Data Science for Causal Inference

Ryan T. Moore

American University

The Lab @ DC

2024-07-15

Table of contents I

Introductions

About Me

- Associate Prof of Government
  (American University)
- Associate Director, Center for Data Science
  (American University)
- Senior Social Scientist
  (The Lab @ DC)
- Fellow in Methodology
  (US Office of Evaluation Sciences: “OES”)

About Me

- Associate Prof of Government (American University)
- Associate Director, Center for Data Science (American University)
- Senior Social Scientist (The Lab @ DC)
- Fellow in Methodology (US Office of Evaluation Sciences: "OES")
- Research agenda: political methodology, causal inference, experimental design, experiments in public policy

About You!

- Name?

About You!

- Name?
- Role?

About You!

- Name?
- Role?
- Interests?

About You!

- Name?
- Role?
- Interests?
- Olympic sport you look forward to?

Plan

- Data Science in Causal Inference

Plan

- Data Science in Causal Inference
  - Models

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification
  - Unobservable parameter

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification
  - Unobservable parameter
  - Unobserved confounders

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification
  - Unobservable parameter
  - Unobserved confounders
- Modern difference-in-difference designs

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification
  - Unobservable parameter
  - Unobserved confounders
- Modern difference-in-difference designs
  - Canonical DiD

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification
  - Unobservable parameter
  - Unobserved confounders
- Modern difference-in-difference designs
  - Canonical DiD
  - Multiple time periods

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification
  - Unobservable parameter
  - Unobserved confounders
- Modern difference-in-difference designs
  - Canonical DiD
  - Multiple time periods
  - Staggered adoption

Plan

- Data Science in Causal Inference
  - Models
  - Heterogeneous treatment effects
  - Variable selection
- Sensitivity
  - Model specification
  - Unobservable parameter
  - Unobserved confounders
- Modern difference-in-difference designs
  - Canonical DiD
  - Multiple time periods
  - Staggered adoption
  - Calloway-Sant'Anna approach

Data Science in Causal Inference

Causal Inference Approaches

The "potential outcomes" framework:

Causal Inference Approaches

The "potential outcomes" framework:

Citizen	Canvass?	Would Enroll if Canvass?	Would Enroll if No Canvass?	Enroll
1	Yes			Yes
2	Yes			Yes
3	No			No
4	No			No

Causal Inference Approaches

The “potential outcomes” framework:

Citizen	Canvass?	Would Enroll if Canvass?	Would Enroll if No Canvass?	Enroll
1	Yes	Yes		Yes
2	Yes			Yes
3	No			No
4	No			No

Causal Inference Approaches

The "potential outcomes" framework:

Citizen	Canvass?	Would Enroll if Canvass?	Would Enroll if No Canvass?	Enroll
1	Yes	Yes		Yes
2	Yes	Yes		Yes
3	No			No
4	No			No

Causal Inference Approaches

The "potential outcomes" framework:

Citizen	Canvass?	Would Enroll if Canvass?	Would Enroll if No Canvass?	Enroll
1	Yes	Yes		Yes
2	Yes	Yes		Yes
3	No		No	No
4	No			No

Causal Inference Approaches

The "potential outcomes" framework:

Citizen	Canvass?	Would Enroll if Canvass?	Would Enroll if No Canvass?	Enroll
1	Yes	Yes		Yes
2	Yes	Yes		Yes
3	No		No	No
4	No		No	No

Causal Inference Approaches

The “potential outcomes” framework:

Citizen	Canvass?	Would Enroll if Canvass?	Would Enroll if No Canvass?	Enroll
1	Yes	Yes	(Yes)	Yes
2	Yes	Yes	(No)	Yes
3	No	(Yes)	No	No
4	No	(No)	No	No

Causal Inference Approaches

The "potential outcomes" framework, more abstractly:

Unit i	Treatment T	$Y(1)$	$Y(0)$	Y^{obs}	True τ $Y(1)-Y(0)$
1	1	10		10	
2	1	20		20	
3	0		15	15	
4	0		5	5	

Causal Inference Approaches

The "potential outcomes" framework, more abstractly:

Unit i	Treatment T	$Y(1)$	$Y(0)$	Y^{obs}	True τ $Y(1) - Y(0)$
1	1	10	(10)	10	0
2	1	20	(10)	20	10
3	0	(40)	15	15	25
4	0	(20)	5	5	15

Causal Inference Approaches

The "potential outcomes" framework, more abstractly:

Unit i	Treatment T	$Y(1)$	$Y(0)$	Y^{obs}	True τ $Y(1) - Y(0)$
1	1	10	(10)	10	0
2	1	20	(10)	20	10
3	0	(40)	15	15	25
4	0	(20)	5	5	15
				ATE $= \bar{\tau} =$	$\frac{50}{4} = 12.5$

Causal Inference Approaches

The "potential outcomes" framework, more abstractly:

Unit i	Treatment T	$Y(1)$	$Y(0)$	Y^{obs}	True τ $Y(1)-Y(0)$
1	1	10	(10)	10	0
2	1	20	(10)	20	10
3	0	(40)	15	15	25
4	0	(20)	5	5	15
				$\text{ATE} = \bar{\tau} =$	$\frac{50}{4} = 12.5$
				$\widehat{ATE} = \hat{\bar{\tau}} =$	$15 - 10 = 5$

Causal Inference Approaches

The “potential outcomes” framework, notation:

- Units indexed by i

Causal Inference Approaches

The “potential outcomes” framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i

Causal Inference Approaches

The "potential outcomes" framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i
- Outcome if treated $Y_i(1)$

Causal Inference Approaches

The "potential outcomes" framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i
- Outcome if treated $Y_i(1)$
- Outcome if control $Y_i(0)$

Causal Inference Approaches

The "potential outcomes" framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i
- Outcome if treated $Y_i(1)$
- Outcome if control $Y_i(0)$
- True treatment effect $\tau_i = Y_i(1) - Y_i(0)$

Causal Inference Approaches

The "potential outcomes" framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i
- Outcome if treated $Y_i(1)$
- Outcome if control $Y_i(0)$
- True treatment effect $\tau_i = Y_i(1) - Y_i(0)$
- True average treatment effect
  $\bar{\tau} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i(1) - Y_i(0)\right)$

Causal Inference Approaches

The "potential outcomes" framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i
- Outcome if treated $Y_i(1)$
- Outcome if control $Y_i(0)$
- True treatment effect $\tau_i = Y_i(1) - Y_i(0)$
- True average treatment effect $\bar{\tau} = \frac{1}{n}\sum_{i=1}^{n}(Y_i(1) - Y_i(0))$
- Pre-treatment covariates $\mathbf{X}$

Causal Inference Approaches

The "potential outcomes" framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i
- Outcome if treated $Y_i(1)$
- Outcome if control $Y_i(0)$
- True treatment effect $\tau_i = Y_i(1) - Y_i(0)$
- True average treatment effect $\bar{\tau} = \frac{1}{n}\sum_{i=1}^{n}(Y_i(1) - Y_i(0))$
- Pre-treatment covariates $\mathbf{X}$

Causal Inference Approaches

The "potential outcomes" framework, notation:

- Units indexed by i
- Treatment T_i or D_i or Z_i
- Outcome if treated $Y_i(1)$
- Outcome if control $Y_i(0)$
- True treatment effect $\tau_i = Y_i(1) - Y_i(0)$
- True average treatment effect $\bar{\tau} = \frac{1}{n} \sum_{i=1}^{n} (Y_i(1) - Y_i(0))$
- Pre-treatment covariates $\mathbf{X}$

(and we'll draw some DAG's, too)

Data Science Approaches

Three tasks of data science:

- Description

Data Science Approaches

Three tasks of data science:

- Description
- Prediction

Data Science Approaches

Three tasks of data science:

- Description
- Prediction
- Causal Inference

Data Science Approaches

Three tasks of data science:

- Description
- Prediction
- Causal Inference

Data Science Approaches

Three tasks of data science:

- Description
- Prediction
- Causal Inference

Models/algorithms central to all three.

Data Science Approaches

Three tasks of data science:

- Description
- Prediction
- Causal Inference

Models/algorithms central to all three.

Hernán, Hsu, and Healy (2019)

Data Science Approaches

Description

- Identifying patterns, etc.

Data Science Approaches

Description

- Identifying patterns, etc.
- E.g., clustering to discover groups

Data Science Approaches

Prediction

- Components

Data Science Approaches

Prediction

- Components
  - Inputs/outputs (predictors/outcomes, features/responses, ...)

Data Science Approaches

Prediction

- Components
  - Inputs/outputs (predictors/outcomes, features/responses, ...)
  - Mapping from inputs to outputs (linear model, decision tree, ...)

Data Science Approaches

Prediction

- Components
  - Inputs/outputs (predictors/outcomes, features/responses, ...)
  - Mapping from inputs to outputs (linear model, decision tree, ...)
  - Metric for evaluating mapping

Data Science Approaches

Prediction

- Components
  - Inputs/outputs (predictors/outcomes, features/responses, ...)
  - Mapping from inputs to outputs (linear model, decision tree, ...)
  - Metric for evaluating mapping
- With these, model machine learning does the work

Data Science Approaches

Prediction

- Components
  - Inputs/outputs (predictors/outcomes, features/responses, ...)
  - Mapping from inputs to outputs (linear model, decision tree, ...)
  - Metric for evaluating mapping
- With these, model machine learning does the work
- E.g., regression, random forests, neural networks, ...

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction
- Requires more than summary statistics, metrics, etc.

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction
- Requires more than summary statistics, metrics, etc.
- Requires some knowledge of causal structure

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction
- Requires more than summary statistics, metrics, etc.
- Requires some knowledge of causal structure
  - Not all inputs treated same

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction
- Requires more than summary statistics, metrics, etc.
- Requires some knowledge of causal structure
  - Not all inputs treated same
  - T v. $\mathbf{X}$ – very different!

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction
- Requires more than summary statistics, metrics, etc.
- Requires some knowledge of causal structure
  - Not all inputs treated same
  - T v. $\mathbf{X}$ – very different!
  - (the more knowledge, the better!)

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction
- Requires more than summary statistics, metrics, etc.
- Requires some knowledge of causal structure
  - Not all inputs treated same
  - T v. $\mathbf{X}$ – very different!
  - (the more knowledge, the better!)
  - (alternative: solve fundamental problem of causal inference!)

Data Science Approaches

Causal Inference

- Potential outcomes/counterfactual/interventionist perspective
- Requires *expertise* different to description/prediction
- Requires more than summary statistics, metrics, etc.
- Requires some knowledge of causal structure
  - Not all inputs treated same
  - T v. $\mathbf{X}$ – very different!
  - (the more knowledge, the better!)
  - (alternative: solve fundamental problem of causal inference!)
- E.g., experiments, observational causal designs, ...

