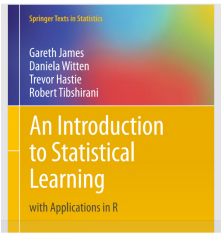
Causal machine learning Turing Masterclass

K. DiazOrdaz, @karlado

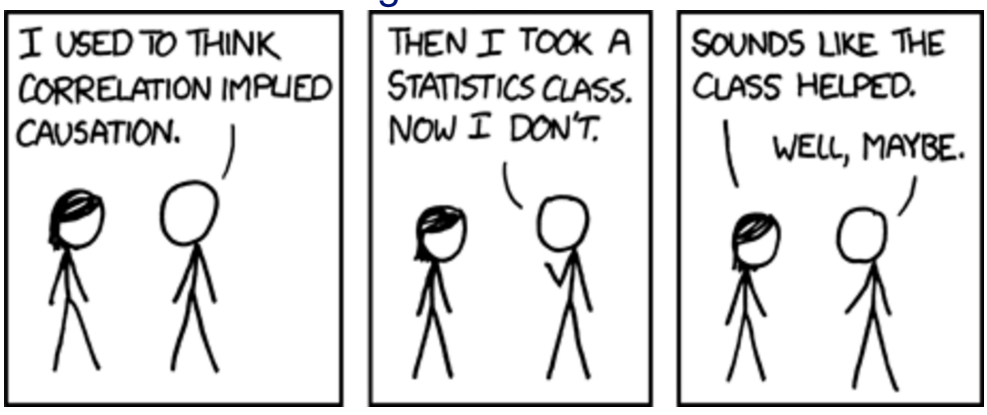
Session 1, Part 1: Introduction

@LSHTMstatmethod

This talk is NOT about



- Providing a tutorial in Machine Learning
- Methods for Learning Causal Structure from the data



This talk IS about

- Using machine learning within a causal model assumed from external knowledge,
- to estimate a causal effect
- using observational data

Why may we want to do this?

- It can help attenuate biases
- Big data settings: machine learning indispensable for variable selection (etc...)
- how to do it correctly

Motivation

- Often we wish to know what would be the outcome if a unit had possibly contrary to fact been exposed vs not exposed to a particular thing
- This is the causal effect of treatment
  - pharmaco-epi: what is the effect of taking a drug on risk of heart attack?
  - Social sciences: effect of parental education on children's low birth weight
  - Policy evaluation: what is the effect of health insurance on infant mortality
- Ideally, we'd randomise the exposure (trials)
- in reality, trials not always feasible: use observational data

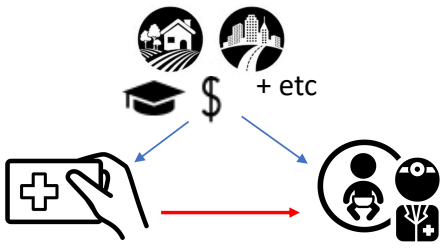
Example: Indonesia Health Insurance Policy

- Policy makers in LMIC : goal of universal health insurance coverage
- Rigorous empirical evidence on the causal impact of publicly provided health insurance on infant health is scarce.
- Infant mortality had decreased, also health care utilisation has increased
- does having insurance make women more likely to give birth assisted by a health professional?

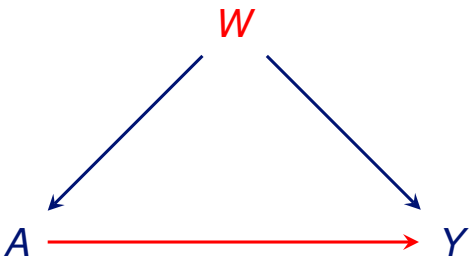
The Data: Indonesian Family Life Survey

What is the causal effect of health insurance on assisted birth by a health professional?

