Big Data and Bayesian Nonparametrics

Matt Taddy, Chicago Booth

faculty.chicagobooth.edu/matt.taddy/research

Big Data

The sample sizes are enormous.

- We'll see 21 and 200 million today.
- Data can't fit in memory, or even storage, on a single machine.

The data are super weird.

- Internet transaction data distributions have a big spike at zero and spikes at other discrete values (e.g., 1 or $99).
- Big tails (e.g., $12 mil/month eBay user spend) that matter.
- The potential feature space is unmanageably large.
- We cannot write down or measure believable models.

Both 'Big' and 'Strange' beg for nonparametrics.

Bayesian nonparametrics

In usual BNP you *model* a complex generative process with flexible priors, then apply that model directly in prediction and inference.

$$\text{e.g.,} \qquad y = f(\mathbf{x}) + \varepsilon \qquad \text{or} \qquad f(y|\mathbf{x}) = f(\mathbf{x}, y)/f(\mathbf{x})$$

'Flexibility' comes from a huge number of variables,

$$\text{e.g.,} \qquad f(\mathbf{z}) = \sum_{k=1}^{\infty} \omega_k \mathrm{N}(\mathbf{z};\ \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k),$$

and learning requires integration over uncertainty on nuisance parameters, e.g., k_i component membership for observation $\mathbf{z}_i$.

For years this integration was done via MCMC. But these algorithms did not scale for big data. Can we do scalable BNP?

Classical distribution-free nonparametrics

Frequentists are great at finding simple procedures (e.g. $[\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'y$) and describing how they work under a variety of possible true DGP.

(DGP = Data Generating Process)

This is 'distribution free' nonparametrics.

1: Find some statistic that is useful regardless of DGP.

2: Derive the distribution for this stat under minimal assumptions.

Practitioners apply the simple stat and feel happy that it will work.

Distribution-free Bayesian nonparametrics

Find some *statistic of the DGP* that you care about:

- derive from first principles, e.g. moment conditions
- *an algorithm* that we know works, e.g. CART
- think about geometric projections, e.g. OLS

Call this statistic $\boldsymbol{\theta}(g)$ where $g(\mathbf{z})$ is the DGP (e.g., for $\mathbf{z} = [\mathbf{x}, y]$).

Then you write down a flexible model for the DGP g, and study properties of the posterior on $\boldsymbol{\theta}(g)$ induced by the posterior over g.

A flexible model for the DGP

Say $\mathbf{z} = [\mathbf{x}, y]$ is a single *independent* data point.

Each data point assumes one of a *finite* number of possible values, $[\zeta_1 \dots \zeta_L]$, with probabilities proportional to $[\theta_1 \dots \theta_L]$.

$$g(\mathbf{z}) = \frac{1}{|\boldsymbol{\theta}|} \sum_{l=1}^{L} \theta_l \mathbb{1}_{[\mathbf{z}=\zeta_l]}$$

We complete specification with a conjugate prior on the weights:

$$\frac{\boldsymbol{\theta}}{|\boldsymbol{\theta}|} \sim \mathrm{Dir}(a) \propto \frac{1}{|\boldsymbol{\theta}|^{L(a-1)}} \prod_l \theta_l^{a-1} \quad \text{where} \quad a, \theta_l > 0.$$

This is the Dirichlet-multinomial sampling model (Ferguson 1973).

Now you've observed some data, say $\mathbf{Z} = \{\mathbf{z}_1 \dots \mathbf{z}_n\}$.
(say every $\mathbf{z}_i = [\mathbf{x}_i, y_i]$ is unique).

The posterior over weights has $\theta_l \overset{ind}{\sim} \mathrm{Exp}\left(a + \mathbb{1}_{[\boldsymbol{\zeta}_l \in \mathbf{Z}]}\right)$.

A convenient limiting case

$a \to 0$ leads to $\mathrm{p}(\theta_l = 0) = 1$ for $\boldsymbol{\zeta}_l \notin \mathbf{Z}$.

In this case, we can focus on only the *observed support* and write the posterior for our DGP

$$g(\mathbf{z}) = \tfrac{1}{|\boldsymbol{\theta}|} \sum_{i=1}^{n} \theta_i \mathbb{1}[\mathbf{z} = \mathbf{z}_i], \quad \theta_i \overset{iid}{\sim} \mathrm{Exp}(1).$$

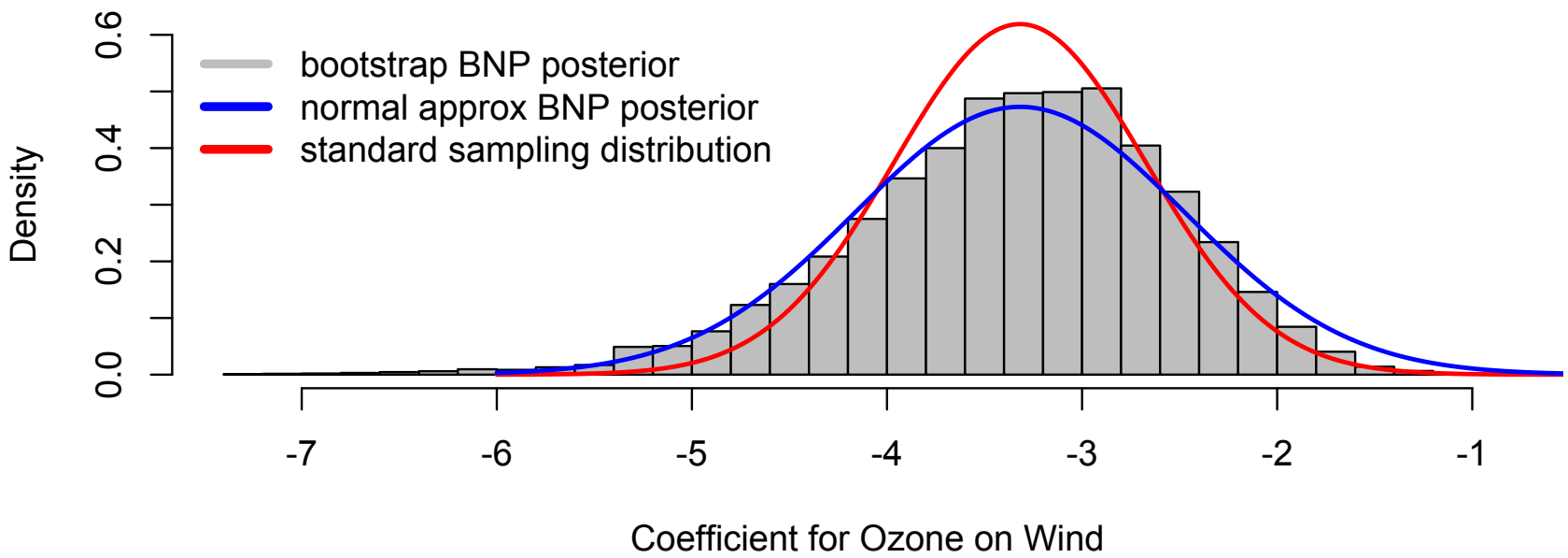
This is just the Bayesian bootstrap. (Rubin 1981)

Example: **Ordinary Least Squares**

Population OLS is a posterior functional

$$\beta = (\mathbf{X}'\mathbf{\Theta}\mathbf{X})^{-1}\mathbf{X}'\mathbf{\Theta}\mathbf{y}$$

where $\mathbf{\Theta} = \mathrm{diag}(\boldsymbol{\theta})$. This is a random variable. (sample via BB)



What is the blue line?

BB sampling is great, but analytic approximations are also useful.