Causal Inference with Machine Learning

Causal Inference with Machine Learning

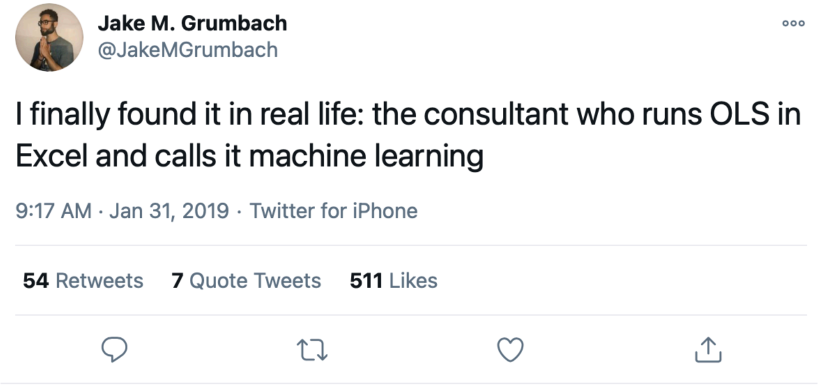
Jake M. Grumbach
@JakeMGrumbach

I finally found it in real life: the consultant who runs OLS in Excel and calls it machine learning

9:17 AM · Jan 31, 2019 · Twitter for iPhone

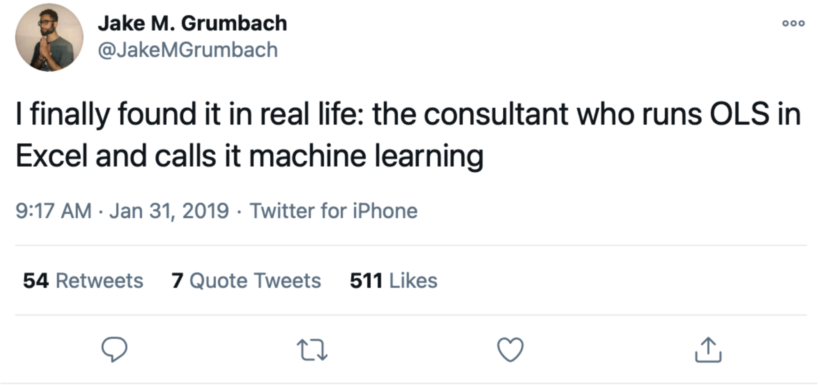
54 Retweets **7** Quote Tweets **511** Likes

Causal Inference with Machine Learning



Don't do this.

Causal Inference with Machine Learning



Jake M. Grumbach
@JakeMGrumbach

I finally found it in real life: the consultant who runs OLS in Excel and calls it machine learning

9:17 AM · Jan 31, 2019 · Twitter for iPhone

54 Retweets **7** Quote Tweets **511** Likes

Don’t do this.

(Not “machine learning”, probably, but *models* at least ...)

Causal Inference with Models

Loaded two datasets:

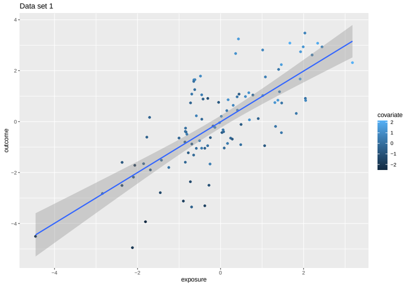
```r
str(df1)
```

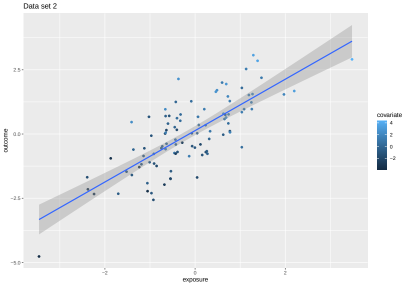
```
tibble [100 x 3] (S3: tbl_df/tbl/data.frame)
 $ covariate: num [1:100] -0.622 1.137 -0.238 1.529 -0.154
 $ exposure : num [1:100] 0.0332 0.3627 0.2422 1.4633 0.779
 $ outcome  : num [1:100] -0.429 2.675 -0.647 2.238 1.044
```

```r
str(df2)
```

```
tibble [100 x 3] (S3: tbl_df/tbl/data.frame)
 $ exposure : num [1:100] 0.4862 0.0653 -1.4021 -0.546 -0.
 $ outcome  : num [1:100] 1.706 0.669 -1.597 -1.733 0.617
 $ covariate: num [1:100] 2.24 0.924 -0.999 -2.343 0.207 ..
```

Causal Inference with Models





Causal Inference with Models

Model each

```
lm_df1 <- lm(outcome ~ exposure, data = df1)
lm_df2 <- lm(outcome ~ exposure, data = df2)
```

```
# A tibble: 4 x 4
  data  term        estimate std.error
  <chr> <chr>          <dbl>     <dbl>
1 df1   (Intercept) -0.00671    0.120
2 df1   exposure     0.996      0.0927
3 df2   (Intercept)  0.133      0.0890
4 df2   exposure     1.00       0.0841
```

Causal Inference with Models

Model each

```r
lm_df1 <- lm(outcome ~ exposure, data = df1)
lm_df2 <- lm(outcome ~ exposure, data = df2)
```

```
# A tibble: 4 x 4
  data  term        estimate std.error
  <chr> <chr>          <dbl>     <dbl>
1 df1   (Intercept) -0.00671    0.120
2 df1   exposure     0.996      0.0927
3 df2   (Intercept)  0.133      0.0890
4 df2   exposure     1.00       0.0841
```

- Both cases: effect of exposure ≈ 1.

Causal Inference with Models

Model each

```r
lm_df1 <- lm(outcome ~ exposure, data = df1)
lm_df2 <- lm(outcome ~ exposure, data = df2)
```

```
# A tibble: 4 x 4
  data  term        estimate std.error
  <chr> <chr>          <dbl>     <dbl>
1 df1   (Intercept) -0.00671    0.120
2 df1   exposure     0.996      0.0927
3 df2   (Intercept)  0.133      0.0890
4 df2   exposure     1.00       0.0841
```

- Both cases: effect of exposure ≈ 1.
- Is this good? Is it correct?

Causal Inference with Models

Model each

```r
lm_df1 <- lm(outcome ~ exposure, data = df1)
lm_df2 <- lm(outcome ~ exposure, data = df2)
```

```
# A tibble: 4 x 4
  data  term        estimate std.error
  <chr> <chr>          <dbl>     <dbl>
1 df1   (Intercept) -0.00671    0.120
2 df1   exposure     0.996      0.0927
3 df2   (Intercept)  0.133      0.0890
4 df2   exposure     1.00       0.0841
```

- Both cases: effect of exposure ≈ 1.
- Is this good? Is it correct?
- What if we adjust for covariate?

Causal Inference with Models

```r
lm_df1_adj <- lm(outcome ~ exposure + covariate, data = df1
lm_df2_adj <- lm(outcome ~ exposure + covariate, data = df2
```

```
# A tibble: 4 x 4
  data  term      estimate std.error
  <chr> <chr>        <dbl>     <dbl>
1 df1   exposure     0.501    0.108
2 df1   covariate    0.970    0.147
3 df2   exposure     0.554    0.0990
4 df2   covariate    0.385    0.0598
```

- Both cases: effect of exposure ≈ 0.5.

Causal Inference with Models

```r
lm_df1_adj <- lm(outcome ~ exposure + covariate, data = df1
lm_df2_adj <- lm(outcome ~ exposure + covariate, data = df2
```

```
# A tibble: 4 x 4
  data  term      estimate std.error
  <chr> <chr>        <dbl>     <dbl>
1 df1   exposure     0.501    0.108
2 df1   covariate    0.970    0.147
3 df2   exposure     0.554    0.0990
4 df2   covariate    0.385    0.0598
```

- Both cases: effect of exposure ≈ 0.5.
- Is this good? Is it correct?

Causal Inference with Models

```
lm_df1_adj <- lm(outcome ~ exposure + covariate, data = df1
lm_df2_adj <- lm(outcome ~ exposure + covariate, data = df2
```

```
# A tibble: 4 x 4
  data  term      estimate std.error
  <chr> <chr>        <dbl>     <dbl>
1 df1   exposure     0.501    0.108
2 df1   covariate    0.970    0.147
3 df2   exposure     0.554    0.0990
4 df2   covariate    0.385    0.0598
```

- Both cases: effect of exposure ≈ 0.5.
- Is this good? Is it correct?
- Which is correct? $\beta = 1$? $\beta = 0.5$?

Causal Inference with Models

```
lm_df1_adj <- lm(outcome ~ exposure + covariate, data = df1
lm_df2_adj <- lm(outcome ~ exposure + covariate, data = df2
```

```
# A tibble: 4 x 4
  data  term       estimate std.error
  <chr> <chr>         <dbl>     <dbl>
1 df1   exposure      0.501    0.108
2 df1   covariate     0.970    0.147
3 df2   exposure      0.554    0.0990
4 df2   covariate     0.385    0.0598
```

- Both cases: effect of exposure ≈ 0.5.
- Is this good? Is it correct?
- Which is correct? $\beta = 1$? $\beta = 0.5$?
- *Should* we adjust for covariate?

Causal Inference with Models

There is nothing in the data that tells us.

