BayesDB

A database for datascience

Richard Weiss

June 21, 2016

The problem

- How much time do you spend preparing data for a task?

The problem

- How much time do you spend preparing data for a task?
- How long do you spend setting up various kinds of models?

The problem

- How much time do you spend preparing data for a task?
- How long do you spend setting up various kinds of models?
- How often have you found that all your efforts came to nothing?

The problem

- How much time do you spend preparing data for a task?
- How long do you spend setting up various kinds of models?
- How often have you found that all your efforts came to nothing?
- And do you remember starting out, and how many techniques were *inaccessible*?

Sapir-Whorf for analysis

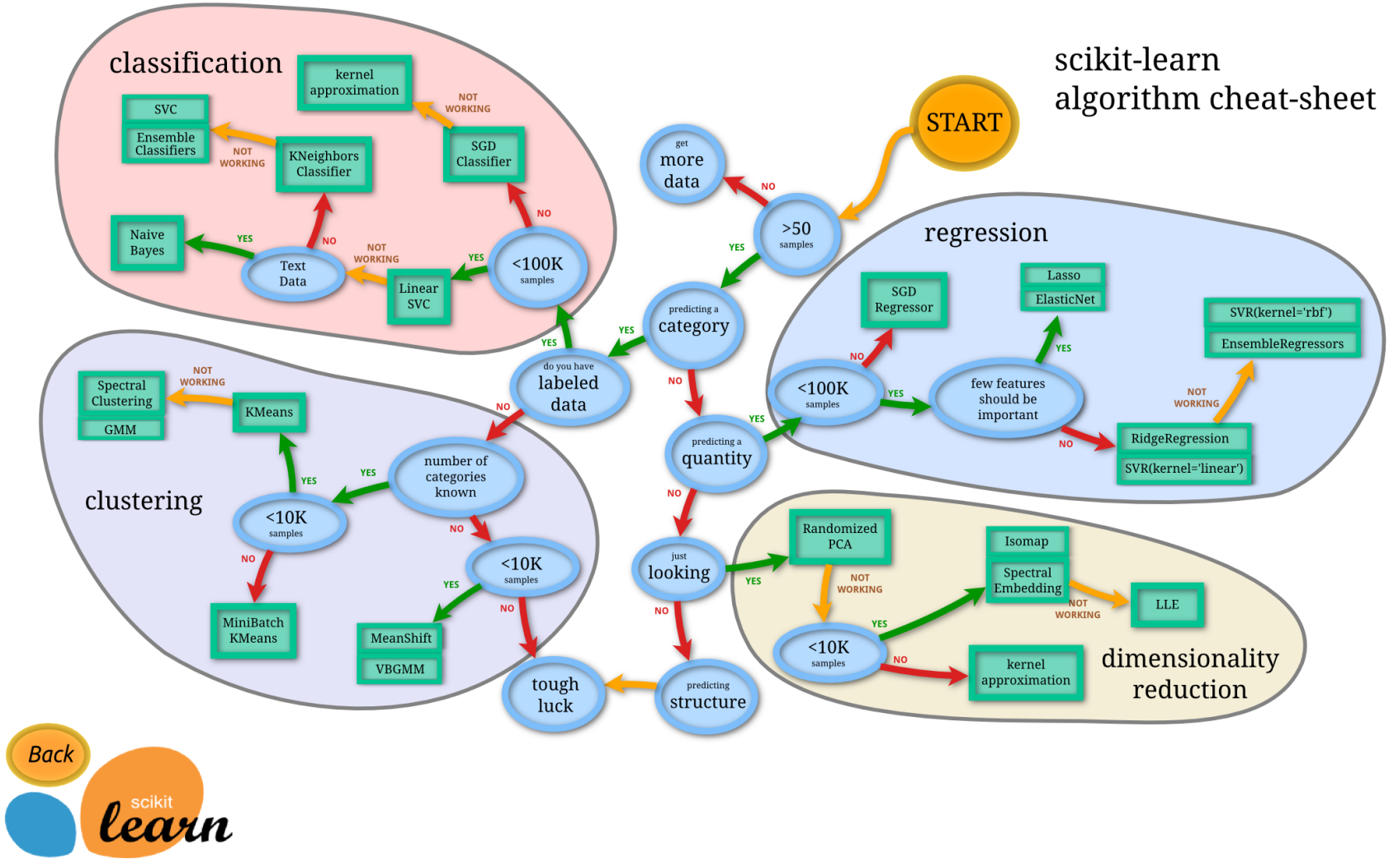
The Sapir Whorf hypothesis: that *language* affects the way you think, and perceive the world.