Consider a first-order Taylor series approximation,

$$\tilde{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}} + \nabla \boldsymbol{\beta}\big|_{\boldsymbol{\theta}=\mathbf{1}}(\boldsymbol{\theta} - \mathbf{1})$$

where $\hat{\boldsymbol{\beta}} = [\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'y = \boldsymbol{\beta}\big|_{\boldsymbol{\theta}=\mathbf{1}}$ is the sample OLS line.

We can derive *exact* posterior moments for $\tilde{\boldsymbol{\beta}}$. e.g.,

$$\mathrm{var}(\boldsymbol{\beta}) \approx \mathrm{var}(\tilde{\boldsymbol{\beta}}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{diag}(\mathbf{e}^2)\mathbf{X}'(\mathbf{X}'\mathbf{X})^{-1}$$

where $\mathbf{e} = \mathbf{y} - \mathbf{X}[\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'y$ are the sample OLS residuals.

This is the economist's Huber-White 'Sandwich' variance formula.

(Lancaster 2003 or Poirier 2011.)

Example: **Heterogeneous Treatment Effects**

eBay runs lots of experiments: they make changes to the marketplace (website) for random samples of users.

Every experiment has response y and treatment d [0/1].
In our illustrative example, $d_i = 1$ for bigger pictures in my eBay.

$i \in$ control : $d_i = 0$

$i \in$ treatment : $d_i = 1$

This is a typical 'A/B experiment'.

What is 'heterogeneity in treatment effects'? (HTE)

Different units [people, devices] respond differently to some treatment you apply [change to website, marketing, policy].

I imagine it exists.

We know $\mathbf{x}_i$ about user i. About 400 features in our example.

- Their previous spend, items bought, items sold...
- Page view counts, items watched, searches, ...
- All of the above, broken out by product, fixed v. auction, ...

Can we accurately measure heterogeneity: index it on $\mathbf{x}$?

Treatment effect regression

Potential outcomes: $\upsilon_i(d)$ is the response for user i if $d_i = d$.

We'd love to model $\mathbb{E}[\upsilon_i(1) - \upsilon_i(0)|\mathbf{x}_i]$, but we can't directly.

Since $d \perp\!\!\!\perp [\mathbf{x}, \boldsymbol{\upsilon}]$, a good alternative is to consider

$$\mathbb{E}[\upsilon_i(1)|d_i = 1, \mathbf{x}_i] - \mathbb{E}[\upsilon_i(0)|d_i = 0, \mathbf{x}_i].$$

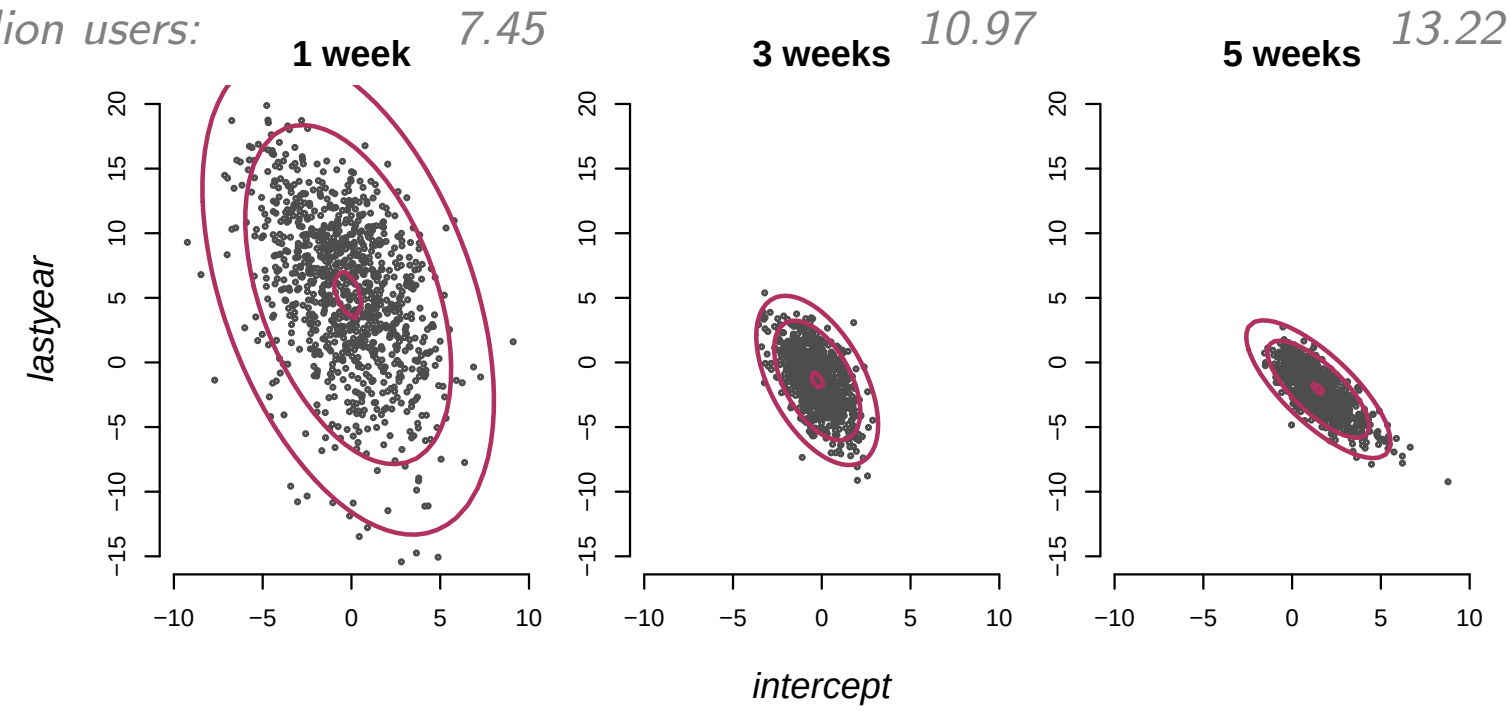
Even simpler, just diff the OLS projections for each group

$$\boldsymbol{\beta}_\mathsf{t} - \boldsymbol{\beta}_\mathsf{c} = (\mathbf{X}_\mathsf{t}'\boldsymbol{\Theta}_\mathsf{t}\mathbf{X}_\mathsf{t})^{-1}\mathbf{X}_\mathsf{t}'\boldsymbol{\Theta}_\mathsf{t}\mathbf{y}_\mathsf{t} - (\mathbf{X}_\mathsf{c}'\boldsymbol{\Theta}_\mathsf{c}\mathbf{X}_\mathsf{c})^{-1}\mathbf{X}_\mathsf{c}'\boldsymbol{\Theta}_\mathsf{c}\mathbf{y}_\mathsf{c}$$

See Taddy, Gardner, Chen, Draper 2014 for alternative linear projections.

Since $\beta_t \perp\!\!\!\perp \beta_c$, our results on BNP analysis of OLS apply directly.
Or you can bootstrap, but it takes a long time.

e.g., coefficient on *[made purchase this year]* vs an intercept:



Sample is from posterior, contours are normal approximation.

$\boldsymbol{\beta}_\mathsf{t} - \boldsymbol{\beta}_\mathsf{c}$ is a statistic we care about, even if the truth is nonlinear.

The most basic application of an A/B experiment is estimating the average treatment effect,

$$\mathsf{ATE} = \mathbb{E}[\upsilon(\mathsf{t})] - \mathbb{E}[\upsilon(\mathsf{c})]$$

The usual estimator is $\bar{y}_\mathsf{t} - \bar{y}_\mathsf{c}$. But many authors advocate instead the 'regression adjusted' $\bar{\mathbf{x}}'(\hat{\boldsymbol{\beta}}_\mathsf{t} - \hat{\boldsymbol{\beta}}_\mathsf{c})$ (Lin, Berk+, Deng+, 2013).

