

# Big Data and Bayesian Nonparametrics

## **Example:** Empirical Bayesian Forests

Matt Taddy, Chicago Booth

`faculty.chicagobooth.edu/matt.taddy/research`

## Challenges of being Big and Bayesian

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 that matter.
- ▶ We cannot write down 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.

Can we Bayesians provide something like this?

## distribution free Bayesian nonparametrics

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

- ▶ Derive it from first principles, e.g. moment conditions.
- ▶ Or a statistic could be an algorithm that we know works.

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

Let's go back to Ferguson '73 and a multinomial sampling model:

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

$\mathcal{Z} = \{\boldsymbol{\zeta}_1 \dots \boldsymbol{\zeta}_L\}$  is the *fixed* support of the DGP and  $\theta_l \stackrel{iid}{\sim} \text{Exp}(a)$ .

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

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

So the posterior has  $g(\mathbf{z}) = \frac{1}{|\boldsymbol{\theta}|} \sum_{l=1}^L \theta_l \mathbb{1}[\mathbf{z} = \mathbf{z}_l]$  with  $\theta_i \sim \text{Exp}(1)$ .

This is just the Bayesian bootstrap.

## Example: Ordinary Least Squares

The *population* OLS projection 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).

We can derive posterior moments for a first-order approx

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

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  $e_i = y_i - \mathbf{x}_i'\hat{\beta}$ .

See Lancaster 2003 or Poirier 2011.

## 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** fits a tree by greedily growing with optimal splits.

Given **parent** node  $\{\mathbf{x}_i, y_i\}_{i=1}^n$ , the optimal split finds  $x_{ij}$  so that

$$\text{left}(x_{ij}) : \{\mathbf{x}_k, y_k : x_{kj} \leq x_{ij}\} \quad \text{and} \quad \text{right}(x_{ij}) : \{\mathbf{x}_k, y_k : x_{kj} > x_{ij}\}$$

**child** sets are as homogeneous in response  $y$  as possible.

e.g., a regression tree recursively splits to minimize

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

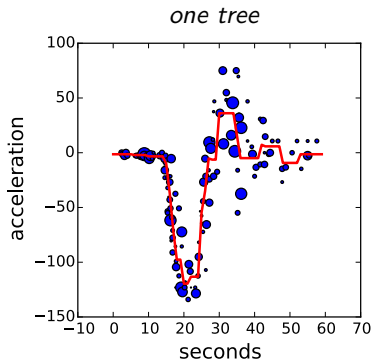
where  $\theta_k$  are observation weights and  $\mu_{c(\mathbf{x})} = \frac{1}{|\boldsymbol{\theta}_{c(\mathbf{x})}|} \sum_{k \in c(\mathbf{x})} \theta_k y_k$ .

If population-CART would predict well, then it's a s.w.c.a.

## Bayesian Forests: Sample the 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  $\mathcal{T}_b = \mathcal{T}(\theta^b)$



This is *very similar* to a random forest. So, RFs  $\approx$  posterior over trees.

## Theoretical **trunk** stability

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

As in simple OLS, look at the posterior for 1st order approx.

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

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,

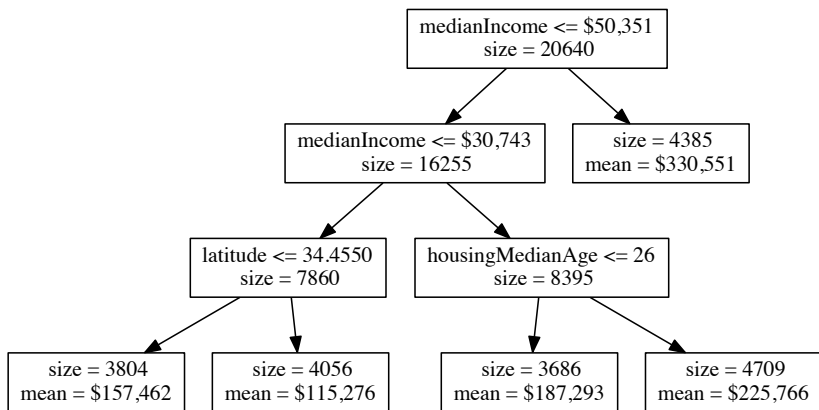
$$p(\text{optimal split matches sample CART}) \gtrsim 1 - \frac{p}{\sqrt{n}} e^{-n},$$

with  $p$  split locations and  $n$  observations.

