Lecture 16: Big data and machine learning

PPHA 34600
Prof. Fiona Burlig

Harris School of Public Policy
University of Chicago

From last time: fuzzy regression discontinuity

As usual, we'd like to run:

$$Y_i = \alpha + \tau D_i + \varepsilon_i$$

The regression discontinuity:

- Suppose D_i is determined by whether or not X_i lies above a cutoff, c
- **Idea**: Having X_i just above or just below c is as good as random...
- ... And there is a discontinuous change in D_i as a result of crossing c

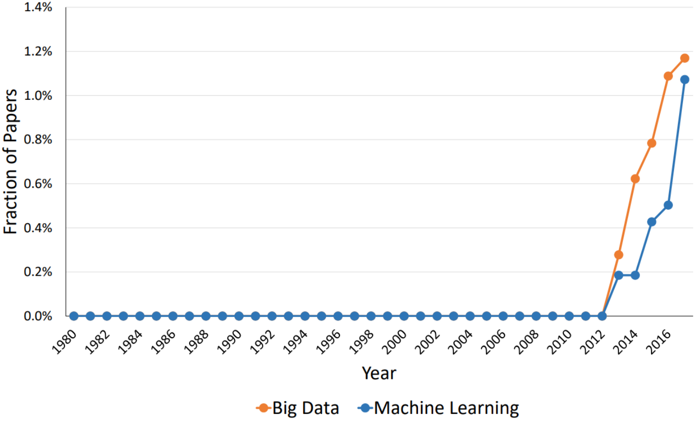
$\rightarrow$ We can compare Y_i for units with X_i just above c to Y_i for units with X_i just below c

- With incomplete changes in D_i from $X_i < c$ to $X_i \geq c$:

$$D_i = \alpha + \gamma \mathbf{1}[X_i \geq c] + f(X_i) + \varepsilon_i \text{ for } c - h < X_i < c + h$$

$$Y_i = \alpha + \tau \hat{D}_i + f(X_i) + \varepsilon_i \text{ for } c - h < X_i < c + h$$

And now for something completely different...



How big is Big?

A computer scientist might say:

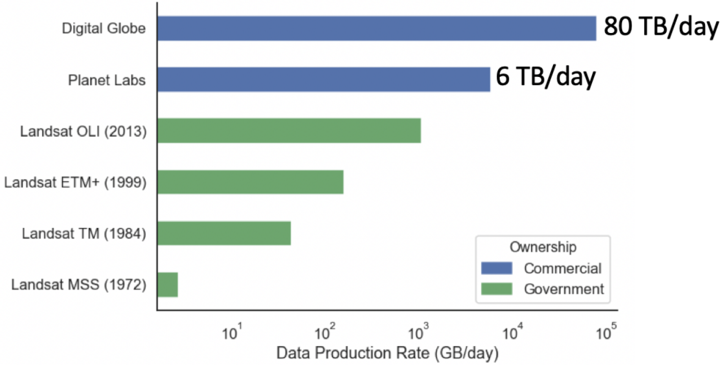
- "Too big to fit on your computer"

An economist might say:

- "I dunno, 15 GB?"

$\rightarrow$ All of this is a bit fuzzy

Big data are getting bigger every day

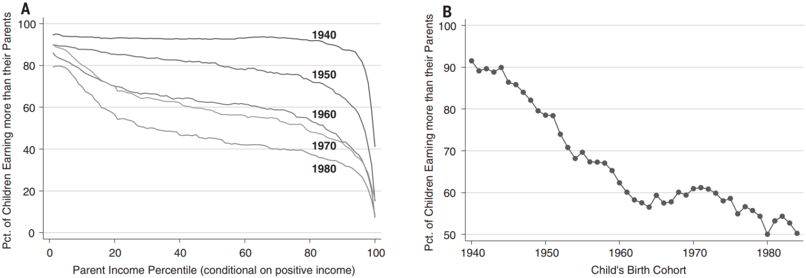


Thinking outside of the box to get new data

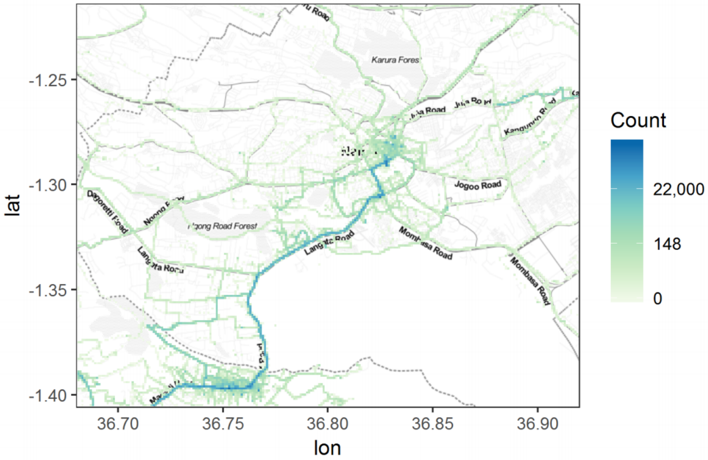
The era of Big Data isn't just good because of small standard errors:

- New data collection methods present opportunities
- We can study previously unanswerable questions
- Sometimes this requires getting a bit creative