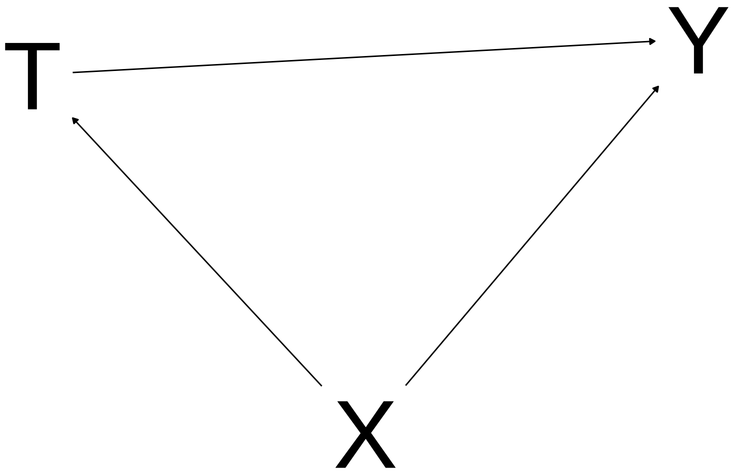
Causal Inference with Models

There is nothing in the data that tells us. ☹

Causal Inference with Models

There is nothing in the data that tells us. ☹

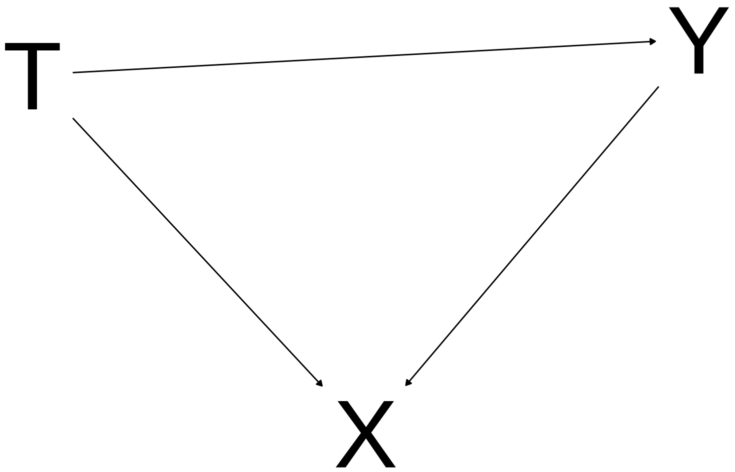
Here are the true structures: First



Causal Inference with Models

There is nothing in the data that tells us. ☹

Here are the true structures: Second



Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

- `df1`: confounding ⇒ **adjust for** X

Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

- `df1`: confounding $\Rightarrow$ **adjust for** X
- `df2`: collider $\Rightarrow$ **do not adjust for** X

Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

- `df1`: confounding $\Rightarrow$ **adjust for** X
- `df2`: collider $\Rightarrow$ **do not adjust for** X

Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

- `df1`: confounding $\Rightarrow$ **adjust for** X
- `df2`: collider $\Rightarrow$ **do not adjust for** X

```
g_conf <- dagitty("dag{ x -> y ; x <- c -> y }")
g_coll <- dagitty("dag{ x -> y ; x -> c <- y }")
```

Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

- `df1`: confounding $\Rightarrow$ **adjust for** X
- `df2`: collider $\Rightarrow$ **do not adjust for** X

```
g_conf <- dagitty("dag{ x -> y ; x <- c -> y }")
g_coll <- dagitty("dag{ x -> y ; x -> c <- y }")
```

```
adjustmentSets(g_conf, "x", "y")
```

```
{ c }
```

Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

- `df1`: confounding ⇒ **adjust for** X
- `df2`: collider ⇒ **do not adjust for** X

```
g_conf <- dagitty("dag{ x -> y ; x <- c -> y }")
g_coll <- dagitty("dag{ x -> y ; x -> c <- y }")
```

```
adjustmentSets(g_conf, "x", "y")
```

```
{ c }
```

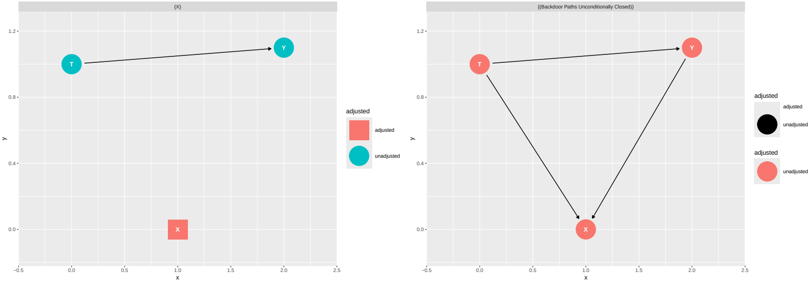
```
adjustmentSets(g_coll, "x", "y")
```

```
 {}
```

Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

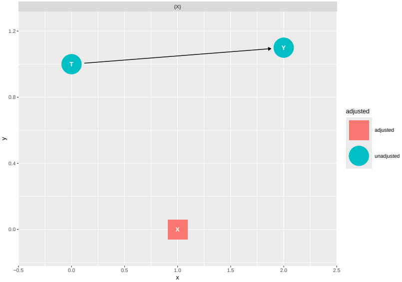
- `df1`: confounding $\Rightarrow$ **adjust for** X
- `df2`: collider $\Rightarrow$ **do not adjust for** X

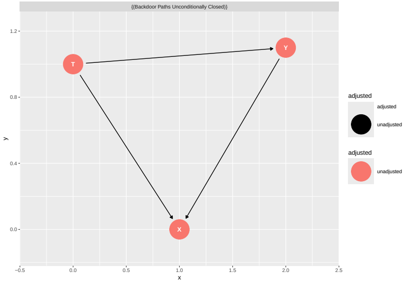


Causal Inference with Models

When know structures, adjustment sets for unbiasedness differ:

- `df1`: confounding $\Rightarrow$ **adjust for** X
- `df2`: collider $\Rightarrow$ **do not adjust for** X





(Data from D'Agostino McGowan (2023))

Causal Inference with Models

- Importance of identifying “pre-treatment covariates”, “proper covariates”; doing “design before analysis”

Causal Inference with Models

- Importance of identifying “pre-treatment covariates”, “proper covariates”; doing “design before analysis”
- Importance of experiments: strong knowledge about (part of) causal structure – assgn mechanism

Causal Inference with Models

- Importance of identifying “pre-treatment covariates”, “proper covariates”; doing “design before analysis”
- Importance of experiments: strong knowledge about (part of) causal structure – assgn mechanism
- Causal inference is critical to scientific questions, and separate from prediction

Causal Inference with Models

- Importance of identifying "pre-treatment covariates", "proper covariates"; doing "design before analysis"
- Importance of experiments: strong knowledge about (part of) causal structure – assgn mechanism
- Causal inference is critical to scientific questions, and separate from prediction
- Though, methods from prediction can aid causal inference

Causal Inference with Models

- Importance of identifying "pre-treatment covariates", "proper covariates"; doing "design before analysis"
- Importance of experiments: strong knowledge about (part of) causal structure – assgn mechanism
- Causal inference is critical to scientific questions, and separate from prediction
- Though, methods from prediction can aid causal inference
- "Causal euphimisms" don't help (Hernán 2018)

Approaches of Prediction and Causal Inference

Two Cultures, (Breiman 2001)

- *Data Models*: our "social science modeling"

Approaches of Prediction and Causal Inference

Two Cultures, (Breiman 2001)

- *Data Models*: our "social science modeling"
- *Algorithmic Models*: our "data science algorithms"

Methods for Prediction and Causal Inference

- Cross-validation
- Regression/Decision trees
- Random forests

James et al. (2021)

Cross-validation

k-fold cross-validation to select method

Cross-validation

k-fold cross-validation to select method

- Randomly partition data into k groups

Cross-validation

k-fold cross-validation to select method

- Randomly partition data into k groups
- Apply method to $k-1$ groups

Cross-validation

k-fold cross-validation to select method

- Randomly partition data into k groups
- Apply method to $k-1$ groups
- Use result to predict for left-out group

Cross-validation

k-fold cross-validation to select method

- Randomly partition data into k groups
- Apply method to $k - 1$ groups
- Use result to predict for left-out group
- Calculate $\text{MSE}_i = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Cross-validation

k-fold cross-validation to select method

- Randomly partition data into k groups
- Apply method to $k-1$ groups
- Use result to predict for left-out group
- Calculate $\mathrm{MSE}_i = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- Calculate test error as average of the k MSE's:

Cross-validation

k-fold cross-validation to select method

- Randomly partition data into k groups
- Apply method to $k-1$ groups
- Use result to predict for left-out group
- Calculate $\text{MSE}_i = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
- Calculate test error as average of the k MSE's:

Cross-validation

k-fold cross-validation to select method

- Randomly partition data into k groups
- Apply method to $k-1$ groups
- Use result to predict for left-out group
- Calculate $\text{MSE}_i = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- Calculate test error as average of the k MSE's:

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k}\text{MSE}_i$$

- Select model that minimises $CV_{(k)}$

CV for Linear Model

```r
library(tidyverse)

## Make data

mk_data <- function(n = 90, n_folds = 10){

  df <- tibble(
    x1 = rnorm(n),
    x2 = rnorm(n),
    x3 = rnorm(n),
    y = 0.1 * x1 + 0.2 * x2 + 0.5 * x3 + rnorm(n),
    cv_fold = sample(rep(1:n_folds, (n / n_folds)))
  )

}

df <- mk_data()
```

CV for Linear Model

```r
head(df)
```

```
# A tibble: 6 x 5
      x1     x2      x3      y cv_fold
   <dbl>  <dbl>   <dbl>  <dbl>   <int>
1 -1.36  -0.778 -0.0372 -2.34        7
2  0.536  0.504  1.21    0.270      10
3 -0.791 -0.142  0.0307 -0.214       7
4 -0.647  0.801 -0.604   0.405       2
5 -1.40   0.984 -0.0544  0.607       3
6 -0.906 -0.936 -0.452   0.497      10
```

CV for Linear Model

```r
head(df)
```

```
# A tibble: 6 x 5
      x1     x2       x3      y cv_fold
   <dbl>  <dbl>    <dbl>  <dbl>   <int>
1 -1.36  -0.778 -0.0372 -2.34        7
2  0.536  0.504  1.21     0.270      10
3 -0.791 -0.142  0.0307 -0.214       7
4 -0.647  0.801 -0.604    0.405       2
5 -1.40   0.984 -0.0544   0.607       3
6 -0.906 -0.936 -0.452    0.497      10
```

```r
table(df$cv_fold)
```

```

 1  2  3  4  5  6  7  8  9 10 
 9  9  9  9  9  9  9  9  9  9 
```

CV for Linear Model

```
cv_lm <- function(data, fmla){

  n_folds <- max(data$cv_fold)
  store_mses <- vector("numeric", length = n_folds)

  for(idx in 1:n_folds){

    df_train <- data |> filter(cv_fold != idx)
    df_test <- data |> filter(cv_fold == idx)

    lm_out <- lm(fmla, data = df_train)

    predictions <- predict(lm_out, newdata = df_test)

    store_mses[idx] <- mean((df_test$y - predictions)^2)}

  test_error_cv_k <- mean(store_mses)
  return(test_error_cv_k)
```

CV for Linear Model

```r
cv_lm(data = df, fmla = y ~ x1 + x2)
```

```
[1] 1.710041
```

CV for Linear Model

```
cv_lm(data = df, fmla = y ~ x1 + x2)
```

```
[1] 1.710041
```

```
df <- mk_data()
cv_lm(df, y ~ x1 + x2)
```

```
[1] 1.520183
```

CV for Linear Model

```r
cv_lm(data = df, fmla = y ~ x1 + x2)
```

```
[1] 1.710041
```

```r
df <- mk_data()
cv_lm(df, y ~ x1 + x2)
```

```
[1] 1.520183
```

```r
df <- mk_data()
cv_lm(df, y ~ x1 + x2 + x3)
```

```
[1] 0.9453955
```

CV for Linear Model

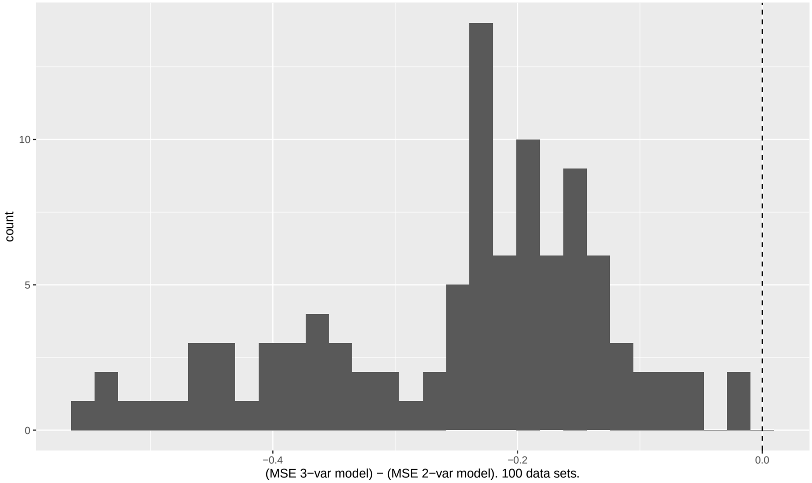


Figure 1: MSE always less (better) for 3-variable model.