Things are pretty stable, until they aren't: as the tree grows, node sizes get smaller and chance of a non-optimal split multiplies.

## 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.

Forests are expensive when data is too big to fit in memory.

Subsampling forests lead to a big drop in performance.

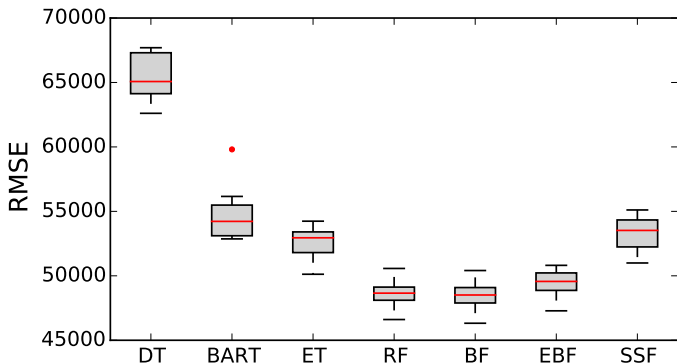
But wait: if the trunks are stable, can we just fit that once and then fit forests at each branch? **Yes!**

## Empirical Bayesian Forests (**EBF**):

- ▶ fit a single tree to a shallow **trunk**.
- ▶ Use this as a mapper to direct full data to each **branch**.
- ▶ Fit a full forest on the smaller branch datasets.

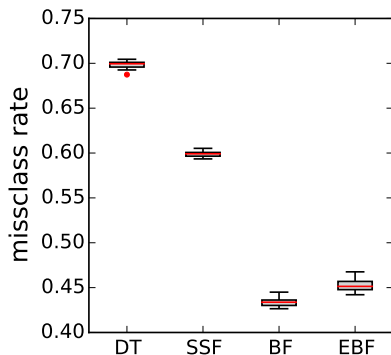
This is classic Empirical Bayes: fix higher levels in a *hierarchical model*, and direct your machinery+data at learning the hard bits.

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

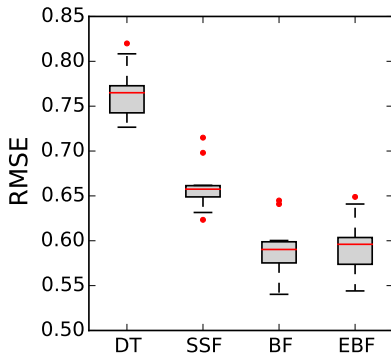


MCR	% WTB	
0.4341	0.0	BF
0.4531	4.4	EBF
0.5989	38.0	SSF
0.6979	60.8	DT

Predicting beer choice from demographics



## 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

or wine rating from chemical profile

## 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 <math>10^3</math></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?

## **EBFs at EBay: predicting Bad Buyer Experiences**

A BBE could be receiving something that is 'significantly not as described', or shipping delays, or any of many other problems.

The models are updated frequently, and information about  $p(\text{BBE})$  is an input to search rankings and more.

The best thing to improve predictions is more data.  
With millions of daily transactions, there's little limit on data.

## EBFs at EBay

Random forest runs take too long on full data.

Subsampling led to a noticeable and big drop in performance.

So: EBFs!

- ▶ trunk can be fit in distribution using Spark MLlib.
- ▶ this trunk acts as a sorting function to map observations to separate locations corresponding to each branch.
- ▶ Forests are then fit on a machine for each branch.

## EBFs at EBay

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.**

A key point: EBFs are not inherently 'better' than forests fit to all of the data. But EBFs can be fit to **more data** in less time.

## 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.

I believe that this is the future for Big Data.

## 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.

**thanks!**