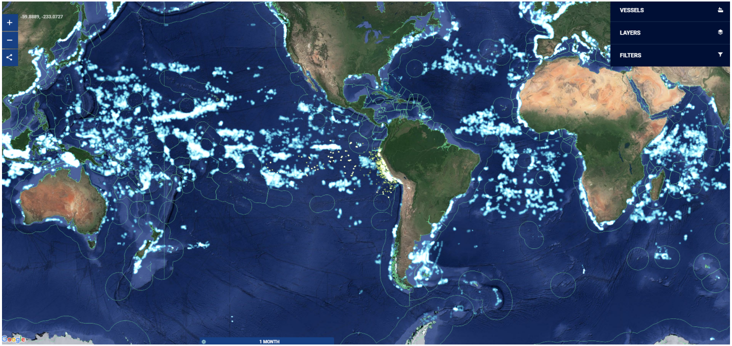
Thinking outside of the box to get new data



Thinking outside of the box to get new data



Thinking outside of the box to get new data



With Big Data come big responsibility

(Especially) with Big Data, we have to be careful:

- Just because a dataset is large doesn't mean it's unbiased
- Large data can also have errors
- And come with additional concerns too (privacy, etc)

With Big Data come big responsibility

(Especially) with Big Data, we have to be careful:

- Just because a dataset is large doesn't mean it's unbiased
- Large data can also have errors
- And come with additional concerns too (privacy, etc)

→ **It's important to understand what we're using**

With Big Data come big responsibility

(Especially) with Big Data, we have to be careful:

- Just because a dataset is large doesn't mean it's unbiased
- Large data can also have errors
- And come with additional concerns too (privacy, etc)

$\rightarrow$ **It's important to understand what we're using**

$\rightarrow$ (The following slides owe credit to Tamma Carleton)

With Big Data come big responsibility

We typically interact with three types of data:

1. Raw, out of the source
2. Processed "in house"
3. Processed "out of the house"

→ All of these data can be used as Y, D, or X (or even Z)

→ Each has its own pros and cons

Raw data

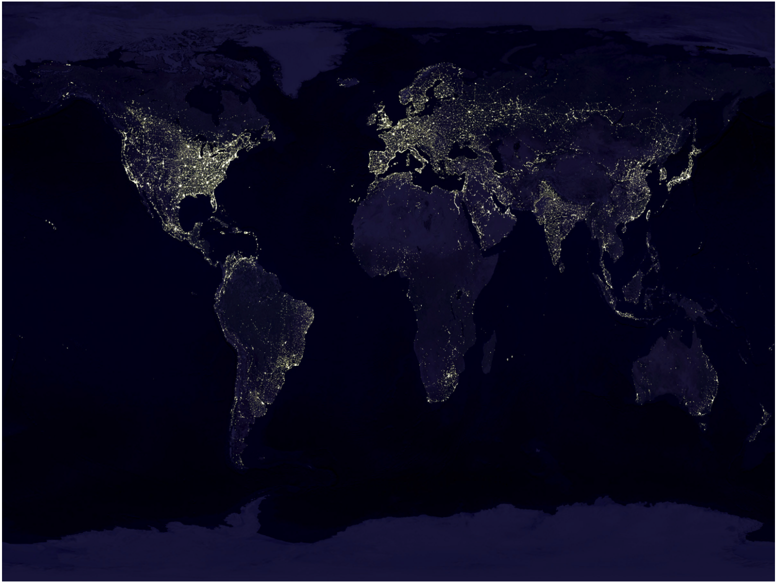
Major pros of raw data:

- We know what we're dealing with
- We get a fighting chance to understand measurement error and bias

Major cons of raw data:

- Raw data are often not exactly what we want
- We have to be careful when we use them as a proxy

Raw data



Processed in house

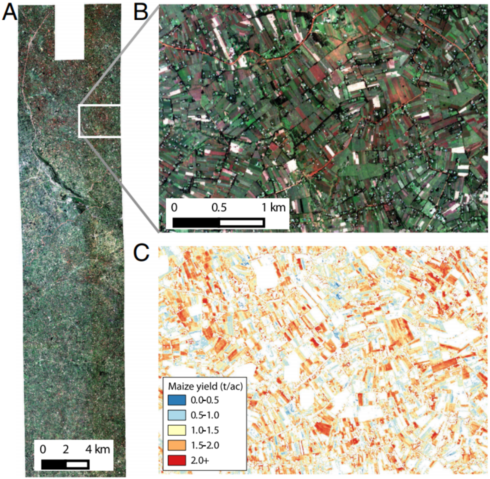
Major pros of home-grown data:

- We know what we're dealing with
- We get a fighting chance to understand measurement error and bias

Major cons of home-grown data:

- This takes a lot of time and effort
- And we don't always have the right toolkit

Processed in house



Processed out of the house

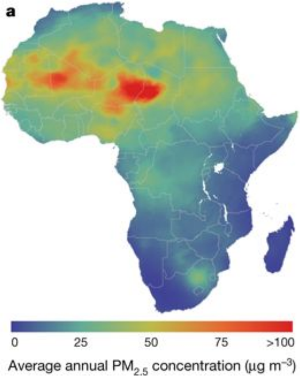
Major pros of outsourced data:

- We leverage external expertise
- We potentially have less measurement error than the in-house version
- This is a lot less work than the

Major cons of outsourced data:

- We don't know exactly what we're measuring
- We can't look "under the hood" to uncover bias

Processed out of the house



Big Data wrap-up

Just like with small data, you need to know what you've got:

- Big Data allow for new possibilities
- But require additional processing tools and time
- A careful combination of in-house and out-of-house work can yield benefits

Estimation vs. prediction

This class has been about asking:

→ What is the causal effect of D on Y?