Regression Trees

- Partition predictor space into regions $R_1, R_2, \ldots, R_J$.

Regression Trees

- Partition predictor space into regions $R_1, R_2, \ldots, R_J$.
- If unit falls in region R_j, use average outcome in R_j as predicted value: $\hat{y}_{R_j}$

Regression Trees

- Partition predictor space into regions $R_1, R_2, \ldots, R_J$.
- If unit falls in region R_j, use average outcome in R_j as predicted value: $\widehat{y}_{R_j}$
- (For *decision* on discrete outcome, count votes in R_j)

Regression Trees

- Partition predictor space into regions $R_1, R_2, \dots, R_J$.
- If unit falls in region R_j, use average outcome in R_j as predicted value: $\hat{y}_{R_j}$
- (For *decision* on discrete outcome, count votes in R_j)
- Goal: minimise residual sum of squares (RSS), just like LS regression:

Regression Trees

- Partition predictor space into regions $R_1, R_2, \dots, R_J$.
- If unit falls in region R_j, use average outcome in R_j as predicted value: $\hat{y}_{R_j}$
- (For *decision* on discrete outcome, count votes in R_j)
- Goal: minimise residual sum of squares (RSS), just like LS regression:

Regression Trees

- Partition predictor space into regions $R_1, R_2, \ldots, R_J$.
- If unit falls in region R_j, use average outcome in R_j as predicted value: $\hat{y}_{R_j}$
- (For *decision* on discrete outcome, count votes in R_j)
- Goal: minimise residual sum of squares (RSS), just like LS regression:

$$\sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j} \right)$$

Regression Trees

How to define regions R_j?

Regression Trees

How to define regions R_j?

- Top-down, greedy recursive binary split

Regression Trees

How to define regions R_j?

- Top-down, greedy recursive binary split
- At each step, find predictor and cut-point that minimise

Regression Trees

How to define regions R_j?

- Top-down, greedy recursive binary split
- At each step, find predictor and cut-point that minimise

Regression Trees

How to define regions R_j?

- Top-down, greedy recursive binary split
- At each step, find predictor and cut-point that minimise

$$\sum_{i:x \in R_1(j,s)} \left(y_i - \hat{y}_{R_1(j,s)}\right)^2 + \sum_{i:x \in R_2(j,s)} \left(y_i - \hat{y}_{R_2(j,s)}\right)^2$$

Regression Trees

- Overfitting is a potential problem

Regression Trees

- Overfitting is a potential problem
- Can we increase predictive quality by only using *part* of a tree?

Regression Trees

- Overfitting is a potential problem
- Can we increase predictive quality by only using *part* of a tree?
- "Pruning"

Regression Trees

Pruning

- Build a large tree

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α
- α: penalty parameter

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α
- α: penalty parameter
- $|T|$: count of terminal nodes of T

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α
- α: penalty parameter
- $|T|$: count of terminal nodes of T
- m: terminal node index

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α
- α: penalty parameter
- $|T|$: count of terminal nodes of T
- m: terminal node index
- Find subtree that minimises

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α
- α: penalty parameter
- $|T|$: count of terminal nodes of T
- m: terminal node index
- Find subtree that minimises

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α
- α: penalty parameter
- $|T|$: count of terminal nodes of T
- m: terminal node index
- Find subtree that minimises

$$\sum_{m=1}^{|T|} \sum_{i: x_i \in R_m} \left(y_i - \hat{y}_{R_m}\right)^2 + \alpha |T|$$

Regression Trees

Pruning

- Build a large tree
- Select the subtree that gives least prediction error (via cross-validation)
- But, many possible subtrees, so penalise larger trees via α
- α: penalty parameter
- $|T|$: count of terminal nodes of T
- m: terminal node index
- Find subtree that minimises

$$\sum_{m=1}^{|T|} \sum_{i: x_i \in R_m} \left(y_i - \hat{y}_{R_m}\right)^2 + \alpha|T|$$

Sum squared pred. error (plus penalty that grows with tree size) across units in region, then regions.

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)
2. For several values of α, find best subtree.

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)
2. For several values of α, find best subtree.
3. Find best value of α via CV. Create K folds. Then

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)
2. For several values of α, find best subtree.
3. Find best value of α via CV. Create K folds. Then
   3a. Do 1 and 2 on all but kth fold

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)
2. For several values of α, find best subtree.
3. Find best value of α via CV. Create K folds. Then
   3a. Do 1 and 2 on all but kth fold
   3b. Predict for kth fold, calculate MSE for several values of α

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)
2. For several values of α, find best subtree.
3. Find best value of α via CV. Create K folds. Then
   3a. Do 1 and 2 on all but kth fold
   3b. Predict for kth fold, calculate MSE for several values of α
   3c. Get avg MSE for each α

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)
2. For several values of α, find best subtree.
3. Find best value of α via CV. Create K folds. Then
   3a. Do 1 and 2 on all but kth fold
   3b. Predict for kth fold, calculate MSE for several values of α
   3c. Get avg MSE for each α
   3d. Pick α to minimise MSE

Regression Trees

But, how to choose α? (Use cross-validation.)

1. Build big tree on training data (with some minimum terminal node size)
2. For several values of α, find best subtree.
3. Find best value of α via CV. Create K folds. Then
   3a. Do 1 and 2 on all but kth fold
   3b. Predict for kth fold, calculate MSE for several values of α
   3c. Get avg MSE for each α
   3d. Pick α to minimise MSE

4. Using that α, select best subtree from Step 2

Example: Regression Tree

```
library(qss)
library(rsample)
library(tree)

data("MPs")
mps <- MPs |> mutate(age = yod - yob,
                     is_labour = if_else(party == "labour",
                     is_london = if_else(region == "Greater
                     is_winner = if_else(margin > 0, 1, 0))
  select(ln.net, age, is_labour, is_london, is_winner) |>
  na.omit()

set.seed(765076184)

mp_split <- initial_split(mps, prop = 0.7)

mp_train <- training(mp_split)
mp_test <- testing(mp_split)
```

Example: Regression Tree

```
tree_mp <- tree(ln.net ~ ., data = mp_train)
plot(tree_mp)
text(tree_mp)
```

Figure 2: The regression tree (for training data)

Example: Regression Tree

Would pruning help?

Example: Regression Tree

Would pruning help?

```
cv_mps <- cv.tree(tree_mp, K = 10)

plot(cv_mps$size, cv_mps$dev, type = "b")
```

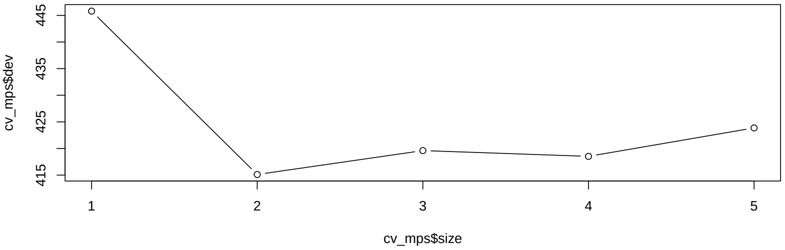


Figure 3: Subtree size 2 minimises SSR

Example: Regression Tree

Would pruning help?

```
cv_mps <- cv.tree(tree_mp, K = 10)

plot(cv_mps$size, cv_mps$dev, type = "b")
```

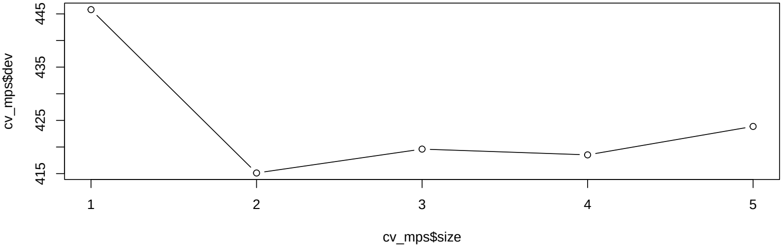


Figure 3: Subtree size 2 minimises SSR

Example: Regression Tree

```r
prune_mps <- prune.tree(tree_mp, best = 2)

plot(prune_mps)
text(prune_mps)
```

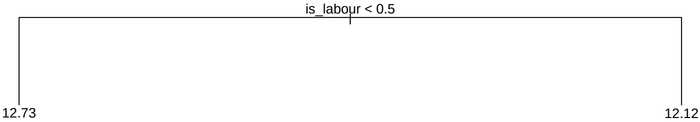


Figure 4: The pruned tree

Example: Regression Tree

Predict for test set:

- MSE for pruned: 1.922
- MSE for full: 1.945

Example: Regression Tree

Predict for test set:

- MSE for pruned: 1.922
- MSE for full: 1.945

So, pruning helped us avoid some overfitting.

Example: Regression Tree

Predict for test set:

- MSE for pruned: 1.922
- MSE for full: 1.945

So, pruning helped us avoid some overfitting.

(Typical pred error of $\sqrt{1.922} \approx 1.386$)

Example: Regression Tree

Predict for test set:

- MSE for pruned: 1.922
- MSE for full: 1.945

So, pruning helped us avoid some overfitting.

(Typical pred error of $\sqrt{1.922} \approx 1.386$)

(Bigger than IQR of 1.183, but range covers $[6.98, 16.3]$.)