- Birth-level dataset ($n = 10985$)
- Outcome Y: assisted birth $n = 8574$
- Treatment A: $n = 1053$ of the women had contributory health insurance
- 34 baseline variables W, (mother's, household and community characteristics)
- uses previous waves (2002, 2007, 2014) to respect temporal ordering



Average treatment effect (ATE)



- Estimand of interest : ATE $= E\left(Y^1\right) - E\left(Y^0\right)$.
- It can be identified from observational data under assumptions:
  - **No interference**
  - **Consistency** $A_i = a \Rightarrow Y_i = Y_i(a)$
    For those who actually received exposure level a, their observed outcome is the same as what it would have been had they received exposure level a via the hypothetical intervention we have in mind
  - **Conditional (mean) exchangeability (given W)**, a.k.a. "no unobserved confounding". Conditional on **W**, the actual exposure level A is independent of each of the POs

ATE

- Consider a single discrete confounder W. Under the identification assumptions:

$$\text{ATE}_w = E\left(Y \,|A=1, W=w\right) - E\left(Y \,|A=0, W=w\right) = \beta_w$$

- ATE_w are conditional causal effects
- the population average causal effect ATE $=$

$$\sum_w \left\{E\left(Y \,|A=1, W=w\right) - E\left(Y \,|A=0, W=w\right)\right\} Pr\left(W=w\right)$$

- this is known as *standardisation* of the conditional effects. Also known as G-computation formula

Estimation 1: g-computation / outcome regression

substitution estimator

- These conditional expectations need to be estimated. In the continuous outcome case by β:

$$E(Y|A,W) = \alpha + \beta A + \gamma^T W$$

- and in non-linear models, we need to marginalise over the distribution of the variable we need to adjust for to get the marginal ATE
- assumes the regression model is correctly specified
- In principle can be checked from the data: add terms, if needed
- Sensitive to extrapolation and over-fitting

G-computation in practice

algorithm

1. Model Y given W separately by treatment level a, e.g. in the treated and then the untreated

$$E(Y|A=a,W=w)$$

2. Use these fitted models to predict the POs Y^a of each individual on the basis of their W.
3. Average these POs over the observed distribution of W.
4. Take the difference of these mean POs to estimate ATE.

Estimation 2: propensity score IPW

- instead of specifying the outcome model $E(Y|A,W)$,
- we will specify a model for $E(A|W)$, known as the propensity score $g(W) = Pr(A = 1|W)$
- To estimate $E(Y^a)$:
  1. evaluate $g(A = a|W)$
  2. average outcomes weighting by $g(A = a|W)^{-1}$
  3. Take the difference of the averages to estimate the ATE.
- assumes positivity $0 < P(A_i = 1|W_i = w) < 1$, for all w.
- relies on the PS being correctly specified
- can be unstable, when weights are large (strong confounding)

"All models are wrong..."

Observation 1: George Box

"All models are wrong, but some are useful."

How to change this?

- Having a “correct model” is an assumption for valid inference
- Can we use machine learning to estimate the conditional expectations needed in either g-computation or IPW and attenuate model mis-specification?

