Section II: developing a marker-based treatment rule

- Non-individualized setting
- Treatment decision rules
- Optimal treatment decision rules
- Estimating optimal treatment decision rule
  - Q-learning (Regression modeling)
  - Direct optimization
  - Super learning

Non-individualized setting

Non-individualized setting

- To simplify presentation, we will primarily focus on the setting that there are two treatments available, namely $A = 0$ and $A = 1$.

- Before beginning our discussion of *individualized* treatments, we will start with the simple case where the same treatment will be given to all individuals in the population of interest.

- **Question:** Which treatment leads to a better outcome D in a given population?
  - We will assume that smaller values of D are preferred.
  - To fix ideas, we will focus on the case that the goal is to optimize the *mean* outcome within the population.

- Examples of outcomes D: Survival time, CD4 count, indicator of no myocardial infarction within 30 days, ...

Non-individualized setting: intervening to set A to 1

- Suppose that you were given access to the whole population of interest and were allowed to intervene on treatment A.
- Say you wanted to learn what the outcome would have been if everyone were to receive treatment 1.
- A natural intervention to perform in this case would be to implement treatment $A = 1$ on the whole population.
- For each individual, you see an outcome $D(1)$, corresponding to the outcome that they have under treatment 1.
- You could then evaluate the population mean outcome when everyone receives treatment 1, namely $E[D(1)]$.

Non-individualized setting: intervening to set A to 0

- Suppose now that, instead of deciding to intervene to set treatment to 1, you had decided to intervene to set treatment to 0 for the whole population.
- In this case, for each individual, you would instead observe their outcome $D(0)$ under treatment 0.
- You could then evaluate the population mean outcome when everyone receives treatment 0, namely $E[D(0)]$.

Non-individualized setting: finding the optimal treatment

- Recall that we wish to administer the treatment in the population with a better (lower) mean outcome.
- If we knew both $E[D(1)]$ and $E[D(0)]$, then
  - treatment 1 would be preferable if $E[D(1)] < E[D(0)]$;
  - treatment 0 would be preferable if $E[D(1)] > E[D(0)]$;
  - the treatments perform equally well[1] otherwise.
- Problem: there's no way to run an experiment in which everyone in the population receives treatment 1, and to also run an experiment in which everyone receives treatment 0!

[1] In terms of mean outcome – there may be other considerations, such as cost, that make one treatment preferable to the other.

Non-individualized setting: a realizable experiment

- There is already a well known solution to this problem from the clinical trials literature: run a randomized experiment!
- For example, could randomly assign 60% of the population to treatment 1 and the remaining 40% to treatment 0.
- For each person, we observe an outcome D.
  - For a person who receives treatment 1, $D = D(1)$.*
  - For a person who receives treatment 0, $D = D(0)$.*
- Note: the outcomes $D(1)$ and $D(0)$ are referred to as *potential outcomes* because they are the outcomes that you would have seen if, possibly contrary to fact, you had implemented treatment 1 or treatment 0 in the population.

* Formally, this is an assumption. This assumption is known as the *stable unit treatment value assumption* in causal inference.

Simple example

Id	$D(0)$	$D(1)$
1	1	0
2	0	0
3	0	0
4	1	1
5	0	1
6	1	0

Simple example

Id	$D(0)$	$D(1)$	A	D
1	1	**0**	1	0
2	**0**	0	0	0
3	**0**	0	0	0
4	1	**1**	1	1
5	**0**	1	0	0
6	1	**0**	1	0

Non-individualized setting: a realizable experiment

- At the end of a trial, we compare the mean outcome among those who received $A = 1$ to the mean outcome among those who received $A = 0$, that is, we compare $E[D|A = 1]$ to $E[D|A = 0]$.

- It's tempting to claim that $E[D|A = 1] = E[D(1)]$, that is, that the mean outcome among those who received treatment 1 is the same as the mean outcome that would have been observed if everyone had received treatment 1 (and similarly for treatment 0).

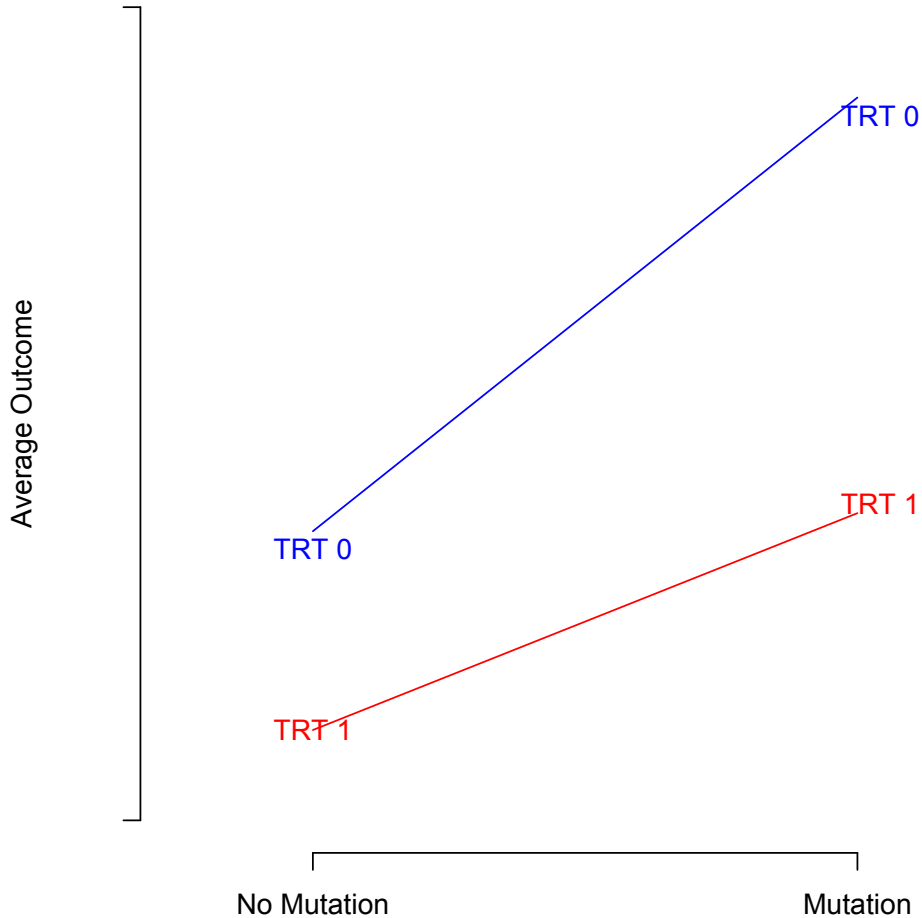
- How can we justify that $E[D|A = a] = E[D(a)]$ for each treatment a? This can be shown by combining the following two observations:
  1. If an individual received treatment a, then $D = D(a)$. So, $E[D|A = a] = E[D(a)|A = a]$.
  2. Treatment assignment is random and, in particular, independent of $D(a)$. Therefore, $E[D(a)|A = a] = E[D(a)]$.

Treatment decision rules

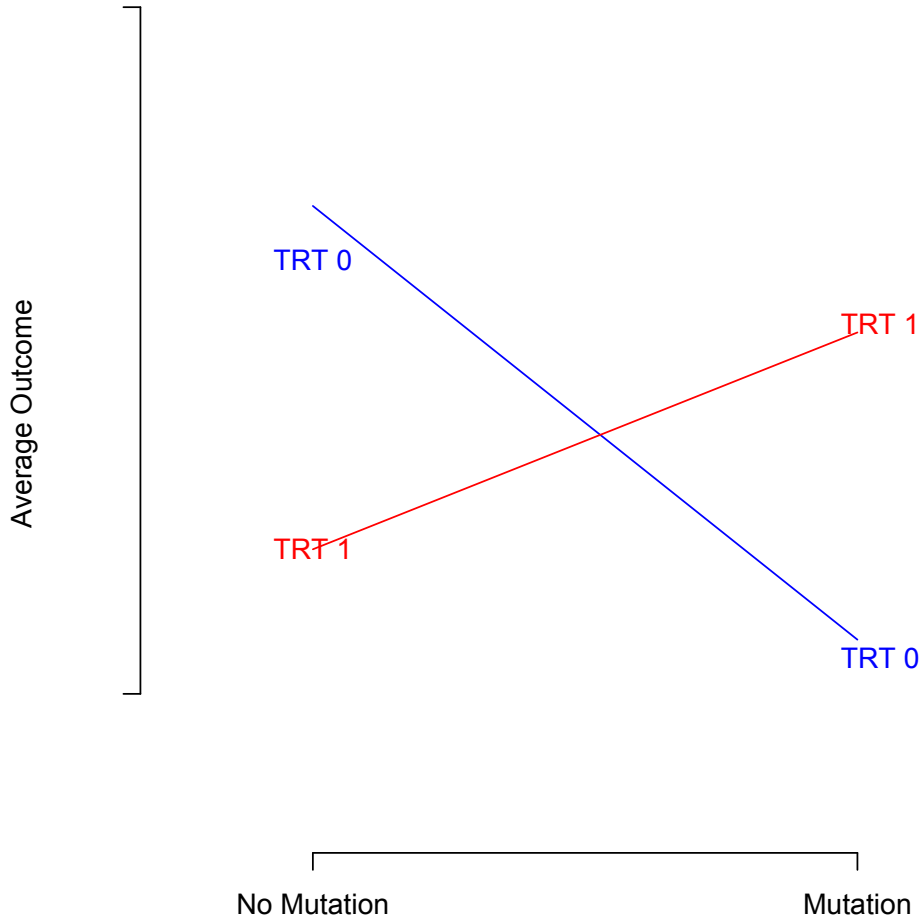
Treatment decision rules

- Suppose that we observe covariates X on all individuals taking values in $\mathcal{X}$.
- The goal will be to make a treatment decision based on these observed covariates.
- Formally, a **treatment decision rule** is a function $d : \mathcal{X} \to \{0, 1\}$.
- Reasonable to expect that the best rules will depend on characteristics (variables , covariates), i.e., X, that exhibit a qualitative interaction with treatment.

Tailoring Variables

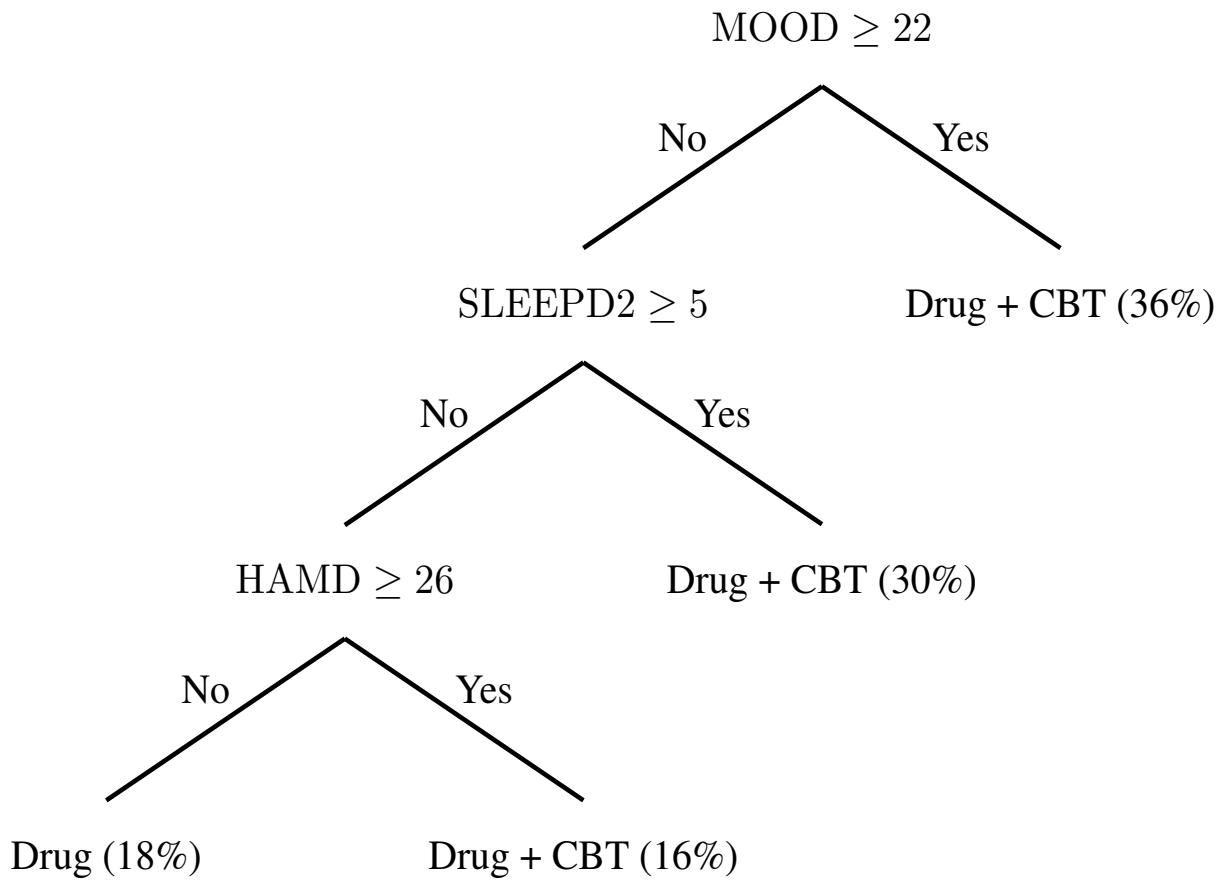


Tailoring Variables



Simple example

A decision rule example:



Simple example

- Even simpler example: If MOOD $\geq 22 \Rightarrow$ Drug + CBT; otherwise $\Rightarrow$ Drug
- *Mathematically:* The formal rule is

$$d(\text{MOOD}) = I(\text{MOOD} > 22)$$

Evaluating the performance of a rule in an ideal setting

- Suppose that, as on slide 4, you are given access to the entire population and wish to evaluate the performance of the treatment rule d.
- In this case, you can assign each individual treatment $d(X)$, where X is their covariate value.
  - If $d(X) = 1$, then you observe the potential outcome $D(1)$.
  - If $d(X) = 0$, then you observe the potential outcome $D(0)$.
  - To simplify notation, we let $D(d)$ denote the potential outcome $D(d(X))$ that you observe.
- Can average observed outcomes resulting from this hypothetical experiment to evaluate the mean outcome $E[D(d)]$.