Example: Regression Tree

Predict for test set:

- MSE for pruned: 1.922
- MSE for full: 1.945

So, pruning helped us avoid some overfitting.

(Typical pred error of $\sqrt{1.922} \approx 1.386$)

(Bigger than IQR of 1.183, but range covers $[6.98, 16.3]$.)

(Pretty good for 1 split!?)

Random Forests

Next: random forest algorithm

Random Forests

Next: random forest algorithm

Ensemble learning algorithms:

Random Forests

Next: random forest algorithm

Ensemble learning algorithms:

- Boosting: models build on prior models $\rightsquigarrow$ pick feature, predict, upweight mispredicted data, ... Do several times and combine.

Random Forests

Next: random forest algorithm

Ensemble learning algorithms:

- Boosting: models build on prior models $\rightsquigarrow$ pick feature, predict, upweight mispredicted data, ... Do several times and combine.
- Bagging: (random select units, model) $\rightarrow$ many times. No building.

Random Forests

Next: random forest algorithm

Ensemble learning algorithms:

- Boosting: models build on prior models $\rightsquigarrow$ pick feature, predict, upweight mispredicted data, ... Do several times and combine.
- Bagging: (random select units, model) $\rightarrow$ many times. No building.

Random Forests

Next: random forest algorithm

Ensemble learning algorithms:

- Boosting: models build on prior models $\rightsquigarrow$ pick feature, predict, upweight mispredicted data, ... Do several times and combine.
- Bagging: (random select units, model) $\rightarrow$ many times. No building.

Random Forests are bagging algorithms.

Random Forests

Next: random forest algorithm

Ensemble learning algorithms:

- Boosting: models build on prior models $\rightsquigarrow$ pick feature, predict, upweight mispredicted data, ... Do several times and combine.
- Bagging: (random select units, model) $\to$ many times. No building.

Random Forests are bagging algorithms.

Bagging: *b*ootstrap *agg*regation

Random Forests

Why bag?

Random Forests

Why bag?

- Trees are low bias, high variance
  (diff answers, depend on data split)

Random Forests

Why bag?

- Trees are low bias, high variance
  (diff answers, depend on data split)
- Bagging averages over data subsets, reducing variance

Random Forests

Why bag?

- Trees are low bias, high variance (diff answers, depend on data split)
- Bagging averages over data subsets, reducing variance
- (Linear regression: lower variance)

Random Forests

Random forests: decorrelated, bagged trees

Random Forests

Random forests: decorrelated, bagged trees

- Take bootstrapped training subsample

Random Forests

Random forests: decorrelated, bagged trees

- Take bootstrapped training subsample
- Build deep tree. At each split, *randomly sample* m of p predictors, build split from only those m.

Random Forests

Random forests: decorrelated, bagged trees

- Take bootstrapped training subsample
- Build deep tree. At each split, *randomly sample* m of p predictors, build split from only those m.
- (Often choose $m \approx \sqrt{p}$)

Random Forests

Random forests: decorrelated, bagged trees

- Take bootstrapped training subsample
- Build deep tree. At each split, *randomly sample* m of p predictors, build split from only those m.
- (Often choose $m \approx \sqrt{p}$)
- So, different splits consider different predictors

Random Forests

Random forests: decorrelated, bagged trees

- Take bootstrapped training subsample
- Build deep tree. At each split, *randomly sample* m of p predictors, build split from only those m.
- (Often choose $m \approx \sqrt{p}$)
- So, different splits consider different predictors
- So, trees will look very different to each other

Example: Random Forests

```r
library(randomForest)

# Full bag:
bag_mps <- randomForest(ln.net ~ ., data = mp_train,
                        ntree = 500, mtry = 4,
                        importance = TRUE)

# Decorrelate:
rf_mps <- randomForest(ln.net ~ ., data = mp_train,
                       ntree = 500, mtry = 2,
                      importance = TRUE)
```

Example: Random Forests

Predict:

```r
preds_bag <- predict(bag_mps, newdata = mp_test)

preds_rf <- predict(rf_mps, newdata = mp_test)
```

- MSE for RF: 1.995
- MSE for full bag: 2.536

Example: Random Forests

Predict:

```
preds_bag <- predict(bag_mps, newdata = mp_test)

preds_rf <- predict(rf_mps, newdata = mp_test)
```

- MSE for RF: 1.995
- MSE for full bag: 2.536

So, decorrelating helped us avoid some overfitting to each bootstrap subsample (and thus, reduced variance).

Example: Random Forests

Predict:

```
preds_bag <- predict(bag_mps, newdata = mp_test)

preds_rf <- predict(rf_mps, newdata = mp_test)
```

- MSE for RF: 1.995
- MSE for full bag: 2.536

So, decorrelating helped us avoid some overfitting to each bootstrap subsample (and thus, reduced variance).

(Typical pred error of $\sqrt{1.995} \approx 1.412$)

Example: Random Forests

Predict:

```r
preds_bag <- predict(bag_mps, newdata = mp_test)

preds_rf <- predict(rf_mps, newdata = mp_test)
```

- MSE for RF: 1.995
- MSE for full bag: 2.536

So, decorrelating helped us avoid some overfitting to each bootstrap subsample (and thus, reduced variance).

(Typical pred error of $\sqrt{1.995} \approx 1.412$)

(Bigger than IQR of 1.183, but range covers $[6.98, 16.3]$.)

Heterogeneous Treatment Effects

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects
- Notationally, often assume $\tau_i = \tau \quad \forall i$

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects
- Notationally, often assume $\tau_i = \tau \quad \forall i$
- But, *heterogeneous* effects often of central interest

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects
- Notationally, often assume $\tau_i = \tau \quad \forall i$
- But, *heterogeneous* effects often of central interest
- Different effects for different groups

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects
- Notationally, often assume $\tau_i = \tau \quad \forall i$
- But, *heterogeneous* effects often of central interest
- Different effects for different groups
  - Subgroup variability (research)

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects
- Notationally, often assume $\tau_i = \tau \quad \forall i$
- But, *heterogeneous* effects often of central interest
- Different effects for different groups
    - Subgroup variability (research)
    - Targeting resources (campaigns, marketing)

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects
- Notationally, often assume $\tau_i = \tau \quad \forall i$
- But, *heterogeneous* effects often of central interest
- Different effects for different groups
  - Subgroup variability (research)
  - Targeting resources (campaigns, marketing)
  - Constituency effects (public policy)

Homogeneous and Heterogeneous Effects

- Most causal inference starts at *average treatment effects*
- Average may be interesting on its own, ...but often masks assumption of *homogeneous* effects
- Notationally, often assume $\tau_i = \tau \quad \forall i$
- But, *heterogeneous* effects often of central interest
- Different effects for different groups
  - Subgroup variability (research)
  - Targeting resources (campaigns, marketing)
  - Constituency effects (public policy)
- Notationally, $\exists i : \tau_i \neq \tau$

Homogeneous and Heterogeneous Effects: Estimation

Homogeneous effects:

$$\text{Outcome} = \beta_0 + \beta_1 \text{Treatment} + \epsilon$$

Homogeneous and Heterogeneous Effects: Estimation

Homogeneous effects:

$$\text{Outcome} = \beta_0 + \beta_1 \text{Treatment} + \epsilon$$

```r
lm_out <- lm(ln.net ~ is_winner, data = mps)
lm_out
```

```
Call:
lm(formula = ln.net ~ is_winner, data = mps)

Coefficients:
(Intercept)    is_winner
    12.2464       0.5176
```

Homogeneous and Heterogeneous Effects: Estimation

Homogeneous effects:

```r
t.test(ln.net ~ is_winner, data = mps)
```

```

    Welch Two Sample t-test

data:  ln.net by is_winner
t = -3.9552, df = 287.65, p-value = 9.636e-05
alternative hypothesis: true difference in means betwee
95 percent confidence interval:
 -0.7751044 -0.2599998
sample estimates:
mean in group 0 mean in group 1
       12.24641        12.76396
```

Homogeneous and Heterogeneous Effects: Estimation

Homogeneous effects:

$$\text{Outcome} = \beta_0 + \beta_1 \text{Treatment} + \sum \beta_j X_j + \epsilon$$

Homogeneous and Heterogeneous Effects: Estimation

Homogeneous effects:

$$\text{Outcome} = \beta_0 + \beta_1 \text{Treatment} + \sum \beta_j X_j + \epsilon$$

```
lm_out <- lm(ln.net ~ is_winner + is_labour +
               is_london + age, data = mps)
lm_out
```

```
Call:
lm(formula = ln.net ~ is_winner + is_labour + is_london + a
    data = mps)

Coefficients:
(Intercept)    is_winner    is_labour    is_london
  12.078838     0.398818    -0.477549     0.161134     0.00
```

Homogeneous and Heterogeneous Effects: Estimation

Homogeneous effects:

```
lm_lin(ln.net ~ is_winner, covariates = ~ is_labour + is_lo
```

```
                          Estimate  Std. Error       t valu
(Intercept)           1.226687e+01 0.078894901 155.4836617
is_winner             3.459885e-01 0.131207672   2.6369536
is_labour_c          -1.613663e-01 0.152608515  -1.0573871
is_london_c           2.427360e-01 0.250214401   0.9701118
age_c                 4.740367e-03 0.007031323   0.6741786
is_winner:is_labour_c -9.104022e-01 0.264395760  -3.4433313
is_winner:is_london_c -8.847770e-02 0.426241818  -0.2075763
is_winner:age_c      -4.778657e-05 0.012753800  -0.0037468
                         CI Lower    CI Upper  DF
(Intercept)           12.111785723 12.42195044 416
is_winner              0.088075873  0.60390123 416
is_labour_c           -0.461346226  0.13861367 416
is_london_c           -0.249106208  0.73457813 416
```

CATEs: Conditional ATEs

- *Conditional average treatment effect* (CATE): avg treatment effect for subset of population

CATEs: Conditional ATEs

- *Conditional average treatment effect* (CATE): avg treatment effect for subset of population
- Sometimes "CACE"

CATEs: Conditional ATEs

- *Conditional average treatment effect* (CATE): avg treatment effect for subset of population
- Sometimes "CACE"
- Inference: not "evidence against $\text{TE} = 0$?", but "evidence against $\text{CATE}_1 = \text{CATE}_2$?"