→ Aka, in

$$Y_i = \alpha + \tau D_i + \varepsilon_i$$

what is $\hat{\tau}$?

- Focus is on **unbiasedness**

Estimation vs. prediction

This class has been about asking:

→ What is the causal effect of D on Y?

→ Aka, in

$$Y_i = \alpha + \tau D_i + \varepsilon_i$$

what is $\hat{\tau}$?

- Focus is on **unbiasedness**

Machine learning instead asks:

→ What is the best guess of some outcome?

→ What is $\hat{Y}$?

- Want to consider a **bias-variance tradeoff**

Estimation vs. prediction

An estimation problem:

"What is the causal effect of my rain dance on rainfall today?"

$\rightarrow$ Estimation problem: rain dances (maybe) affect rainfall

Estimation vs. prediction

An estimation problem:

"What is the causal effect of my rain dance on rainfall today?"

$\rightarrow$ Estimation problem: rain dances (maybe) affect rainfall

A prediction problem:

"Do I need an umbrella today?"

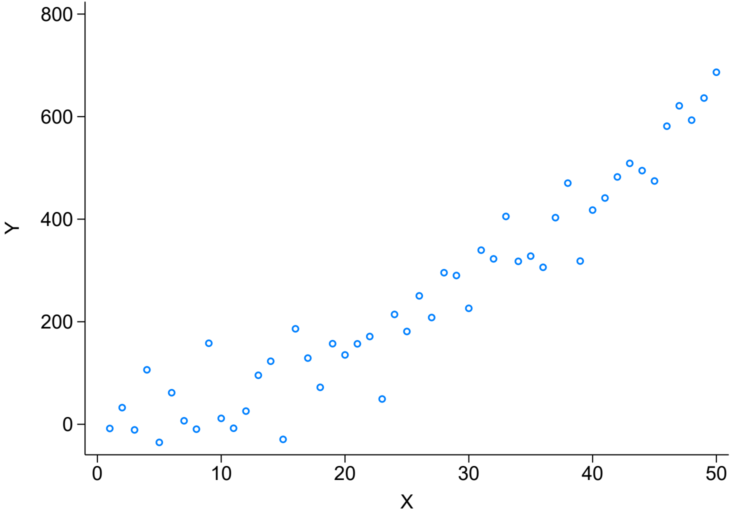
$\rightarrow$ Prediction problem: rainfall doesn't depend on umbrellas

Basics of machine learning

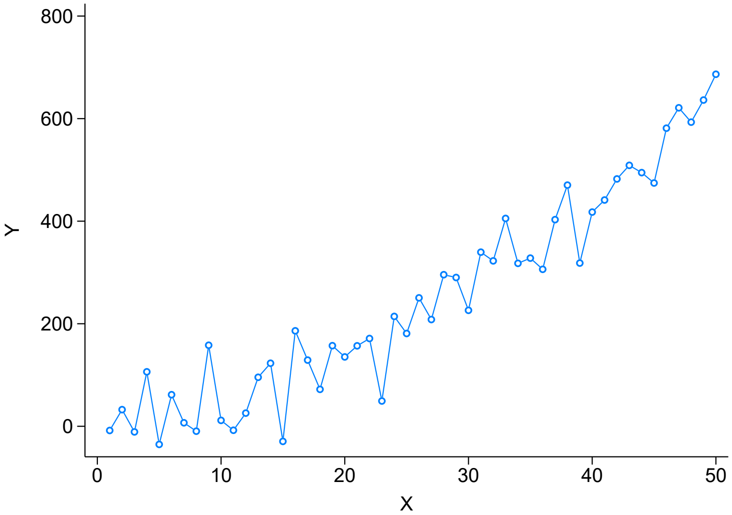
Machine learning is just methods trying to generate predicions:

- Given a dataset with outcome Y and covariates $\mathbf{X}$, what function $f(\mathbf{X})$ best predicts Y?
- Note the difference between this and causal inference!

Predicting $\hat{Y}$



Predicting $\hat{Y}$



Predicting $\hat{Y}$ without overfitting

We want to come up with a good estimate of $\hat{Y}$...
... But we need to watch out for overfitting!

Predicting $\hat{Y}$ without overfitting

We want to come up with a good estimate of $\hat{Y}$...
... But we need to watch out for overfitting!

Machine learning typically uses three steps:

1. **In-sample prediction**:
    - Use an algorithm to generate the best in-sample prediction
    - Many ways to do this, but imagine running thousands of OLS regressions

Predicting $\hat{Y}$ without overfitting

We want to come up with a good estimate of $\hat{Y}$...
... But we need to watch out for overfitting!

Machine learning typically uses three steps:

1. **In-sample prediction**:
   - Use an algorithm to generate the best in-sample prediction
   - Many ways to do this, but imagine running thousands of OLS regressions
2. **Cross-validation**:
   - Instead of using the whole sample for step (1)...
   - Split the sample into pieces...
   - Do step (1) on one part, and predict $\hat{Y}$ on the other part
   - Record how well the model fits (eg $Y - \hat{Y}$)

Predicting $\hat{Y}$ without overfitting

We want to come up with a good estimate of $\hat{Y}$...
... But we need to watch out for overfitting!

