

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.
- ▶ Our familiar MCMC algorithms take too long.

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 dimension of the feature space is enormous.
- ▶ We cannot write down or measure believable models.

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

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

$$\text{e.g., } y = f(\mathbf{x}) + \epsilon, \text{ or even just } f(y|\mathbf{x})$$

However averaging over all of the nuisance parameters we introduce to be 'flexible' is a hard computational problem.

Can we do scalable BNP?

Frequentists are great at finding simple procedures (e.g. $[\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'\mathbf{y}$) and showing that they will 'work' regardless of the true DGP.

(DGP = Data Generating Process)

This is classical '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 $\theta(g)$ 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 $\theta(g)$ induced by the posterior over g .

A flexible model for the DGP

$$g(\mathbf{z}) = \frac{1}{|\boldsymbol{\theta}|} \sum_{l=1}^L \theta_l \mathbb{1}[\mathbf{z} = \zeta_l], \quad \theta_l / |\boldsymbol{\theta}| \stackrel{iid}{\sim} \text{Dir}(\mathbf{a})$$

After observing $\mathbf{Z} = \{\mathbf{z}_1 \dots \mathbf{z}_n\}$, posterior has $\theta_l \sim \text{Exp}(a + \mathbb{1}_{\zeta_l \in \mathbf{Z}})$.
(say every $\mathbf{z}_i = [\mathbf{x}_i, y_i]$ is unique).

$a \rightarrow 0$ leads to $p(\theta_l = 0) = 1$ for $\zeta_l \notin \mathbf{Z}$.

$$\Rightarrow g(\mathbf{z} \mid \mathbf{Z}) = \frac{1}{|\boldsymbol{\theta}|} \sum_{l=1}^L \theta_l \mathbb{1}[\mathbf{z} = \mathbf{z}_l], \quad \theta_i \sim \text{Exp}(1)$$

This is just the Bayesian bootstrap.

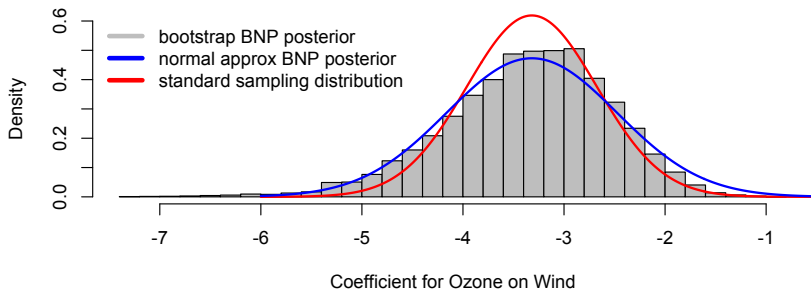
Ferguson 1973, Rubin 1981

Example: Ordinary Least Squares

Population OLS is a posterior functional

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

where $\Theta = \text{diag}(\theta)$. 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,

$$\tilde{\beta} = [\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'y + \nabla\beta|_{\theta=\mathbf{1}}(\theta - \mathbf{1})$$

We can derive *exact* posterior moments for $\tilde{\beta}$ under $\theta_i \stackrel{iid}{\sim} \text{Exp}(1)$.

e.g., $\text{var}(\tilde{\beta}) \approx (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\text{diag}(\mathbf{e})^2\mathbf{X}'(\mathbf{X}'\mathbf{X})^{-1}$, where $e_i = y_i - \mathbf{x}'_i\hat{\beta}$.

This is the familiar Huber-White 'Sandwich' variance formula.

See Lancaster 2003 or Poirier 2011.

Example: User-Specific Behavior in Experiments

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.

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



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



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} ?

Example: an HTE statistic that we care about.

Potential outcomes:

- ▶ $v_i(d)$ is the response for user i if $d_i = d$.
- ▶ The *treatment effect* is $v_i(1) - v_i(0)$
- ▶ We only observe one of $v_i(1)$ and $v_i(0)$: ' y '.