Homogeneous and Heterogeneous Effects: Estimation

Heterogeneous effects:

$$\text{Outcome} = \beta_0 + \beta_1 \text{Treatment} + \beta_2 \text{Group} + \beta_3 \text{Treatment} \cdot \text{Group} + \epsilon$$

Homogeneous and Heterogeneous Effects: Estimation

Heterogeneous effects:

$$\text{Outcome} = \beta_0 + \beta_1 \text{Treatment} + \beta_2 \text{Group} + \beta_3 \text{Treatment} \cdot \text{Group} + \epsilon$$

- β_1 gives TE for `Group == 0`

Homogeneous and Heterogeneous Effects: Estimation

Heterogeneous effects:

$$\text{Outcome} = \beta_0 + \beta_1 \text{Treatment} + \beta_2 \text{Group} + \beta_3 \text{Treatment} \cdot \text{Group} + \epsilon$$

- β_1 gives TE for `Group == 0`
- $\beta_1 + \beta_3$ gives TE for `Group == 1`

Homogeneous and Heterogeneous Effects: Estimation

Heterogeneous effects:

```
lm_out <- lm(ln.net ~ is_winner * is_labour +
               is_london + age, data = mps)
coef(lm_out) |> round(3)
```

```
        (Intercept)           is_winner           is_labou
             11.959               0.780              -0.16
                age is_winner:is_labour
              0.005              -0.914
```

Causal Forests

- Our regression trees had terminal nodes ("leaves") that were sufficiently homogeneous for prediction.

Causal Forests

- Our regression trees had terminal nodes ("leaves") that were sufficiently homogeneous for prediction.
- Use $\hat{y}_{R_j}$ as pred value for obs in R_j

Causal Forests

- Our regression trees had terminal nodes ("leaves") that were sufficiently homogeneous for prediction.
- Use $\hat{y}_{R_j}$ as pred value for obs in R_j
- (Tory, winner, London $\rightarrow$ 13.84)

Causal Forests

- Our regression trees had terminal nodes ("leaves") that were sufficiently homogeneous for prediction.
- Use $\hat{y}_{R_j}$ as pred value for obs in R_j
- (Tory, winner, London $\rightarrow$ 13.84)
- $$\hat{y}_{R_j} = \frac{1}{|R_j|} \sum_{i \in R_j} Y_i$$

Causal Forests

- Similarly, consider each leaf R_j small, homogeneous enough that potential outcomes independent

Causal Forests

- Similarly, consider each leaf R_j small, homogeneous enough that potential outcomes independent
- Treateds in R_j provide good estimates of what controls in R_j would have done under treatment

Causal Forests

- Similarly, consider each leaf R_j small, homogeneous enough that potential outcomes independent
- Treateds in R_j provide good estimates of what controls in R_j would have done under treatment
- Controls in R_j provide good estimates of what treateds in R_j would have done under control

Causal Forests

- Similarly, consider each leaf R_j small, homogeneous enough that potential outcomes independent
- Treateds in R_j provide good estimates of what controls in R_j would have done under treatment
- Controls in R_j provide good estimates of what treateds in R_j would have done under control
- Assignment w/in leaf R_j is as-good-as-random

Causal Forests

- Similarly, consider each leaf R_j small, homogeneous enough that potential outcomes independent
- Treateds in R_j provide good estimates of what controls in R_j would have done under treatment
- Controls in R_j provide good estimates of what treateds in R_j would have done under control
- Assignment w/in leaf R_j is as-good-as-random
- I.e., each leaf contains an experiment

Causal Forests

- Similarly, consider each leaf R_j small, homogeneous enough that potential outcomes independent
- Treateds in R_j provide good estimates of what controls in R_j would have done under treatment
- Controls in R_j provide good estimates of what treateds in R_j would have done under control
- Assignment w/in leaf R_j is as-good-as-random
- I.e., each leaf contains an experiment

Causal Forests

- Similarly, consider each leaf R_j small, homogeneous enough that potential outcomes independent
- Treateds in R_j provide good estimates of what controls in R_j would have done under treatment
- Controls in R_j provide good estimates of what treateds in R_j would have done under control
- Assignment w/in leaf R_j is as-good-as-random
- I.e., each leaf contains an experiment

$$Y(0), Y(1) \perp\!\!\!\perp T | \mathbf{X}$$

Causal Forests

- Let $\{T, R_j\} = \{i : T_i = 1, i \in R_j\}$ (Tr obs in R_j)

Causal Forests

- Let $\{T, R_j\} = \{i : T_i = 1, i \in R_j\}$ (Tr obs in R_j)
- Let $\{C, R_j\} = \{i : T_i = 0, i \in R_j\}$ (Co obs in R_j)

Causal Forests

- Let $\{T, R_j\} = \{i : T_i = 1, i \in R_j\}$ (Tr obs in R_j)
- Let $\{C, R_j\} = \{i : T_i = 0, i \in R_j\}$ (Co obs in R_j)
- Natural estimation of

Causal Forests

- Let $\{T, R_j\} = \{i : T_i = 1, i \in R_j\}$ (Tr obs in R_j)
- Let $\{C, R_j\} = \{i : T_i = 0, i \in R_j\}$ (Co obs in R_j)
- Natural estimation of

Causal Forests

- Let $\{T, R_j\} = \{i : T_i = 1, i \in R_j\}$ (Tr obs in R_j)
- Let $\{C, R_j\} = \{i : T_i = 0, i \in R_j\}$ (Co obs in R_j)
- Natural estimation of

$$\hat{\hat{\tau}}_{R_j} = \frac{1}{|\{T, R_j\}|} \sum_{\{T, R_j\}} Y_i - \frac{1}{|\{C, R_j\}|} \sum_{\{C, R_j\}} Y_i$$

Causal Forests

- Let $\{T, R_j\} = \{i : T_i = 1, i \in R_j\}$ (Tr obs in R_j)
- Let $\{C, R_j\} = \{i : T_i = 0, i \in R_j\}$ (Co obs in R_j)
- Natural estimation of

$$\hat{\hat{\tau}}_{R_j} = \frac{1}{|\{T, R_j\}|} \sum_{\{T, R_j\}} Y_i - \frac{1}{|\{C, R_j\}|} \sum_{\{C, R_j\}} Y_i$$

So, we can use RF methods to estimate conditional (heterogeneous) treatment effects, CATEs.

Causal Forests

- Let $\{T, R_j\} = \{i : T_i = 1, i \in R_j\}$ (Tr obs in R_j)
- Let $\{C, R_j\} = \{i : T_i = 0, i \in R_j\}$ (Co obs in R_j)
- Natural estimation of

$$\hat{\hat{\tau}}_{R_j} = \frac{1}{|\{T, R_j\}|} \sum_{\{T, R_j\}} Y_i - \frac{1}{|\{C, R_j\}|} \sum_{\{C, R_j\}} Y_i$$

So, we can use RF methods to estimate conditional (heterogeneous) treatment effects, CATEs.

☺

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$
- But **not both**!

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$
- But **not both**!
- One way: "Double-sample" causal trees

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$
- But **not both**!
- One way: "Double-sample" causal trees
  - Split training data into $\mathcal{I}$ and $\mathcal{J}$

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$
- But **not both**!
- One way: "Double-sample" causal trees
  - Split training data into $\mathcal{I}$ and $\mathcal{J}$
  - Splits chosen to maximise variance on $\hat{\hat{\tau}}$ for $i \in \mathcal{J}$

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$
- But **not both**!
- One way: “Double-sample” causal trees
  - Split training data into $\mathcal{I}$ and $\mathcal{J}$
  - Splits chosen to maximise variance on $\hat{\hat{\tau}}$ for $i \in \mathcal{J}$
  - Splitting cannot use y_i from $\mathcal{I}$

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$
- But **not both**!
- One way: "Double-sample" causal trees
  - Split training data into $\mathcal{I}$ and $\mathcal{J}$
  - Splits chosen to maximise variance on $\hat{\hat{\tau}}$ for $i \in \mathcal{J}$
  - Splitting cannot use y_i from $\mathcal{I}$
  - Prediction, estimation of $\hat{\hat{\tau}}$ uses only $\mathcal{I}$

Causal Forests: Honesty

- But need one more thing for asymptotics to work out ...
- Each tree must be *honest*
- Each unit i *either*
  - used to determine tree splits, *or*
  - used to estimate $\hat{\hat{\tau}}_{R_j}$
- But **not both**!
- One way: "Double-sample" causal trees
  - Split training data into $\mathcal{I}$ and $\mathcal{J}$
  - Splits chosen to maximise variance on $\hat{\hat{\tau}}$ for $i \in \mathcal{J}$
  - Splitting cannot use y_i from $\mathcal{I}$
  - Prediction, estimation of $\hat{\hat{\tau}}$ uses only $\mathcal{I}$
- Build a random forest (decorrelated deep trees picking from m predictors) of causal trees

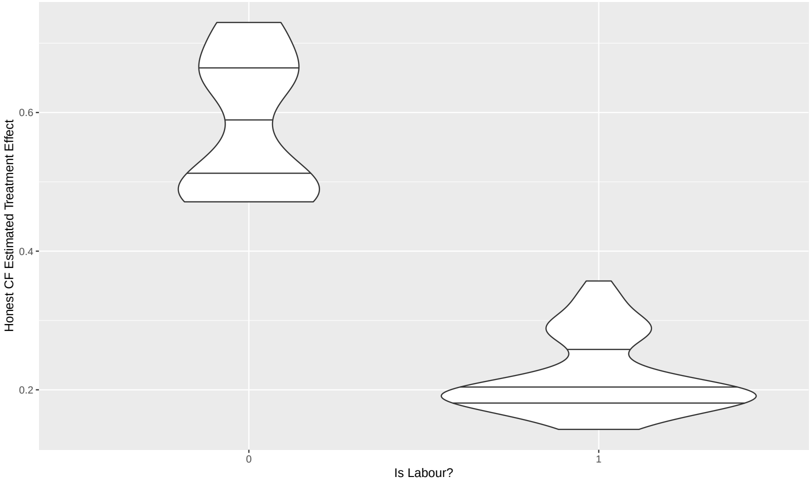
Example: Causal Forests

```r
library(grf)

X <- mp_train |> select(age, is_labour, is_london)

W <- mp_train |> select(is_winner) |>
  unlist() |> as.numeric()

Y <- mp_train |> select(ln.net) |> unlist()

cf_out <- causal_forest(X, Y, W)
```