Sapir-Whorf for analysis

The Sapir Whorf hypothesis: that *language* affects the way you think, and perceive the world.

The Sapir Whorf hypothesis for programming and analysis: the easier it is to *express* a program, the more likely you are to conceive it and write it.

Thanks Kenneth E. Iverson!


scikit-learn
algorithm cheat-sheet
START
get more data
NO
>50 samples
YES
predicting a category
YES
NO
do you have labeled data
YES
NO
predicting a quantity
YES
NO
just looking
YES
NO
predicting structure
tough luck
<100K samples
YES
NO
Linear SVC
NOT WORKING
Text Data
NO
YES
Naive Bayes
KNeighbors Classifier
NOT WORKING
Ensemble Classifiers
SVC
SGD Classifier
NOT WORKING
kernel approximation
classification
<100K samples
YES
NO
SGD Regressor
few features should be important
YES
NO
Lasso
ElasticNet
RidgeRegression
NOT WORKING
SVR(kernel='rbf')
EnsembleRegressors
SVR(kernel='linear')
regression
number of categories known
YES
NO
<10K samples
YES
NO
KMeans
NOT WORKING
Spectral Clustering
GMM
MiniBatch KMeans
<10K samples
YES
NO
MeanShift
VBGMM
clustering
Randomized PCA
YES
NOT WORKING
<10K samples
YES
NO
Isomap
Spectral Embedding
NOT WORKING
LLE
kernel approximation
dimensionality reduction
Back
scikit learn

SQL is amazing.

SQL is one of the only languages where you say what you want.

BayesDB addresses this problem

BayesDB tries to extend this "say what you want" model to datascience.
With more expressive power: more thoughts.

Why are relational database management systems good?

- They let you use SQL.
- They handle all the messy storage problems (there are more than you think).
  Consider: the web programmer and the DBA.
- They handle caching, memory management, concurrency, etc.
- They let you enforce security policies.
- They scale better than your code (probably).

Why are RDBMS bad?

- We're limited in the questions we can ask
- In fact, most of the time, our analysis is building ersatz models. Consider: `average`, `group by`, `percentile_cont`, etc.

Wouldn't it be nice if we had a way to do this in the database?

Enter, BayesDB

Bayes DB gives SQL a bunch of extra tricks:

- Simulation
- Classification
- Regression
- Imputation
- Clustering
- Relationship discovery

Data loading and a caveat

Data loading can be a little bit tricky. Remember BDB is research software, so its a bit rough around the edges.
And, as it is research software, they've decided to log people's usage, to understand users better.

Do not load sensitive data into BayesDB

Some examples

Using the 2013 federal election, and 2011 census data, I've set up `votes` with

- Population
- Gender distribution
- Age distribution
- Income distribution
- Number of children
- Employment rate

We'll see what relationships these things have with each other and with the two party preferred vote.
All these data and the notebook are available at
`https://github.com/ririw/data-talk-2016-06`

So I loaded it all in...
Then I told BayesDB to go learn how to generate data...

