10 - High-Dimensional Heterogeneous Effects

ml4econ, HUJI 2025

Itamar Caspi

June 8, 2025 (updated: 2025-06-08)

Replicating this Presentation

```r
library(tidyverse)
library(tidymodels)
library(htetree)
library(experimentdatar)
library(rpart.plot)
library(broom)
library(knitr)
library(xaringan)
```

Outline

- Heterogenous Treatment Effects (HTE)
- Challenges in Estimating HTE
- Introducing Causal Trees (and Forests)
- Empirical Illustration

Heterogeneous Treatment Effects

Treatment and Potential Outcomes: Rubin (1974, 1977)

- Treatment Definition

$$D_i = \begin{cases} 1, & \text{if unit } i \text{received the treatment} \\ 0, & \text{otherwise.} \end{cases}$$

- Potential Outcomes

$$\begin{aligned} Y_{0i} & \quad \text{denotes the potential outcome for unit } i \text{ without the treatment } (D_i = 0) \\ Y_{1i} & \quad \text{denotes the potential outcome for unit } i \text{ with the treatment } (D_i = 1) \end{aligned}$$

Treatment and Potential Outcomes: Rubin (1974, 1977) Cont.

- Observed Outcome: Under the Stable Unit Treatment Value Assumption (SUTVA - no leakage, same dose), we calculate the outcome for unit i as

$$Y_i = Y_{1i}D_i + Y_{0i}(1 - D_i)$$

- Individual Treatment Effect: We quantify this as the difference between unit i's potential outcomes:

$$\tau_i = Y_{1i} - Y_{0i}$$

> The Fundamental Problem of Causal Inference (Holland, 1986): We cannot simultaneously observe both Y_{1i} and Y_{0i}, and as a result, τ_i is unobservable.

Assumption of Random Treatment Assignment

In this discussion, we operate under the assumption that treatments are randomly assigned. This implies that D_i is *independent* of potential outcomes, stated as:

$$\{Y_{1i}, Y_{0i}\} \perp D_i$$

Remember, randomized control trials (RCTs) allow us to calculate the Average Treatment Effect (ATE) using the average difference in outcomes based on treatment status:

$$\mathbf{ATE} = \mathbb{E}\left[Y_i | D_i = 1\right] - \mathbb{E}\left[Y_i | D_i = 0\right]$$

and its corresponding sample estimate:

$$\hat{\tau} = \frac{1}{n_T} \sum_{i \in Treatment} Y_i - \frac{1}{n_C} \sum_{i \in Control} Y_i$$

In essence, the difference in average outcomes between the treatment and control groups provides an unbiased estimate.

The Importance of Understanding HTE

- There are often reasons to suspect that a treatment may impact individuals differently, for instance,
    - Younger subjects might have a better response to a certain medication.
    - Short-term unemployed individuals may benefit more from job-training programs.
- Consequently, gaining insights into treatment effect heterogeneity allows for more effective treatment allocation:
    - Treatments can be tailored and targeted towards those who stand to gain the most.

Treatment Effect Heterogeneity

Let's recall the definition of Average Treatment Effect (ATE):

$$\tau = \mathbb{E}[Y_{i1} - Y_{i0}]$$

We define the Conditional Average Treatment Effect (CATE) as:

$$\tau(x) = \mathbb{E}[Y_{i1} - Y_{i0} | X_i = x]$$

where x denotes a specific value of X_i or a range of values (representing a subspace of the feature space).

Challenges in Estimating HTE

Moving the Goalpost

Our primary interest lies in determining "personalized" treatment effects.

We can consider Conditional Average Treatment Effects (CATE) as a balance between ATE and personalized treatment effects.

CATEs represent ATEs for specific subgroups of individuals. We classify these subgroups based on the X_i's. This can be formally expressed as:

$$\text{CATE} = \tau(x) = \mathbb{E}[Y_{1i} - Y_{0i} | X_i = x, x \in \mathbb{X}]$$

Here, x denotes a partition of the feature space $\mathbb{X}$.

> For instance, x might represent a subgroup consisting of individuals under 18 years old who weigh more than 75 kg.

Striking a Balance: The Bias-Variance Trade-off in HTEs

In an ideal scenario, we want to uncover "personalized" treatment effects, meaning the effect of treatment on an individual with a specific set of features $X_i = x$.

If your goal is to learn an individual's effect τ_i, using the population or subgroup average is increasingly biased the further you move away from that individual.

$$Bias(\hat{\tau}) > Bias(\hat{\tau}(x)) > Bias(\hat{\tau}_i)$$

However, with increased personalization, the noise in our estimate rises:

$$Var(\hat{\tau}) < Var(\hat{\tau}(x)) < Var(\hat{\tau}_i)$$

> Bottom line: Moving from the overall ATE to more personalised effects can reduce bias with respect to an individual's true effect, but at the cost of higher variance and stronger identification assumptions. The practical goal is to pick a level of granularity (or amount of smoothing/shrinkage) that minimises mean-squared error.

Estimating CATE Using Linear Regression

The most prevalent approach is to estimate the *best linear projection* (BLP) for $\mu = \mathbb{E}[Y_i | X_i = x]$, incorporating interaction terms between the treatment and the set of features.