Evaluating the performance of a rule in a trial setting

- Earlier, we showed that, in the absence of covariates, data from a trial can be used to estimate $E[D|A = a]$, which is in turn equal to the causal quantity $E[D(a)]$ of interest.
  - The key to showing this was that the treatment A is randomized, and is therefore independent of $D(a)$.
- We will see that $E[D(d)]$ can be similarly learned from a randomized trial.
  - The key additional observation is that we can write

$$E[D(d)] = \sum_x E[D(d)|X = x]p(x)$$

- The marginal distribution of covariates (in the trial population) is easily learned, so the question is whether we can learn $E[D(d)|X = x]$.

Evaluating the performance of a rule in a trial setting

- Learning $E[D(d)|X = x]$ is a matter of applying the same arguments from slide 10, but within the stratum of individuals for whom $X = x$.
- In particular, we can show that $E[D|A = d(x), X = x] = E[D(d)|X = x]$ as follows:
  1. If an individual received treatment $d(x)$, then $D = D(d)$. So, $E[D|A = d(x), X = x] = E[D(d)|A = d(x), X = x]$.
  2. Treatment assignment is random and, in particular, independent of $D(d)$ conditionally on $X = x$. Therefore, $E[D(d)|A = d(x), X = x] = E[D(d)|X = x]$.

Important condition for evaluating a rule d

- We have seen that, to evaluate a rule d, we can evaluate $E[D|A = d(x), X = x]$ and then average over realizations of the covariates x.
- This conditional expectation is only well-defined if $P(A = d(x)|X = x) > 0$.
- To ensure that all rules d can be evaluated, it suffices to have **positivity**:

$$P(A = 1|X = x) > 0, \text{ and}$$
$$P(A = 0|X = x) > 0.$$

Optimal treatment decision rules

Considerations

- Identify the subset that are good tailoring variables
- Rule $d(X)$: a function of X
- There are many possible rules d:

  $\mathcal{D}$: class of all possible treatment decision rules

- Can we find the optimal treatment decision rule in $\mathcal{D}$?
- Optimal treatment decision rule : If followed by *all patients* in the population, would lead to *smallest expected outcome* among all rules in $\mathcal{D}$

Optimal decision rule

- The optimal treatment decision rule $d^* \in \mathcal{D}$ minimizes the expected outcome

$$d^* = \underset{d \in \mathcal{D}}{\operatorname{argmin}}\, E\{D(d)\}$$

- That is, $E\{D(d^*)\} \leq E\{D(d)\}$ for all $d \in \mathcal{D}$
- Also, $E\{D(d^*)|X = x\} \leq E\{D(d)|X = x\}$ for all $d \in \mathcal{D}$ and for all patient subgroups defined by x.
- $d^*(X) = I[E\{D(1)|X\} < E\{D(0)|X\}]$.
- From our earlier arguments, we can also write this as:

$$d^*(X) = I[E\{D|A = 1, X\} < E\{D|A = 0, X\}].$$

Example

- $X \sim$Uniform$[-1, 1]$, A is binary {-1, 1} with probability $1/2$, $D \sim N(1 - (X - 1/3)^2 A, 1)$
- Consider the rule $d(x) = I(x - 1/3 \geq 0)$. What is the expected outcome of this rule?

$$\begin{aligned} E\{D(d)\} &= E[E(D|A = 1, X)I(X - 1/3 \geq 0) \\ &\quad + E(D|A = 0, X)I(X - 1/3 < 0)] \\ &= \int_{1/3}^{1} \left\{ \frac{1 - (x - 1/3)^2}{2} \right\} dx + \int_{-1}^{1/3} \frac{1}{2} dx = 73/81 \end{aligned}$$

- What is the optimal treatment rule?

Example

- $d^*(x) = 1$
- What is the expected outcome of the optimal rule?

$$\begin{aligned} E\{D(d)\} &= E[E(D|A=1,X)I\{d(X)=1\} \\ &= \int_{-1}^{1} \frac{1-(x-1/3)^2}{2} dx = 45/81 \end{aligned}$$

Optimal Rule

- Optimal Rule:

$$E(D|X, A = 1) \leq E(D|X, A = 0) \Rightarrow d^*(X) = 1$$
$$E(D|X, A = 1) > E(D|X, A = 0) \Rightarrow d^*(X) = 0$$

- d^* provides a treatment recommendation to every individual given their X
- If $E(D|X, A)$ were known, we could find d^*.
- Problem: $E(D|X, A)$ is unknown.

Estimating optimal treatment decision rule

- **Q-learning (Regression modeling)**
- Direct optimization
- Super Learning

Q-learning (Regression modeling)

- If we had an iid sample of data $(X_i, A_i, D_i), i = 1, \ldots, n$, we can posit a regression model

$$E(D|A, X) = \mu(A, X; \beta)$$

and estimate $\hat{\beta}$ using e.g. least squares/logistic regression.

- The estimate of the optimal treatment decision rule is:

$$\hat{d}_n(x) = \begin{cases} 1, & \text{if } \mu(1, x; \hat{\beta}_n) \leq \mu(0, x; \hat{\beta}_n) \\ 0, & \text{otherwise.} \end{cases}$$

Alternatives

- Use flexible models for the outcome.
- Other methods, e.g., modeling contrast
  - A more robust method for estimating the optimal treatment decision rule
  - One does not need to know the entire function $E(D|A, X)$.
  - It suffices to only consider the contrast function

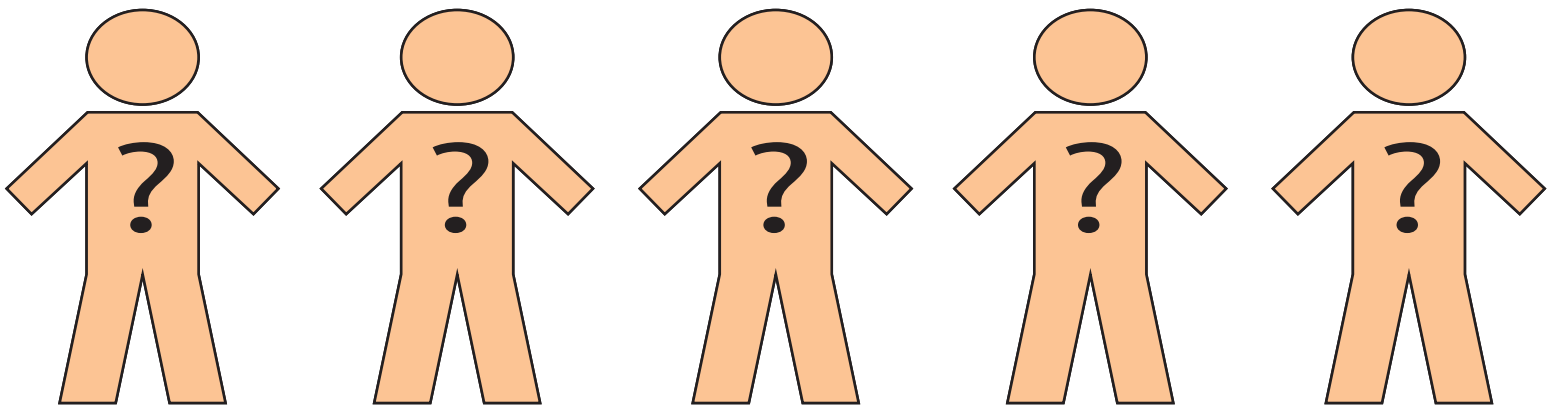
$$\Delta(X) = E(D|A = 0, X) - E(D|A = 1, X)$$

  - $d^*(x) = I\{\Delta(x) \geq 0\}$.

Murphy (*JRSSB*, 2003); Tian et al (*JASA*, 2014)

Contrast function specifies optimal resource-constrained allocation

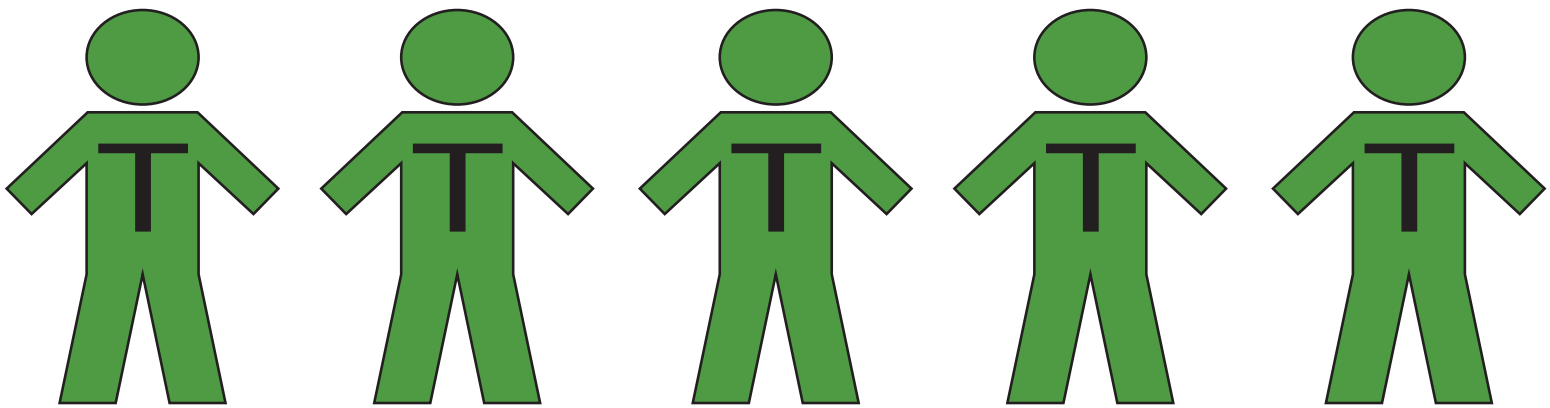
- Suppose treatment is beneficial to everyone...



Luedtke & van der Laan (Int J Biostat, 2016); vanderWeele et al. (arXiv 1802.09642, 2018)

Contrast function specifies optimal resource-constrained allocation

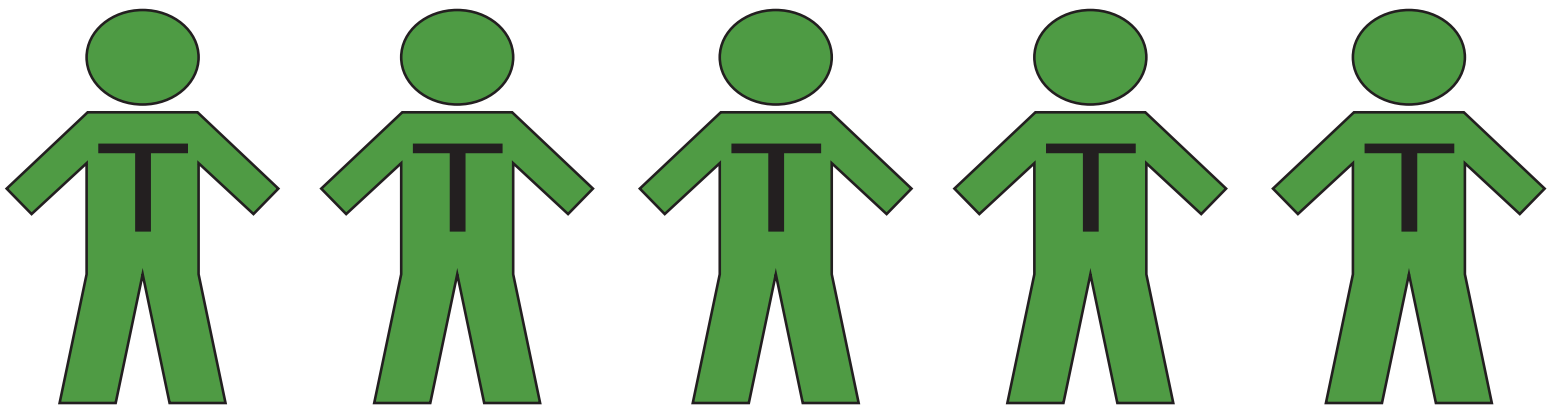
- Suppose treatment is beneficial to everyone...