Supervised machine learning (ML) in 1 slide

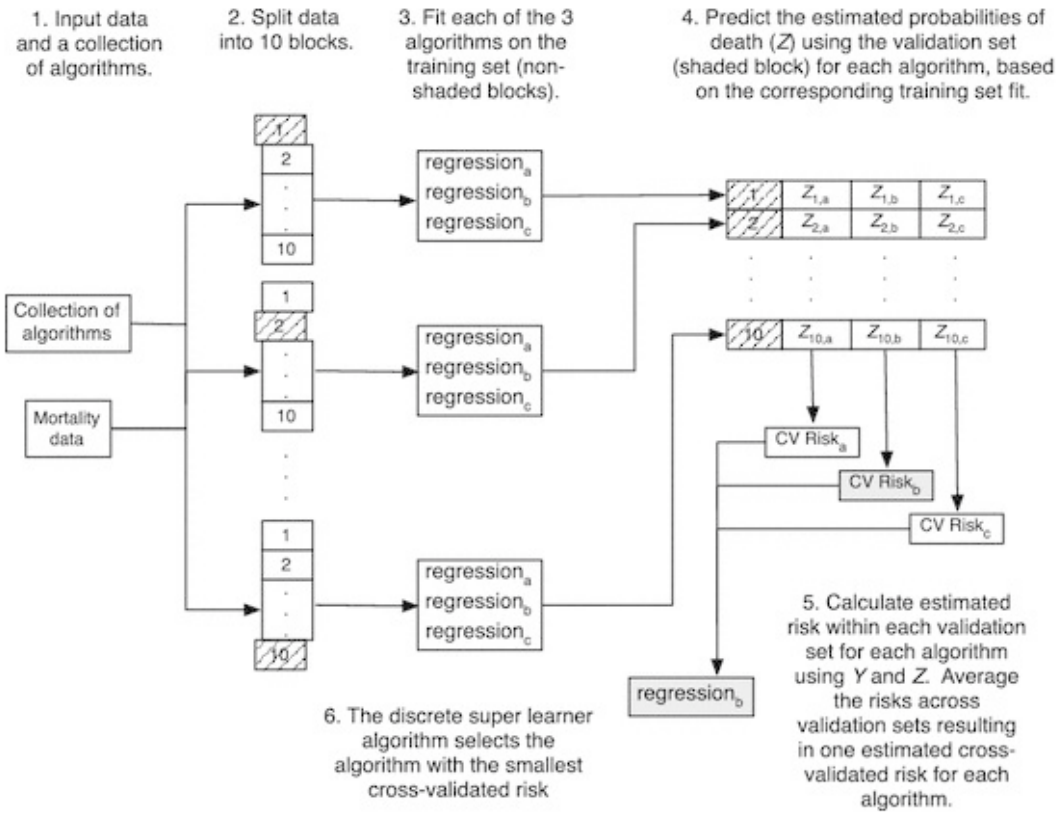
- Supervised ML aims to fit a prediction algorithm $f(W)$ for the “target” (outcome) Y using the features (covariates) W
- A good prediction algorithm $f(W)$ accurately predicts Y for future (out-of-sample) observations
- Aim: to minimise the prediction error, e.g. the expected Mean Squared Error among data not used for fitting the model
- we want to prevent over-fitting, so ML algorithms usually involve a *regulariser*
- there are many ML algorithms, which one to use depends mostly on the type of outcome we are trying to predict.

How to change this?

- Let's use machine learning to estimate the conditional expectations needed in either g-computation or IPW
- instead of choosing the best ML algorithm
- I will do this using stacked generalisation, in particular the Super Learner

Super Learner (SL) : Ensemble or stacking learner

Taken from "Targeted learning" book, vdL 2011



Super Learner (SL)

- uses cross-validation to estimate the performance of multiple algorithms
- Performance: e.g. MSE, on the validation set (risk)
- Discrete SL chooses the algorithm with smallest CV-risk
- SL runs a stacked regression: e.g. using non-negative least squares and constraining coef. to sum to 1

$$E[Y] = \alpha_1 \widehat{Y}_{ml_1} + \alpha_2 \widehat{Y}_{ml_2} + \alpha_3 \widehat{Y}_{ml_3}$$

- this creates an optimal weighted average of the predictions obtained by each learner included in its library using the test data performance
- SL is asymptotically as accurate as the best possible prediction algorithm included in the library

Machine learning applied to the Indonesia example

- We could use Super Learner to predict the POs for all units under the treatment and control (insured vs not)
- Then “plug” these estimates into the g-computation formula
- and also to predict the propensity score of being insured
- Note: We use an extra layer of cross validation around the SL procedure to (a) tune the learners parameters and (b) estimate of the performance of the ensemble itself