We'd love to solve for ' γ ' from the *moment condition*

$$\mathbb{E} [\mathbf{x}(v(t) - v(c) - \mathbf{x}'\gamma)] = \mathbf{0}$$

But randomization implies $\mathbb{E}[\mathbf{x}v(d)] = \mathbb{E}[\mathbf{x}v(d)|d]$, so:

$$\gamma = \mathbb{E} [\mathbf{x}\mathbf{x}']^{-1} \left(\mathbb{E}[\mathbf{x}y|d = 1] - \mathbb{E}[\mathbf{x}y|d = 0] \right)$$

This is a sort of OLS projection for treatment effects.

As we did with OLS, consider a first-order approximation

$$\tilde{\gamma} = \hat{\gamma} + \nabla \gamma|_{\theta=1}(\theta - \mathbf{1}).$$

where

$$\hat{\gamma} = \gamma|_{\theta=1} = n(\mathbf{X}'\mathbf{X})^{-1} \left(\frac{\mathbf{X}'_t \mathbf{y}_t}{n_t} - \frac{\mathbf{X}'_c \mathbf{y}_c}{n_c} \right).$$

This yields an approximate variance

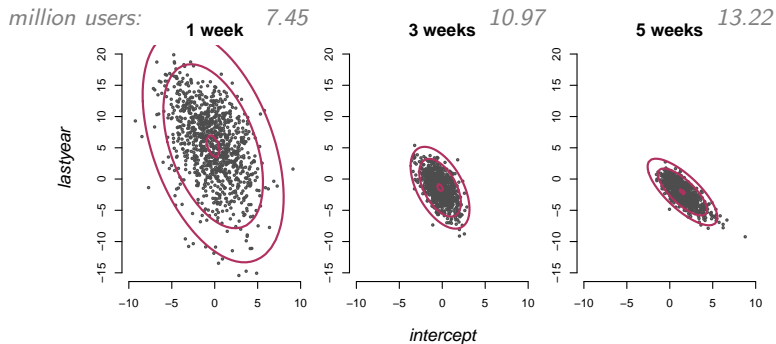
$$\text{var}(\tilde{\gamma}) \approx (\mathbf{X}'\mathbf{X})^{-1} \mathbf{X}' \text{diag}(\mathbf{e}^*) \mathbf{X} (\mathbf{X}'\mathbf{X})^{-1}$$

with ‘treatment effect residuals’

$$\mathbf{e}_i^* = \left(\frac{\mathbb{1}_{[i \in t]}}{n_t} - \frac{\mathbb{1}_{[i \in c]}}{n_c} \right) n y_i - \mathbf{x}_i' \hat{\gamma}.$$

Or you can bootstrap, but it takes a long time.

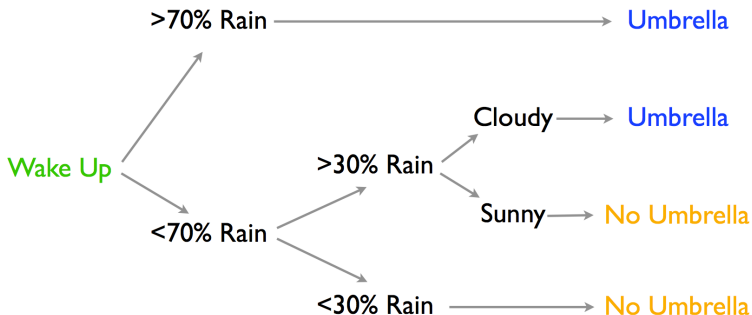
e.g., coefficient on purchase within last-year vs an intercept:



Sample is from posterior, contours are normal approximation.
This is a statistic we care about, even if the truth is nonlinear.

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 θ , find x to minimize

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

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

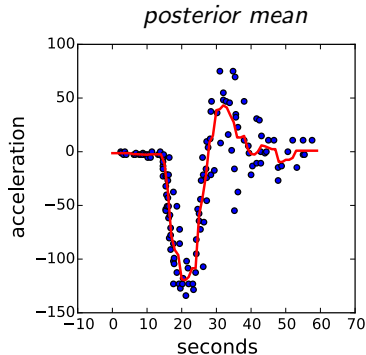
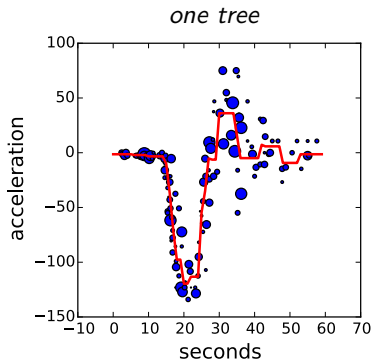
Population-CART might be a statistic we care about.