Luedtke & van der Laan (Int J Biostat, 2016); vanderWeele et al. (arXiv 1802.09642, 2018)

Contrast function specifies optimal resource-constrained allocation

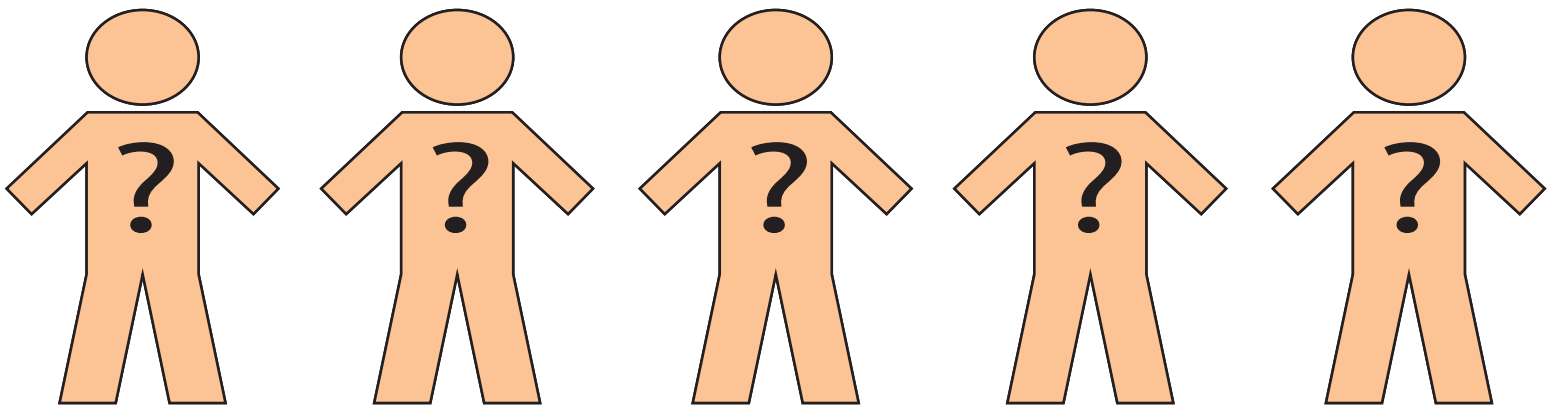
- Suppose treatment is beneficial to everyone...
- But resources are limited so can only treat 40% of population



Luedtke & van der Laan (Int J Biostat, 2016); vanderWeele et al. (arXiv 1802.09642, 2018)

Contrast function specifies optimal resource-constrained allocation

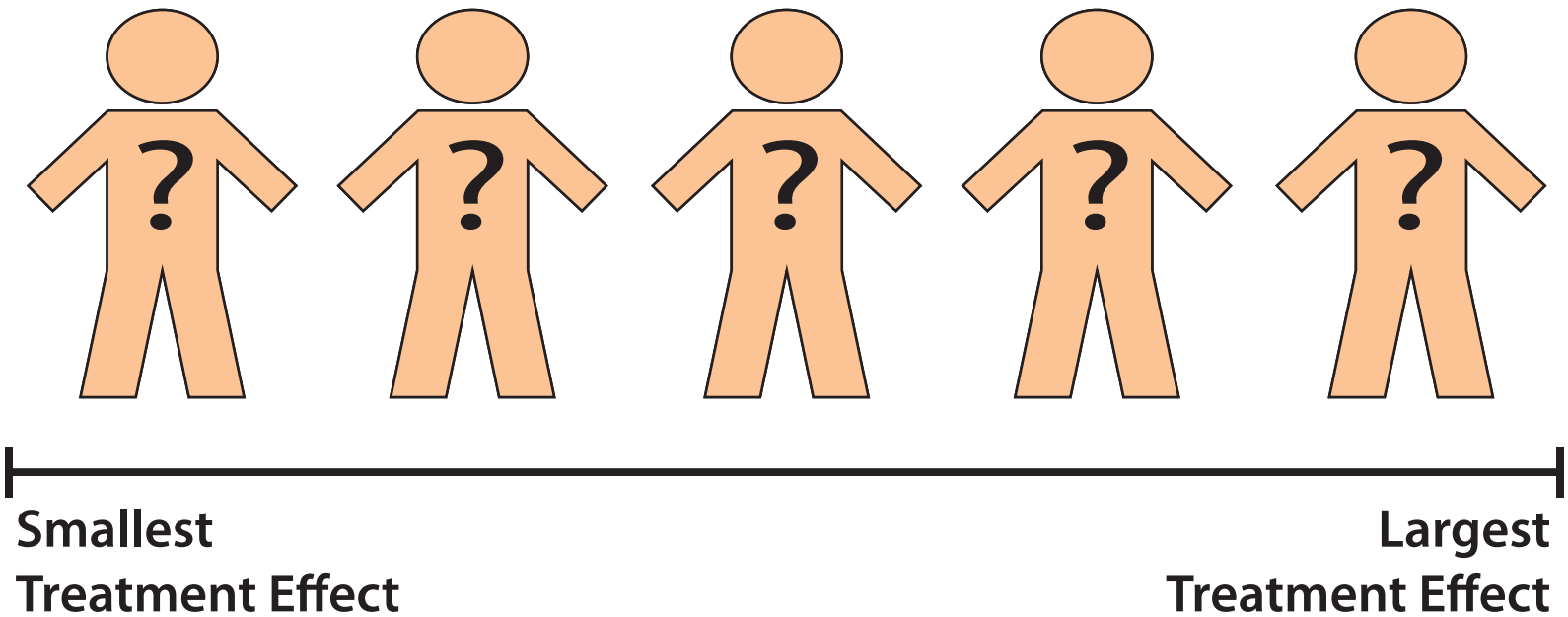
- Suppose treatment is beneficial to everyone...
- But resources are limited so can only treat 40% of population



Luedtke & van der Laan (Int J Biostat, 2016); vanderWeele et al. (arXiv 1802.09642, 2018)

Contrast function specifies optimal resource-constrained allocation

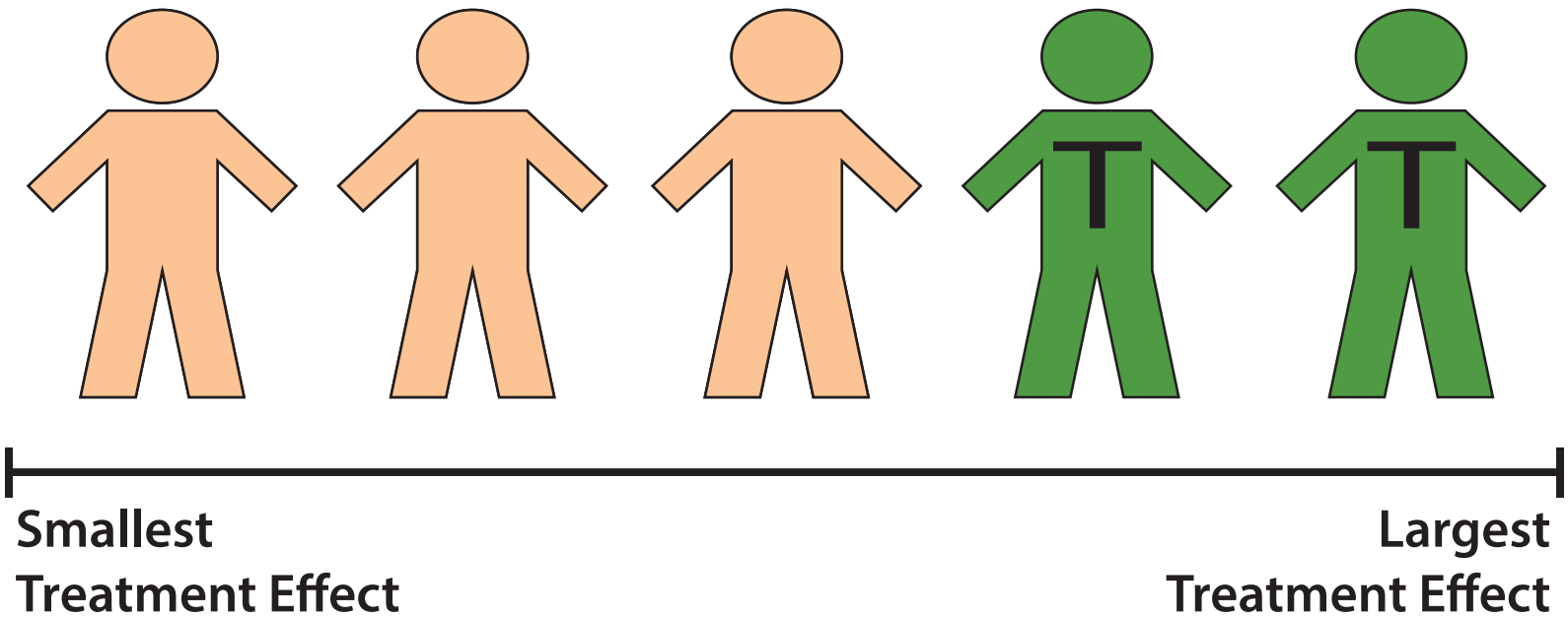
- Suppose treatment is beneficial to everyone...
- But resources are limited so can only treat 40% of population



Luedtke & van der Laan (Int J Biostat, 2016); vanderWeele et al. (arXiv 1802.09642, 2018)

Contrast function specifies optimal resource-constrained allocation

- Suppose treatment is beneficial to everyone...
- But resources are limited so can only treat 40% of population

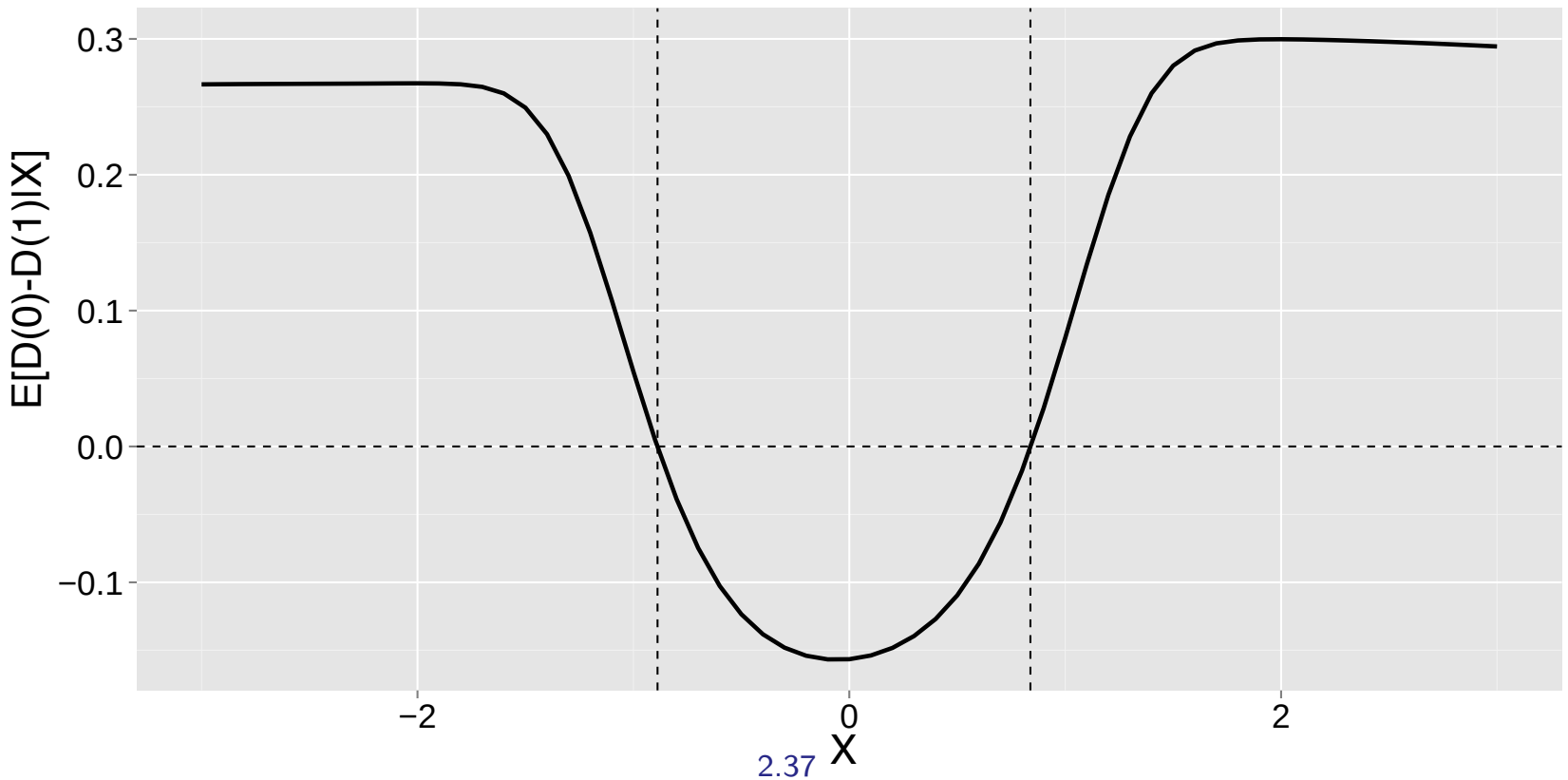


Luedtke & van der Laan (Int J Biostat, 2016); vanderWeele et al. (arXiv 1802.09642, 2018)

If our regression model is misspecified, is our rule reasonable?