Both are [frequentist] unbiased.

But quantifying uncertainty about the regression adjustment, and any variance reduction it provides, is tough without assumptions.

From our BNP persepective, both the usual and adjusted ATE estimators are random functions of the DGP.

$$\begin{aligned}
\text{ATE}_g^{\text{obs}} &= \frac{1}{|\boldsymbol{\theta}_\text{t}|}\boldsymbol{\theta}_\text{t}'\mathbf{y}_\text{t} - \frac{1}{|\boldsymbol{\theta}_\text{c}|}\boldsymbol{\theta}_\text{c}'\mathbf{y}_\text{c} \\
\text{ATE}_g^{\text{lin}} &= \frac{1}{|\boldsymbol{\theta}|}\boldsymbol{\theta}'\mathbf{X}\left((\mathbf{X}_\text{t}'\boldsymbol{\Theta}_\text{t}\mathbf{X}_\text{t})^{-1}\mathbf{X}_\text{t}'\boldsymbol{\Theta}_\text{t}\mathbf{y}_\text{t} - (\mathbf{X}_\text{c}'\boldsymbol{\Theta}_\text{c}\mathbf{X}_\text{c})^{-1}\mathbf{X}_\text{c}'\boldsymbol{\Theta}_\text{c}\mathbf{y}_\text{c}\right).
\end{aligned}$$

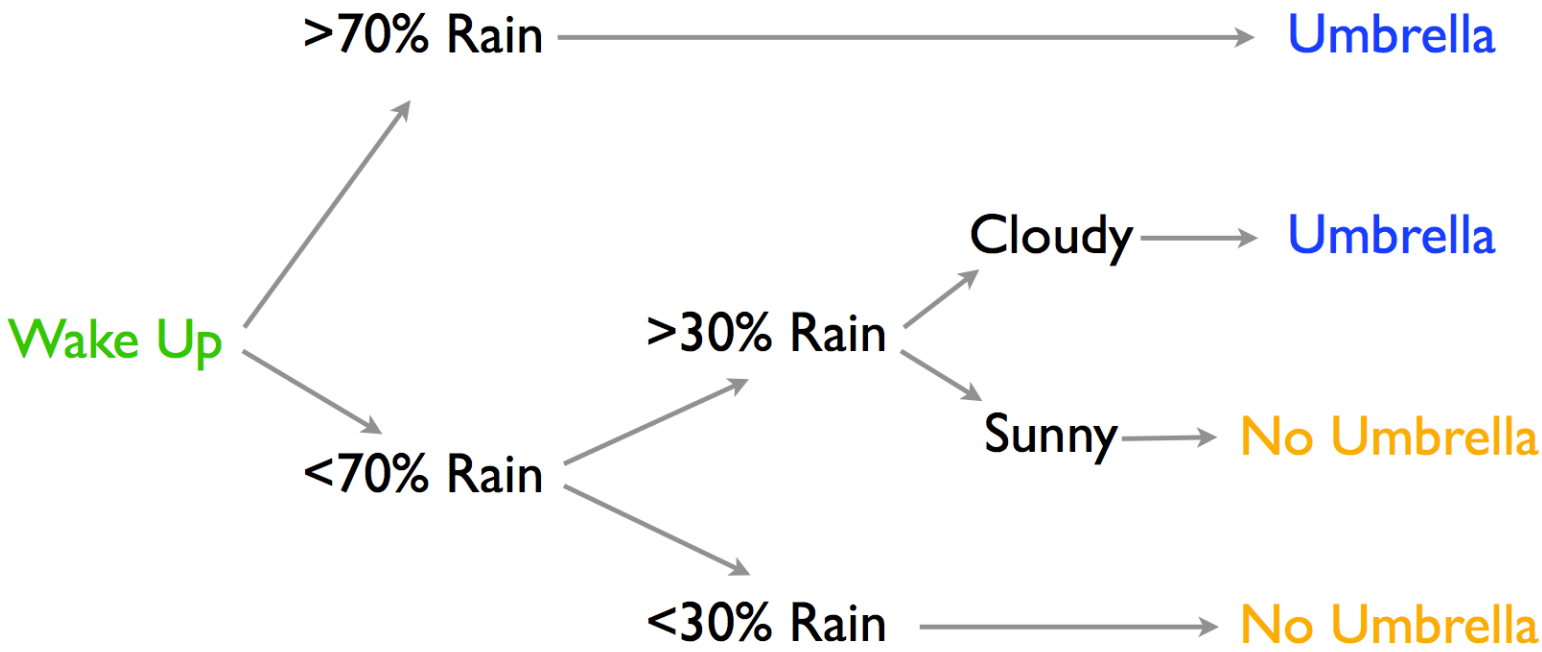
Using our usual tricks, we can resolve any *controversy*:

$$\text{var}\left(\text{ATE}_g^{\text{lin}}\right) \approx \text{var}(\text{ATE}_g^{\text{obs}}) - \left(\frac{R_\text{t}^2 s_{\mathbf{y}_\text{t}}^2}{n_\text{t}^2} + \frac{R_\text{c}^2 s_{\mathbf{y}_\text{c}}^2}{n_\text{c}^2}\right),$$

so that big variance reduction comes only for small sample n_d or large $R_\text{d}^2 = \text{cor}(y_\text{d}, \hat{y}_\text{d})$. In digital experiments we have big samples and low signal, so we shouldn't expect much variance reduction.

Example: **Decision Trees**

Trees are great: nonlinearity, deep interactions, heteroskedasticity.



The 'optimal' decision tree is a statistic we care about (s.w.c.a).

CART: greedy growing with optimal splits

Given node $\{\mathbf{x}_i, y_i\}_{i=1}^n$ and DGP weights $\boldsymbol{\theta}$, find x to minimize

$$|\boldsymbol{\theta}|\sigma^2(x, \boldsymbol{\theta}) = \sum_{k \in \text{left}(x)} \theta_k (y_k - \mu_{\text{left}(x)})^2 + \sum_{k \in \text{right}(x)} \theta_k (y_k - \mu_{\text{right}(x)})^2$$

for a regression tree. Classification impurity can be Gini, etc.