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 $b = 1 \dots B$:

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



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

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

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

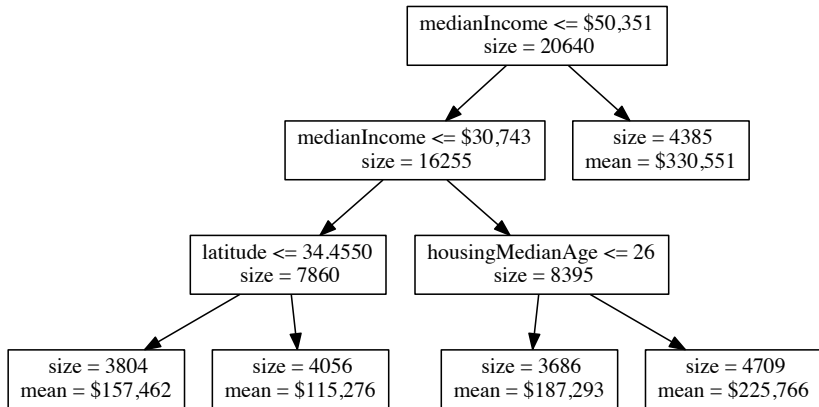
Based on this approx, we can say that for data at a given node,

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

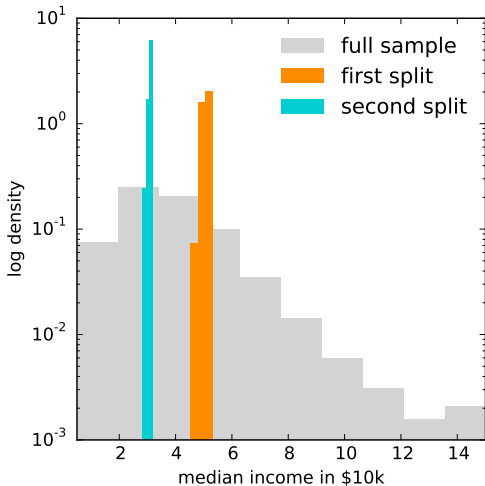
with p split locations and n observations.

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.

Empirical Bayesian Forests (**EBF**)

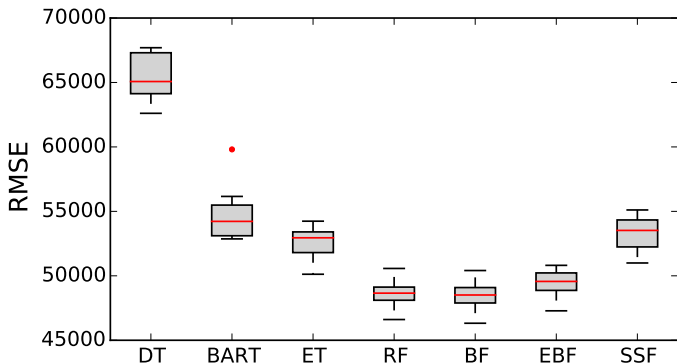
RFs are expensive. Sub-sampling hurts bad.

Instead:

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

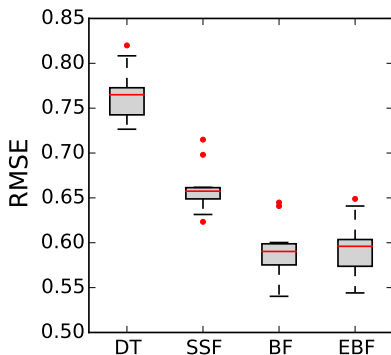
Empirical Bayes: fix plug-in estimates at high levels in a hierarchical model, focus effort at learning the hard bits.