As an example, for a binary treatment D_i and a single feature X_i, we estimate the following regression via Ordinary Least Squares (OLS):

$$Y_i = \alpha + \tau D_i + \beta X_i + \gamma D_i X_i + u_i,$$

Here, the coefficient γ denotes the interaction effect and reflects the difference between ATE and the effect of D_i among individuals with $X_i = x$.

> NOTE: The parameter γ holds a causal interpretation only when X_i is randomly assigned.

Limitations of the Best Linear Projection (BLP) Approach

1. This method becomes unmanageable when the quantity of attributes and interaction terms significantly outweighs the number of observations.

2. While Lasso can assist when $k \gg n$, it may introduce omitted variable bias. For example, Lasso might eliminate some of the main effects.

3. Managing and keeping track of numerous variables and their interactions (bookkeeping) can become cumbersome.

Introducing Causal Trees (and Forests)

Notation Summary: Data and Observations

Data

- Y_i: observed outcome for individual i
- X_i: attribute vector for individual i
- D_i: binary treatment indicator $\{0, 1\}$

Sample

- $\mathcal{S}$: the full sample
- $\mathcal{S}^{tr}$: training sample
- $\mathcal{S}^{te}$: test sample
- $\mathcal{S}^{est}$: estimation sample
- $\mathcal{S}_{treat}$: treatment group
- $\mathcal{S}_{control}$: control group

Observations

- N: total number of observations
- N^{tr}: size of the training sample
- N^{te}: size of the test sample
- N^{est}: size of the estimation sample

Notation Summary: Trees and CATE

Tree

- $\mathbb{X}$: attribute space
- $\mathbf{\Pi}$: a partitioned tree
- $\#(\mathbf{\Pi})$: number of partitions
- ℓ_j: a leaf of $\mathbf{\Pi}$, such that $\cup_{j=1}^{\#(\mathbf{\Pi})} \ell_j = \mathbb{X}$
- $\ell(x; \mathbf{\Pi})$: Leaf assignment function. For any covariate vector $x \in \mathcal{X}, \ell(x; \mathbf{\Pi}) \in \mathbf{\Pi}$ denotes the unique leaf such that $x \in \ell(x; \mathbf{\Pi})$

Treatment

- $\tau(\ell_j)$: CATE within leaf ℓ_j
- p: marginal treatment probability, represented as $P(D_i = 1)$

Hypothetical Scenario: Observable τ_i

Let's imagine we have data on τ_i and X_i for $i = 1, \ldots, N$.

Our goal is to predict τ_i for an individual with X_i equal to some x, in an out-of-sample scenario.

A simple approach could be to fit a regression tree to the data. Splits would be based on in-sample fit, expressed as:

$$\frac{1}{N}\sum_{i=1}^{N}(\tau_i - \hat{\tau}(X_i|\mathcal{S}^{tr}, \Pi))^2$$

We would use cross-validation for regularization (also known as pruning).

Causal tree (Athey and Imbens, PNAS 2016)

Goal: Estimate Conditional Average Treatment Effects (CATE) $\tau(x)$.

Basic Idea: Use a regression tree to partition the attribute space $\mathbb{X}$.

Challenges:

1. Traditional trees split leaves based on Y_i. However, we are interested in τ_i, which is unobserved.
2. How to determine the regularization criteria?
3. How to form confidence intervals?

Solutions:

1. Split the tree based on the heterogeneity and accuracy of $\tau(x)$.
2. Regulate based on treatment effect heterogeneity and accuracy within leaves.
3. Implement sample splitting: Build the tree on one sample and estimate CATE on a different, independent sample.

The Naive Approach

We could use an off-the-shelf CART algorithm to:

1. Estimate two separate trees to predict outcomes Y_i, one for treated and one for control
2. Estimate a single tree for Y_i, focusing on splits in D_i (hoping they exist).

> **Problem:** The naive approaches (tree construction and cross-validation) are optimized for outcome heterogeneity, not treatment heterogeneity. They implicitly rely on the assumption that treatment is highly correlated with the X_i's.

Concrete toy illustration

Data-generating process:

$$Y_i = 100 - 15 \times \mathrm{Age}_i + \underbrace{10 \cdot 1\{\mathrm{Age}_i < 30\}}_{\text{treatment effect if treated}} D_i + \varepsilon_i$$

- Age is a huge predictor of Y (slope -15).
- The treatment effect is non-zero only for young people, but its magnitude (10) is small compared with outcome swings driven by age.

> A naïve CART (either recipe) will usually split on Age at roughly 40 y (biggest reduction in Y-variance), ending up with one giant "young" leaf and one giant "old" leaf in both treated and control trees. Inside each leaf the treated-control gap is nearly constant, so no heterogeneity is uncovered. The true effect-the extra 10 units for the under-30s only-is never isolated.

Approach #1: Transformed Outcome Trees (TOT)

Suppose we have an RCT with a 50% probability of receiving the treatment. Define

$$Y_i^* = \begin{cases} 2Y_i & \text{if } D_i = 1, \\ -2Y_i & \text{if } D_i = 0. \end{cases}$$

In this case, Y_i^* becomes an unbiased estimate for individual i's τ_i.

