Characterizing Effect Heterogeneity 2

Cyrus Samii

2019-06-07

Contents

1 Estimating Conditional Treatment Effects (CATES) 2
- 1.1 Overview and Goals . . . 2
- 1.2 Data preparation . . . 2

2 Trees 4
- 2.1 Trees for potential outcomes . . . 4
- 2.2 Trees for effect heterogeneity . . . 10

3 Random Forests 10

4 Elastic Net 10

5 Analyzing Features of CATES (Chernozhukov et al, 2017) 10
- 5.1 Goals . . . 10
- 5.2 Setting . . . 11
- 5.3 Methods . . . 11
- 5.4 Choosing an ML method . . . 12
- 5.5 Applications . . . 12

1 Estimating Conditional Treatment Effects (CATES)

1.1 Overview and Goals

We consider three types of high-dimensional methods for estimating heterogenous treatment effects: trees, random forests, and "elastic net" regularized regression. The focus initially will be on point predictions, rather than inference on such predictions. This is because the applications that we use work with summaries of the point predictions rather than the point predictions in and of themselves. See the section below on "features of CATES", referencing Chernozhukov et al. (2017, arxiv), for more on this point.

We are also interested in methods that work well when we entertain a high-dimensional covariate vector. I say "entertain" because it may be that, in fact, the covariates that predict heterogeneity are few, but this is something that we do not know *a priori*, and rather we have at our disposal many covariates that we want to consider as candidates for predicting effect heterogeneity. This leads us to machine learning approaches that use regularization to balance that ability to make very fine grained predictions with penalties for overfitting.

We consider the following algorithms:

- Trees
- Random Forests
- Elastic Net

1.2 Data preparation

Bringing in Data (note that we need to harmonize data wrt to section 1):

```
mex_data <- as.data.frame(import("progresa_mat.dta"))
covs <- setdiff(names(mex_data),
                c('indiv_id',
                  'treatment',
                  'enrolled',
                  'treated_adj',
                  'y_adj',
                  'ml_single_parent',
                  'group'))
```

Checking the data:

```
misCheck <- as.matrix(apply(mex_data, 2, function(x){sum(is.na(x))}))
colnames(misCheck) <- "Number missing"
kable(misCheck)
```

	Number missing
indiv_id	0
enrolled	0
ml_age	0
ml_n_child	0
ml_male	0
ml_hh_head_male	0
ml_single_parent	0
ml_al_father	0
ml_al_mother	0
ml_lw_father	0
ml_lw_mother	0
ml_literacy	0
ml_toilet	0

	Number missing
ml_water_piped	0
ml_lights	0
ml_tv	0
ml_car	0
ml_refrige	0
ml_n_total	0
ml_yrs_educ	0
ml_hh_head_edu	0
ml_father_educ	0
ml_mother_educ	0
ml_male_mi	0
ml_al_father_mi	0
ml_al_mother_mi	0
ml_literacy_mi	0
ml_toilet_mi	0
ml_water_piped_mi	0
ml_lights_mi	0
ml_tv_mi	0
ml_car_mi	0
ml_refrige_mi	0
ml_yrs_educ_mi	0
ml_father_educ_mi	0
ml_mother_educ_mi	0
treatment	0

2 Trees

2.1 Trees for potential outcomes

We start with using a regression tree to model potential outcomes. This helps us to understand how the trees work and the settings that we need to fix. We just use all available covariates. The `rpart()` function loads when we load the `causalTree` package. We will use 10-fold cross validation (CV) to estimated CV-error that we will minimize in selecting the complexity parameter to prune the tree. We evaluate covariate importance using the built-in evaluation function. For a classification problem such that this, a given covariate's importance is based on the sum of standardized goodness of classification values for all splits that involves it.