Machine learning typically uses three steps:

1. **In-sample prediction**:
   - Use an algorithm to generate the best in-sample prediction
   - Many ways to do this, but imagine running thousands of OLS regressions
2. **Cross-validation**:
   - Instead of using the whole sample for step (1)...
   - Split the sample into pieces...
   - Do step (1) on one part, and predict $\hat{Y}$ on the other part
   - Record how well the model fits (eg $Y - \hat{Y}$)
3. **Repeat**:
   - Do this several times over different sample splits
   - Pick the final model that does best

The in-sample prediction step

It's worth unpacking this a bit further:

- **Goal:** Produce the best guess at $\hat{f}(\mathbf{X})$
- This typically involves being very flexible: interactions between Xs
- We know we want to avoid over-fitting
- Just running a ton of OLS regressions is a slow way to do this

The in-sample prediction step

Function class $\mathcal{F}$ (and its parametrization)	*Regularizer $R(f)$*
Global/parametric predictors	
Linear $\beta' x$ (and generalizations)	Subset selection$\|\|\beta\|\|_0 = \sum_{j=1}^{k} 1_{\beta_j \neq 0}$
	LASSO $\|\|\beta\|\|_1 = \sum_{j=1}^{k} \|\beta_j\|$
	Ridge $\|\|\beta\|\|_2^2 = \sum_{j=1}^{k} \beta_j^2$
	Elastic net $\alpha\|\|\beta\|\|_1 + (1-\alpha)\|\|\beta\|\|_2^2$
Local/nonparametric predictors	
Decision/regression trees	Depth, number of nodes/leaves, minimal leaf size, information gain at splits
Random forest (linear combination of trees)	Number of trees, number of variables used in each tree, size of bootstrap sample, complexity of trees (see above)
Nearest neighbors	Number of neighbors
Kernel regression	Kernel bandwidth
Mixed predictors	
Deep learning, neural nets, convolutional neural networks	Number of levels, number of neurons per level, connectivity between neurons
Splines	Number of knots, order
Combined predictors	
Bagging: unweighted average of predictors from bootstrap draws	Number of draws, size of bootstrap samples (and individual regularization parameters)
Boosting: linear combination of predictions of residual	Learning rate, number of iterations (and individual regularization parameters)
Ensemble: weighted combination of different predictors	Ensemble weights (and individual regularization parameters)

A few cautions with ML models

ML is a great tool, but we have to be careful!

- **Do NOT try to interpret the function!**
- The ML model gives you $\hat{Y}$...but *not* $\hat{\tau}$!
- We're not recovering causal effects
- And we don't get standard errors
- And the models are typically unstable

Using machine learning for causal inference

Machine learning is not designed for $\hat{\tau}$:

- We can't directly use ML for what we want to estimate
- But does this mean ML is useless for us?

Using machine learning for causal inference

Machine learning is not designed for $\hat{\tau}$:

- We can't directly use ML for what we want to estimate
- But does this mean ML is useless for us?

$\rightarrow$ **No.**

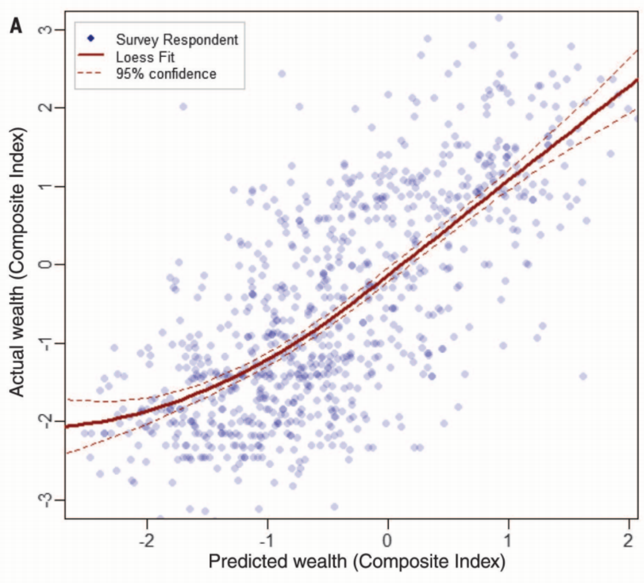
$\rightarrow$ We just need to be a little bit creative!

Using machine learning for causal inference

There are three main ways to use ML for causal inference:

1. **Data generation**
2. **Heterogeneity analysis**
3. **Estimating $\hat{\tau}$**

ML for data generation



ML for heterogeneity analysis

We often want to find heterogeneous effects:

- The FPCI gets in our way for this too
- We need to compare treated vs. untreated units...
- ... *conditional* on $X_i = x$
- But there might be many X_i
- And they might even be continuous

ML for heterogeneity analysis

We often want to find heterogeneous effects:

- The FPCI gets in our way for this too
- We need to compare treated vs. untreated units...
- ... *conditional* on $X_i = x$
- But there might be many X_i
- And they might even be continuous

$\rightarrow$ Which X_i have interesting heterogeneity in $\tau_i(X_i)$?