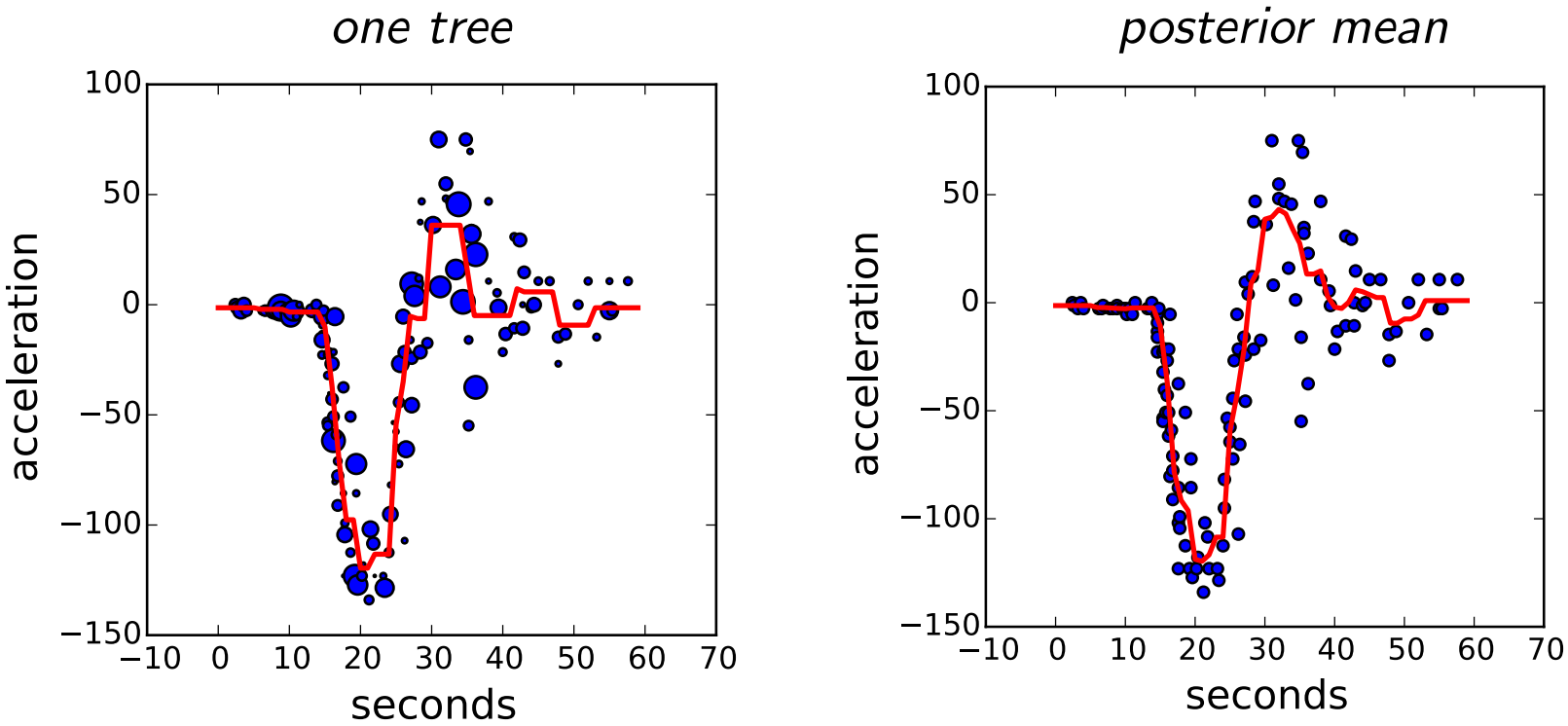
Population-CART might be a statistic we care about.

Or, in settings where greedy CART would do poorly (big p), a randomized splitting algorithm might be a better s.w.c.a.

Bayesian Forests: a posterior for CART trees

For $b = 1 \ldots B$:

- draw $\boldsymbol{\theta}^b \overset{iid}{\sim} \mathrm{Exp}(\mathbf{1})$
- run weighted-sample CART to get $\mathcal{T}_b = \mathcal{T}(\boldsymbol{\theta}^b)$



Random Forest $\approx$ Bayesian forest $\approx$ posterior over CART fits.

Theoretical trunk stability

Given forests as a posterior, we can start talking about *variance*.

Consider the first-order approximation

$$\begin{aligned}\sigma^2(x, \boldsymbol{\theta}) \approx \quad & \sigma^2(x, \mathbf{1}) + \nabla\sigma^2\big|_{\boldsymbol{\theta}=\mathbf{1}}(\boldsymbol{\theta} - \mathbf{1}) \\ & = \tfrac{1}{n}\textstyle\sum_i \theta_i \left[y_i - \bar{y}_i(x)\right]^2\end{aligned}$$

with $\bar{y}_i(x)$ the sample mean in i's node when splitting on x.

Based on this approx, we can say that for data at a given node,

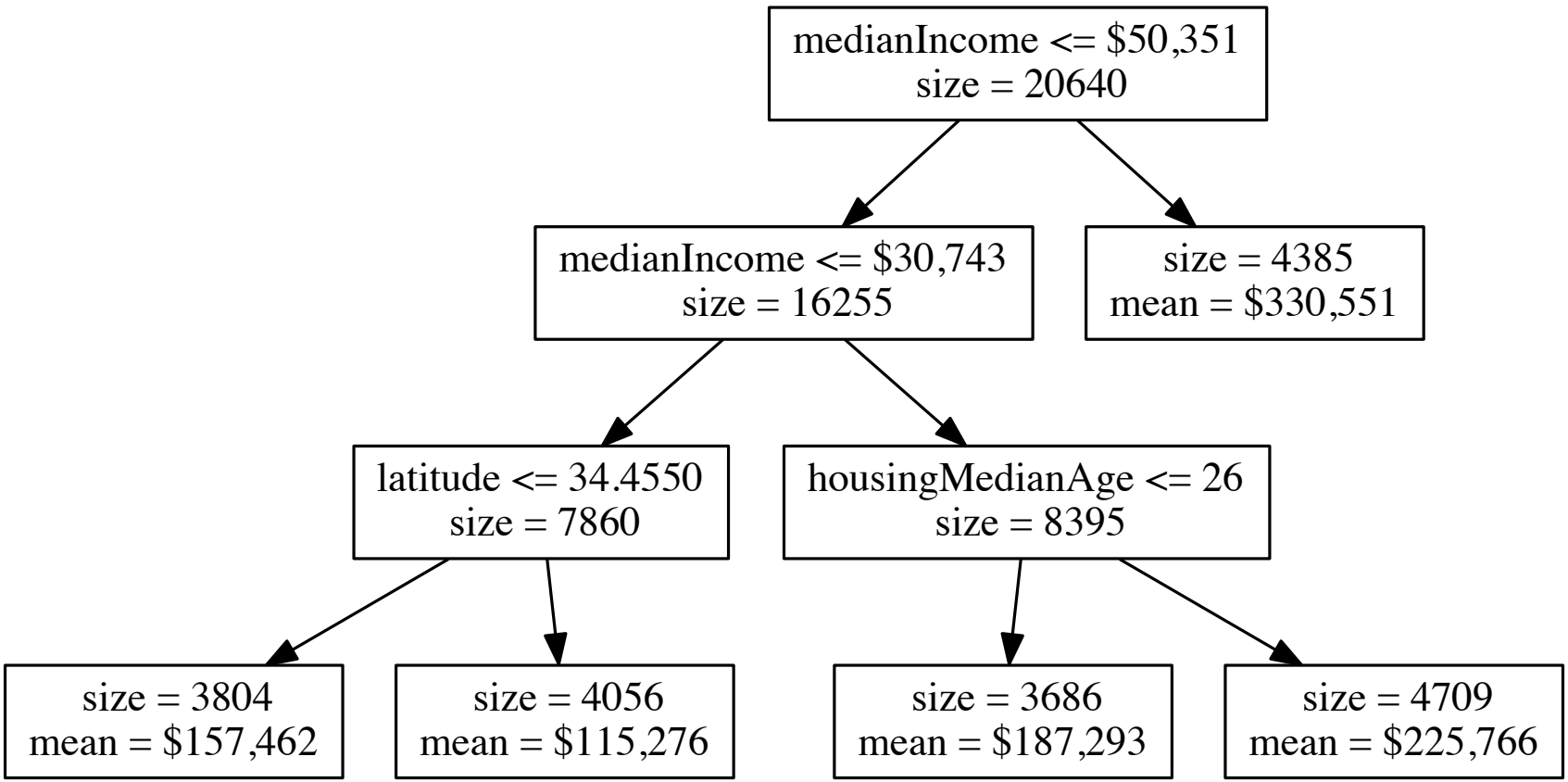
$$\mathrm{p}\left(\text{optimal split matches sample CART}\right) \gtrsim 1 - \frac{p}{\sqrt{n}} e^{-n},$$

with p split locations and n observations.