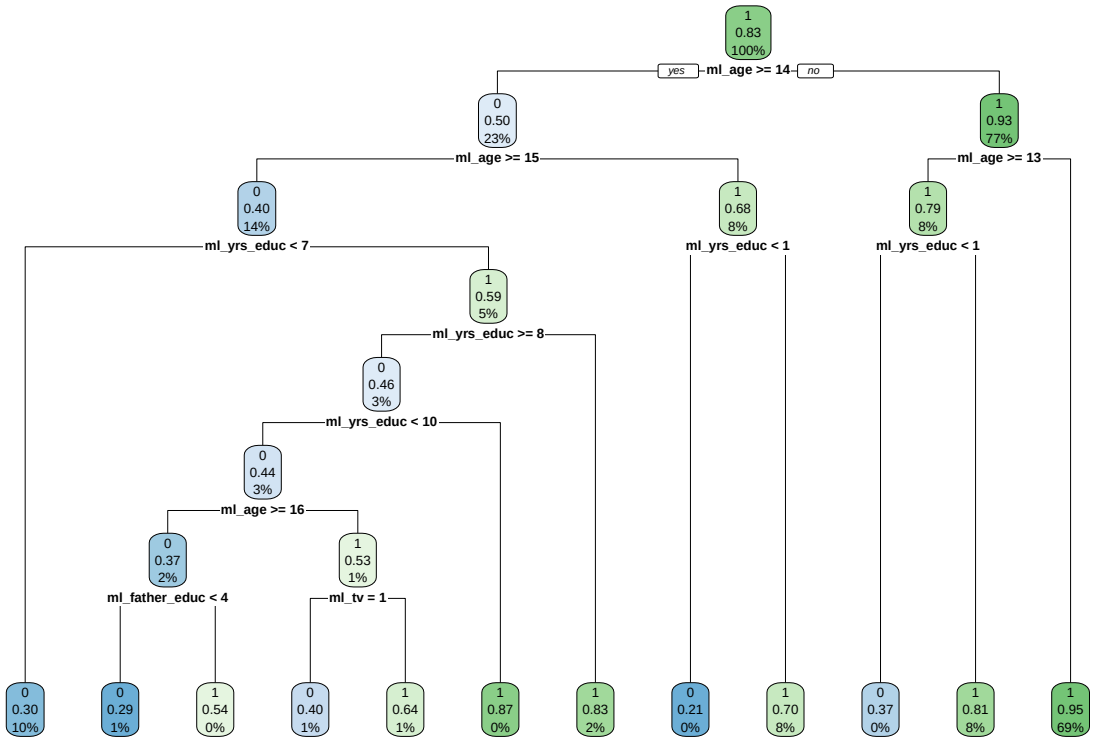
Here is the function:

```
print(rpartOp)
```