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



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

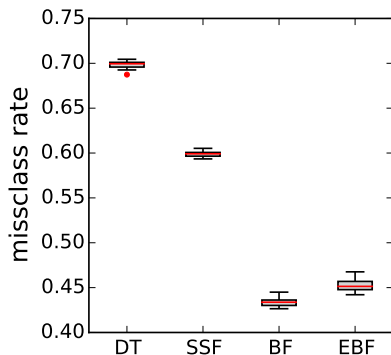
EBFs work all over the place



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



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		
<i>Min Leaf Size in 10^3</i>	6	3	1.5	2	1	0.5	20	10	5
<i>% Worse Than Best</i>	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.

$p(\text{BBE})$ is an input to search rankings.

Best way to improve prediction is more data.

EBFs via Spark: more data in less time.

On 12 million transactions, EBF with 32 branches yields a 1.3% drop in misclassification over the SSF alternatives.

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

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

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

This works because

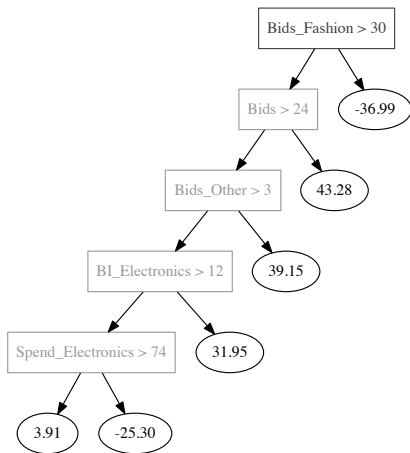
$$\mathbb{E}[y_i^* | \mathbf{v}_i] = v_i(1) - v_i(0)$$

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

We only get to observe one of these: $y_i = v_i(d_i)$.

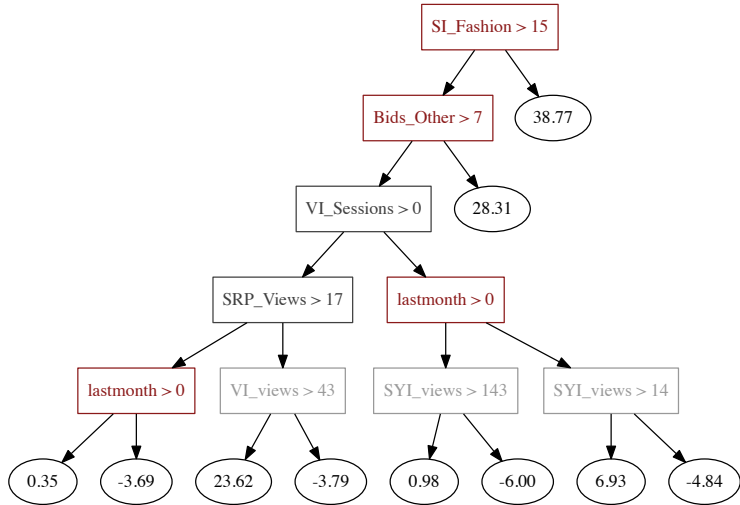
We can apply the Bayesian bootstrap (i.e., fit a BF) to assess *posterior uncertainty* about these treatment-effect trees.

e.g., sample depth-5 CART with min-leaf-size- 10^5 after one week:



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

After 5 weeks the tree is much more stable.



We can quantify uncertainty about all sorts of structure.

Posterior probability of CART splitting on variable

Week 5	depth in tree				
	1	≤ 2	≤ 3	≤ 4	≤ 5
<i>SI Fashion</i>	.45	.50	.50	.50	.50
<i>Bids Other</i>	.30	.75	.75	.75	.75
<i>VI sessions</i>	.05	.05	.05	.10	.35
<i>lastmonth</i>	.05	.10	.15	.40	.65
<i>SRP views</i>	.00	.10	.15	.25	.40
<i>VI views</i>	.00	.00	.10	.20	.25
<i>SYI views</i>	.00	.00	.05	.15	.25

⇒ easy scalable uncertainty quantification for complex algorithms.

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.

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.

thanks!