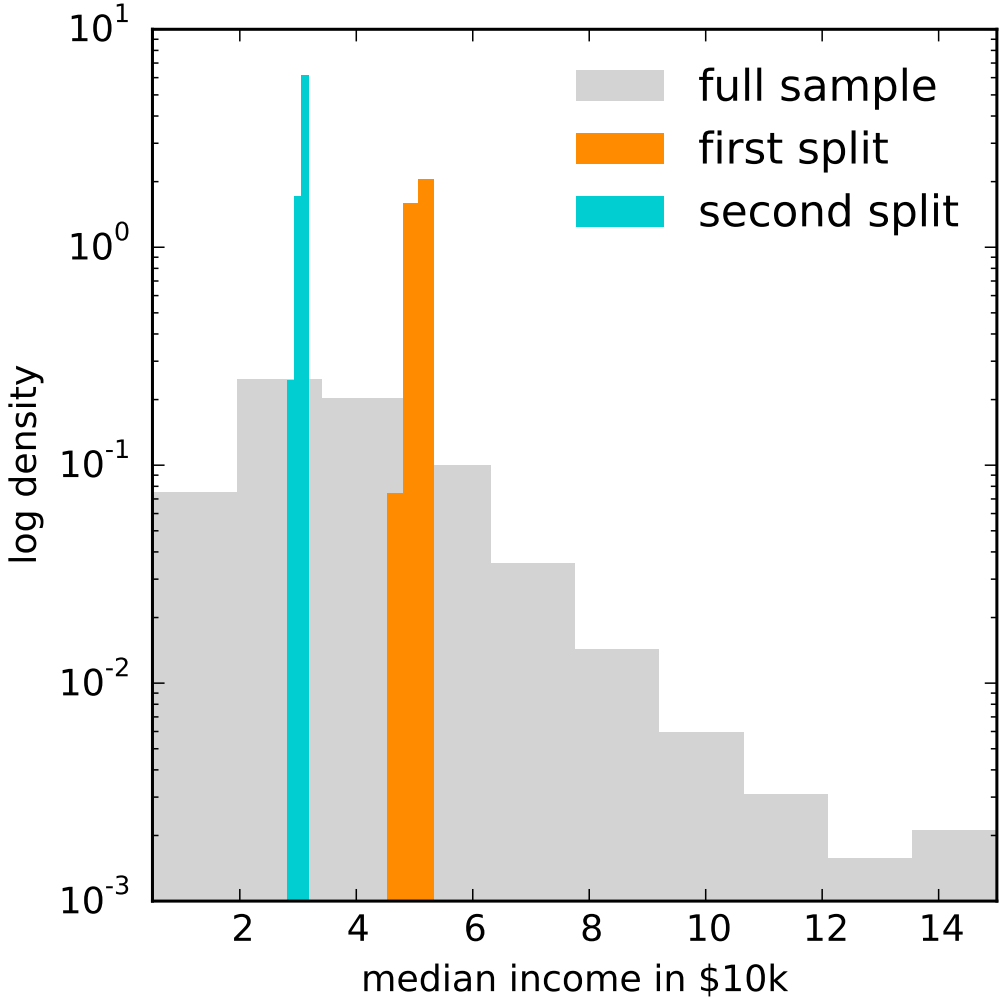
(Taddy, Chen, Yu, Wyle 2015)

California Housing Data

20k observations on median home prices in zip codes.



Above is the trunk you get setting min-leaf-size of 3500.



- sample tree occurs 62% of the time.
- 90% of trees split on income twice, and then latitude.
- 100% of trees have 1st 2 splits on median income.

Empirically and theoretically: trees are stable, at the trunk.

Random Forest bottlenecks

RFs (or BFs) are awesome, but when the data are too big to fit in memory or on a single machine they get extremely expensive.
(e.g., Google PLANET, Panda 2009).

A common solution is a 'sub-sampling forest': instead of drawing with-replacement, or re-weighting, draw $m \ll n$ sub-samples.

This defeats the whole purpose of trees: these are rules that are designed to grow in complexity with the amount of data available.
(that's why we bother storing so much data!)

If you starve the individual trees of data you lose.

Data Distribution and Big Data

Distributed: independent computations on many datasets.

For truly Big Data we need distributed algorithms.

The trick is to find strategies for distribution so that local-machine calculations are as relevant as possible to the global inference question: only data that needs to be together is together.

Empirical Bayesian Forests (EBF)

RFs are expensive. Sub-sampling hurts bad.

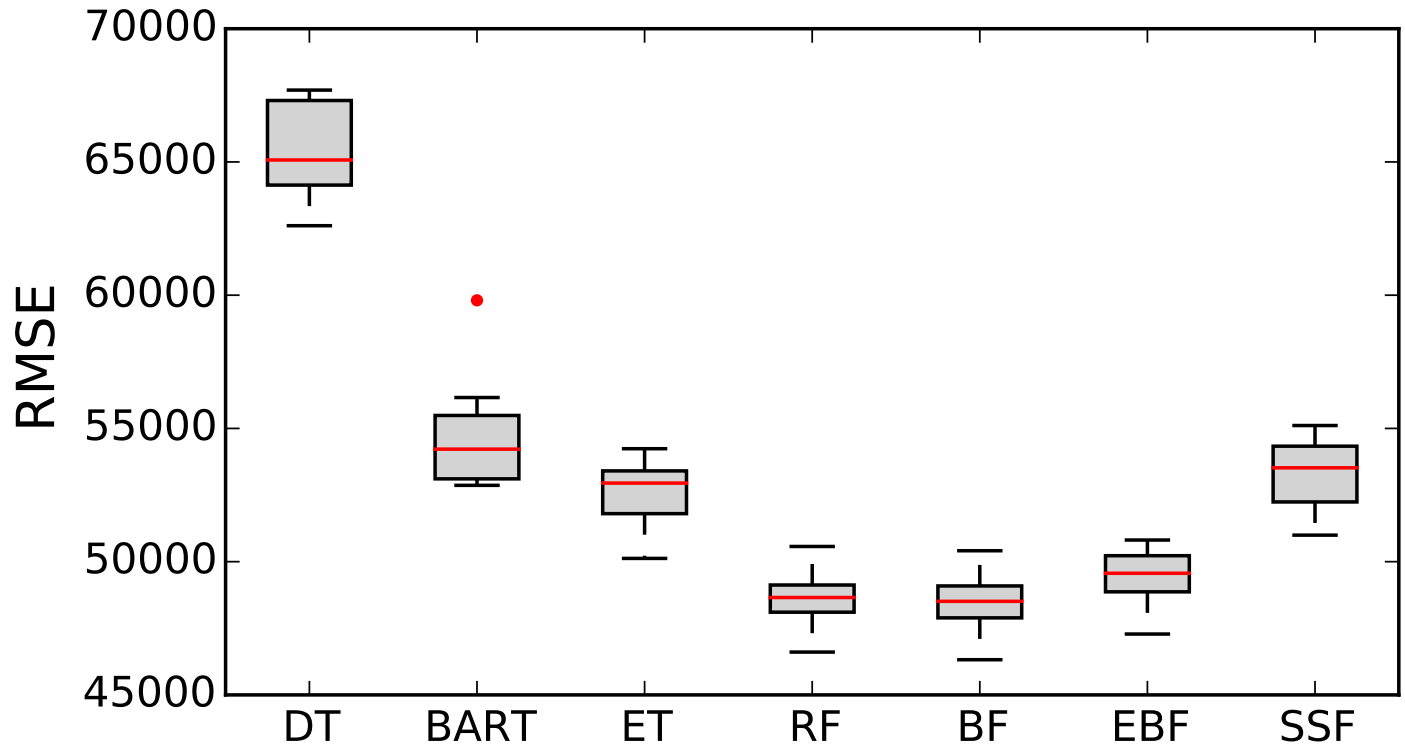
Instead:

- fit a single tree to a shallow trunk.
- Map data to each branch.
- Fit a full forest on the smaller branch datasets.

