of suspending users), and don’t have any instrumental variables

Comparing causal models.

Which causal model?

- There is a wealth of causal inference packages available, first and foremost **DoWhy/EconML**, also CausalML, UpliftML and several others
    - Inconsistent APIs
    - Each model comes with its own set of quirks
    - Most models require choice of hyperparameters and also of ‘regular’ ML regressors/classifiers as components

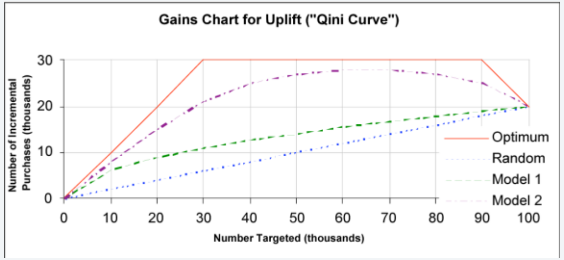


- Most importantly, **how do you compare** different models after fitting?
    - How do you do **out-of-sample** testing for **counterfactual** estimates?

Out of sample scoring of causal models

To compare causal models, we need to score **out of sample**.

We consider two main method families:



1. **Qini/AUC**: cumulative curves for outcomes sorted by estimated impact, similar to ROC for classifiers. Theoretically nicer, but **harder to interpret**
2. **Policy value**, aka **ERUPT** (Estimated Response Under Proposed Treatment): **unbiased estimator of the outcome** if we treated every customer for whom a model predicts (say) a positive CATE: more **interpretable**

$$\hat{\Pi}(d) = \sum_{i=1}^{N} \left(\frac{1 - W_i}{1 - e(X_i)} (1 - d(X_i)) \cdot \pi_i(0) + \frac{W_i}{e(X_i)} d(X_i) \cdot \pi_i(1) \right)$$

We also support the **r-scorer**, but don’t really like its complexity ;)

Image source: N. Radcliffe (2007), formula: Hitsch and Misra (2018) https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3111957

What policy to choose for ERUPT?

- If the treatment effect is **variable**, but **positive for every customer**, the naïve "treat if CATE > 0" policy will involve treating everybody, which makes it **hard to differentiate** between models

- To correct for this, we offer a **normalized ERUPT** score, by adding a treatment cost equal to the average treatment effect - thus forcing the models to compete on predicting **impact variability**

- In real-world problems, you should use a policy that's **as realistic as possible**. For example, "send promotion to customer if predicted impact of a promotion on revenue minus promotion cost is positive"

Automated model selection.

Building blocks for automated model selection

DoWhy

- CATE models:
  - Most **EconML** CATE estimators
  - Transformed Outcome
  - Dummy model: average treatment effect + randomness
  - Easy to add others – let us know if you have favorites!
- **DoWhy** for a consistent high-level interface to individual estimators
- **FLAML regression** for regression component models
- **DummyClassifier** or **FLAML** (user choice) for propensity to treat
- **FLAML**, again, for estimator and hyperparameter search
  - First run all chosen estimators with default settings, then sample

It’s yours to use right now!

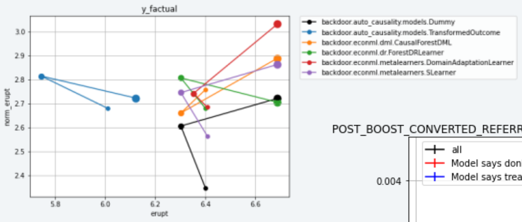
https://github.com/transferwise/auto-causality

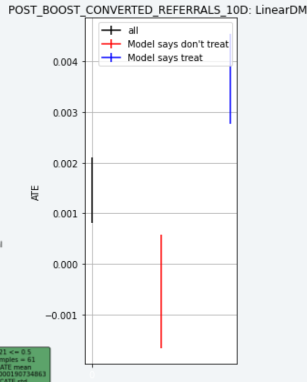
Current state, tested and used:

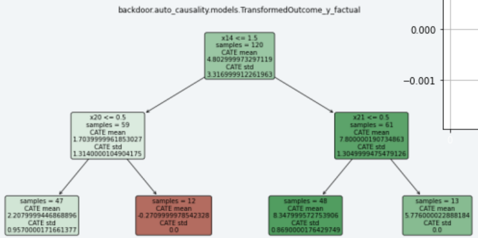
- Extensively tested for binary treatment, random assignment
- Example notebook with the full fitting and analysis cycle

Coming soon (almost works, bar testing and bugfixes):

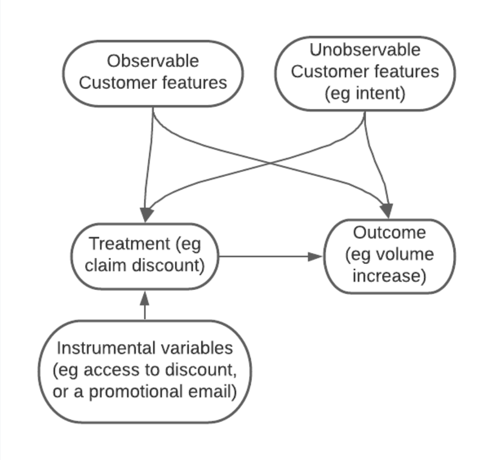
- Multi-valued discrete treatment
- More testing for FLAML classifier option for propensity to treat
- Component models’ time budget as part of the search space
- Reuse of component models across fits, for efficiency







Next steps



Next big milestone (planned for Summer 2022):

- Extension to EconML instrumental variable models

Could use some help from the EconML team:

- Review of hyperparameter search spaces for the EconML models
- OrthoForest inference on 500K+ points doesn't finish after running for days, although training only takes tens of minutes – is that expected?
- Early stopping option for tree-based models
- What are good ways to score instrumental variable models out of sample?