Suppose we use the model $\mu(a, x; \hat{\beta}) = \beta_0 + \beta_1 X + \beta_2 A + \beta_3 XA$, so that our rule takes the form

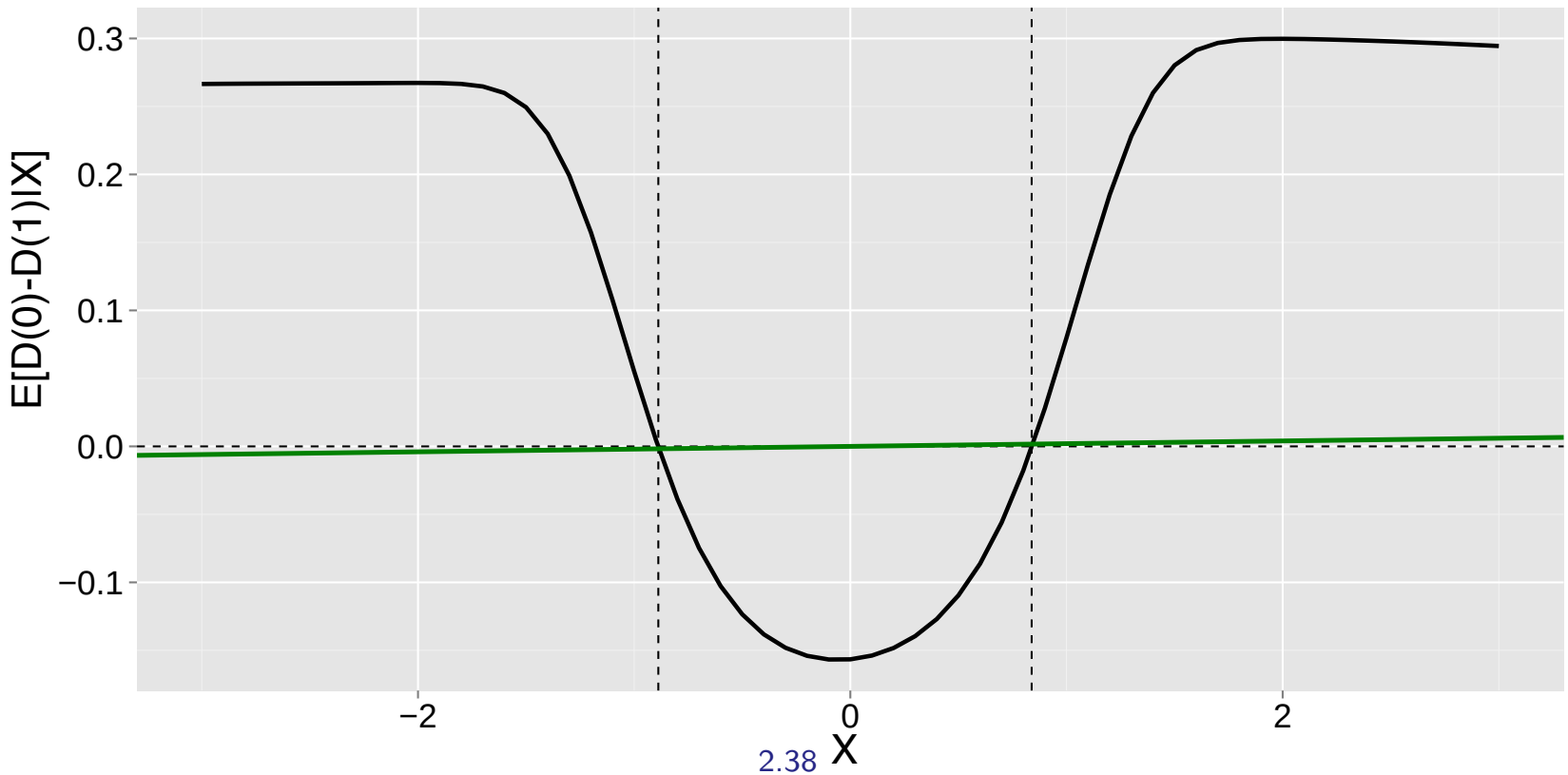
$$\hat{d}_n(x) = \begin{cases} 1, & \text{if } \mu(0, x; \hat{\beta}) - \mu(1, x; \hat{\beta}) = -\hat{\beta}_2 - \hat{\beta}_3 X \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$



If our regression model is misspecified, is our rule reasonable?

Suppose we use the model $\mu(a, x; \hat{\beta}) = \beta_0 + \beta_1 X + \beta_2 A + \beta_3 XA$, so that our rule takes the form

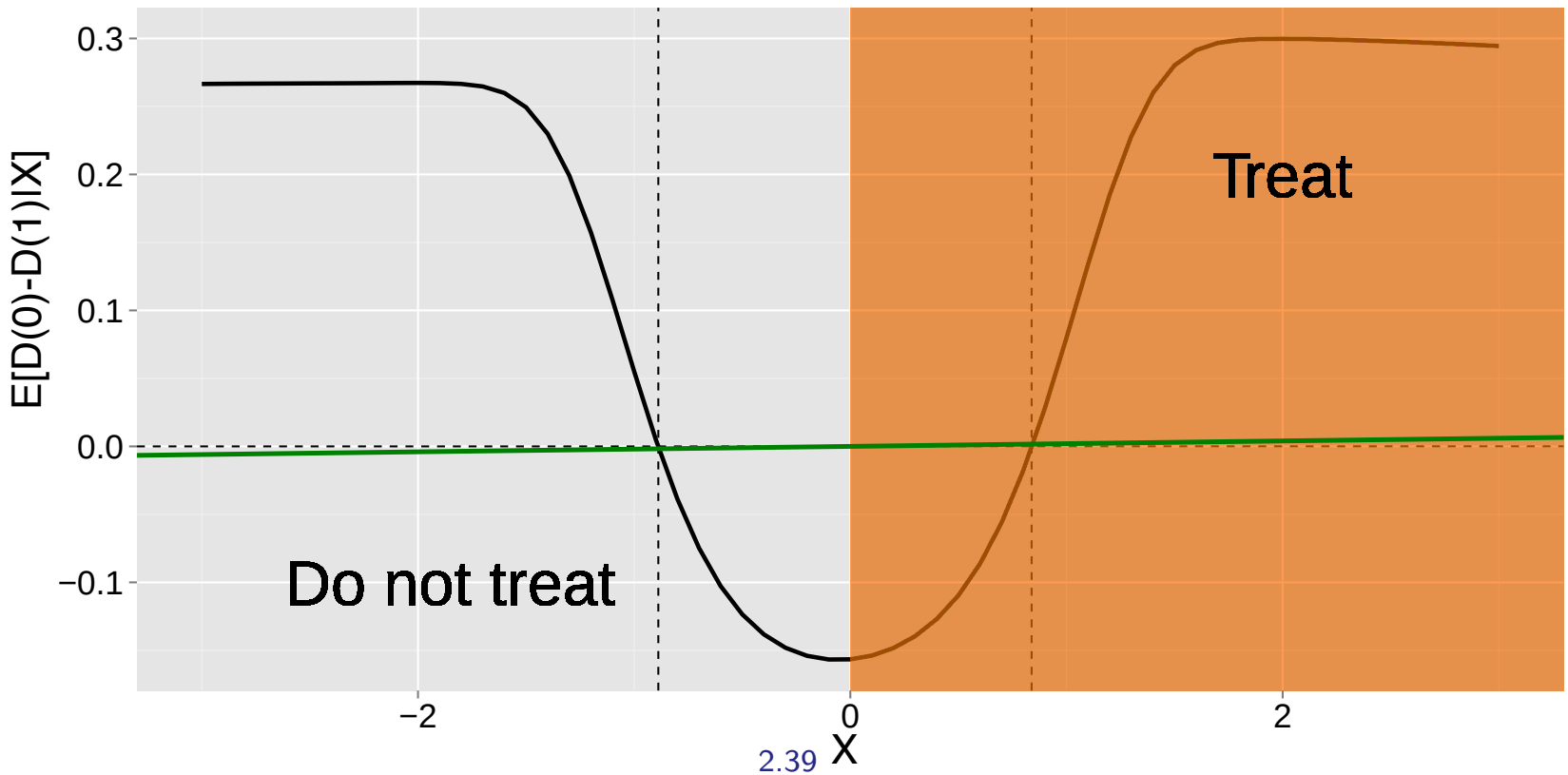
$$\hat{d}_n(x) = \begin{cases} 1, & \text{if } \mu(0, x; \hat{\beta}) - \mu(1, x; \hat{\beta}) = -\hat{\beta}_2 - \hat{\beta}_3 X \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$



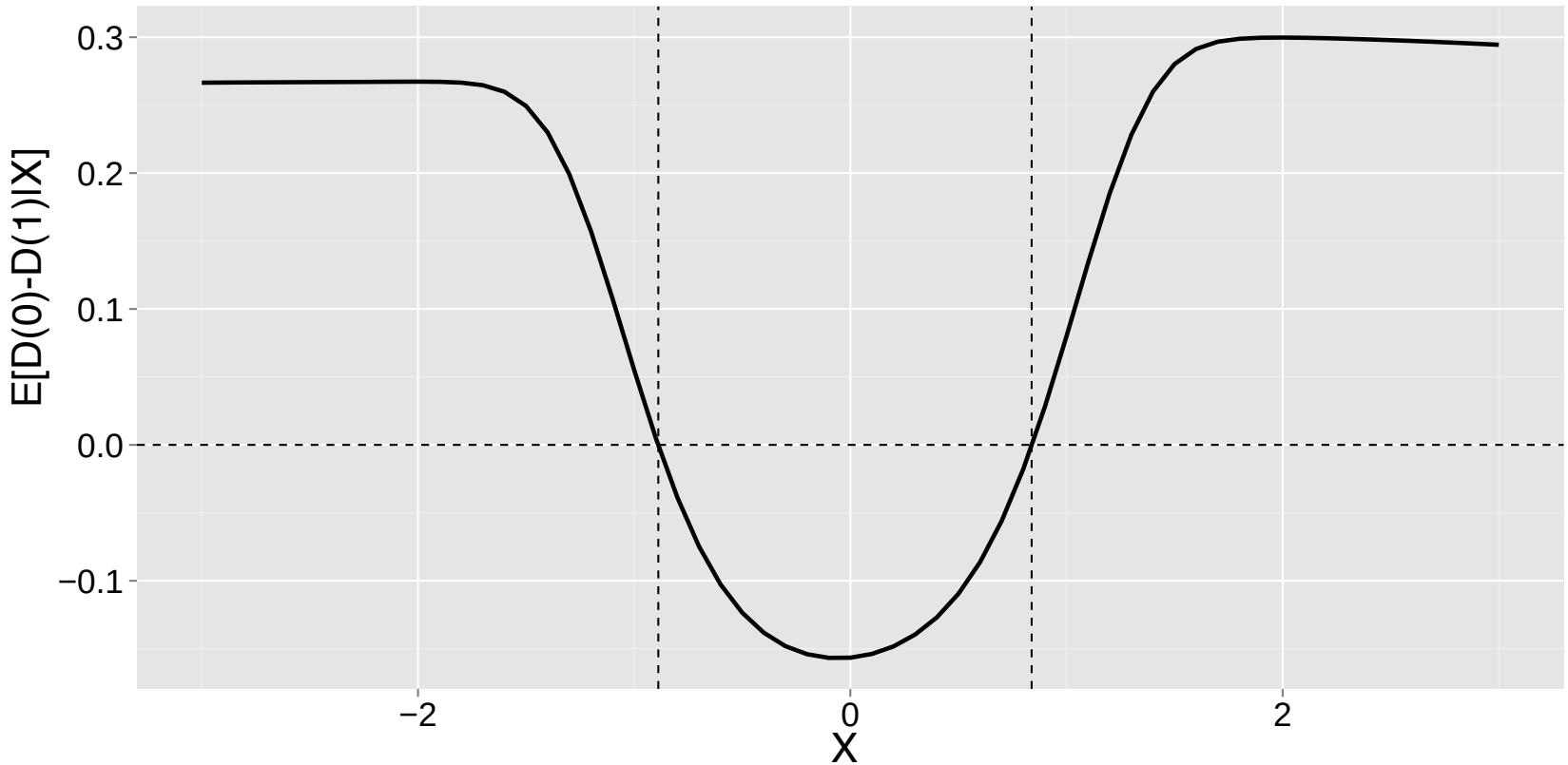
If our regression model is misspecified, is our rule reasonable?