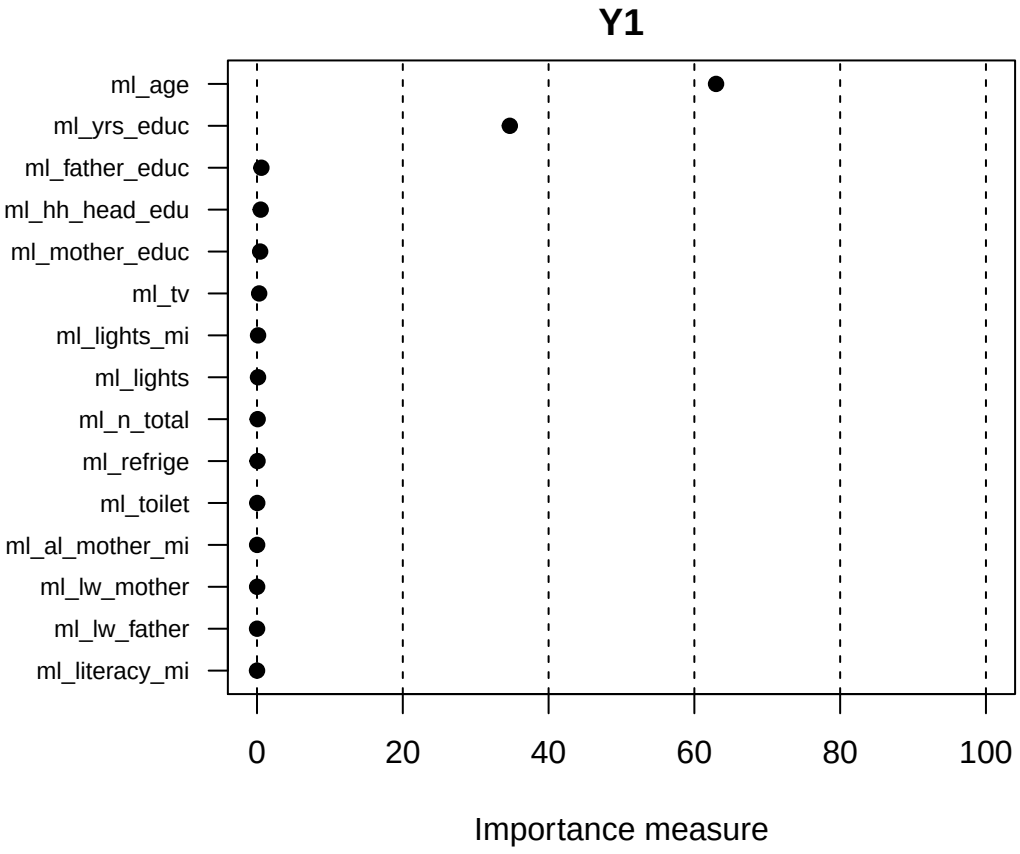
```
## function (formulaIn = NULL, methodIn = NULL, dataIn = NULL, xvalIn = NULL,
##     minbucketIn = NULL, cp_incIN = NULL)
## {
##     treeOut <- rpart(formulaIn, method = methodIn, data = dataIn,
##         control = rpart.control(xval = xvalIn, minbucket = minbucketIn,
##             cp = cp_incIN), model = TRUE)
##     opcp <- treeOut$cptable[, 1][which.min(treeOut$cptable[,
##         4])]
##     optreeOut <- prune(treeOut, opcp)
##     resList <- list(tree = optreeOut, varimp = 100 * (optreeOut$variable.importance/sum(optreeOut$va
##     return(resList)
## }
```

And now results:

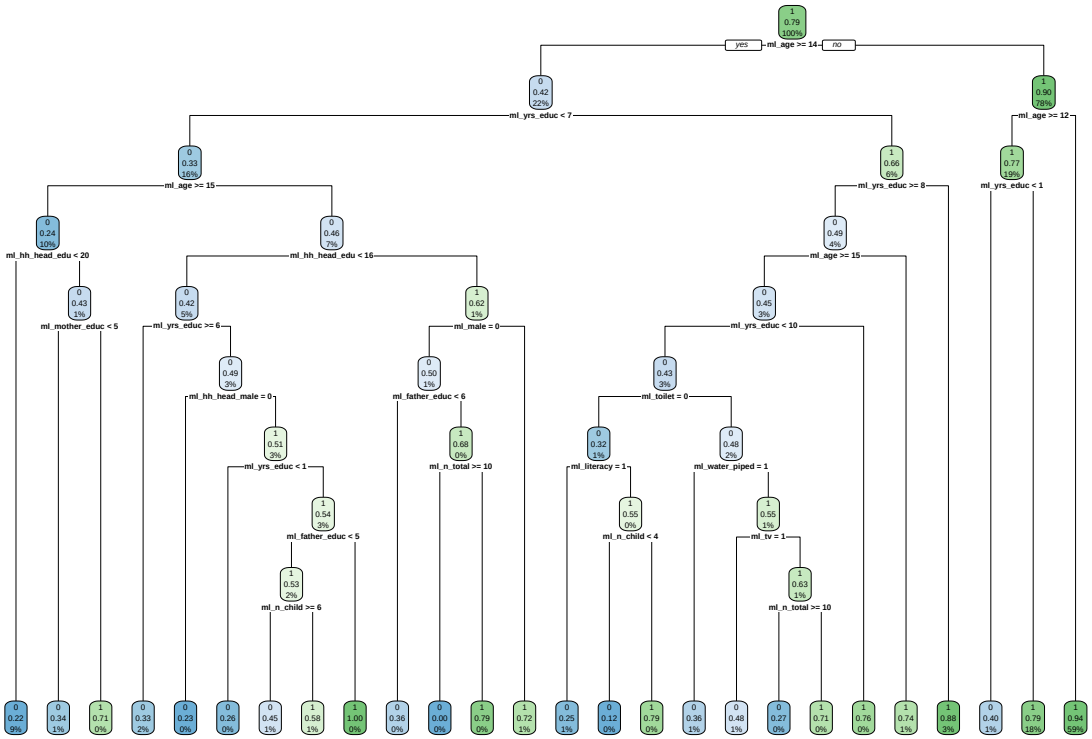
```
mexFormula <- as.formula(paste("enrolled ~", paste(covs, collapse="+")))
mex_data1 <- subset(mex_data, treatment==1)
mex_data0 <- subset(mex_data, treatment==0)
tree_y1 <- rpartOp(formulaIn=mexFormula,
                   methodIn="class",
                   dataIn=mex_data1,
                   xvalIn=10,
                   minbucketIn=2,
                   cp_incIN=0)
rpart.plot(tree_y1$tree)
```