Proof: Given that we're in a 50-50 RCT,

$$\mathbb{E}[Y_i] = \frac{1}{2}\mathbb{E}[Y_{1i}] + \frac{1}{2}\mathbb{E}[Y_{0i}]$$

The expectation is with respect to the *probability of being treated*. Similarly,

$$\begin{aligned} \mathbb{E}[Y_i^*] &= 2\left(\frac{1}{2}\mathbb{E}[Y_{1i}] - \frac{1}{2}\mathbb{E}[Y_{0i}]\right) \\ &= \mathbb{E}[\tau_i]. \end{aligned}$$

Not 50-50 Assignment

More generally, if the probability of treatment assignment is denoted by p, then

$$Y_i^* = \frac{D_i - p}{p(1-p)} Y_i = \begin{cases} \frac{1}{p} Y_i & \text{if } D_i = 1 \\ -\frac{1}{1-p} Y_i & \text{if } D_i = 0 \end{cases}$$

In observational studies, p can be estimated based on the X's, i.e., use $\hat{p}(X)$ instead of setting a constant p for all i.

Once Y_i^* is defined, we can proceed with standard tree methods for prediction:

1. Use a conventional algorithm (e.g., `rpart`) to fit a tree to predict Y_i^*.
2. Use the mean of Y_i^* within each leaf as the estimate for $\tau(x)$.

Problems with the TOT Approach

PROBLEM: The Transformed Outcome Trees (TOT) approach estimates CATE as the average Y_i^* within each leaf, and not as the difference in average outcome between the treatment and control groups.

EXAMPLE: Consider a leaf ℓ with 7 treated and 10 untreated. In this case, $\mathbf{CATE}(\ell)$ will be the average of Y_i^*, for $i = 1, \ldots, 17$.

Ideally, what we want is to calculate the difference between the average of Y_i^* in the Treatment group and the average of Y_i^* in the Control group:

$$\hat{\tau}(\ell) = \bar{Y}_{\ell,1} - \bar{Y}_{\ell,0}, \quad \widehat{\mathrm{SE}}[\hat{\tau}(\ell)] = \sqrt{\frac{S^2_{\ell,1}}{n_{\ell,1}} + \frac{S^2_{\ell,0}}{n_{\ell,0}}}$$

(NOTE: As we will discuss later, the `causalTree` package estimates $\hat{\tau}$ within each leaf instead of $\widehat{Y}^*$.)

An Aside: Sample Splitting and Honest Estimation

Sample Splitting: This involves dividing the data in half, computing the sequence of models on one half, and then evaluating their significance on the other half.

COST: This approach can lead to a significant loss of power, unless the sample size is large.

BENEFIT: It provides valid inference due to independent subsamples.

In the context of causal trees, sample splitting involves constructing a tree using the training sample $\mathcal{S}^{tr}$ and estimating the effect using the estimation sample $\mathcal{S}^{est}$.

Approach #2: Causal Tree (CT)

> Where do the covariates make the treatment effect different?

Solution: We define $\hat{\tau}_i$ as the Average Treatment Effect (ATE) within the leaf.

Athey and Imbens propose two splitting rules:

1. Adaptive Causal Tree (CT-A):

$$-\widehat{\text{MSE}}_\tau\left(\mathcal{S}^{\text{tr}}, \mathcal{S}^{\text{tr}}, \Pi\right) = \underbrace{\frac{1}{N^{\text{tr}}} \sum_{i \in \mathcal{S}^{\text{tr}}} \hat{\tau}^2\left(X_i | \mathcal{S}^{\text{tr}}, \Pi\right)}_{\text{Variance of treatment}}$$

In other words: The CT-A splitting rule chases splits that magnify the variance of the estimated treatment effect across leaves; it cuts the feature space wherever the treatment looks most different for different kinds of units.

Approach #2: Causal tree (CT)

1. Honest causal tree (CT-H) which uses sample splitting:

$$\widehat{\text{EMSE}}_\tau\left(\mathcal{S}^{\text{tr}},\Pi\right) \equiv \frac{1}{N^{\text{tr}}}\sum_{i\in\mathcal{S}^{\text{tr}}}\hat{\tau}^2\left(X_i;\mathcal{S}^{\text{tr}},\Pi\right)$$

$$-\frac{2}{N^{\text{tr}}}\cdot\sum_{\ell\in\Pi}\left(\frac{S^2_{\mathcal{S}^{\text{tr}}_{\text{treat}}}(\ell)}{p}+\frac{S^2_{\mathcal{S}^{\text{tr}}_{\text{control}}}(\ell)}{1-p}\right).$$

where $S^2_{\mathcal{S}^{tr}_{\text{control}}}(\ell)$ is the within-leaf variance on outcome Y for $\mathcal{S}^{tr}_{\text{control}}$ control group in leaf ℓ, and $S^2_{\mathcal{S}^{tr}_{\text{treat}}}(\ell)$ is the counter part for $\mathcal{S}^{tr}_{\text{treat}}$ treatment group.

In words: CT-H keeps a candidate split only when “the jump in treatment effects” it reveals is large enough to outweigh “the extra statistical noise” created by having smaller leaves—measured on a sample that will not be used to estimate the effects.

More Intuition about CT

- Big leaves, tiny variance – no penalty, but maybe no heterogeneity found.
- Tiny leaves, huge variance – penalty overwhelms, split is rejected.
- Moderate split that pulls treatment effects apart and still leaves decent sample sizes – passes the EMSE test and is kept.