Since the trunks are all similar for each tree in a full forest, our EBF looks nearly the same at a fraction of computational cost.

It's an updated version of classical Empirical Bayes:
use plug-in estimates at high levels in a hierarchical model, focus effort at full Bayesian learning for the the hard bits.

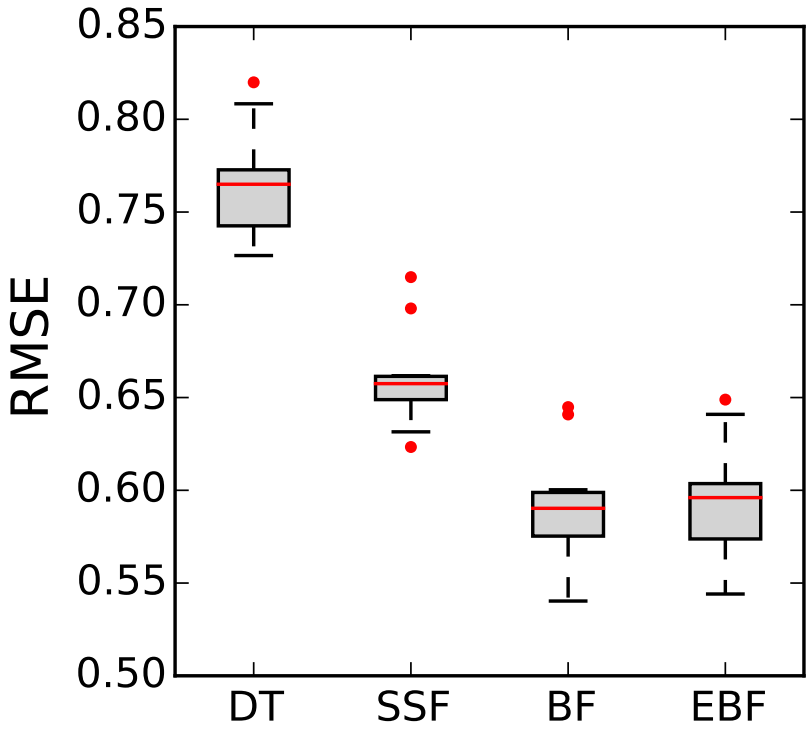
OOS predictive performance on California Housing



Here EBF and BF give nearly the same results. *SSF does not.*

EBFs crunch more data faster without hurting performance.

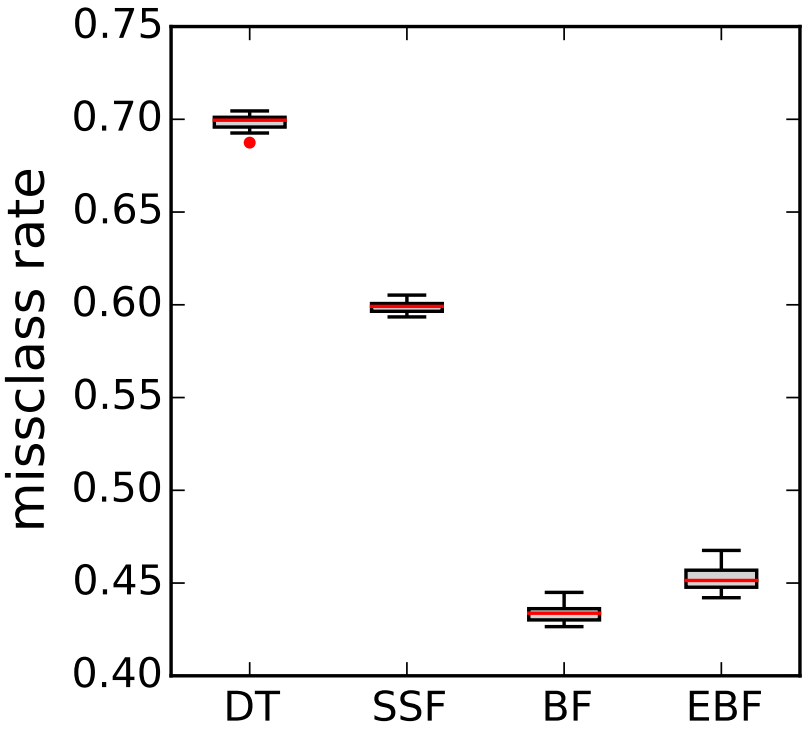
EBFs work all over the place



$\overline{\text{RMSE}}$	% WTB	
0.5905	0.0	BF
0.5953	0.8	EBF
0.6607	11.9	SSF
0.7648	29.5	DT

Predicting wine rating from chemical profile

EBFs work all over the place



$\overline{\text{MCR}}$	% WTB	
0.4341	0.0	BF
0.4531	4.4	EBF
0.5989	38.0	SSF
0.6979	60.8	DT

or beer choice from demographics

Choosing the trunk depth

Distributed computing perspective: fix only as deep as you must!

How big is each machine? Make that your branch size.

	CA housing			Wine			Beer		
Min Leaf Size in 10^3	6	3	1.5	2	1	0.5	20	10	5
% Worse Than Best	1.6	2.4	4.3	0.3	0.8	2.2	1.0	4.4	7.6

Still, open questions: e.g., more trees vs shallower trunk?

Catching Bad Buyer Experiences at eBay

BBE: 'not as described', delays, etc.

p(BBE) is a key input to search rankings, and elsewhere.

Big Data axiom: best way to improve prediction is more data.

(a.k.a. 'data beats model')

On 200 million transactions, EBF with 32 branches yields a 1.5% drop in misclassification over the SSF alternatives.

EBFs via Spark $\Rightarrow$ more data in less time.

Putting it into production requires some careful engineering, but this really is a very simple algorithm. Big gain, little pain.

Back to HTE: Treatment Effect Trees

Recall our earlier example: A/B experiment with user covariates $\mathbf{x}_i$.

Athey+Imbens propose indexing user HTE by fitting CART to

$$y_i^\star = y_i \frac{d_i - q}{q(1-q)} = \begin{cases} y_i/(1-q) & \text{if } d_i = 0 \\ y_i/q & \text{if } d_i = 1 \end{cases}$$

where q is the probability of treatment (2/3 in our example).

This works because

$$\mathbb{E}[y_i^\star | \boldsymbol{v}_i] = v_i(1) - v_i(0).$$

Recall: $v_i(d)$ is the potential outcome for user i if $d_i = d$.

A+I use Cross Validation to prune a single CART tree.

But: CV doesn't work well for selection of *unstable* models.

Breiman 1996 is the classic paper on this and it's the motivation behind Random Forests in Breiman 2001: averaging beats pruning.