Suppose we use the model $\mu(a, x; \hat{\beta}) = \beta_0 + \beta_1 X + \beta_2 A + \beta_3 XA$, so that our rule takes the form

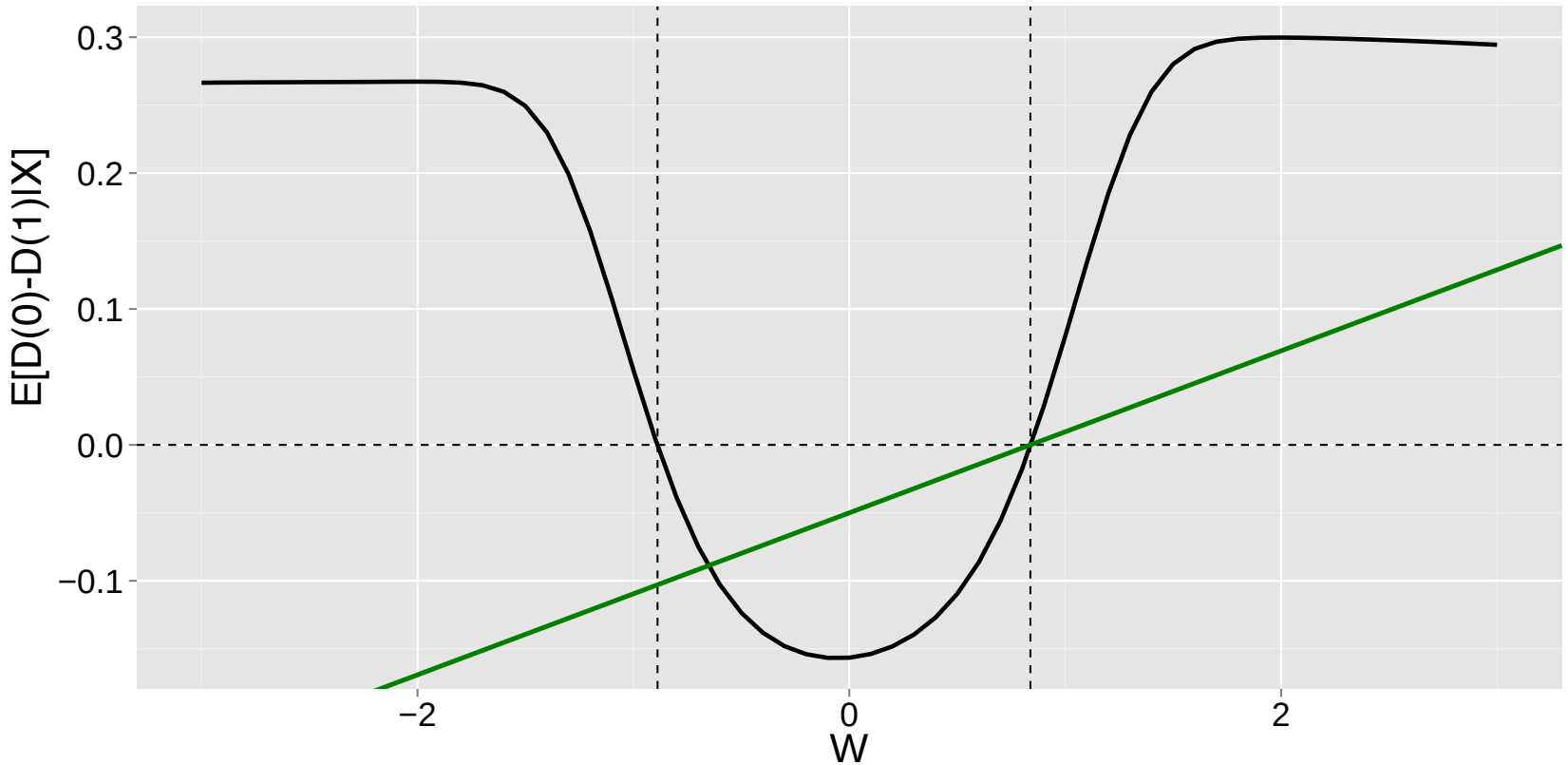
$$\hat{d}_n(x) = \begin{cases} 1, & \text{if } \mu(0, x; \hat{\beta}) - \mu(1, x; \hat{\beta}) = -\hat{\beta}_2 - \hat{\beta}_3 X \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$



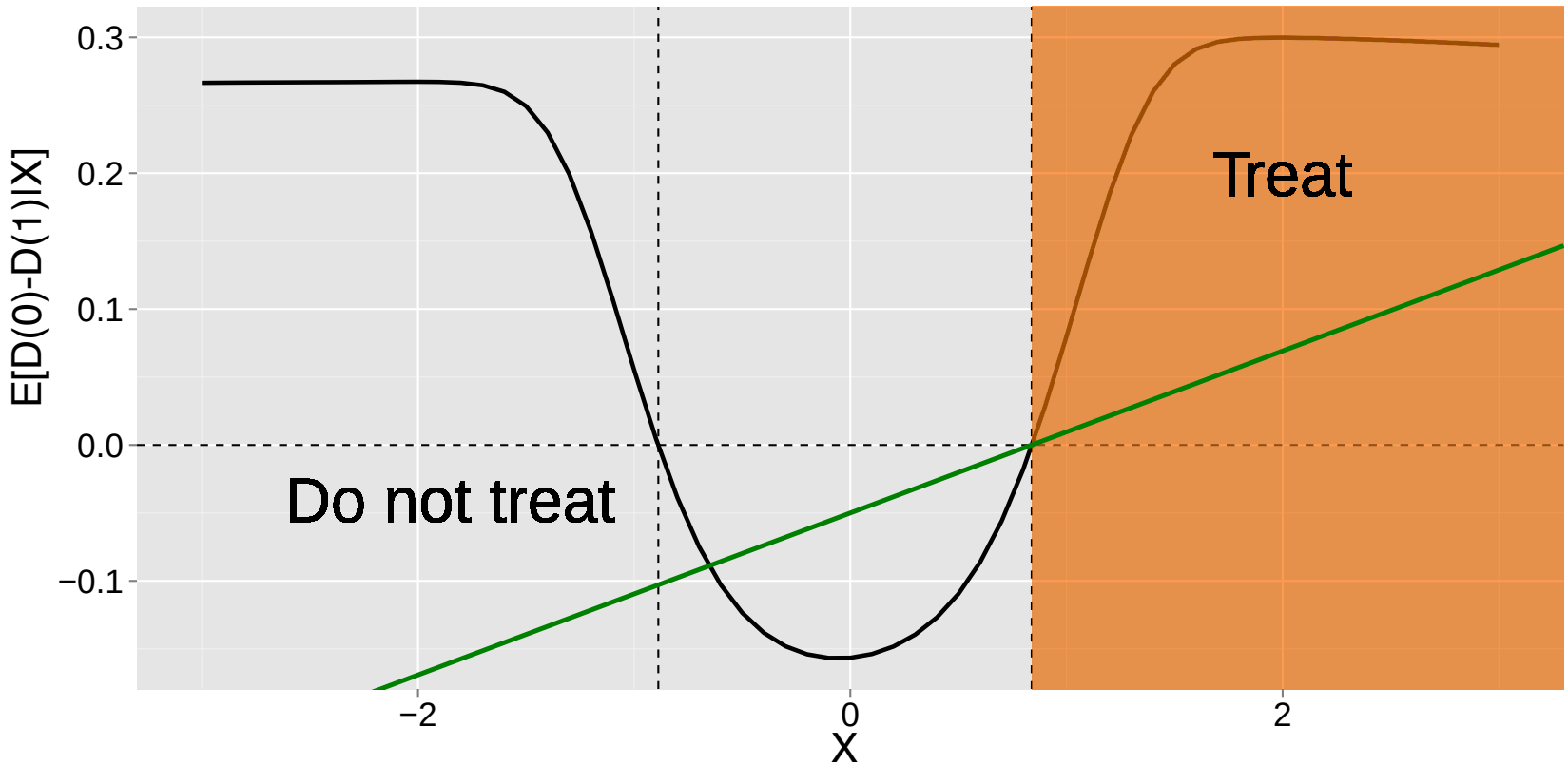
Is there a better linear rule?



Is there a better linear rule?



Is there a better linear rule?



Yes - but how do we learn this rule from the data?

Estimating optimal treatment decision rule

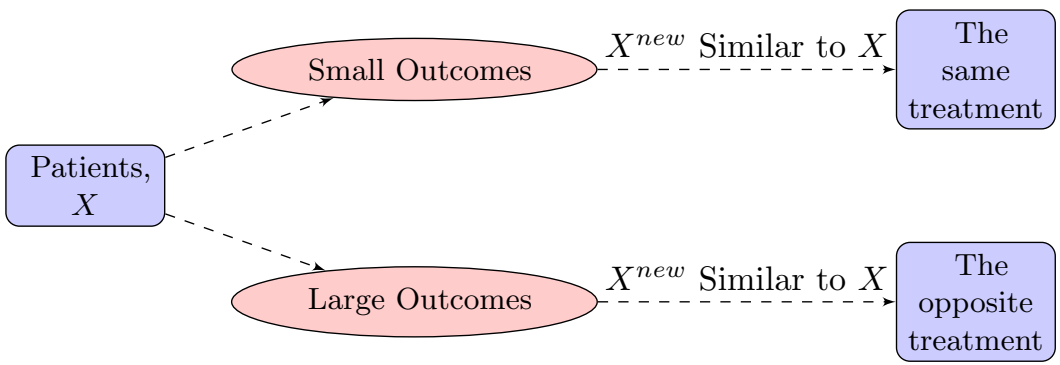
- Q-learning (Regression modeling)
- **Direct optimization**
- Super Learning

Direct Optimization: Classification Perspective

Intuition: Classification

Given a new observation X^{new}, predict the class label $d^{*,\text{new}}$.

- No direct information on the true class labels, d^*.
- Can we assign the right treatment based on the observed information?



Directly Estimating the Optimal Rule

Thought: Minimize a "*good*" estimator for $E[D(d)]$

- $\pi(X) = P(A = 1|X)$ is the propensity score for treatment
- $\pi(X)$ known in a randomized study; Can also be estimated using the data $(A_i, X_i), i = 1, \ldots, n$, e.g., logistic regression $\pi(X; \gamma)$ and estimate γ by $\hat{\gamma}$.
- The propensity of receiving treatment consistent with $d(X)$

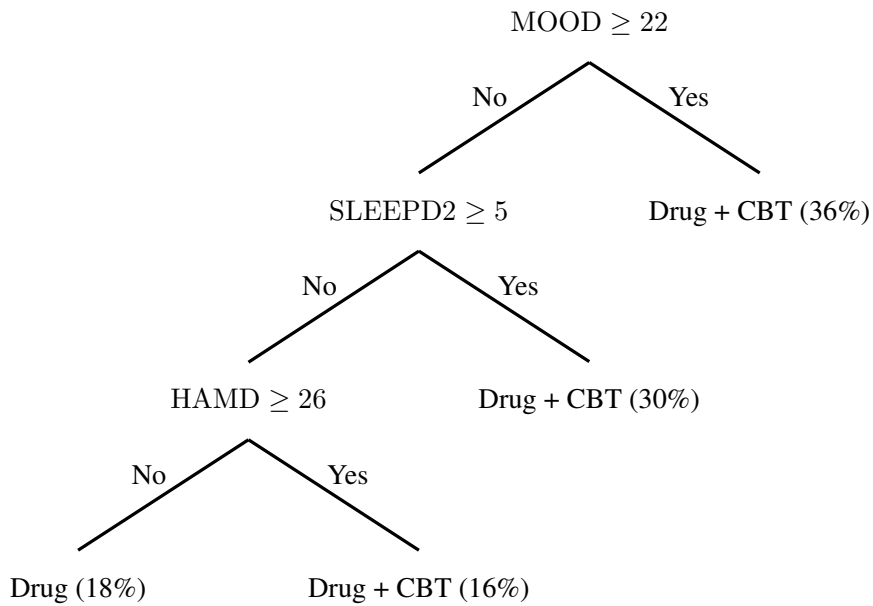
$$P\{d(X)|X\} = \begin{cases} \pi(X), & \text{if } d(X) = 1 \\ 1 - \pi(X), & \text{if } d(X) = 0. \end{cases}$$

Direct Optimization: Optimal Restricted Rule

- Optimize the objective within a restricted class of rules, e.g.
  - Linear rules

$$d_\eta(x) = \begin{cases} 1, & \text{if } \eta_0 + \eta_1 X_1 + \eta_2 X_2 > 0 \\ 0, & \text{otherwise.} \end{cases}$$

  - Binary decision trees of depth at most 3, each decision parameterized by a linear rule



Zhang et al. (*Biometrics* 2012)

Inverse Probability Weighted Estimator for Mean Outcome of Rule

Identify estimators for $E[D(d)]$:

- Using that

$$E[E\{D(d)|A = d(X), X = x\}] = E\left[\frac{I\{A = d(X)\}}{P\{d(X)|X\}}D\right],$$

we arrive at the inverse probability weighted estimator

$$IPWE(d) = n^{-1}\sum_{i=1}^{n}\frac{I\{A_i = d(X_i)\}D_i}{P\{d(X_i)|X, \hat{\gamma}\}}. \qquad (1)$$

- Consistent for $E[D(d)]$ if $\pi(X;\gamma)$, and hence $P\{d(X_i)|X, \hat{\gamma}\}$, is correctly specified

Outcome Weighted Learning (OWL)

- Minimize $IPWE(d)$ (1)
- For any rule d, $2d(X) - 1 = \text{sign}\{f(X)\}$ for some function f.
- Hence, minimize:

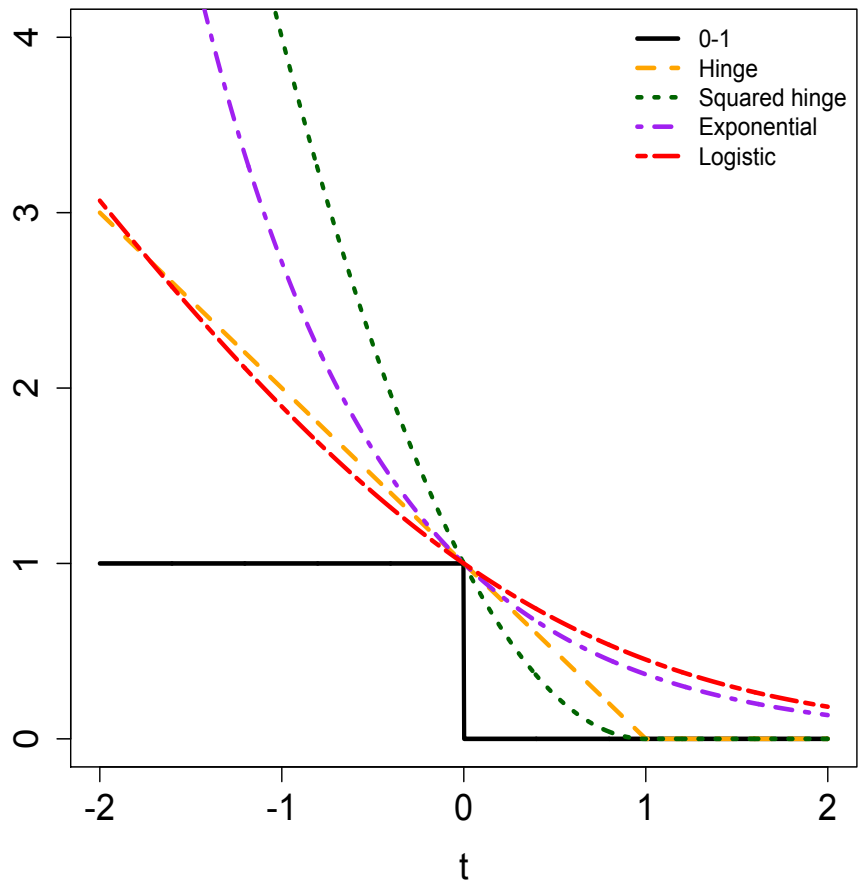
$$n^{-1} \sum_{i=1}^{n} \frac{-D_i}{P\{d(X_i)|X, \hat{\gamma}\}} I\{(2A_i - 1) \neq \text{sign}(f(X_i)\}.$$

- Can be treated as recoding $\mathcal{A} = \{-1, 1\}$

Zhao et al. (*JASA* 2012)

Computational challenges: non-convexity and discontinuity of 0-1 loss

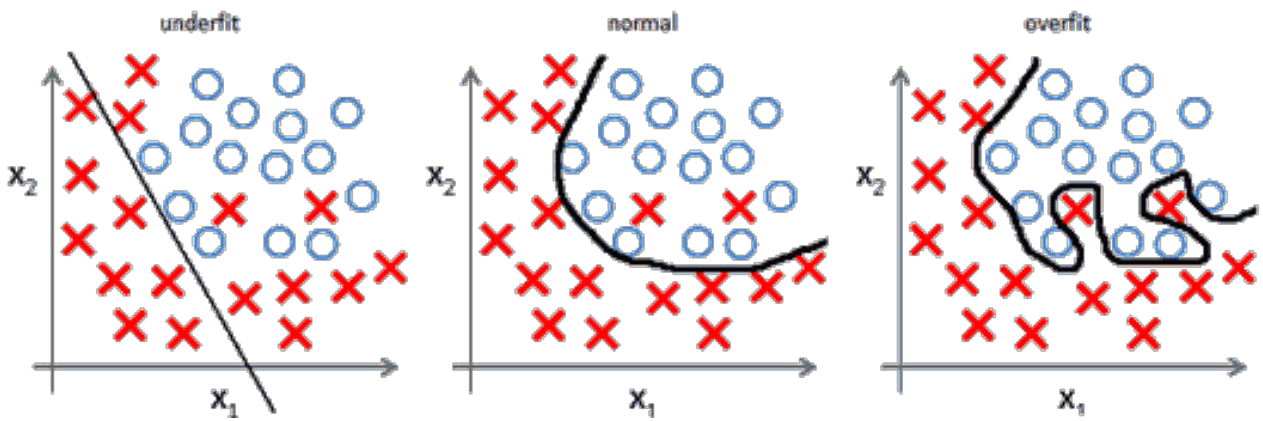
- Solution: replace the indicator that $(2A_i - 1) \neq \text{sign}(f(X_i)\}$ by a smoother function ϕ



Directly estimating the optimal rule is prone to overfitting

Consider the simple case that D is a binary event indicator and treatment probability is always $1/2$ so that we minimize

$$-\frac{1}{n}\sum_{i=1}^{n} D_i I\{(2A_i - 1) \neq \text{sign}(f(X_i))\}$$



X is a patient who was untreated and event-free *or* treated and had the event

O is a patient who was treated and event-free *or* untreated and had the event

Image source: http://mlwiki.org/index.php/Overfitting

Avoid overfitting by adding penalties

$$\min_f \frac{1}{n}\sum_{i=1}^{n} \frac{-D_i}{P\{d(X_i)|X,\hat{\gamma}\}}\phi\{(2A_i-1)f(X_i))\} + \lambda_n\|f\|^2. \qquad (2)$$

- $\|f\|$ is some norm for f, and λ_n controls the severity of the penalty on the functions.
- A linear decision rule: $f(X) = X^T\beta + \beta_0$, with $\|f\|$ as the Euclidean norm of β.
- **Estimated treatment rule**:

$$\hat{d}_n(X) = \text{sign}(\hat{f}_n(X)),$$

where $\hat{f}_n$ is the solution to (2).

More Efficient form of Outcome Weighted Learning

- Residual weighted learning: use residuals (after subtracting main effects) instead of the original outcomes as the weights.
- Efficient augmentation and relaxation learning: use an improved estimator of $E[D(d)]$
  - Doubly robust augmented inverse probability weighted estimator: model both the propensity score and the outcome
  - Consistent if either the propensity score or the expected outcome conditional on treatment and covariates is consistently estimated
  - Outcome weighted learning is a special case.

Zhou et al. (*JASA* 2017)

Summary on Direct Optimization Approach

- Direct optimization: conceptual appeal / robustness
- How to implement, e.g. surrogate loss function, form of penalties for variable selection, depends on the context
- Disadvantage of direct optimization relative to Q-learning: more difficult to interpret the final output
  - Does not give magnitude of treatment effect
  - Need to add additional constraints if want to derive resource-constrained rule
  - Rule is a "black box": does not characterize contributions of variables to treatment effect or treatment rule

Estimating optimal treatment decision rule

- Q-learning (Regression modeling)
- Direct optimization
- **Super Learning**

What is Super-Learning?

- Suppose want to estimate the regression $E[D|A, X]$ "as well as possible"
  - e.g., minimize mean-squared error (MSE)
  - MSE performance can be related to the performance of the estimated optimal treatment rule that treats if and only if $E[D|A = 1, X] < E[D|A = 0, X]$

- How could we do this?
  1. Linear regression
  2. Maybe add some interactions
  3. Maybe add a Lasso penalty on the coefficients
  4. Or some other penalty
  5. If X lower dimensional, maybe run kernel regression or nearest neighbors
  6. What about Random Forests?
  7. Or neural networks?

Given all of these options, what should we do?

- One option is to just pick one *a priori*
- This strategy can never do better than the oracle selector, i.e. the best choice of any one algorithm
- Unlikely you will perform as well as the oracle selector

Objective 1

Perform as well as the oracle selector.

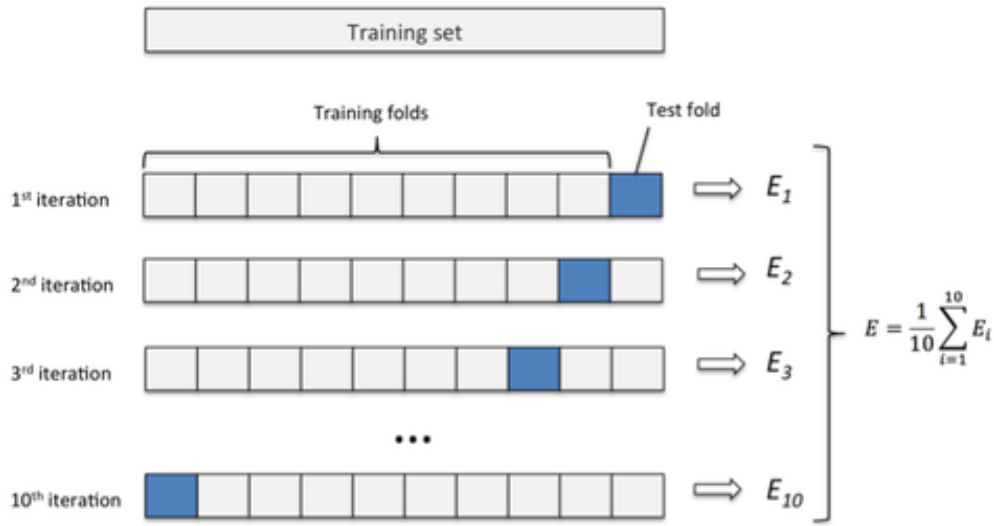
- You've probably seen an algorithm attaining Objective 1 before, though you may not have been aware of its optimality properties

Objective 2

Outperform the oracle selector.

Objective 1: Matching the Oracle

- Could use *V*-fold cross-validation to select the rule minimizing MSE:



- In what sense?

$$\Big(\text{CV-MSE of Selector}\Big) \leq 1.1 \times \Big(\text{CV-MSE of Oracle}\Big) + C\frac{\log(\#\ \text{Alg})}{n}$$

image source: https://sebastianraschka.com

Objective 1: An Illustration

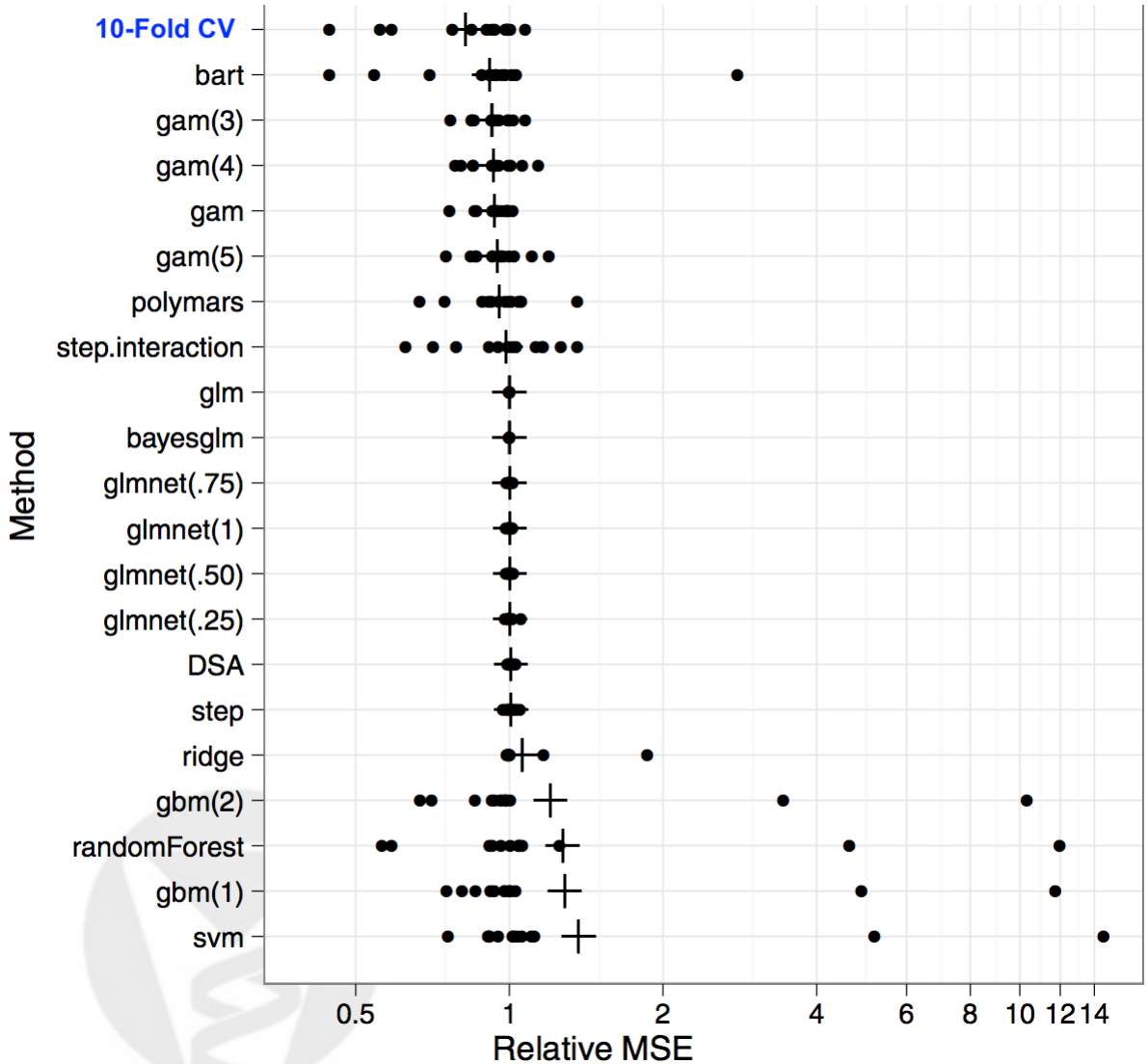


image source: Polley & van der Laan (2010)

Objective 2: A Better Oracle

- So far, we've argued that we can do as well as the best candidate in our library
- Can we do better?