Causal inference in Wise.

A lot of analytics questions are questions about causality

In progress:

- Analysis of CRM campaign impacts
- **Improved targeting of referral reward programs**

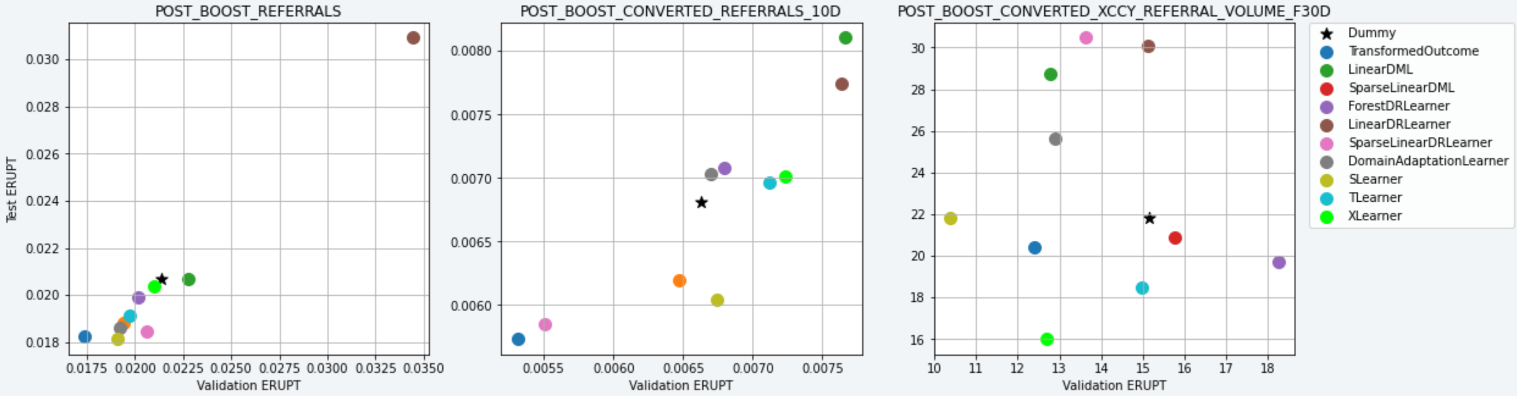
Planned:

- Improved analysis of A/B test results for product features
- Impact of customer support turnaround time on retention and future volumes

Extra referral rewards impact analysis

- 470K customers in the sample, some kind of extra reward for referring new customers was offered to 360K of them (all grouped into one ‘treatment’ here for simplicity)
- Simulations run on an r5a.4xlarge (128GB RAM)
- Component model time budget was set to 10min, total time budget to 3h
- Features include reward base currency, customer ‘age’, host’s recent transaction volume

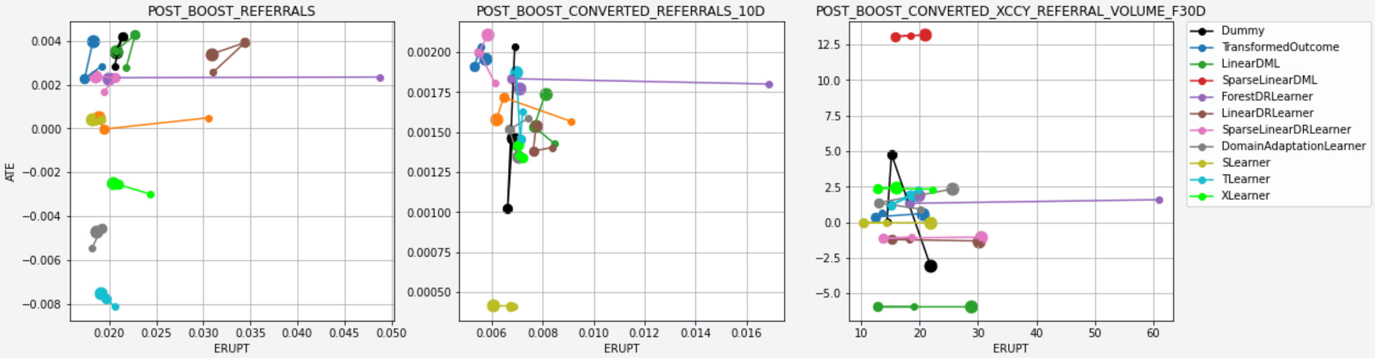
Quite variable out-of-sample model performance depending on model and target:



Results continued

- Different models' ATE estimates are all over the place
- DML and Sparse flavors most likely to generate wildly off ATE estimates occasionally (excluded from graphs for readability)
- ForestDRLearner tends to overfit the training set (need early stopping?)