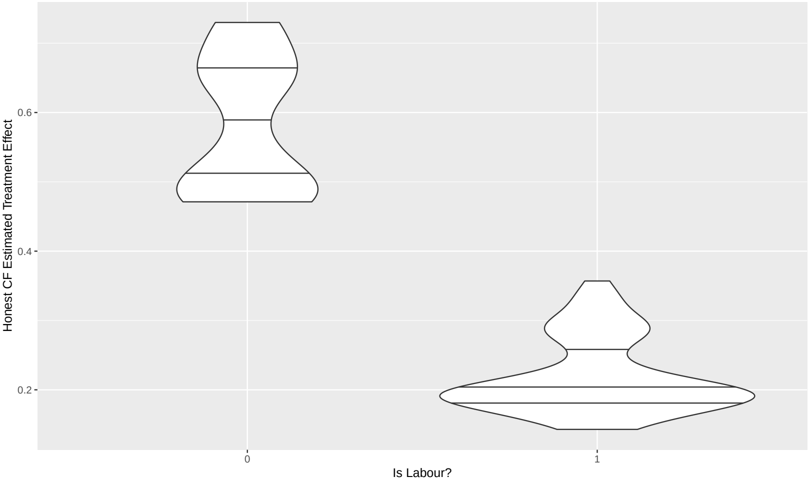
Example: Causal Forests

```
X_test <- mp_test |> select(age, is_labour, is_london)

cf_pred_est_var <- predict(cf_out, X_test,
                           estimate.variance = TRUE)

cf_preds <- cf_pred_est_var$predictions

df_cf <- tibble(X_test,
                cf_te = cf_preds,
                cf_se = sqrt(cf_pred_est_var$variance.e
                te_1se_lower = cf_te - cf_se,
                te_1se_upper = cf_te + cf_se)
```

Example: Causal Forests Results, Party

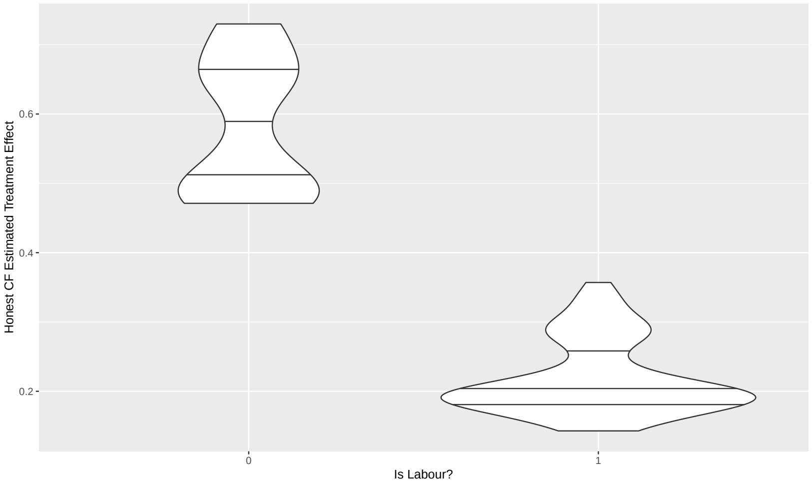


Example: Causal Forests Results, Party



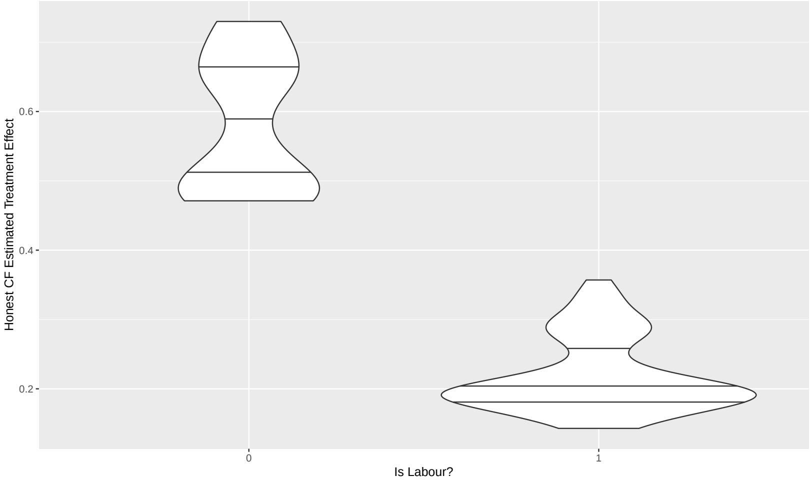
- Mean CF TE, Tory: 0.58

Example: Causal Forests Results, Party



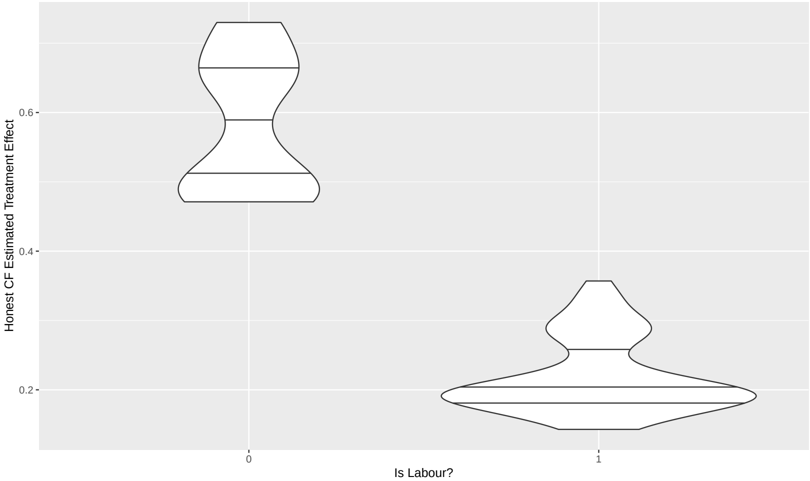
- Mean CF TE, Tory: $0.58 \rightsquigarrow £192,000$

Example: Causal Forests Results, Party



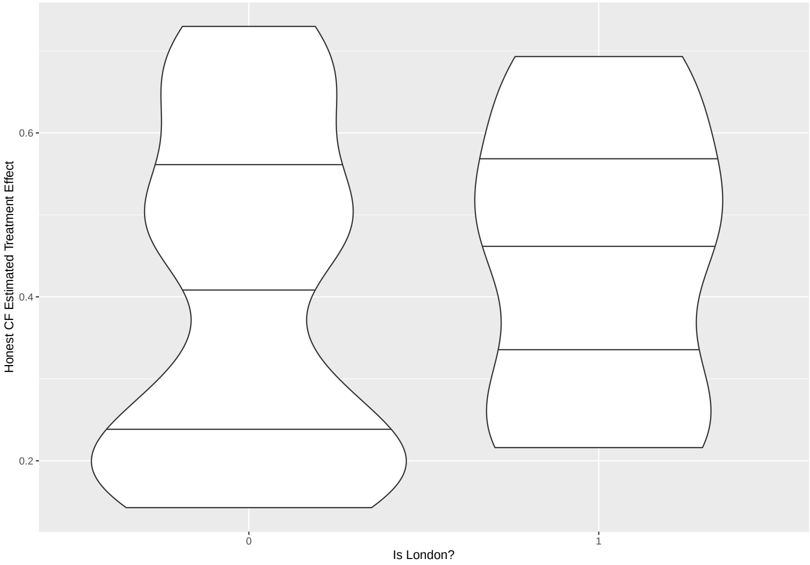
- Mean CF TE, Tory: $0.58 \rightsquigarrow$ £$192,000$
- Mean CF TE, Labour: 0.219

Example: Causal Forests Results, Party

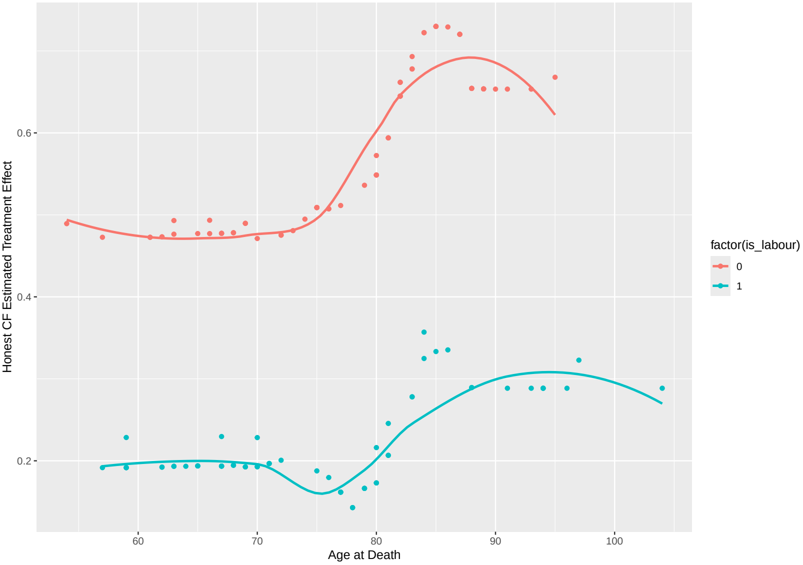


- Mean CF TE, Tory: $0.58 \rightsquigarrow$ £$192,000$
- Mean CF TE, Labour: $0.219 \rightsquigarrow$ £$60,000$

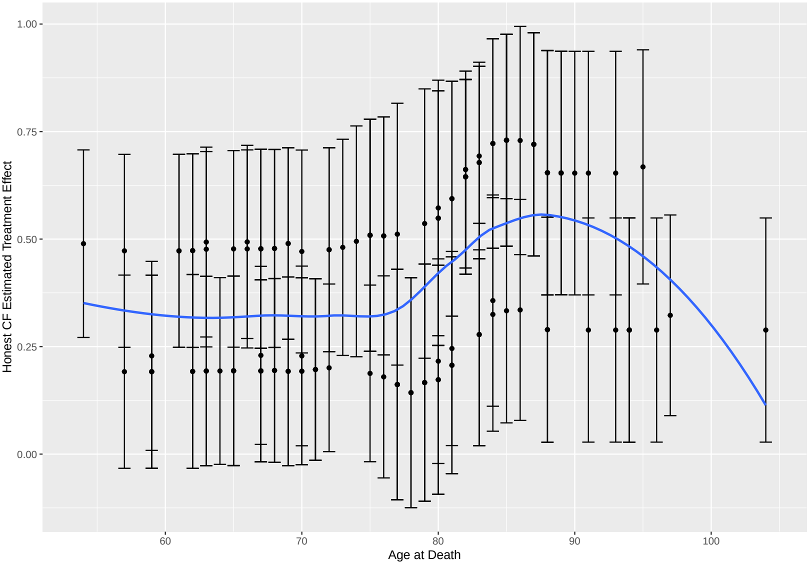
Example: Causal Forests Results, London



Example: Causal Forests Results, Age



Example: Causal Forests Results, Age



Variable Selection

Feature Selection

- Wrappers: pick subset of covars, train on data (estimate model), test on hold-out, score predictions. Keep best-scoring subset.

Feature Selection

- Wrappers: pick subset of covars, train on data (estimate model), test on hold-out, score predictions. Keep best-scoring subset.
- Filters: correlate covars with outcome. Keep strongest.