ML for heterogeneity analysis

Reframe this into a prediction question:

- What is predicted $\hat{Y}_i(X_i)$?
- That is, which X_is give you *different* $\hat{Y}_i$?
- For this to be the same as heterogeneity in $\tau_i(X_i)$...
- ... we need random assignment to treatment
- Under random assignment, there is a 1:1 mapping between $\hat{Y}_i$ and $\hat{\tau}_i$

ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample

ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample
2. Training sample only: Predict $\hat{Y}$ for all units

ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample
2. Training sample only: Predict $\hat{Y}$ for all units
3. Split into X_i groups, looking for the largest difference in $\hat{Y}_i$

ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample
2. Training sample only: Predict $\hat{Y}$ for all units
3. Split into X_i groups, looking for the largest difference in $\hat{Y}_i$
4. Split into X_i subgroups, looking for the largest difference in $\hat{Y}_i$

ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample
2. Training sample only: Predict $\hat{Y}$ for all units
3. Split into X_i groups, looking for the largest difference in $\hat{Y}_i$
4. Split into X_i subgroups, looking for the largest difference in $\hat{Y}_i$
5. Split into X_i subsubgroups, looking for the largest difference in $\hat{Y}_i$

ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample
2. Training sample only: Predict $\hat{Y}$ for all units
3. Split into X_i groups, looking for the largest difference in $\hat{Y}_i$
4. Split into X_i subgroups, looking for the largest difference in $\hat{Y}_i$
5. Split into X_i subsubgroups, looking for the largest difference in $\hat{Y}_i$
6. Stop splitting based on some rule

ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample
2. Training sample only: Predict $\hat{Y}$ for all units
3. Split into X_i groups, looking for the largest difference in $\hat{Y}_i$
4. Split into X_i subgroups, looking for the largest difference in $\hat{Y}_i$
5. Split into X_i subsubgroups, looking for the largest difference in $\hat{Y}_i$
6. Stop splitting based on some rule
7. Testing sample only: using the identified groups, estimate $\hat{\tau}_g$

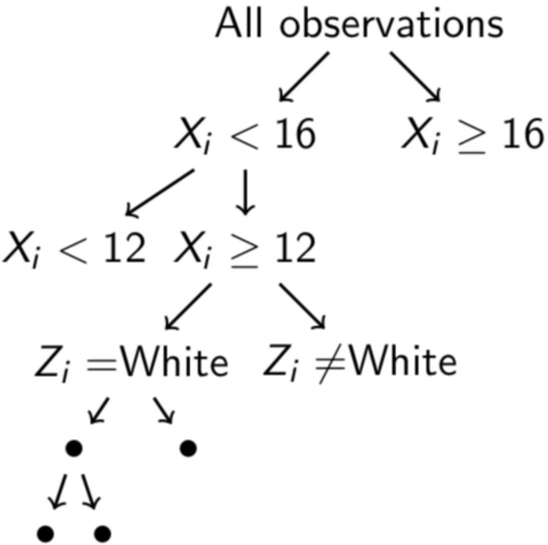
ML for heterogeneity analysis

The most common approach is the **causal tree**:

1. Select a training and a test sample
2. Training sample only: Predict $\hat{Y}$ for all units
3. Split into X_i groups, looking for the largest difference in $\hat{Y}_i$
4. Split into X_i subgroups, looking for the largest difference in $\hat{Y}_i$
5. Split into X_i subsubgroups, looking for the largest difference in $\hat{Y}_i$
6. Stop splitting based on some rule
7. Testing sample only: using the identified groups, estimate $\hat{\tau}_g$

$\rightarrow$ Repeat with different training samples to construct a causal forest

Causal trees



ML for causal inference

Now we're really pushing the frontier:

1. ML with selection on observables
2. ML with selection on unobservables

$\rightarrow$ We need to reframe our questions as prediction problems

ML with selection on observables

Consider an underlying model:

$$Y_i = \alpha + \tau D_i + f(\mathbf{X}_i) + \varepsilon_i$$

where $E[\varepsilon|D, \mathbf{X}] = 0$: **Conditional** on $\mathbf{X}$, D is as good as random

- But which $\mathbf{X}$s matter?
- And what is the right $f(\mathbf{X})$?
- We can use ML to help us figure this out

ML with selection on observables

Consider an underlying model:

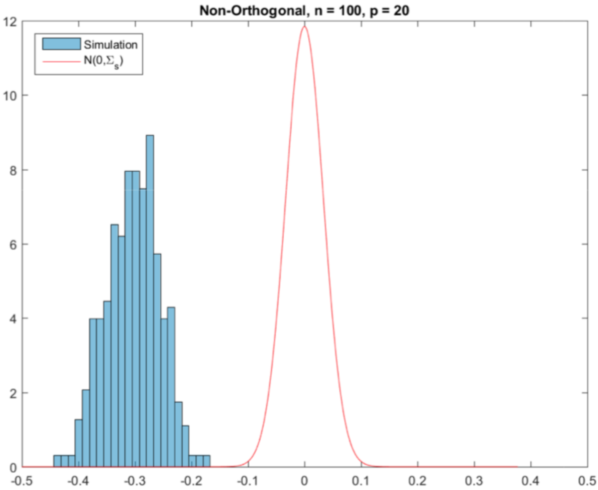
$$Y_i = \alpha + \tau D_i + f(\mathbf{X}_i) + \varepsilon_i$$

where $E[\varepsilon | D, \mathbf{X}] = 0$: **Conditional** on $\mathbf{X}$, D is as good as random

- But which $\mathbf{X}$s matter?
- And what is the right $f(\mathbf{X})$?
- We can use ML to help us figure this out

→ **Simple guess:** simply predict $\hat{Y}$ based on D and $\mathbf{X}$; interpret coefficient on D as τ

ML with selection on observables



ML with selection on observables

The simple guess doesn't work!