Additional Splitting Rules

Athey and Imbens (2016) propose two additional splitting rules:

1. Fit-Based Trees: The split is based on the goodness-of-fit of the *outcome*, where fitting takes into account the binary treatment indicator D_i.

2. Squared T-Statistic Trees: The split is performed according to the largest value of the square of the t-statistic for testing the null hypothesis that the average treatment effect is the same in the two potential leaves.

Please refer to the Athey and Imbens (2016) paper for more details.

Enhancing Causal Trees: Cross-Validation and Pruning

- Cross-validation in causal trees utilizes the *out-of-sample* version of the *goodness-of-fit* rule. This rule is essential for tree construction.

- Specifically, we split the training sample into two sets:

  - A **training set**, denoted as $\mathcal{S}^{tr,tr}$.
  - A **validation set**, denoted as $\mathcal{S}^{tr,cv}$.

- The tree's construction is based on the training set, $\mathcal{S}^{tr,tr}$, and its validation uses the set $\mathcal{S}^{tr,cv}$.

- Based on validation performance, we can **prune** the tree. This step simplifies the model, enhances interpretability, and potentially improves prediction performance.

Summarizing the Causal Tree Algorithm

1. Randomly split the sample $\mathcal{S}$ into two halves, forming:
   - A **training sample**, $\mathcal{S}^{tr}$
   - An **estimation sample**, $\mathcal{S}^{est}$
2. Using only $\mathcal{S}^{tr}$, construct a tree. Each split follows a criteria aiming to maximize:
   - The *variability* of treatment effect estimates across the resulting subgroups, thereby increasing treatment heterogeneity.
   - The *accuracy* of these estimates, thus reducing estimate variance.
3. Using only $\mathcal{S}^{est}$, calculate $\tau(x \in \ell)$ within each terminal leaf, ℓ.

Implementing Causal Trees: Key Considerations

- The `{causalTree}` package (by Athey) provides a convenient implementation of the causal tree algorithm.

- Better to use `{hteree}`

- Users must select:

    - `minsize`: This defines the minimum number of treatment and control observations in each leaf.
    - `bucketNum`: At every step, shift chunks big enough (≥ this many treated & controls) so the score doesn’t yo-yo on single observations.
    - `bucketMax`: Even in huge leaves, don’t create more than this many chunks; that keeps computation reasonable.

Causal Forests (Wager and Athey, JASA 2018)

Causal Forests tackle the issue of noise in individual causal trees. They can reduce variance through forest creation. Here's an overview of the causal forest algorithm:

1. Draw a subsample b without replacement from the N observations in the dataset.
2. Split b randomly in half to form: A training sample, $\mathcal{S}_b^{tr}$ and an estimation sample, $\mathcal{S}_b^{est}$
3. Using only $\mathcal{S}_b^{tr}$, grow a tree Π_b. Each split follows criteria aiming to maximize:
    - The *variability* of treatment effect estimates across the resulting subgroups (increasing treatment heterogeneity)
    - The *accuracy* of these estimates (minimizing estimate variance)
4. Use $\mathcal{S}_b^{est}$ only to calculate $\hat{\tau}_b(x \in \ell)$ within each terminal leaf.
5. Return to the full sample N and assign for each i, based on where it is located in Π_b.
6. Repeat steps 1-5 B times.
7. Define subject i's Conditional Average Treatment Effect (CATE) as $B^{-1} \sum_{j=1}^{B} \hat{\tau}_b$.

Implementing Causal Forests: Key Considerations

- The `{grf}` package (Tibshirani, Athey, and Wager) offers a handy implementation of the causal forest algorithm.

- Users must select:

  - The number of trees.
  - The subsample size.
  - The minimum number of treatment and control observations in each leaf.
  - The number of variables considered at each split (`mtry`).

- For an excellent practical application of causal forests, refer to Davis and Heller (2017). They applied causal forests to an RCT that evaluated the impact of a summer jobs program on disadvantaged youth in Chicago.

Empirical Illustration

{experimentdatar}

A description from the `{experimentdatar}` GitHub repository:

> *"The `experimentdatar` data package contains publicly available datasets that were used in Susan Athey and Guido Imbens' course "Machine Learning and Econometrics" (AEA continuing Education, 2018). The datasets are conveniently packed for R users."*

You can install the *development* version from GitHub

```r
install.packages("pak")
pak::pak("itamarcaspi/experimentdatar")
```

The `social` dataset

The data is from Gerber, Green, and Larimer (2008)'s paper "Social Pressure and Voter Turnout: Evidence from a Large-Scale Field Experiment".