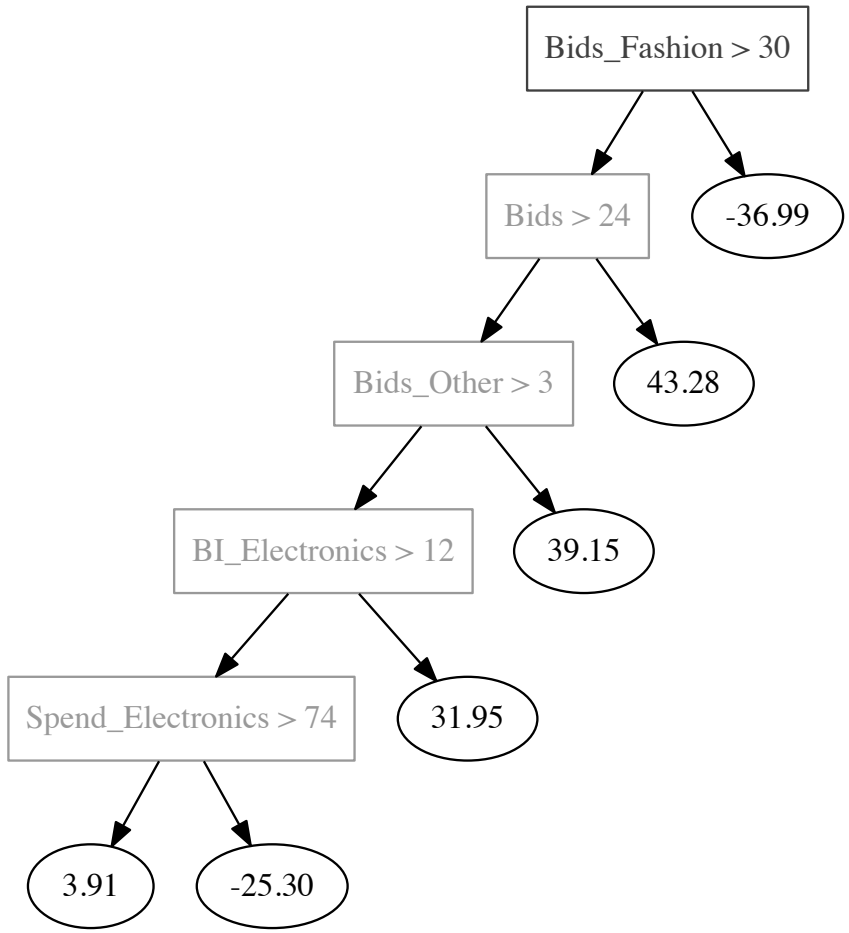
Of course, for many applications you need a single tree: e.g., in HTE modeling you can't show users an average of many websites.

You need to choose!

So we're stuck with an imperfect method for choosing a single tree.

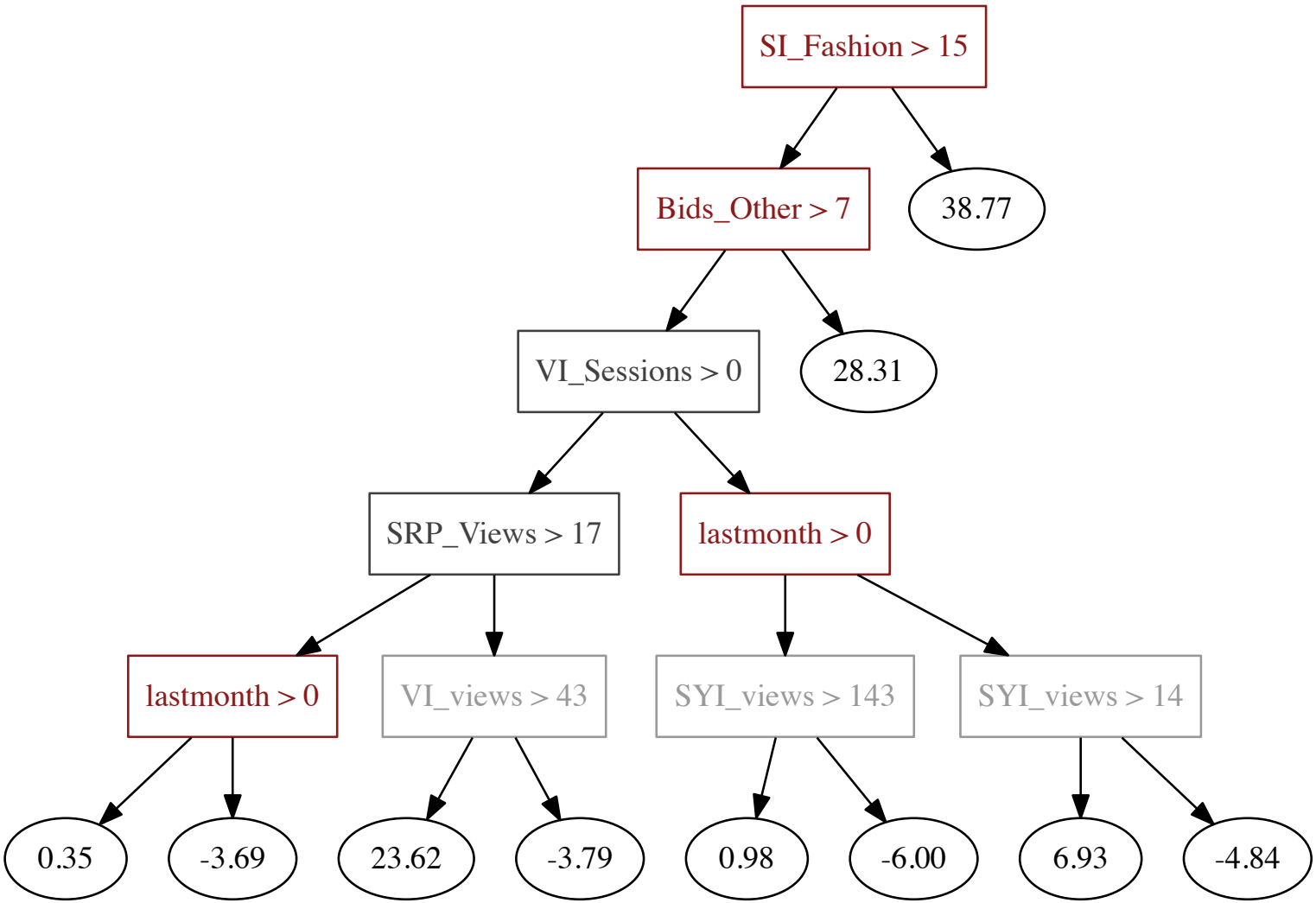
But we can at least apply the Bayesian bootstrap (i.e., fit a BF) to assess *posterior uncertainty* about these treatment-effect trees.

e.g., Pruned CART with after one week:



prob variable is node in tree: $\mathrm{p} < \frac{1}{3}$, $\mathrm{p} \in \left[\frac{1}{3}, \frac{1}{2}\right)$, and $\mathrm{p} \geq \frac{1}{2}$.

After 5 weeks the tree is much more stable.



We can quantify uncertainty about all sorts of structure.