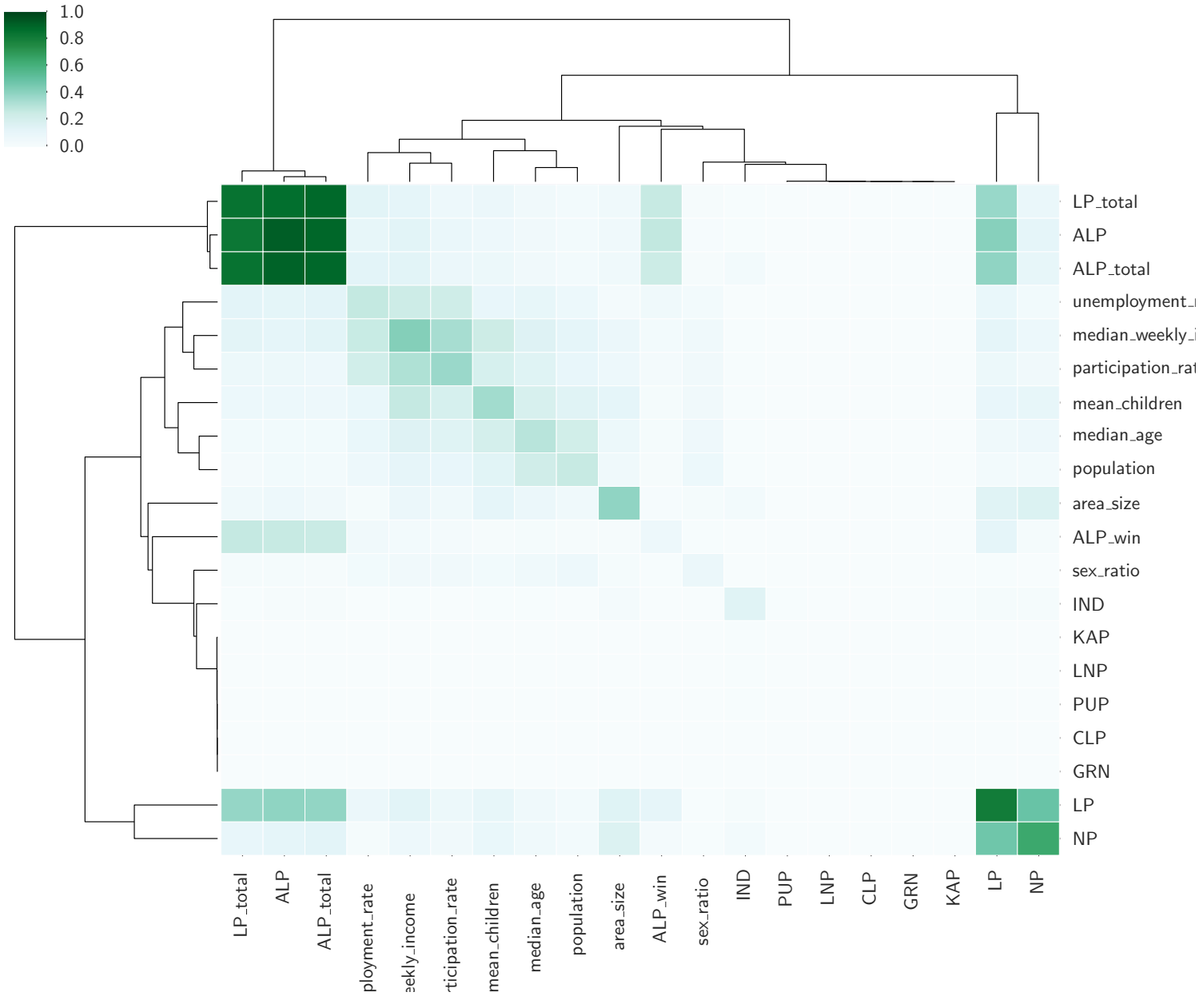
```
CREATE GENERATOR "votes_cc"
IF NOT EXISTS FOR "votes" USING crosscat(
    GUESS(*),
    GRN numerical, IND numerical,
    CLP numerical, KAP numerical,
    LNP numerical, LP  numerical,
    NP  numerical, PUP numerical,
    median_age            numerical,
    mean_children         numerical,
    median_weekly_income numerical)
```

A short while later...