For this illustration, we will make use of the `social` dataset

```
data(social)
```

The following command will open a link to Gerber, Green, and Larimer (2008)'s paper

```
dataDetails("social")
```

Design of the Voter Experiment

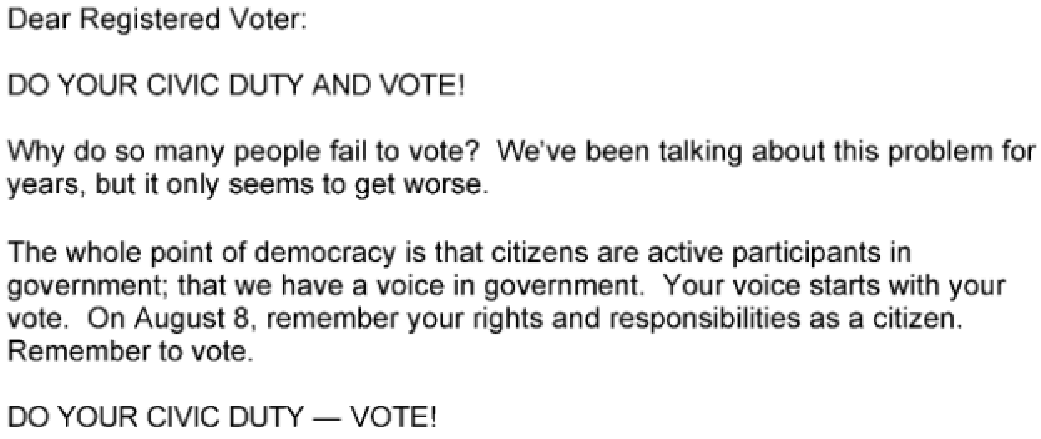
Researchers wanted to see whether a little social pressure makes people more likely to vote. So, just before Michigan's August 2006 primary election, they randomly split about 180,000 households into five groups and sent five kinds of postcards

We will focus on a sample of voters underwent random assignment into two groups:

- Treatment group $(D_i = 1)$: This group received a message stating that, post-election, the recent voting record of everyone in their household would be shared with their neighbors.
- Control group $(D_i = 0)$: This group did not receive any message.

The goal of this study is to investigate a potential "social pressure" effect on voter turnout.

The treatment and control messages



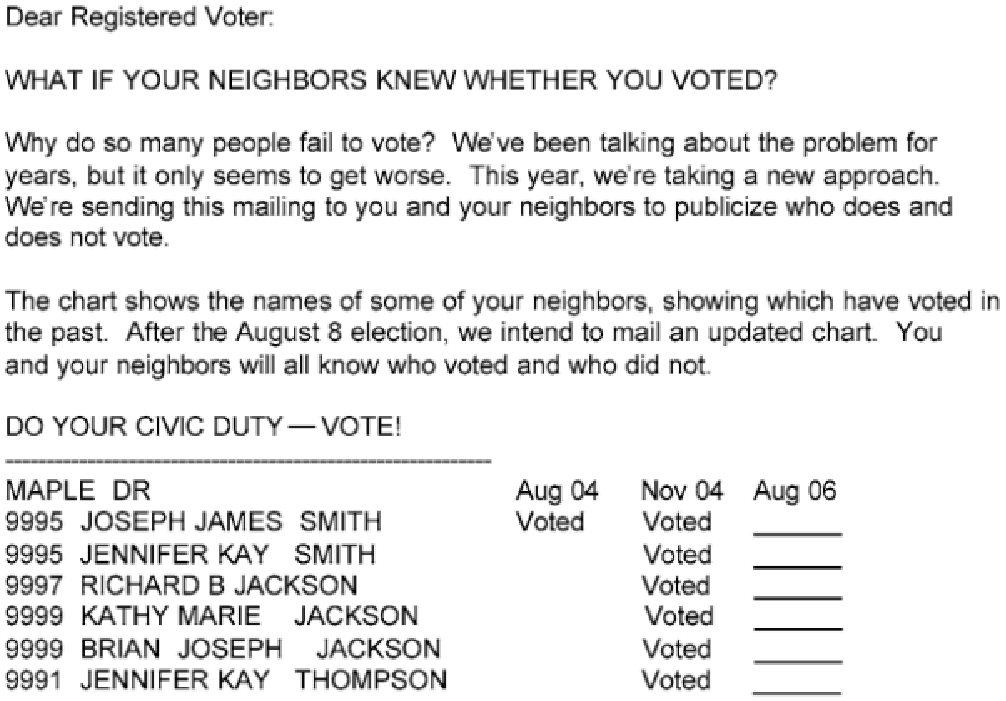
Dear Registered Voter:

DO YOUR CIVIC DUTY AND VOTE!

Why do so many people fail to vote? We've been talking about this problem for years, but it only seems to get worse.

The whole point of democracy is that citizens are active participants in government; that we have a voice in government. Your voice starts with your vote. On August 8, remember your rights and responsibilities as a citizen. Remember to vote.

DO YOUR CIVIC DUTY — VOTE!



Dear Registered Voter:

WHAT IF YOUR NEIGHBORS KNEW WHETHER YOU VOTED?

Why do so many people fail to vote? We've been talking about the problem for years, but it only seems to get worse. This year, we're taking a new approach. We're sending this mailing to you and your neighbors to publicize who does and does not vote.

The chart shows the names of some of your neighbors, showing which have voted in the past. After the August 8 election, we intend to mail an updated chart. You and your neighbors will all know who voted and who did not.

DO YOUR CIVIC DUTY — VOTE!