- The overall best fit ignores the SOO assumption
- Some X_i that are important for D_i may be left out

$\rightarrow$ ML will choose X_i that are important for Y_i

$\rightarrow$ These aren't necessarily the same as those that matter for D_i

ML with selection on observables

The simple guess doesn't work!

- The overall best fit ignores the SOO assumption
- Some X_i that are important for D_i may be left out

→ ML will choose X_i that are important for Y_i

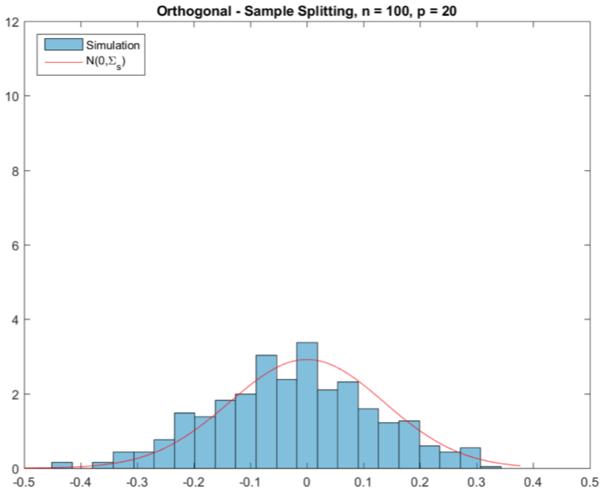
→ These aren't necessarily the same as those that matter for D_i

We can do better!

1. Predict $\hat{Y}$ as a function of X
2. Also predict $\hat{D}$ as a function of X
3. Estimate treatment effects using both sets of covariates

→ This only works when you do steps (1) and (2) with LASSO

ML with selection on observables



ML with selection on observables

The most up-to-date approach is:

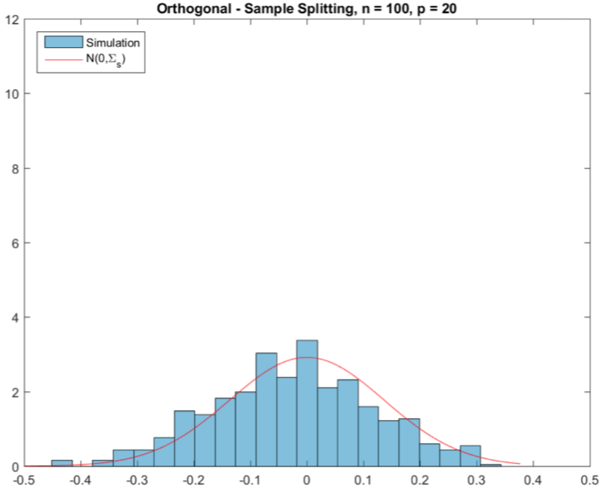
1. Predict $\hat{Y}$ as a function of X
2. Predict $\hat{D}$ as a function of X
3. Compute residuals: $Y^R = Y - \hat{Y}$ and $D^R = D - \hat{D}$
4. Recover $\hat{\tau}$ by regressing:

$$Y_i^R = \alpha + \tau D^R + \varepsilon_i$$

→ You can do steps (1) and (2) with any ML method

→ Note that this approach still needs the SOO assumptions

ML with selection on observables



ML with selection on unobservables

Just like with SOO, ML can be useful to pick covariates:

- ML helps pick X_i when we don't need them for identification
- This works for RCTs, DD, IV, etc **with exogenous covariates only**

ML with selection on unobservables

Just like with SOO, ML can be useful to pick covariates:

- ML helps pick X_i when we don't need them for identification
- This works for RCTs, DD, IV, etc **with exogenous covariates only**
- We can use ML when we want a better fit

Great application: first stage of IV:

- Conditional on (a) good instrument(s), we just want a good fit
- Nothing wrong with using ML to improve the first stage...
- As long as you're only using exogenous covariates

Can we use ML to build a better counterfactual?

<u>We need to reframe as a prediction exercise:</u>

- What is a(n estimated) counterfactual?
- Just a guess at what would've happened without treatment
- This is a simple prediction exercise
- We can potentially use ML to help us generate this counterfactual
- (Subject to all of the standard selection / identification issues)

An example: Energy efficiency in California schools

Policy issue:

- Lots of money is being spent on EE upgrades
- But are they effective?

Approach:

- Look at hourly data from 2,000 public K-12 California schools
- Some schools decided to implement EE upgrades
- This was not randomized, so we use an FE approach

→ Leverage high-resolution data for an ML-augmented FE method

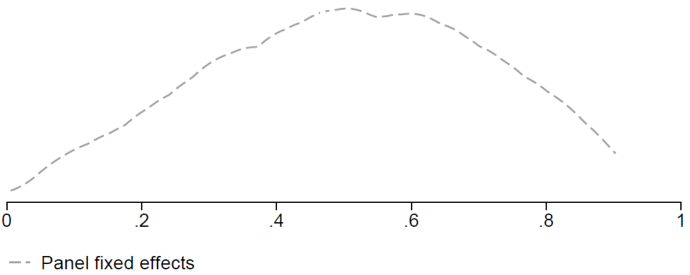
Estimating the effects of EE upgrades: version A

- We compare:
  - Consumption at schools that retrofitted to those that didn't
  - Consumption before and after retrofits
- We progressively add a series of control variables (school, hour and month-of-sample fixed effects, plus interactions):

$$Y_{ith} = \tau D_{it} + \alpha_i + \kappa_h + \gamma_t + \varepsilon_{ith}$$

Interpretation of τ: Average reduction in KWh at treated schools.