For POST_BOOST_REFERRALS, best model's ERUPT and ATE estimates are inconsistent

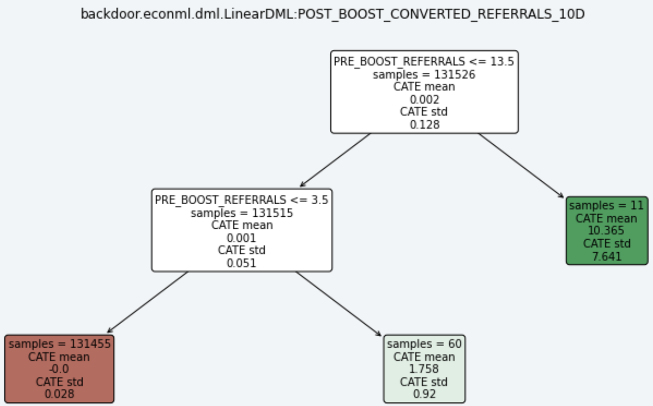


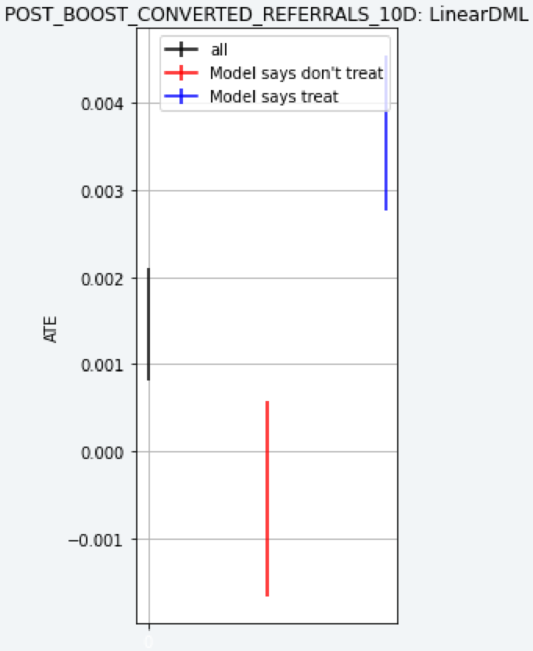
ERUPT vs ATE for training, validation, and test sets (denoted by marker size from smallest to largest)

Out-of-sample segmentation

Let's compare the estimated treatment effects in the **test set** split by the **best model's recommendations:**

On the other hand**, the simple tree policy derived from the model's recommendations is not very useful**





Final approach

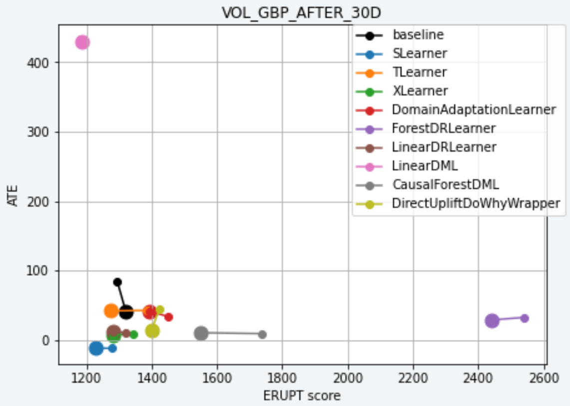
- The **number of converted referrals** post-boost was the best-modeled target, choose that to train a causal inference model

- Augment it with a **conditional** model of how much volume a new customer would do with us **if converted,** given the boost program the host was offered and the other host's features, to **quantify a reward program's payoff**.
  No causal inference needed for this stage, just regular regression on the much smaller dataset of hosts who had at least one post-boost conversion

- Re-run optimization with **policy based on the conditional model**

Lessons learned

- Consider **different target variables** as well as **different models**
- Check **consistency** between **ERUPT and ATE** estimates, and between **validation and test performance**
- Choose your **metric** wisely
- **Need quite large instances** to run the fits
  - Now looking at using Ray to parallelize
- **Beware of causality leakage!**
  If treatment is correlated with features, but you use a naïve propensity-to-treat model, can get GREAT out-of-sample scores that aren't real



Conclusion.

So how can causal inference supercharge your A/B testing?

- **Use same A/B testing data**, enriched with customer features
- Estimate causal impact as function of features, allowing you to **segment customers by impact**
- Learn fine-grained impact also from **biased** random sampling, allowing you to **optimize** and **keep learning** at the same time
  - For example, could do a kind of **Thompson sampling on customer segments**
- Thanks to the magic of **DoWhy**, **EconML**, and **FLAML**, combined in **auto-causality**, you can do all this with **no prior expertise** in causal inference

Many thanks to

Pere Planell Morell
Mark Harley
Timo Flesh
Guy Durant
Wen Kho
Ziyang Zhang

For helping to bring the package this far!

Questions?

egor.kraev@wise.com