Feature Selection

- Wrappers: pick subset of covars, train on data (estimate model), test on hold-out, score predictions. Keep best-scoring subset.
- Filters: correlate covars with outcome. Keep strongest.
- Embeds: select features and estimate model at same time. Penalize using more predictors.

Regularization Methods

OLS reminder

Minimize SSR:

$$\mathrm{argmin}_{\beta} \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2$$

$$\mathrm{argmin}_{\beta} \sum_{i=1}^{n} \left(\mathbf{y} - \mathbf{X}\hat{\beta}\right)^2$$

Embedded Regularization Methods

L1 regularization: the LASSO (Least Absolute Shrinkage and Selection Operator)

$$\text{argmin}_{\beta} \left[\sum_{i=1}^{n} \left(y_i - \mathbf{X}\hat{\beta} \right)^2 + \lambda \sum_{j=1}^{k} |\hat{\beta}_j| \right]$$

Embedded Regularization Methods

L1 regularization: the LASSO (Least Absolute Shrinkage and Selection Operator)

$$\text{argmin}_\beta \left[\sum_{i=1}^{n} \left(y_i - \mathbf{X}\hat{\beta} \right)^2 + \lambda \sum_{j=1}^{k} |\hat{\beta}_j| \right]$$

L2 regularization: Ridge regression

$$\text{argmin}_\beta \left[\sum_{i=1}^{n} \left(y_i - \mathbf{X}\hat{\beta} \right)^2 + \lambda \sum_{j=1}^{k} \hat{\beta}_j^2 \right]$$

Embedded Regularization Methods

Mix L1 and L2: Elastic net

$$\operatorname{argmin}_\beta \left(\frac{\sum_{i=1}^{n} \left(y_i - \mathbf{X}\hat{\beta}\right)^2}{2n} + \lambda \left[\alpha \sum_{j=1}^{k} |\hat{\beta}_j| + \frac{1-\alpha}{2} \sum_{j=1}^{k} \hat{\beta}_j^2 \right] \right)$$

Embedded Regularization Methods

Mix L1 and L2: Elastic net

$$\operatorname{argmin}_{\beta}\left(\frac{\sum_{i=1}^{n}\left(y_i-\mathbf{X}\hat{\beta}\right)^2}{2n}+\lambda\left[\alpha\sum_{j=1}^{k}|\hat{\beta}_j|+\frac{1-\alpha}{2}\sum_{j=1}^{k}\hat{\beta}_j^2\right]\right)$$

Regularized trees, ...

Embedded Regularization Methods

How to choose λ, α?

Embedded Regularization Methods

How to choose λ, α?

The LASSO

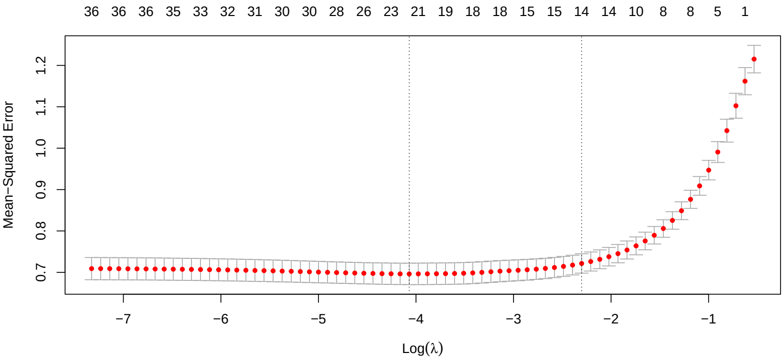
The LASSO

Cross-validation for λ:

```
df_lasso <- read_csv("~/Desktop/lasso.csv")

X <- as.matrix(df_lasso[, 2:ncol(df_lasso)])

Y <- as.matrix(df_lasso[, "y"])

library(glmnet)

cv_lasso <- cv.glmnet(X, Y, alpha = 1)
```

The LASSO

```
plot(cv_lasso)
```




```
cv_lasso$lambda.min
```

```
[1] 0.0170891
```

The LASSO

Implement:

```
lasso_out <- glmnet(X, Y, alpha = 1,
                    lambda = cv_lasso$lambda.min)

lasso_out
```

```
Call:  glmnet(x = X, y = Y, alpha = 1, lambda = cv_lasso$la

  Df  %Dev  Lambda
1 21 45.32 0.01709
```

The LASSO

Coefficients:

```r
coef_lasso <- coef(lasso_out)
round(coef_lasso, 3)
```

```
37 x 1 sparse Matrix of class "dgCMatrix"
                   s0
(Intercept)  0.000
x1           0.112
x2           0.095
x3           0.086
x4           0.147
x5           0.002
x6           0.063
x7           0.051
x8           0.074
x9           0.042
x10          .
x11
```

The LASSO

Coefficients:

```
round(coef_lasso[, ], 3)
```

```
(Intercept)          x1          x2          x3          x4
      0.000       0.112       0.095       0.086       0.147
         x6          x7          x8          x9         x10
      0.063       0.051       0.074       0.042       0.000
        x12         x13         x14         x15         x16
      0.039       0.000       0.026       0.000       0.000
        x18         x19         x20         x21         x22
      0.010       0.127      -0.015       0.030       0.000
        x24         x25         x26         x27         x28
      0.000       0.000       0.000       0.000      -0.010
        x30         x31         x32         x33         x34
      0.000       0.028       0.032       0.000      -0.041
        x36
      0.048
```

The LASSO

Implement, alternative λ:

```r
lasso_1se <- glmnet(X, Y, alpha = 1,
                    lambda = cv_lasso$lambda.1se)

coef(lasso_1se)
```

```
37 x 1 sparse Matrix of class "dgCMatrix"
                       s0
(Intercept) -0.0003034087
x1           0.1051188782
x2           0.0898842045
x3           0.0742522801
x4           0.1513883536
x5           .
x6           0.0603811184
x7           0.0389489143
x8           0.0575738993
x9           0.0374420416
```

The LASSO

Coefficients:

```r
round(coef(lasso_1se)[, ], 3)
```

```
(Intercept)          x1          x2          x3          x4
      0.000       0.105       0.090       0.074       0.15
         x6          x7          x8          x9          x1
      0.060       0.039       0.058       0.037       0.00
        x12         x13         x14         x15         x1
      0.029       0.000       0.008       0.000       0.00
        x18         x19         x20         x21         x2
      0.004       0.039       0.000       0.000       0.00
        x24         x25         x26         x27         x2
      0.000       0.000       0.000       0.000       0.00
        x30         x31         x32         x33         x3
      0.000       0.000       0.013       0.000       0.00
        x36
      0.030
```

The Double LASSO for Treatment Effects

The idea:

- covariates may $\rightsquigarrow Y$ or $\rightsquigarrow T$

The Double LASSO for Treatment Effects

The idea:

- covariates may $\rightsquigarrow Y$ or $\rightsquigarrow T$
- $\approx$ "double robust", "AIPW" estimators

The Double LASSO for Treatment Effects

The idea:

- covariates may $\rightsquigarrow Y$ or $\rightsquigarrow T$
- $\approx$ "double robust", "AIPW" estimators
- (different to just "doing LASSO twice" for regularization + shrinkage)

The Double LASSO for Treatment Effects

1. Model $Y = f(X)$ using LASSO

The Double LASSO for Treatment Effects

1. Model $Y = f(X)$ using LASSO
2. Model $T = f(X)$, using LASSO

The Double LASSO for Treatment Effects

1. Model $Y = f(X)$ using LASSO
2. Model $T = f(X)$, using LASSO
3. Let X_{LASSO} be set of imp covariates identified s.t. each $\beta_{X_{\text{LASSO}}} > 0$

The Double LASSO for Treatment Effects

1. Model $Y = f(X)$ using LASSO
2. Model $T = f(X)$, using LASSO
3. Let X_{LASSO} be set of imp covariates identified s.t. each $\beta_{X_{\text{LASSO}}} > 0$
4. Model $Y = T + X_{\text{LASSO}}$

The Double LASSO for Treatment Effects

1. Model $Y = f(X)$ using LASSO
2. Model $T = f(X)$, using LASSO
3. Let X_{LASSO} be set of imp covariates identified s.t. each $\beta_{X_{\text{LASSO}}} > 0$
4. Model $Y = T + X_{\text{LASSO}}$

The Double LASSO for Treatment Effects

```
library(hdm)
```

```
data(social)

df_social <- social |> mutate(is_male = if_else(sex == "mal
                              age = 2006 - yearofbirth,
                              is_neighbors = if_else(messag
  filter(messages %in% c("Neighbors", "Control"))

rlasso_out <- rlassoATE(primary2006 ~ age + is_male + prima
```

The Double LASSO for Treatment Effects

```r
summary(rlasso_out)
```

```
Estimation and significance testing of the treatment effect
Type: ATE
Bootstrap: not applicable
     coeff.      se. t-value p-value
TE 0.080091 0.002625   30.51  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```

The Double LASSO for Treatment Effects

```r
X <- as.matrix(df_social[, c("age", "is_male", "primary2004

Y <- as.matrix(df_social[, "primary2006"])

D <- as.matrix(df_social[, "is_neighbors"])

summary(rlassoEffects(X, Y, method = "double selection"))
```

```
[1] "Estimates and significance testing of the effect of ta
              Estimate. Std. Error t value Pr(>|t|)
age           0.0038449  0.0000681  56.456  < 2e-16 ***
is_male       0.0086763  0.0018889   4.593 4.36e-06 ***
primary2004   0.1474364  0.0019924  74.000  < 2e-16 ***
hhsize        0.0004260  0.0012618   0.338    0.736
is_neighbors  0.0802361  0.0026278  30.534  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```

R packages for Regularization, etc.

- `glmnet`
- `caret`

See also `tidymodels`, `parsnip`, ...

Thanks!

rtm@american.edu
www.ryantmoore.org

References I

Breiman, Leo. 2001. "Statistical Modeling: The Two Cultures." *Statistical Science* 16 (3): 199–215. http://www.jstor.org/stable/2676681.

D'Agostino McGowan, Lucy. 2023. *quartets: Datasets to Help Teach Statistics*. https://r-causal.github.io/quartets/.

Hernán, Miguel A. 2018. "The c-Word: Scientific Euphemisms Do Not Improve Causal Inference from Observational Data." *American Journal of Public Health* 108 (5): 616–19.

Hernán, Miguel A., John Hsu, and Brian Healy. 2019. "A Second Chance to Get Causal Inference Right: A Classification of Data Science Tasks." *CHANCE* 32 (1): 42–49. https://doi.org/10.1080/09332480.2019.1579578.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2021. *An Introduction to Statistical Learning with Applications in R*. 2nd ed. New York, NY: Springer.