Panel FE results are unstable



Can machine learning help?

- Panel FE models aren't properly specified.
- Schools are very heterogeneous (e.g., climate, size, school calendar).
    - Ideally, introduce school-specific coefficients and trends in a very flexible manner.
- We easily came up with $\sim$6,000,000 candidate control variables by making them school-hour specific!
- No clear *ex ante* optimal choice.

Machine Learning: Advantages in this application

- Exogenous weather variation and predictable weekly and seasonal patterns drive variation in electricity consumption.
- Schools are relatively stable consumption units:
  - as opposed to single households that move around, unobservably buy a new appliance, expand family size, etc.
  - as opposed to businesses and manufacturing plants, exposed to macroeconomic shocks.

Machine Learning: Advantages in this application

- Exogenous weather variation and predictable weekly and seasonal patterns drive variation in electricity consumption.
- Schools are relatively stable consumption units:
  - as opposed to single households that move around, unobservably buy a new appliance, expand family size, etc.
  - as opposed to businesses and manufacturing plants, exposed to macroeconomic shocks.

Prediction can do well!

Machine Learning: Approach

Step 1

- Use *pre-treatment data* to predict electricity consumption as a function of flexible co-variates, *for each school separately*.

Machine Learning: Approach

Step 1

- Use *pre-treatment data* to predict electricity consumption as a function of flexible co-variates, *for each school separately*.
  - For control schools, determine a "pre-treatment period" randomly.

Machine Learning: Approach

Step 1

- Use *pre-treatment data* to predict electricity consumption as a function of flexible co-variates, *for each school separately*.
  - For control schools, determine a "pre-treatment period" randomly.
  - Use LASSO method (penalized regression).
    - Minimizing the sum of the squared errors plus $\lambda \cdot \sum_{j=1}^{p} |\beta_j|$.
    - Larger "tuning parameters" lead to fewer coefficients.
    - Use bootstrapped cross-validation with training and holdout samples *within pre-treatment*.

Machine Learning: Approach

Step 1

- Use *pre-treatment data* to predict electricity consumption as a function of flexible co-variates, *for each school separately*.
  - For control schools, determine a "pre-treatment period" randomly.
  - Use LASSO method (penalized regression).
    - Minimizing the sum of the squared errors plus $\lambda \cdot \sum_{j=1}^{p} |\beta_j|$.
    - Larger "tuning parameters" lead to fewer coefficients.
    - Use bootstrapped cross-validation with training and holdout samples *within pre-treatment*.
  - Include a wide range of school-specific variables, and also consumption at control schools (a la synthetic control).
  - Also consider other alternatives (random forests).

Machine Learning: Approach

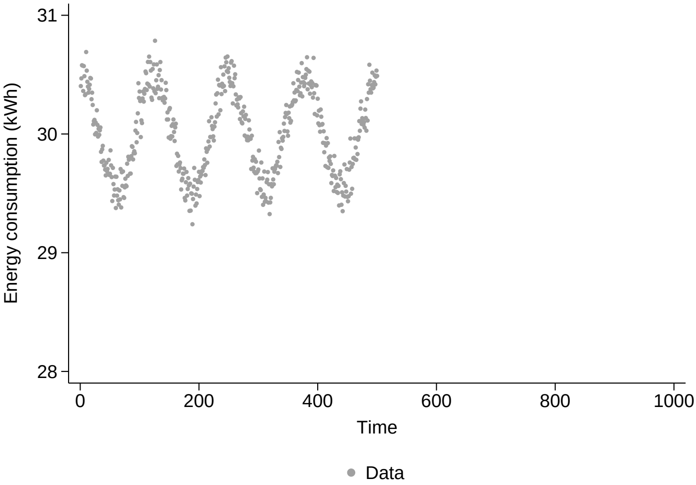
Step 2

- Regress *prediction errors* on treatment and controls.

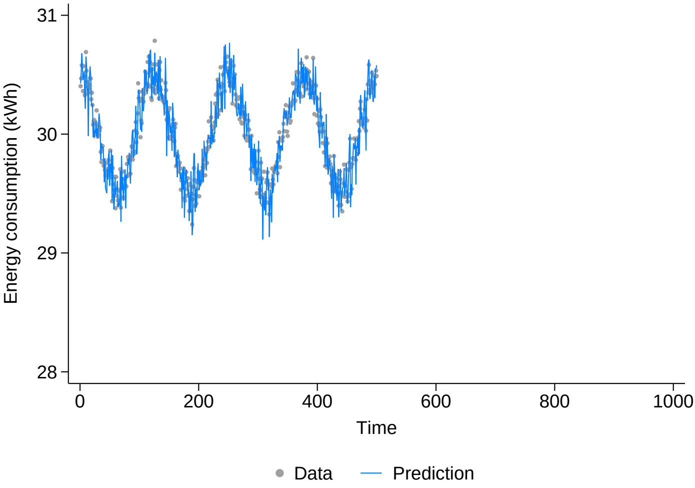
$$Y_{ith} = \tau D_{it} + \alpha_i + \kappa_h + \gamma_t + \varepsilon_{ith}$$