MAPLE DR	Aug 04	Nov 04	Aug 06
9995 JOSEPH JAMES SMITH	Voted	Voted	______
9995 JENNIFER KAY SMITH		Voted	______
9997 RICHARD B JACKSON		Voted	______
9999 KATHY MARIE JACKSON		Voted	______
9999 BRIAN JOSEPH JACKSON		Voted	______
9991 JENNIFER KAY THOMPSON		Voted	______

`social`: Outcome, treatment and attributes

- `outcome_voted`: Dummy where 1 indicates voted in the August 2006
- `treat_neighbors`: Dummy where 1 indicates *Neighbors mailing* treatment
- `sex`: male / female
- `yob`: Year of birth
- `g2000`: voted in the 2000 general
- `g2002`: voted in the 2002 general
- `p2000`: voted in the 2000 primary
- `p2002`: voted in the 2002 primary
- `p2004`: voted in the 2004 primary
- `city`: City index
- `hh_size`: Household size
- `totalpopulation_estimate`: City population
- `percent_male`: % males in household
- `median_age`: Median age in household
- `median_income`: Median income in household
- `percent_62yearsandover`: % of subjects of age higher than 62 yo
- `percent_white`: % white in household
- `percent_black`: % black in household
- `percent_asian`: % Asian in household
- `percent_hispanicorlatino`: % Hispanic or Latino in household
- `employ_20to64`: % of employed subjects of age 20 to 64 yo
- `highschool`: % having only high school degree
- `bach_orhigher`: % having bachelor degree or higher

Data Preprocessing

First, we define the outcome, treatment and other covariates

```r
Y <- "outcome_voted"

D <- "treat_neighbors"

X <- c("yob", "city", "hh_size",
       "totalpopulation_estimate",
       "percent_male", "median_age",
       "percent_62yearsandover",
       "percent_white", "percent_black",
       "percent_asian", "median_income",
       "employ_20to64", "highschool",
       "bach_orhigher", "percent_hispanicorlatino",
       "sex","g2000", "g2002", "p2000",
       "p2002", "p2004")
```

NOTE: The `social` dataset includes a much more diverse set of features, as well as additional treatments.

Data wrangling

Rename the outcome and treatment variables

```
df <- social %>%
  select(Y, D, X) %>%
  rename(Y = outcome_voted, D = treat_neighbors)
```

For efficiency, we'll use only a subset of the sample:

```
set.seed(1203)

df_smpl <- df %>%
  sample_n(50000)
```

Data Splitting: Training, Estimation, and Test Sets

Before we begin, we need to split our sample into training and estimation sets. The training set will be used to construct the tree, while the estimation set will enable honest estimation of $\tau(x)$:

```
split    <- initial_split(df_smpl, prop = 0.5)

df_train <- training(split)
df_estim <- testing(split)
```

The `causalTree` Package

A description from the `{causalTree}` GitHub repository

> *"The `causalTree` function builds a regression model and returns an `rpart` object, which is the object derived from `rpart` package, implementing many ideas in the CART (Classification and Regression Trees), written by Breiman, Friedman, Olshen and Stone. Like `rpart`, `causalTree` builds a binary regression tree model in two stages, but focuses on estimating heterogeneous causal effect."*

To install the package, run the following commands:

```
install.packages("devtools")
devtools::install_github("susanathey/causalTree")
```

Estimating the Causal Tree

Now, we proceed to estimate the tree using the CT-H (Honest Causal Tree) approach:

```
tree <- honest.causalTree(
  formula = "I(Y) ~ . - D",

  data      = df_train,
  treatment = df_train$D,

  est_data      = df_estim,
  est_treatment = df_estim$D,

  split.Rule   = "CT",
  split.Honest = TRUE,

  cv.option = "CT",
  cv.Honest = TRUE,

  minsize = 100,
  HonestSampleSize = nrow(df_estim),
  cp=0
)
```

Tree Pruning Based on (Honest) Cross-Validation

First, extract a table of cross-validated values by tuning parameter:

```
cptable <- as.data.frame(tree$cptable)
```

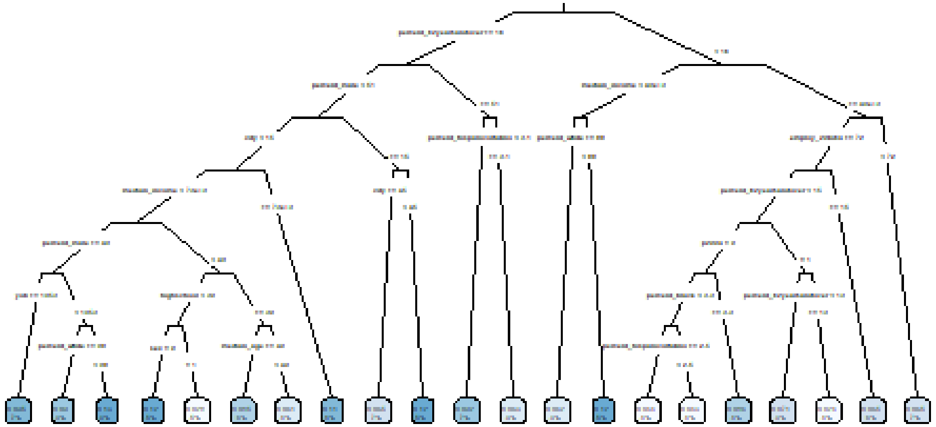
Then, obtain the optimal cp to prune the tree:

```
min_cp       <- which.min(cptable$xerror)
optim_cp_ct <- cptable[min_cp, "CP"]
```