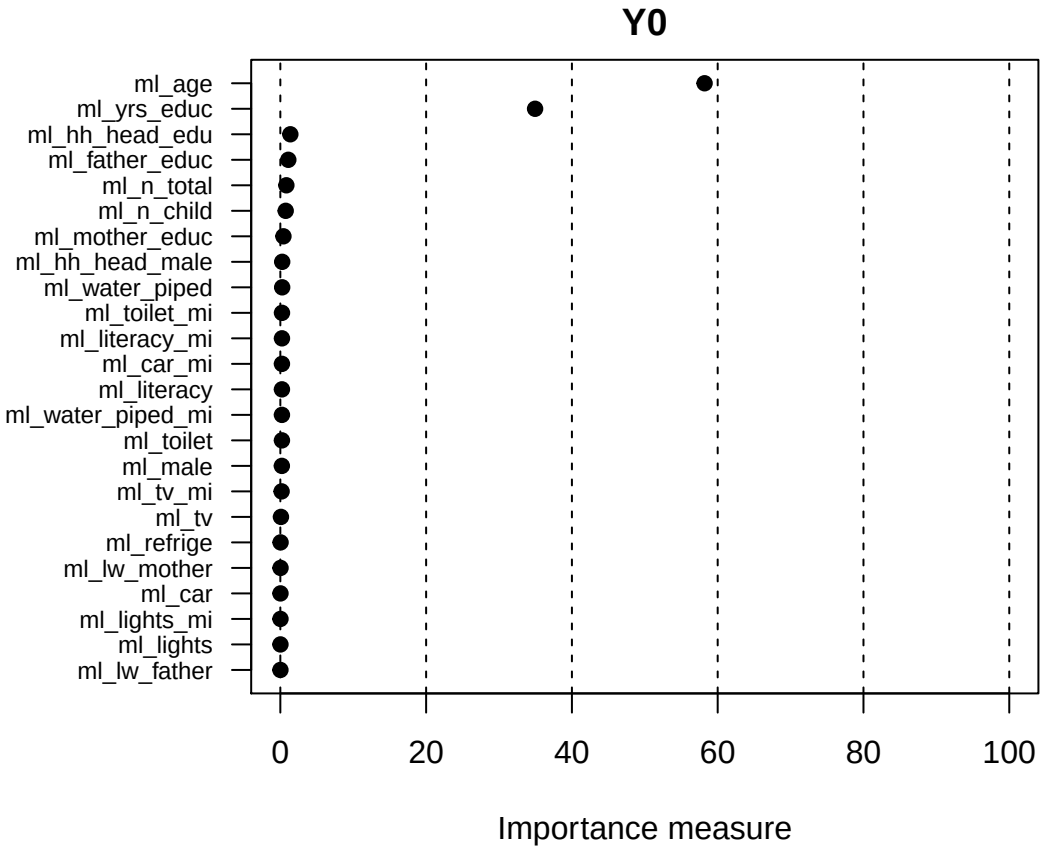

```
par(mar=c(4,10,2,2))
rpart_impplot(treeIn=tree_y1, mainIn="Y1", labelScale=0.75)
```



```
tree_y0 <- rpartOp(formulaIn=mexFormula,
                   methodIn="class",
                   dataIn=mex_data0,
                   xvalIn=10,
                   minbucketIn=2,
                   cp_incIN=0)
rpart.plot(tree_y0$tree)
```




```
par(mar=c(4,10,2,2))
rpart_impplot(treeIn=tree_y0, mainIn="Y0", labelScale=0.75)
```



Looking at the results for the treated and control outcomes, we see that in both cases, the optimal trees gain predictive traction almost entirely by splitting on age and years of education. The other variables contribute to predictive accuracy only to a neglible degree.