In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.

– Wikipedia (2018)

Objective 2: A Better Oracle

- How can we hope to outperform the best candidate?
- Could consider all linear combinations of candidate algorithms:

$$\widehat{E}[D|A,X] = \sum_{i=1}^{\#\text{ Alg}} \alpha_i \widehat{E}_i[D|A,X],$$

  where α_i is a real number and $\widehat{E}_i[D|A,X]$ are candidate estimates
- Issue with this choice of combination is that it may be unstable (candidate estimates will be highly correlated)
  - To stabilize regression, restrict α to be a convex combination
- General combination approaches called stacking in the literature
- Weighted sums known as ensemble averaging
- Using convex combination known as super-learning

Objective 2: A Better Oracle

- In the remainder, we refer to the oracle selector as the selector that returns the best convex combination of estimators, rather than the best estimator
- Have the same oracle inequality as before:

$$\Big(\text{CV-MSE of Selector}\Big) \leq 1.1 \times \Big(\text{CV-MSE of Oracle}\Big) + C\frac{\log n}{n}$$

- Do at least as well as the best candidate algorithm
  - Only exception is if one can *a priori* correctly specify a parametric model, in which case perform slightly worse

Better Oracle: An Illustration

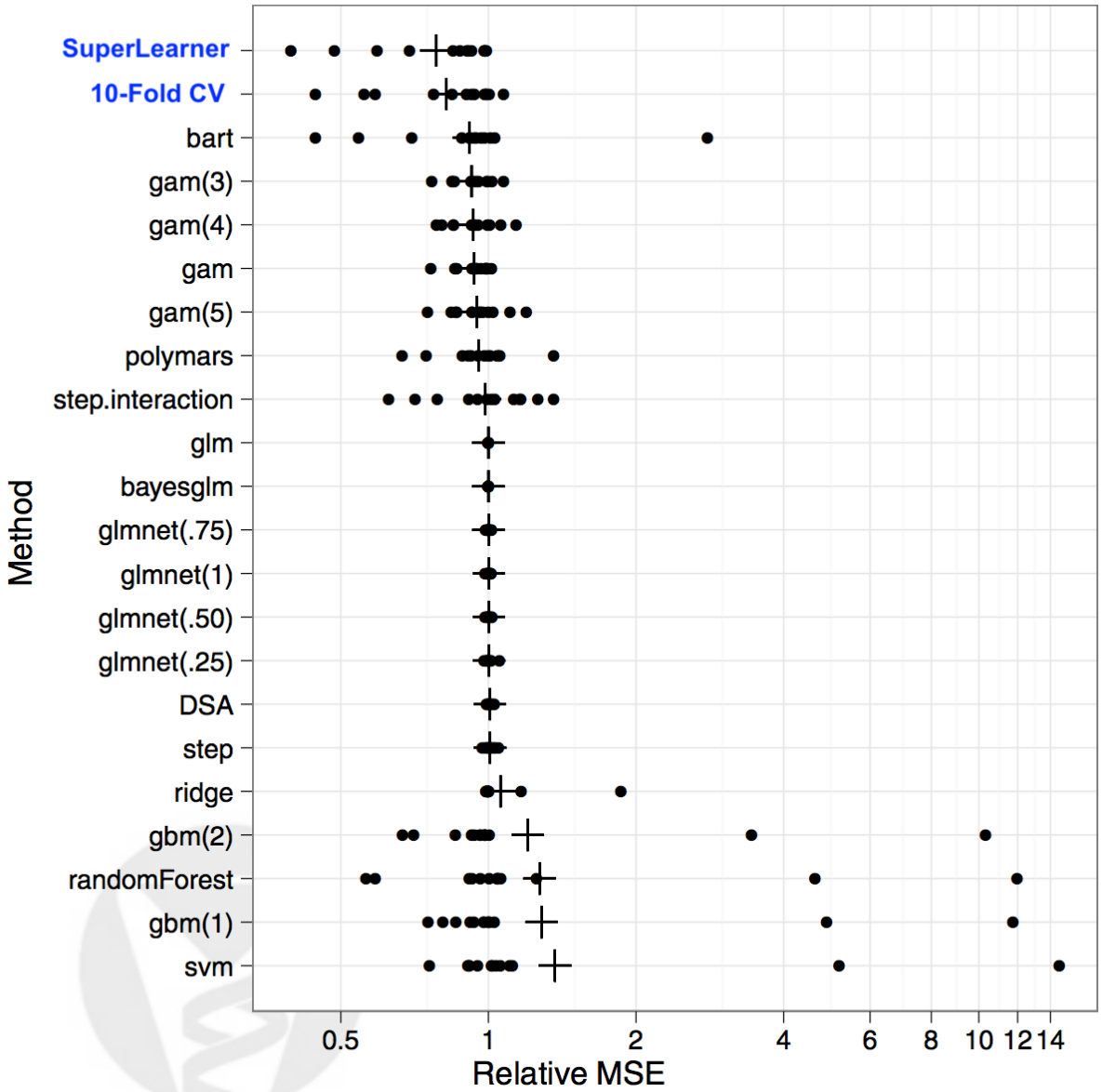


image source: Polley & van der Laan (2010)

SuperLearner to estimate the contrast function

- Can directly estimate the contrast function $\Delta(X) = E[D|A=0,X] - E[D|A=1,X]$ using SuperLearner
  - Allows us to focus exclusively on estimating how X influences the treatment effect
  - In a linear model, this would correspond to estimating the interaction term without needing to estimate the main effect
  - Can make it much easier to estimate Δ
- The approach involves defining a pseudo-outcome Y:

$$Y = \frac{1-2A}{P(A|X,\hat{\gamma})} D$$

and regressing this pseudo-outcome against X only (not A)

- A more efficient approach uses pseudo-outcome

$$\frac{1-2A}{P(A|X,\hat{\gamma})} \left(D - \hat{E}[D|A,X] \right) + \hat{E}[D|A=0,X] - \hat{E}[D|A=1,X],$$

where $\hat{E}[D|A,X]$ is an estimate of $E[D|A,X]$

Luedtke & van der Laan (*Int J Biostat*, 2016)

SuperLearner Summary

- Advantages:
  - Can give optimal estimates of $E[D|A, X]$ by optimally selecting from a user-specified collection of modeling approaches, which in turn provides gurantees about the quality of the treatment rule[1]
  - Estimated magnitude of effect for a stratum X can be computed
  - Also can directly estimate the contrast function or perform direct optimization using the SuperLearner framework[2]
- Disadvantage:
  - Because SuperLearner allows for very flexible regression models, the models may be difficult to interpret

[1] Qian & Murphy (*AoS*, 2011)
[2] Luedtke & van der Laan (*Int J Biostat*, 2016)

- Examples

Depression Data

- Compare drug therapy ($A = 0$) with drug + behavioral therapy ($A = 1$)
- Five covariates: Age, Gender, HAMABase (pre-treatment total Hamilton Anxiety Rating Scale score), Sleep (sleep disturbance score), Mood (mood cognition score)
- Response: 24-item Hamilton Rating Scale for Depression
- Number of patients: 436

Analyzing Depression Data

- Q-learning: model the depression score using the covariate, the treatment and their interactions

$$D \sim 1 + X + A + XA$$

- Efficient Augmentation and Relaxation Learning: will model both the outcome and the propensity score
  - Logistic loss: $\phi(t) = \log(1 + e^{-t})$
  - Outcome model: $D \sim 1 + X + A + XA$
  - Propensity model: $A \sim X$

Results

- Q-learning: $\hat{d}(X) = I(-0.83 + 0.01Age - 0.55Gender + 0.06HAMABase + 0.01Sleep - 0.04Mood < 0)$.
- Efficient Augmentation and Relaxation Learning: $\hat{d}(X) = I(-0.94 + 0.00Age - 0.33Gender + 0.05HAMABase + 0.02Sleep - 0.01Mood < 0)$.

Simulation Example

- $X_1, \ldots, X_5 \sim Uniform(-1, 1)$
- $A \sim \{0, 1\}$ w.p. 0.5
- $D \sim 3 + X_1^2 + X_2^2 + (2X_1 + X_3 - 1)A + N(0, 1)$
- The optimal rule: $d^*(x) = I(2x_1 + x_3 < 1)$

Simulation Example

- $X_1, \ldots, X_5 \sim Uniform(-1, 1)$
- $A \sim \{0, 1\}$ w.p. 0.5
- $D \sim 3 + X_1^2 + X_2^2 + (2X_1 + X_3 - 1)A + N(0, 1)$
- The optimal rule: $d^*(x) = I(2x_1 + x_3 < 1)$

Simulation Example

```
set.seed(1111)

n  = 300
p  = 5
X  = matrix(runif(n*p,-1,1),n,p)
A  = rbinom(n,1,0.5)
mX = 3 + X[,1]^2 + X[,2]^2
cX = 2*X[,1]+ X[,3] -1
D  = mX + A*cX + rnorm(n,1)

## optimal rule
dstar  = (cX<0)
> table(dstar)
dstar
FALSE  TRUE
   85   215
```

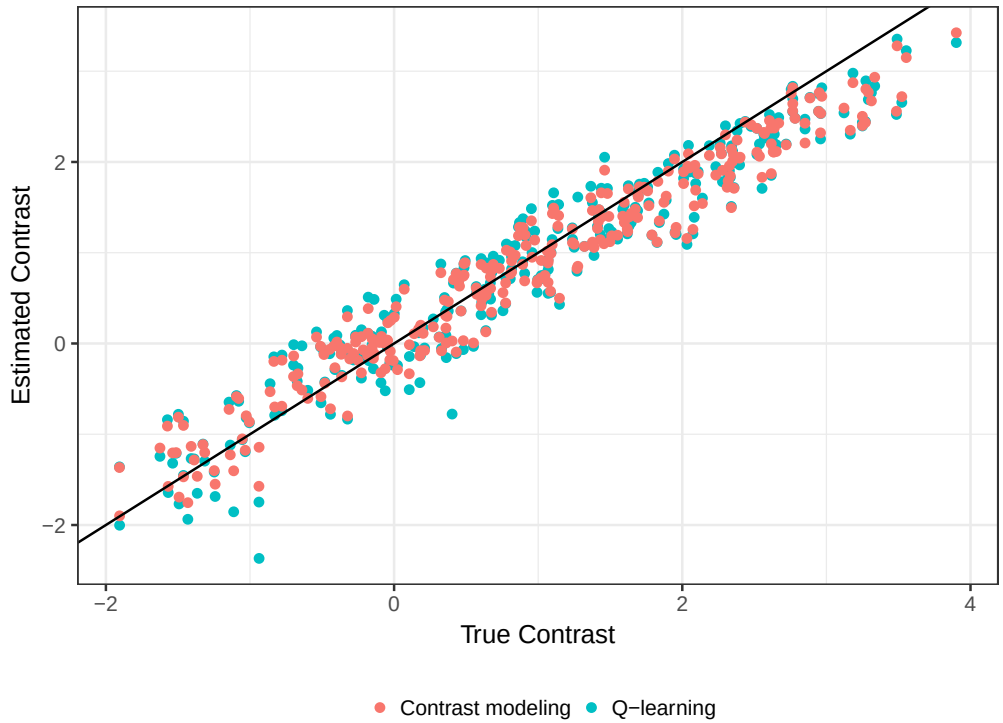
Simulation Example: Q learning (regression modeling)

```
library(SuperLearner)

# candidate algorithms: run "listWrappers()" to see more
SL.library = c("SL.glm","SL.glm.interaction","SL.nnet",
  "SL.cforest","SL.gam","SL.glmnet")

# SuperLearner calls for E[D|A=0,X] and E[D|A=1,X]
SL.out0 = SuperLearner(D[A==0],data.frame(X)[A==0,],
  newX=data.frame(X),SL.library=SL.library,family=gaussian())
SL.out1 = SuperLearner(D[A==1],data.frame(X)[A==1,],
  newX=data.frame(X),SL.library=SL.library,family=gaussian())

# Q estimates
Q0 = SL.out0$SL.predict[,1]
Q1 = SL.out1$SL.predict[,1]

# contrast function as estimated by Q-learning
Q.contrast = Q0-Q1

# Q-learning rule
QTrtRec = as.numeric(Q.contrast>0)
QTrtRec
  0   1
 80 220
```