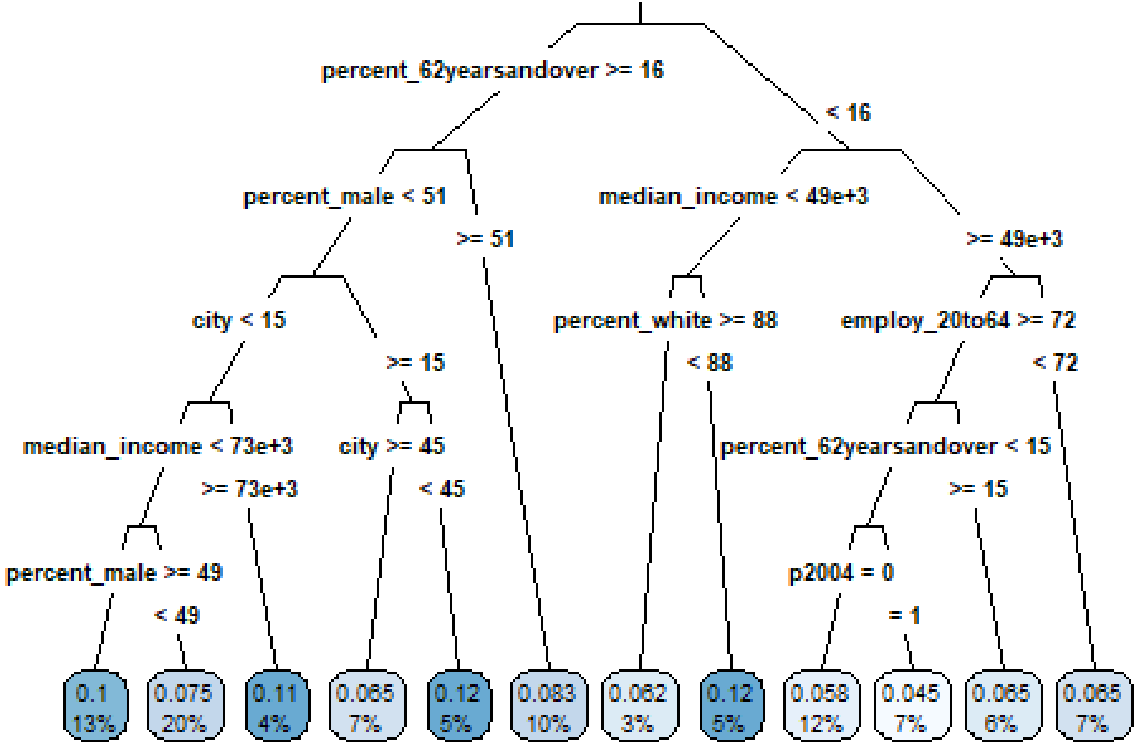
Finally, prune the tree at the optimal cp:

```
pruned_tree <- prune(tree = tree, cp = optim_cp_ct)
```

The Estimated Tree



Pruned Tree


percent_62yearsandover >= 16
< 16
percent_male < 51
>= 51
median_income < 49e+3
>= 49e+3
city < 15
>= 15
percent_white >= 88
< 88
employ_20to64 >= 72
< 72
median_income < 73e+3
>= 73e+3
city >= 45
< 45
percent_62yearsandover < 15
>= 15
percent_male >= 49
< 49
p2004 = 0
= 1
0.1
13%
0.075
20%
0.11
4%
0.065
7%
0.12
5%
0.083
10%
0.062
3%
0.12
5%
0.058
12%
0.045
7%
0.065
6%
0.065
7%

Assigning Each Observation to a Specific Leaf

First, form a tibble that holds both the training and estimation samples:

```r
df_all <- tibble(
  sample = c("training", "estimation"),
  data   = list(df_train, df_estim)
)
```

Then, assign each observation in the training and estimation sets to a leaf based on `tree`:

```r
df_all_leaf <- df_all %>%
  mutate(leaf = map(data, ~ predict(pruned_tree,
                        newdata = .x,
                        type = "vector"))) %>%
  mutate(leaf = map(leaf, ~ round(.x, 3))) %>%
  mutate(leaf = map(leaf, ~ as.factor(.x))) %>%
  mutate(leaf = map(leaf, ~ enframe(.x, name = NULL, value = "leaf"))) %>%
  mutate(data = map2(data, leaf, ~ bind_cols(.x, .y)))
```