  - Data pooled across schools
  - Replicates diff-in-diff approach, but Y variable is now the prediction error

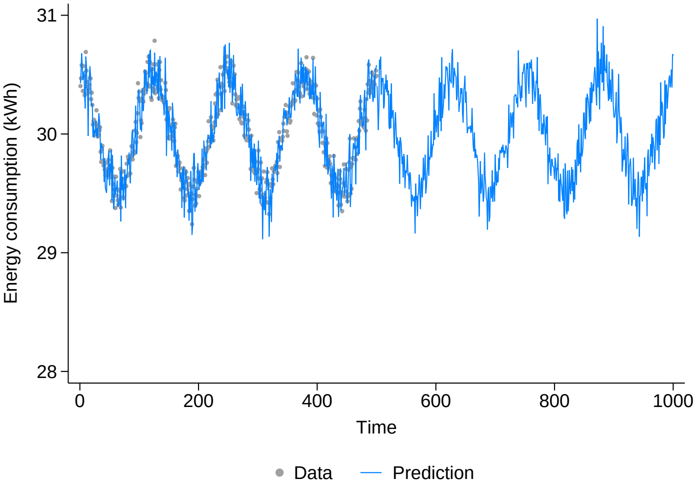
Machine Learning: Graphical intuition



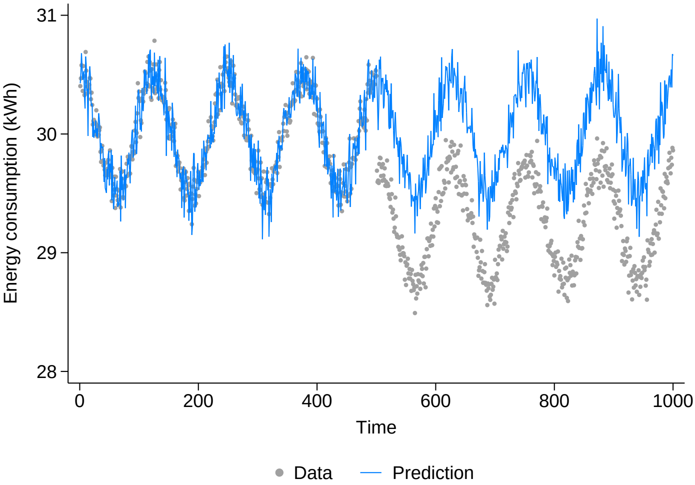
Machine Learning: Graphical intuition



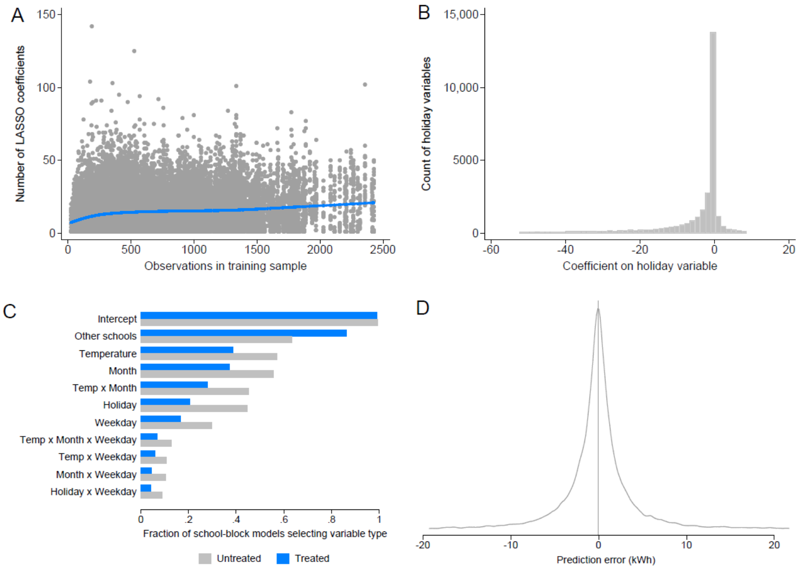
Machine Learning: Graphical intuition



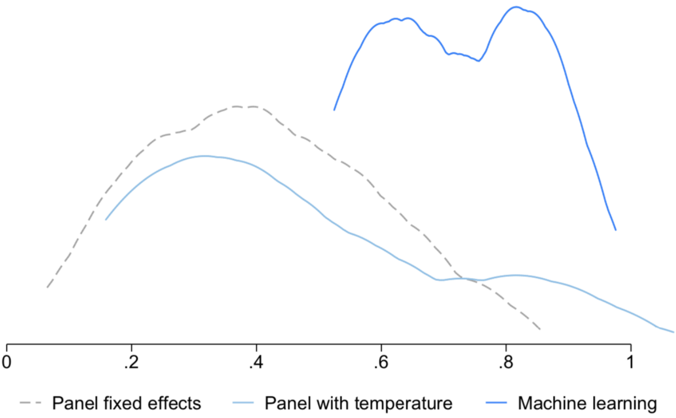
Machine Learning: Graphical intuition



ML diagnostics



ML results are stable across estimators



Recap

TL;DR:

1. New datasets open new questions
2. Machine learning offers opportunity
3. Both require some careful consideration or tweaks to be useful for us