What are the implications of a covariate's importance for predicting potential outcomes when it comes to predicting effect heterogeneity? We have

$$E[Y(1) - Y(0)|X] = E[Y(1)|X] - E[Y(0)|X]$$

by the linearity of expectations. That being the case, if $E[Y(1)|X] = E[Y(1)]$ and $E[Y(0)|X] = E[Y(0)]$, then it must be that $E[Y(1) - Y(0)|X] = E[Y(1) - Y(0)]$. As such, being a variable that is predictive of either of the potential outcomes is a *necessary* condition for a covariate for it to be predictive of effects. That said, variables that are predictive of potential outcomes may not be predictive of effects. E.g., we could have that

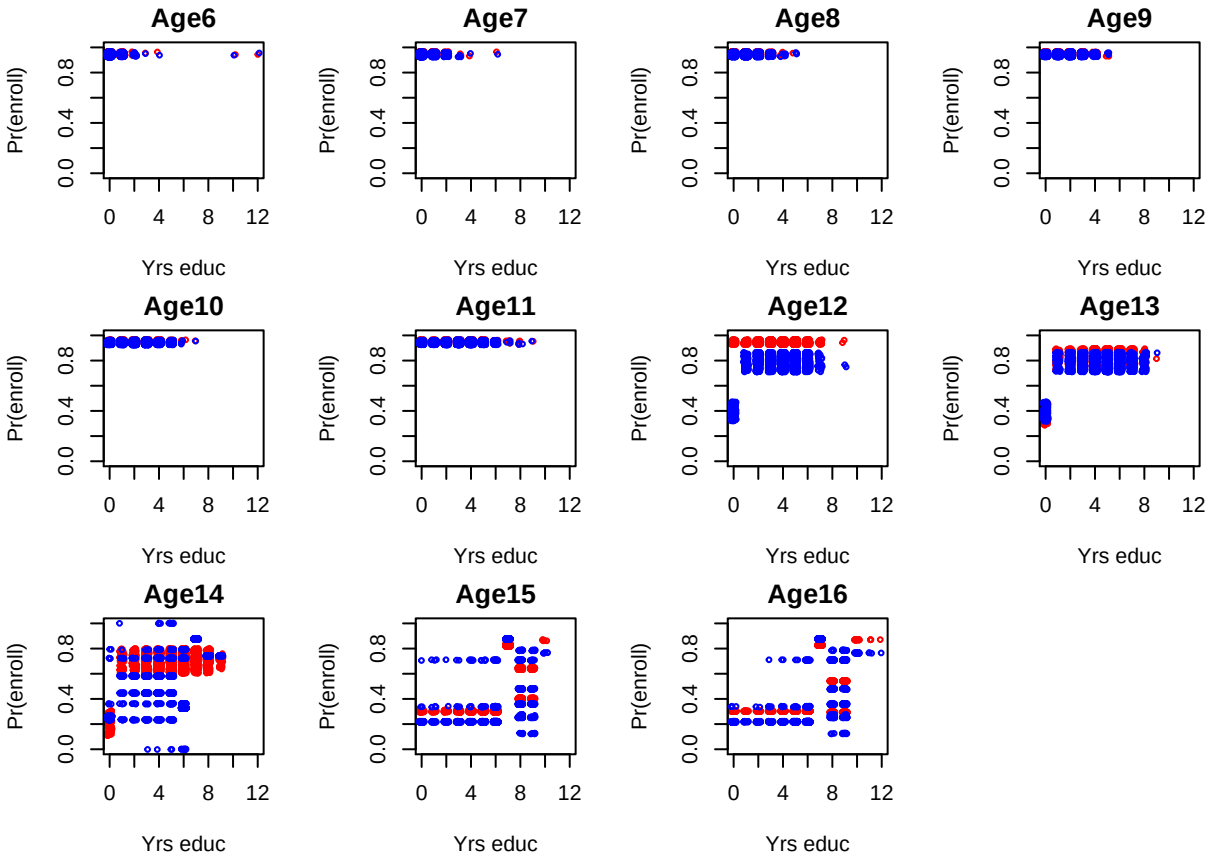
$$E[Y(t)|X] = \alpha + \beta t + \gamma X$$