Estimating CATE Using the Causal Tree

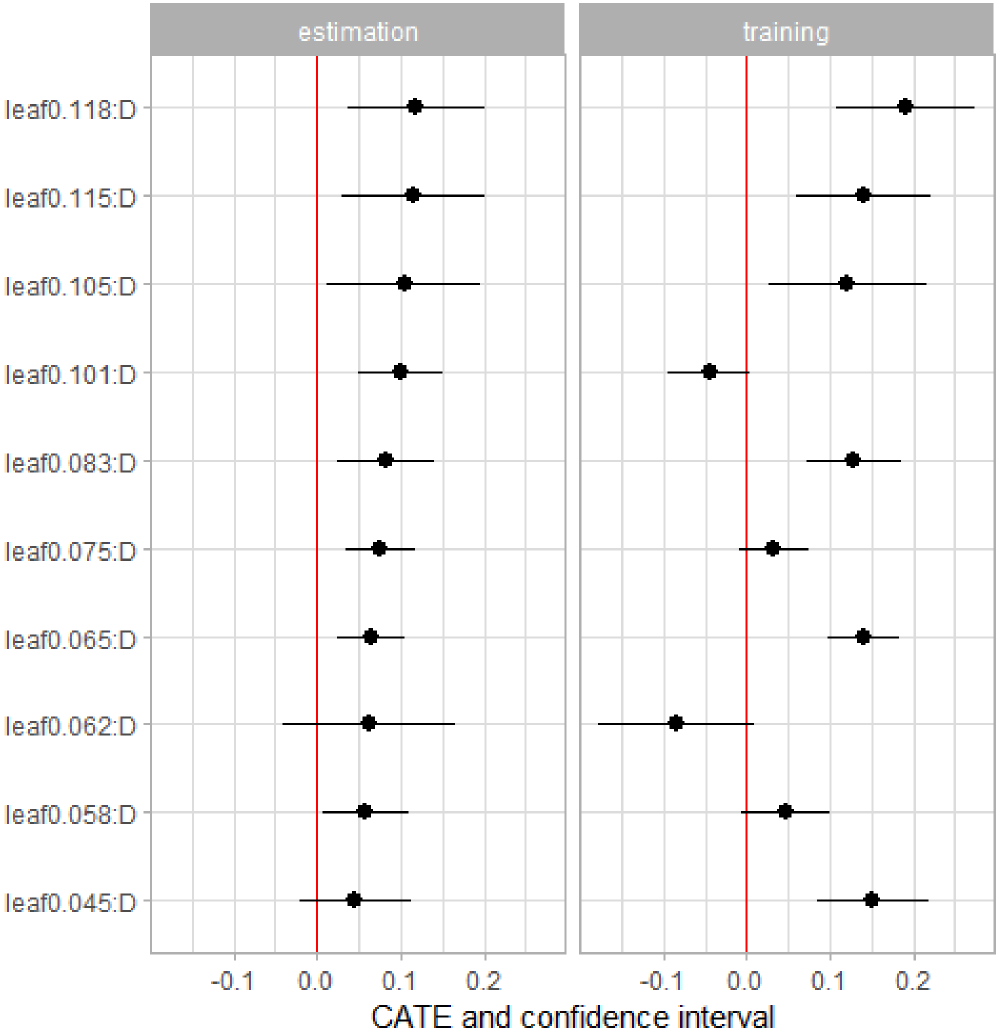
Employ the `lm()` function with interaction terms, for instance:

`lm(Y ~ leaf + D * leaf - D - 1)`

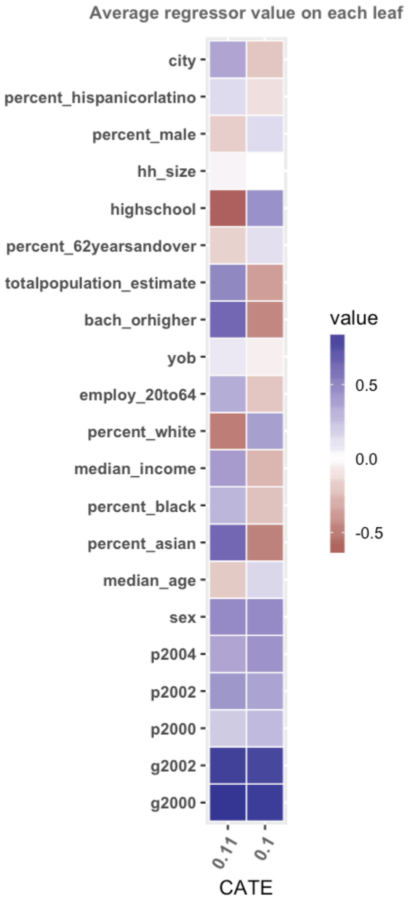
This allows estimation of the average treatment effect within each leaf and provides confidence intervals.

```r
df_all_lm  <-
  df_all_leaf %>%
  mutate(model = map(data, ~ lm(Y ~ leaf + D * leaf
                                - D - 1, data = .x))) %>%
  mutate(tidy = map(model, broom::tidy, conf.int = TRUE)) %>%
  unnest(tidy)
```

Visualizing Coefficients and Confidence Intervals



Interpreting Causal Trees



Source: https://drive.google.com/open?id=1FuF4_q4HCzbU_ImFoLW4r4Gop6A0YsO_

```
slides |> end()
```

Source code

Selected References

- Athey, S., & Imbens, G. (2016). Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27), 7353-7360.

- Athey, S., Imbens, G. W., Kong, Y., & Ramachandra, V. (2016). An Introduction to Recursive Partitioning for Heterogeneous Causal Effects Estimation Using `causalTree` package. 1–15.

- Davis, J.M. V & Heller, S.B., 2017. Using Causal Forests to Predict Treatment Heterogeneity: An Application to Summer Jobs. *American Economic Review: Papers & Proceedings*, 107(5), pp.546–550.

- Lundberg, I., 2017. Causal forests: A tutorial in high-dimensional causal inference. Available here.

- Wager, S., & Athey, S. (2018). Estimation and Inference of Heterogeneous Treatment Effects using Random Forests. *Journal of the American Statistical Association*, 113(523), 1228–1242.