ML algorithms used in SL library in the example

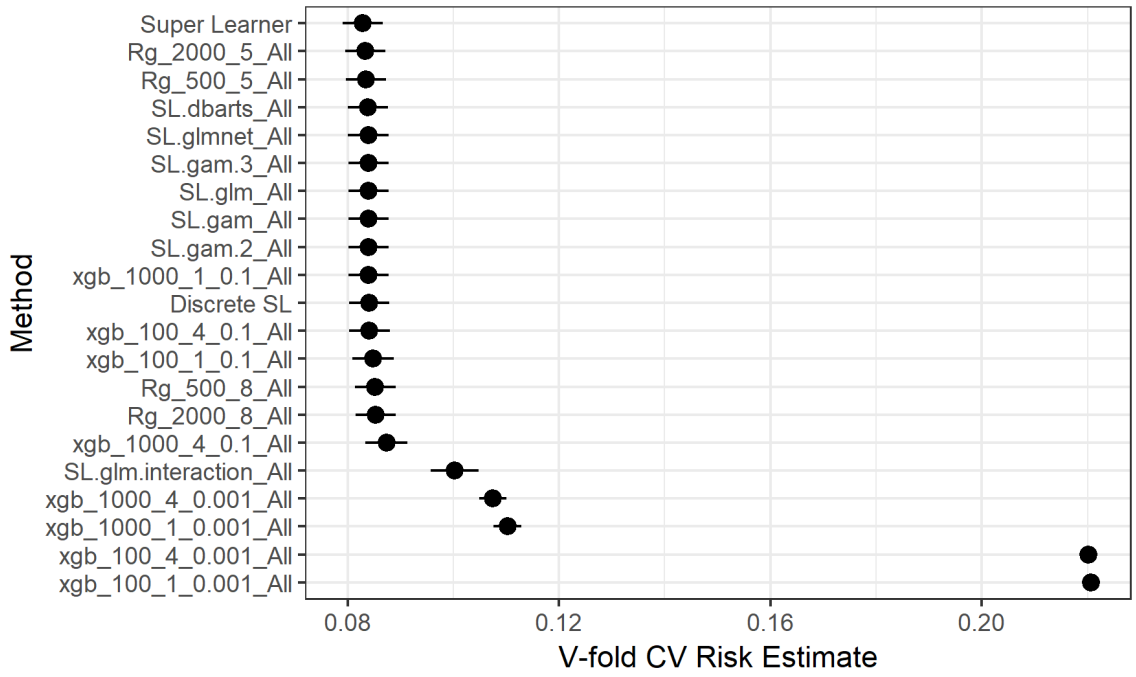
Algorithm	Description
glm	GLM with logit link for PS and outcome models
gam	Generalised additive models with 2, 3, 4, and 5 knots.
Lasso	*budget* c s.t. regression coefficients $\sum_{j=1}^{d} \|\beta_j\|_1 = c$.
Random forests	number of trees (500, 2000), number of covariates to split on (5 and 8),
boosting	number of trees (100 and 1000), shrinkage (0.001 and 0.1) and maximum tree depth (1 and 4)
BART	Bayesian additive regression trees

Application to the Indonesia data

All baseline variables included in the models

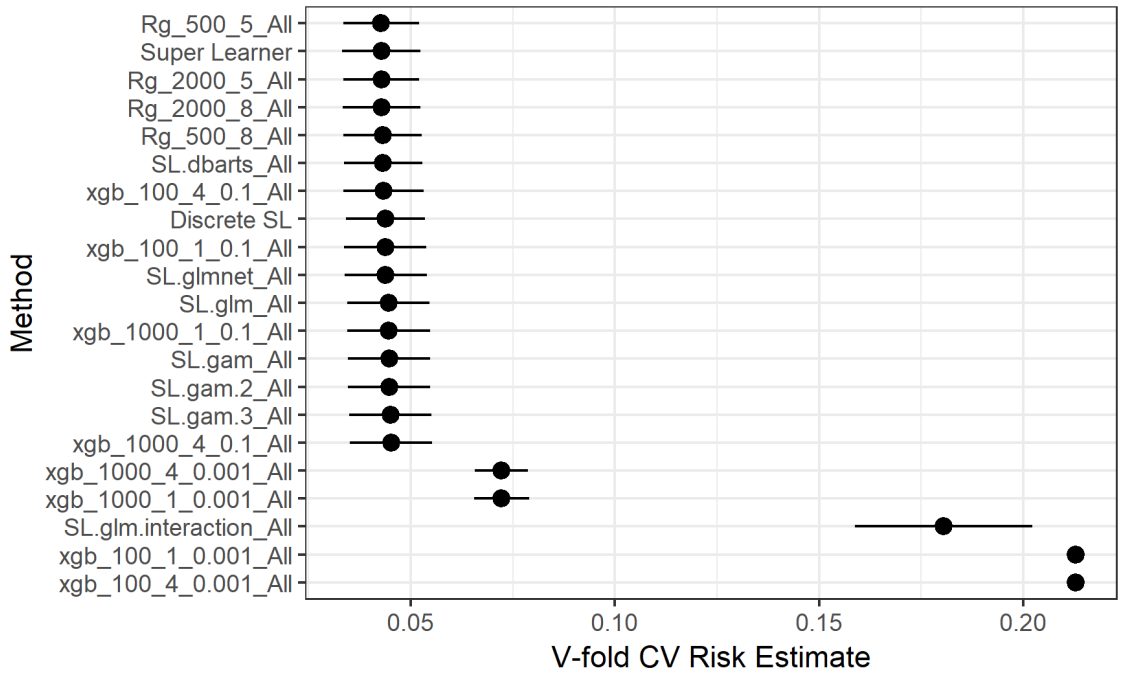
1. Unadjusted ATE (naive)
2. G-computation (outcome regression)
3. IPW with weights obtained via logistic PS (main terms)
4. SL-plug in G-computation
5. IPW SL-PS