for $t = 0, 1$, in which case

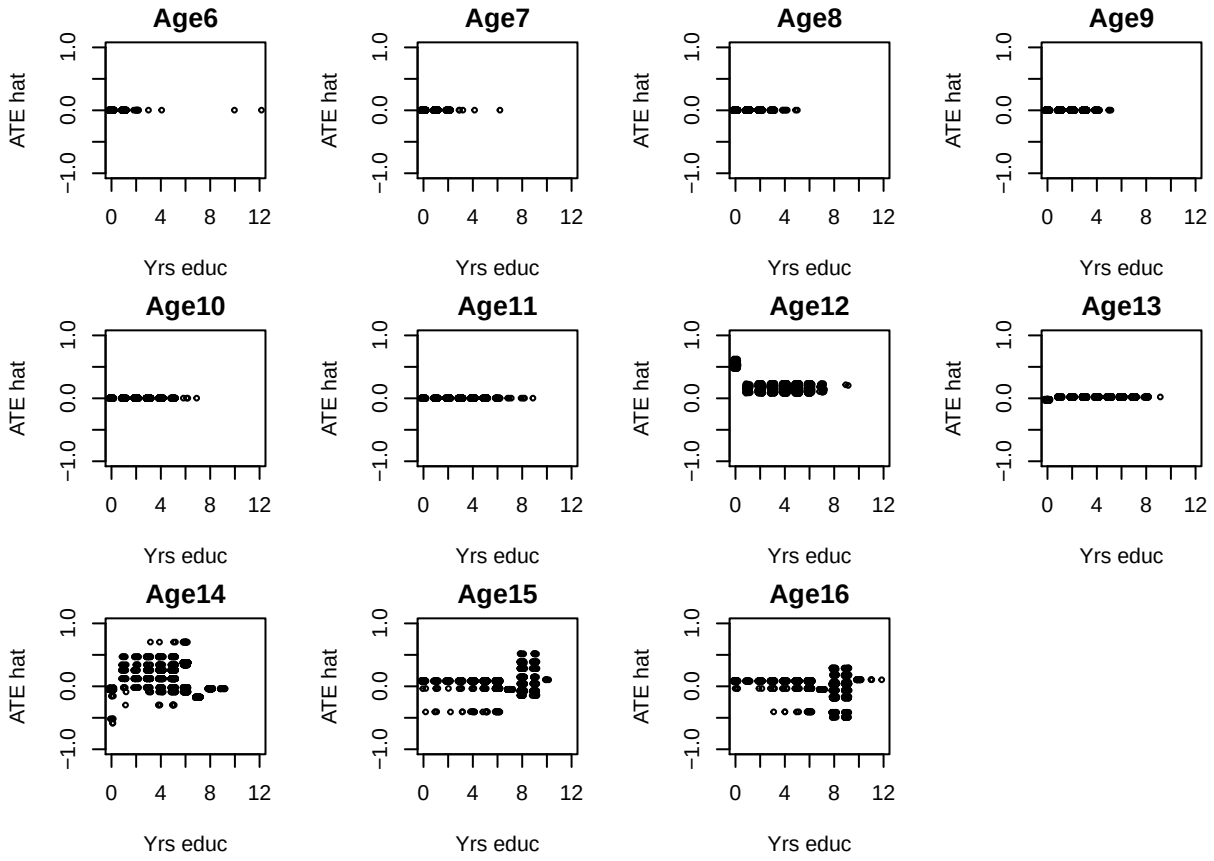
$$E[Y(1) - Y(0)|X] = (\alpha + \beta + \gamma X) - (\alpha + \gamma X) = \beta,$$

which does not depend on X. That said, a starting point for selecting variables to consider as contributors to effect heterogeneity would those that have some predictive relationship with potential outcomes. (Note that this is similar to what is done in Imai and Strauss 2011).