Simulation Example: Directly modeling the contrast

```
library(SuperLearner)

# candidate algorithms: run "listWrappers()" to see more
SL.library = c("SL.glm","SL.glm.interaction","SL.nnet",
  "SL.cforest","SL.gam","SL.glmnet")

# Defining a data frame of X
Xdf = data.frame(X)

PA1givenX = predict(glm(A~.,data=Xdf,family=binomial),type="response")

# Use AIPW pseudo-outcome
pseudoOutcome = (1-2*A)*(D - A*Q1 - (1-A)*Q0)/(A*PA1givenX + (1-A)*PA1givenX) + Q0-Q1

# Run SL. Specifying "family=gaussian()" because outcome is continuous
# and this will to minimize mean-squared error
SL.out = SuperLearner(pseudoOutcome,Xdf,SL.library=SL.library,family=gaussian())

# Contrast function estimates
direct.contrast = SL.out$SL.predict[,1]

# Contrast estimation rule
contrastTrtRec = as.numeric(direct.contrast>0)

table(ContrastTrtRec)
contrastTrtRec
  0   1
 70 230
```

Comparison of contrast function estimates




```
library(ggplot2)

# Comparison of contrast function estimates at the /observed/ X's
df = data.frame(X=c(-cX,-cX),val=c(Q.contrast,direct.contrast),
  method=rep(c("Q-learning","Contrast modeling"),each=n))

ggplot(data=df,aes(x=X,y=val,colour=method)) + theme_bw() +
  geom_point() + geom_abline(a=0,b=1) + xlab("True Contrast") +
  ylab("Estimated Contrast") +
  theme(legend.title=element_blank(),legend.position="bottom")
```

Simulation Example: Restricted Rule

- R package: DynTxRegime, methods for Estimating Optimal Dynamic Treatment Regimes, including single decision setup
- For restricted regime:

```
## A doubly robust Augmented Inverse Propensity Weighted Estimator (AIPWE) or Inverse
Propensity Weighted Estimator (IPWE) for population mean outcome is optimized over a
restricted class of regimes. Methods are available for both single-decision-point and multiple-
decision-point regimes. This method requires the rgenoud package.

Usage

optimalSeq(..., moPropen, moMain, moCont, data, response, txName, regimes,
           fSet = NULL, refit = FALSE, iter = 0, verbose = TRUE)
```

Simulation Example: OWL/EARL

- For OWL/EARL:

```
##Estimation of optimal treatment regime using efficient augmentation and relaxation
learning (EARL). The method is limited to single-decision-point scenarios with binary
treatment options.

## by setting moMain and moCont to NULL, the function is to estimate the optimal
treatment regime using outcome weighted learning (OWL).


Usage

earl(..., moPropen, moMain, moCont, data, response, txName, regime,
     iter = 0L, lambdas = 0.5, cvFolds = 0L, surrogate = "hinge",
     guess = NULL, verbose = TRUE)
```

- There is also a function on OWL implementation with more features (R function: owl). See help for details.

Simulation Example: Restricted Rule

```
library(DynTxRegime)

# implementation to estimate the optimal restricted rule
# Data Preparation
 data <- data.frame(X, A, D)
 colnames(data) <- c("x1", "x2", "x3", "x4", "x5","a","D")


# Define the propensity for treatment model and methods.
 moPropen<- buildModelObj(model =  ~ x1 + x2 + x3 + x4 + x5,
                                   solver.method = 'glm',
                                   solver.args = list('family'='binomial'),
                                   predict.method = 'predict.glm',
                                   predict.args = list(type='response'))


# Create modelObj object for main effect component
  moMain <- buildModelObj(model = ~ x1 + x2 + x3 + x4 + x5,
                           solver.method = 'lm')

# Create modelObj object for contrast component
  moCont <- buildModelObj(model = ~ x1 + x2 + x3 + x4 + x5,
                           solver.method = 'lm')
```

Simulation Example: Restricted Rule

```
# treatment regime rules at each decision point.
  regimes <- function(a,b,c,d, e, f, data){
               as.numeric( a + b*data$x1 + c*data$x2 + d*data$x3 + e*data$x4 + f*data$x5 > 0)
    }

# genoud requires some additional information
  c1 <- c(-1,-1,-1,-1,-1,-1)
  c2 <- c( 1, 1, 1, 1, 1, 1)
  Domains <- cbind(c1,c2)
  starts <- c(0,0,0,0,0,0)

#!! A LARGER VALUE FOR POP.SIZE IS RECOMMENDED
#!! THIS VALUE WAS CHOSEN TO MINIMIZE RUN TIME OF EXAMPLES
  pop.size <- 50
```

Simulation Example: Restricted Rule

```
 estAIPWE <- optimalSeq(moPropen = moPropen,
                        moMain = moMain,
                        moCont = moCont,
                        data = data,
                        response = -data$D,
                        txName =  "a",
                        regimes = regimes,
                        iter=0L,pop.size = pop.size, starting.values = starts,
                        Domains = Domains, solution.tolerance = 0.0001)

> regimeCoef(estAIPWE)
           a            b            c            d            e            f
4.506975e-01 -7.614161e-01 -5.267877e-05 -5.334575e-01  3.900155e-03 -1.398331e-01

 AIPWTrtRec<- optTx(estAIPWE)

> table(AIPWTrtRec)
AIPWTrtRec
  0   1
 70 230
```

Simulation Example: EARL

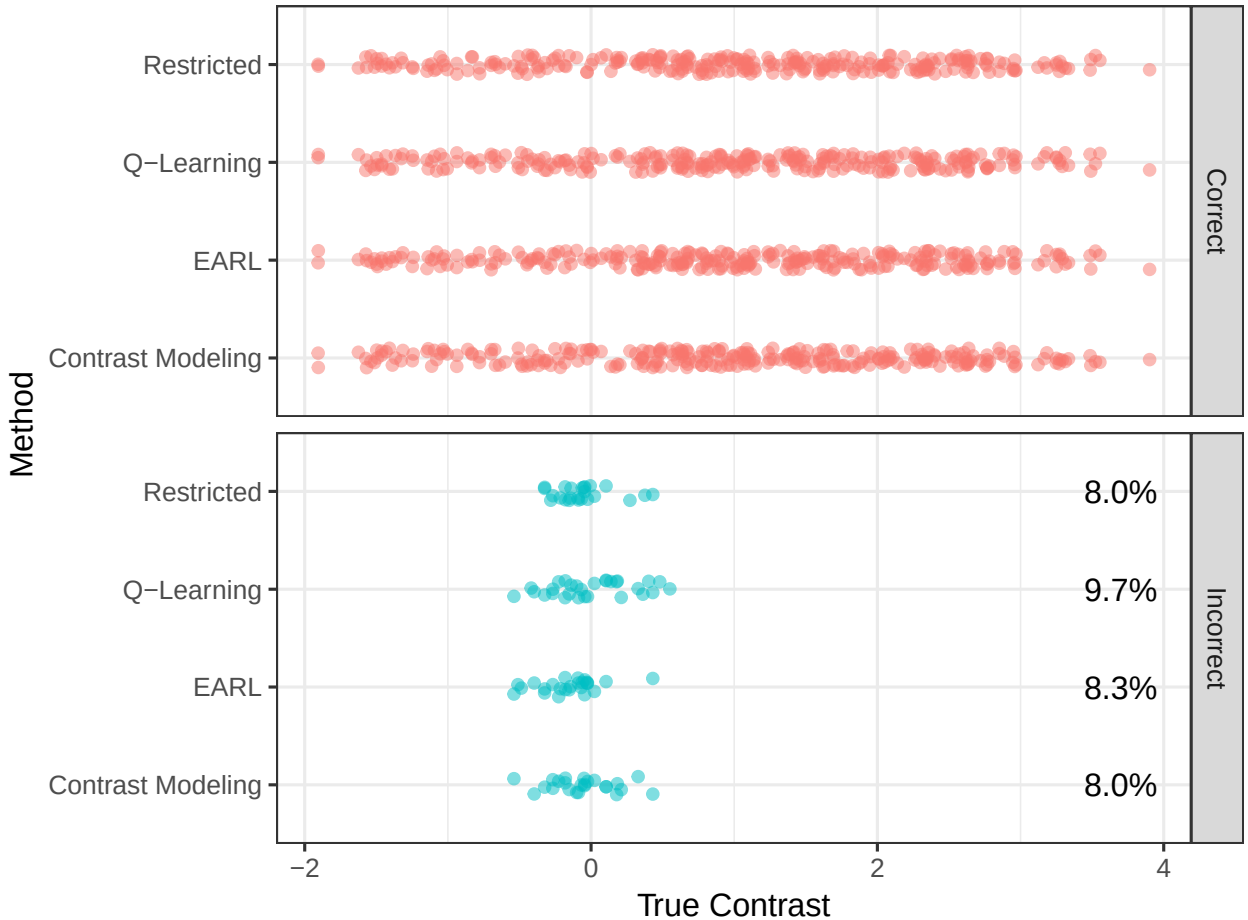
```
library(DynTxRegime)

# Data Preparation
 data <- data.frame(X, A, D)
 colnames(data) <- c("x1", "x2", "x3", "x4", "x5","a","D")


# Define the propensity for treatment model and methods.
 moPropen<- buildModelObj(model =  ~ x1 + x2 + x3 + x4 + x5,
                                   solver.method = 'glm',
                                   solver.args = list('family'='binomial'),
                                   predict.method = 'predict.glm',
                                   predict.args = list(type='response'))


# Create modelObj object for main effect component
  moMain <- buildModelObj(model = ~ x1 + x2 + x3 + x4 + x5,
                           solver.method = 'lm')

# Create modelObj object for contrast component
  moCont <- buildModelObj(model = ~ x1 + x2 + x3 + x4 + x5,
                           solver.method = 'lm')
```

Simulation Example: EARL

```
 earlRes <- earl(moPropen = moPropen, moMain = moMain,
                   moCont = moCont,
                   data = data, response = -data$D, txName = "a", surrogate = 'logit',
                   regime = ~ x1 + x2 + x3 + x4 + x5, lambdas=2^seq(-5,5,1), cvFolds = 5)

> regimeCoef(earlRes)
[1]  0.39663271 -0.59853084 -0.14985610 -0.34259186  0.00478191 -0.02726041


EARLTrtRec <-  optTx(earlRes)$optimalTx

EARLTrtRec <- (EARLTrtRec + 1)/2  ## change coding from (-1,1) to (0,1)

> table(EARLTrtRec)
EARLTrtRec
  0   1
 64 236
```

Simulation Example: Performance Comparison

- Compare predictions to those of optimal rule



Percentages indicate percent discrepancy with true optimal rule in our data set.

- Can further validate performance on an independent data set.

Summary

- Active research area.
- Regression modeling: easy to implement; model may be misspecified.
- Direct optimization: more robust.
- SuperLearner provides a means to learn from the data which method best estimates the optimal treatment rule for the given setting

Extra slides

Simulation Example: OWL

```
library(DynTxRegime)

# Data Preparation
 data <- data.frame(X, A, D)
 colnames(data) <- c("x1", "x2", "x3", "x4", "x5","a","D")


# Define the propensity for treatment model and methods.
 moPropen<- buildModelObj(model =  ~ x1 + x2 + x3 + x4 + x5,
                                   solver.method = 'glm',
                                   solver.args = list('family'='binomial'),
                                   predict.method = 'predict.glm',
                                   predict.args = list(type='response'))
```

Simulation Example: OWL

```
owlRes <-  earl(moPropen = moPropen, moMain = NULL, moCont = NULL,
                data = data, response = -data$D, txName = "a", surrogate = 'logit',
                regime = ~ x1 + x2 + x3 + x4 + x5, lambdas=2^seq(-5,5,1), cvFolds = 5)
```

```
> regimeCoef(owlRes)
[1]  0.42115454 -0.65789664 -0.25178980 -0.33182440 -0.09571889 -0.03276892
```

```
 OWLTrtRec <-  optTx(owlRes)$optimalTx

 OWLTrtRec <- (OWLTrtRec + 1)/2  ## change coding from (-1,1) to (0,1)
```

```
> table(OWLTrtRec)
OWLTrtRec
  0   1
 67 233
```