CV SL for the PS model $g(W)$



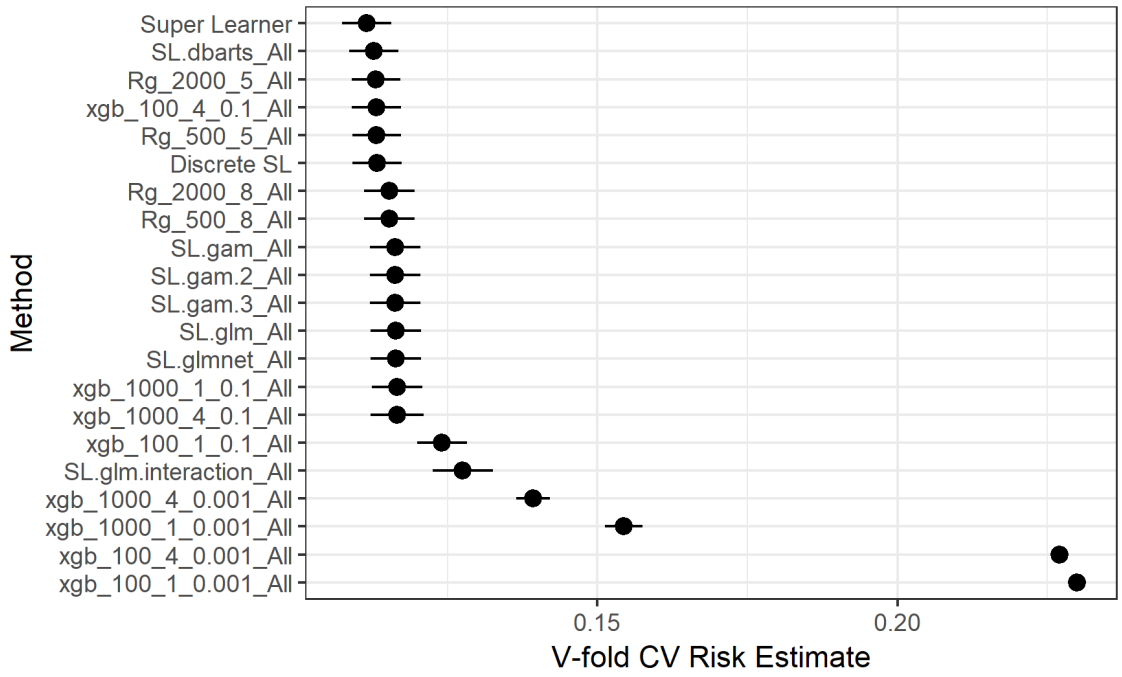
Risk is based on: Mean Squared Error

CV SL for the Outcome model in the Exposed



Risk is based on: Mean Squared Error

CV SL for the Outcome model in the Unexposed



Risk is based on: Mean Squared Error

Results

	ATE	95 % CI L	95 % CI U
Unadjusted (naive)	0.13	0.11	0.16
G-computation*	0.05	0.02	0.07
IPW logistic*	0.06	0.03	0.14
SL G-computation	0.08	bootstrap	not valid
IPW SL	0.11	bootstrap	not valid

* bootstrapped CIs

Conclusions so far

- Unadjusted ATE is biased as we believe there is confounding
- both parametric g-computation and IPW give similar results, IPW less efficient
- SL plug-in in g-computation: don't know how to get CIs
- Using the SL for the PS:
  1. introduced bias!: close to the unadjusted
  2. SL predicts the treatment assignment very well, but this is not what is needed for a valid PS analysis
  3. don't know how to get CIs
- We will see later why "naive" ML plug-ins are not valid
- and find estimators/conditions where machine learning can attenuate reliance on parametric models

End of first part

Questions?

Covariate balance across PS estimators in the example