Let us view the predicted potential outcome trees, plotting over the two covariates that seem the matter:



From these plots we can see that for lower age groups, the predicted means are very close, and so effects are basically zero. Then, we have variation for higher age groups. We can see all of this more clearly by plotting the estimated conditional treatment effects themselves:



While this approach is useful for getting a handle on effect heterogeneity, it is not optimized with respect to treatment effects per se. We will turn to such methods next.

2.2 Trees for effect heterogeneity

We now turn to trees for effect heterogeneity. There are different ways to go about this.

We now turn to the "causal tree," as per Athey and Imbens. We use their recommended causal tree (CT) splitting rule. We do not use their honest splitting or CV rules because we are not interested in in-sample inference.

First is a test run, where we will work with a smaller version of the dataset:

```
covShort <- covs[1:10]
mexShort <- mex_data[1:5000,c("enrolled", "treatment", covShort)]
#mexFormula <- as.formula(paste("enrolled ~", paste(covs, collapse="+")))
mexFormula <- as.formula(paste("enrolled ~", paste(covShort, collapse="+")))
treeOut <- causalTree(mexFormula,
                      data=mexShort,
                      treatment=mexShort$treatment,
                      split.Rule="CT",
                      split.Honest=F,
                      cv.option="CT",
                      cv.Honest=F,
                      cp=0.0001,
                      split.alpha=1)
```

Checking the cross-validated prediction error (`xerror`) for the different depth trees:

```
treeOut$cptable
```

We can view cross-validated error minimizing tree:

```
opcp <- treeOut$cptable[,1][which.min(treeOut$cptable[,4])]
opTree <- prune(treeOut, opcp)
rpart.plot(opTree)
```

3 Random Forests

Link to the Athey & Wager paper: JASA

They start with the same HT scaling result, expressing it in a different (but equivalent) manner (keeping with the notation from above):

$$E\left[Y\left(\frac{D}{p(Z)}-\frac{1-D}{1-p(Z)}\right)\middle|Z\right]=s_0(Z).$$

Now I am going to move over to Aaron's code.

4 Elastic Net

5 Analyzing Features of CATES (Chernozhukov et al, 2017)

Link to the paper: arxiv

5.1 Goals

The goal is to learn about effect heterogeneity in a way that does not overfit and to provide "uniformly valid inference" on conditional average treatment effects (CATEs), or at least *features* of such CATEs. The

approach is two-step: (i) build a machine learning (ML) proxy predictor of the CATE, then (ii) do inference on features of this proxy predictor. The features considered are first, best linear predictor (BLP), second, sorted group average effects (GATES), which are ATEs by heterogeneity groups, and third classification analysis (CLAN), which are the average effects of the most and least affected units.

5.2 Setting

Some notation:

- Potential outcomes, $Y(1), Y(0)$.
- Covariates Z.
- Baseline conditional average (BCA): $b_0(Z) = E[Y(0)|Z]$.
- CATE: $s_0(Z) = E[Y(1)|Z] - E[Y(0)|Z]$.
- CIA holds: $D \perp\!\!\!\perp (Y(1), Y(0))|Z$.
- Subvector of stratifying covariates, $Z_1 \subseteq Z$, such that
- propensity score is $p(Z) = P[D = 1|Z_1]$, with $0 < p_0 \leq p(Z) \leq p_1 < 1$.
- Observe $Y = DY(1) + (1 - D)Y(0)$, in which case,
- $Y = b_0(Z) + Ds_0(Z) + U$, with $E[U|Z, D] = 0$, where
- $b_0(Z) = E[Y|D = 0, Z]$ and $s_0(Z) = E[Y|D = 1, Z] - E[Y|D = 0, Z]$. CIA allows us to write b_0 and s_0 in terms of observables.
- Observe N iid draws of (Y, Z, D) with law P. Draws are indexed by $i = 1, ..., N$.