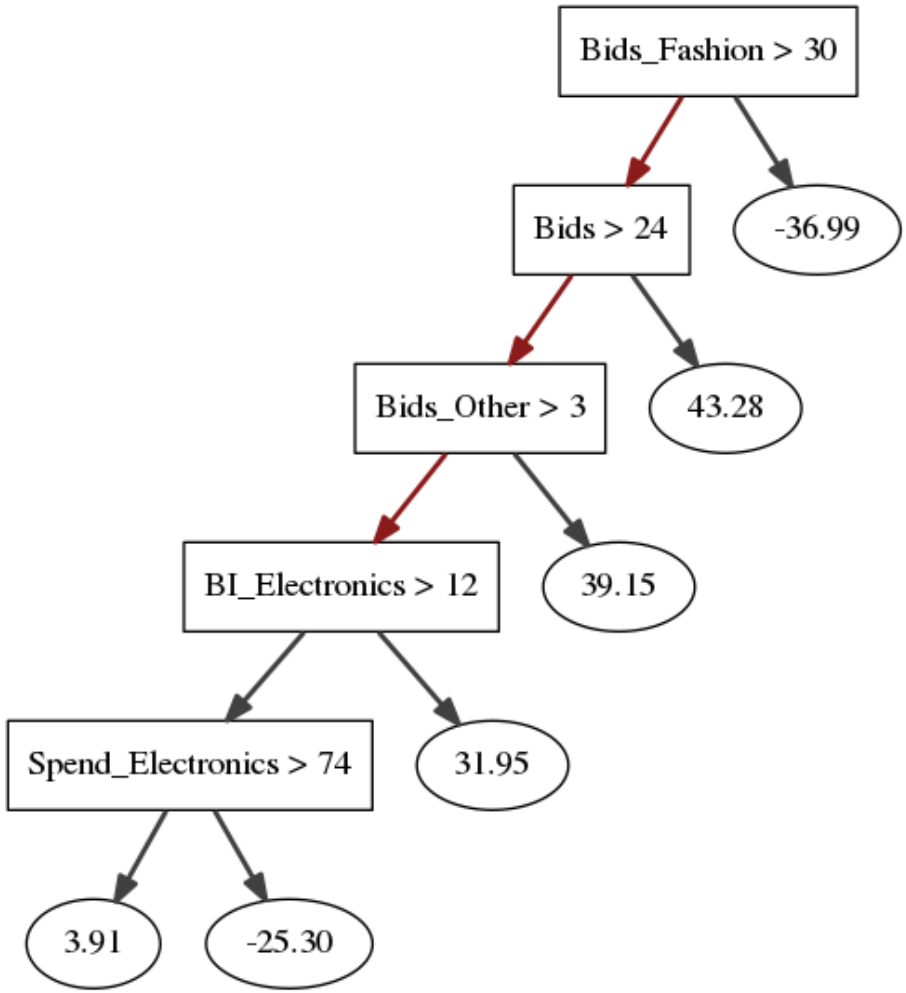
A Bayesian Forest implies a proximity matrix:

> $n \times n$ with i, j element the proportion of trees
> that have observations i and j in the same leaf.

This proximity matrix maps from from the full sample
to each observation's predicted value from the full tree.

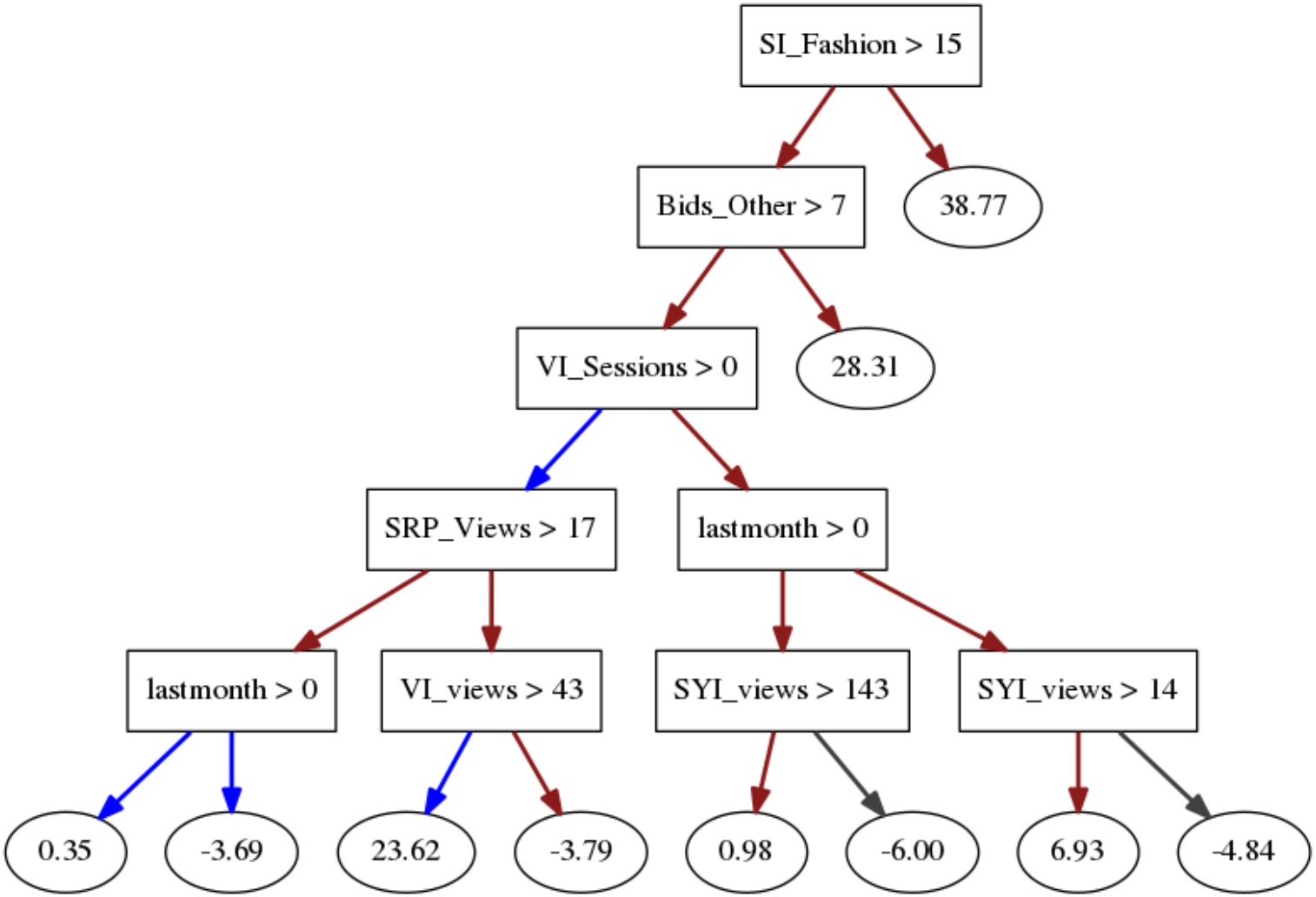
The average proximity between observations at each node
measures how well a given tree is replicating this structure.

e.g., Pruned CART with after one week:



Average proxy: $\mathrm{p} < \frac{1}{3}$, $\mathrm{p} \in \left[\frac{1}{3}, \frac{1}{2}\right)$, $\mathrm{p} \in \left[\frac{1}{2}, \frac{3}{4}\right]$, and $\mathrm{p} \geq \frac{3}{4}$.

Again: after 5 weeks the tree is much more stable.



Average proxy: $p < \frac{1}{3}$, $p \in \left[\frac{1}{3}, \frac{1}{2}\right)$, $p \in \left[\frac{1}{2}, \frac{3}{4}\right]$, and $p \geq \frac{3}{4}$.

Can we do better?

BNP and BFs provide scalable uncertainty quantification for trees.

An open question: can we use it to find a better single tree?

Posterior mean is usually best for prediction, but the *median* tends to be a better option than the mode ($\approx$ the single CART tree).

Can we find a tree that is approximately in the middle of our forest? Or, consider the graph defined by thresholding proximity at 0.5: can we recover a tree that yields a similar partitioning?

Big Data and distribution-free BNP

I think about BNP as a way to analyze (and improve) algorithms. Decouple action/prediction from the full generative process model.

This type of analysis is key for taking these algorithms from ML to fields like Economics that really care about uncertainty.

Efficient Big Data analysis

To cut computation without hurting performance, we need to think about what portions of the 'model' are hard or easy to learn.

Once we figure this out, we can use a little bit of the data to learn the easy stuff and direct our full data at the hard stuff.

This is *the* future for Big Data.

thanks!