A result to keep in mind (already noted in Athey and Imbens PNAS, I believe):
Horvitz-Thompson scaling Define

$$H = H(D, Z) = \frac{D - p(Z)}{p(Z)(1 - p(Z))}.$$

Given the assumptions above, we have,

$$\begin{aligned} E[YH|Z] &= E\left[\frac{Y(D - p(Z))}{p(Z)(1 - p(Z))}\middle| Z\right] \\ &= \frac{1}{p(Z)(1 - p(Z))} E\left[D(D - p(Z))Y(1) + (1 - D)(D - p(Z))Y(0)|Z\right] \\ &= E[Y(1)|Z] - E[Y(0)|Z] \\ &= s_0(Z). \end{aligned}$$

We can use YH as an "unbiased signal" about $s_0(Z)$. It is, however, a noisy signal, and so in using this, Chernozhukov et al. make adjustments (e.g., for their second BLP method). Nonetheless, it does provide a target that one can use in trying to tune methods for estimating CATEs (see below the section on choosing an ML method).

5.3 Methods

For Chernozhukov et al., directly learning about s_0 in a generic way seems impossible at the moment When P is high dimensional, ML methods will not be consistent. Moreover, inference is complicated for adaptive estimators, particularly in trying to bound biases. Finally, tuning ML models is sort of free for all at the moment, with little to guide.

For these reasons Chernozhukov et al. focus on inference for low dimensional features of s_0. First, partition the indices $\{1, ..., N\}$ into two sets, M and A for "main" and "auxiliary", respectively, to yield a main sample, Data_M, and auxiliary sample, Data_A. Suppose for the time being that the two sets are about the same size. From sample A obtain ML estimates of b_0 and s_0, to be called "proxy predictors":

$$z \mapsto B(z) = B(z; \text{Data}_A) \text{ and } z \mapsto S(z) = S(z; \text{Data}_A).$$

These proxies need not be consistent. Then construct the features of interest (BLP, GATES, CLAN) in sample M. Then do inference in a way that accounts for estimation uncertainty given sample A and splitting uncertainty given the split into M and A. Their precise method is to use many splits and then take the median of the feature estimates and then the medians of the random conditional confidence sets, adjusting them to account for splitting uncertainty. They refer to this inferential approach as "variational estimation and inference" (VEIN).

BLP method 1 involves the following:

Consider the weighted linear projection:

$$Y = \alpha_1 + \alpha_2 B(Z) + \beta_1(D - p(Z)) + \beta_2(D - p(Z))(S(Z) - E[S(Z)]) + \epsilon,$$

with weights,

$$w(Z) = \frac{1}{p(Z)(1 - p(Z))}.$$

Chernozhulov et al. explain that "the interaction $(D - p(Z))(S - E[S])$ is orthogonal to $D - p(Z)$ under the weight $w(Z)$". This is due to LIE – cf. p. 34 of the paper.

I am going to leave the BLP alone for now. I can see how it may provide for a test of heterogeneity, but not sure how useful it is for our purposes.

Sorted group ATE is quite relevant to us. We want to put units into groups based on their predicted treatment effects. E.g., suppose we want a partition,

$$G - k = \{S \in I_k\}, k = 1, ..., K,$$

where $I_k = [\ell_{k-1}, \ell_k)$ are intervals that divide the support of S. Then, we want the conditional average effects within these bins,

$$E[s_0(Z)|G_k], k = 1, ..., K.$$

given the monotonicity restriction,

$$E[s_0(Z)|G_1] \leq ... \leq E[s_0(Z)|G_K].$$

5.4 Choosing an ML method

Options that they cover: (i) "ability to predict YH using BH and S" or (ii) "ability to predict Y using B and $(D - p(Z))(S - E[S])$ given $w(Z)$.

5.5 Applications

The application that is most relevant for us is the analysis of effect heterogeneity in the Morocco microfinance study. In this case, $p(Z) = 1/2$ for everyone. They try Random Forest, Elastic Net, Boosted Tree, and Neural Network, finding that the first two do best when judged using metrics geared toward the BLP and GATES analyses.