Clusters in the data

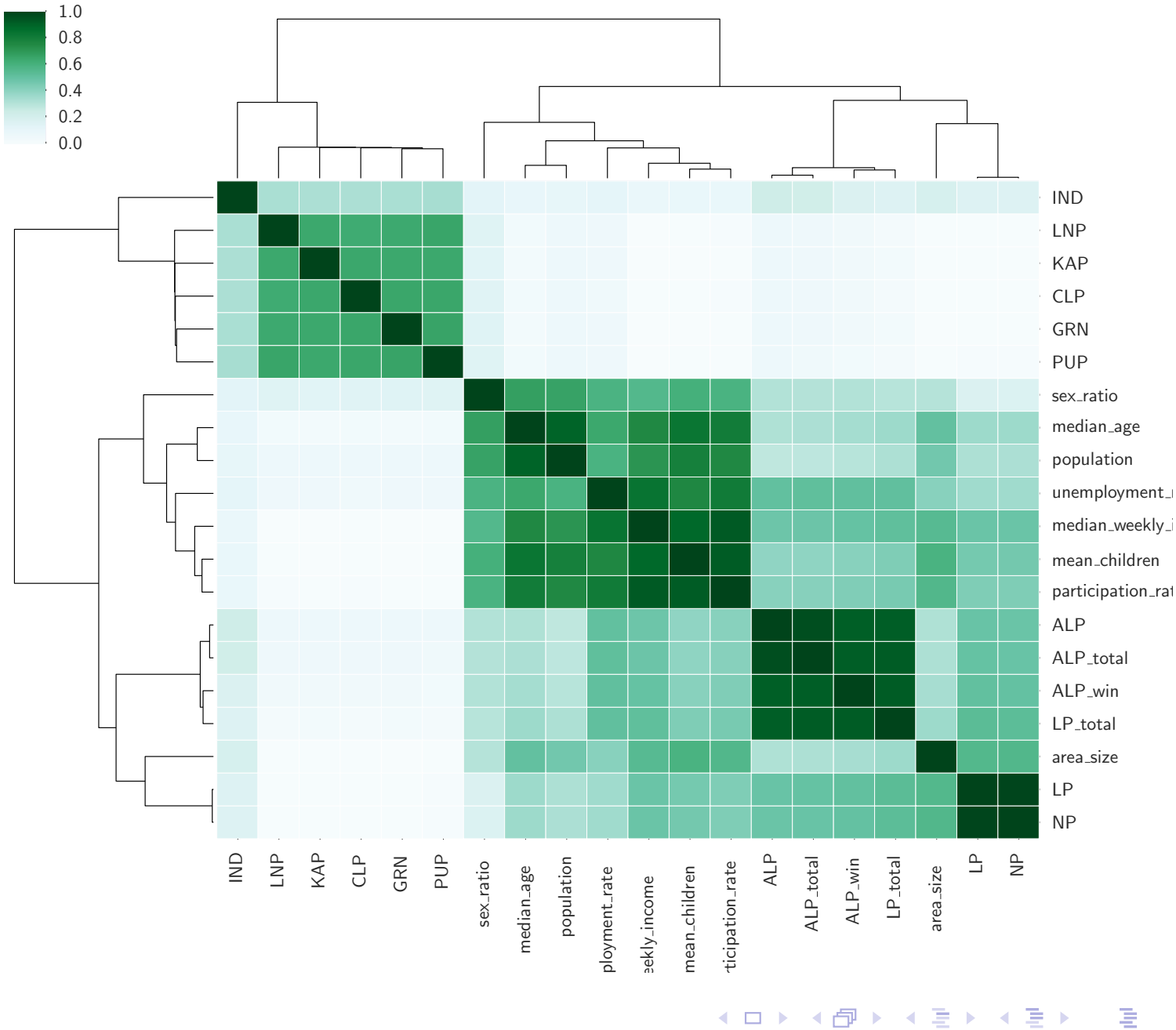
Inter-column dependencies: how much does one column tell you about another?
This is the point of "mutual information", and it's measured in bits.

```
votes.heatmap(
    votes.q('''
        ESTIMATE MUTUAL INFORMATION
        FROM PAIRWISE COLUMNS OF votes_cc
     ''')
)
```


1.0
0.8
0.6
0.4
0.2
0.0
LP_total
ALP
ALP_total
unemployment_
median_weekly_
participation_rat
mean_children
median_age
population
area_size
ALP_win
sex_ratio
IND
KAP
LNP
PUP
CLP
GRN
LP
NP
LP_total
ALP
ALP_total
ployment_rate
ekly_income
ticipation_rate
mean_children
median_age
population
area_size
ALP_win
sex_ratio
IND
PUP
LNP
CLP
GRN
KAP
LP
NP

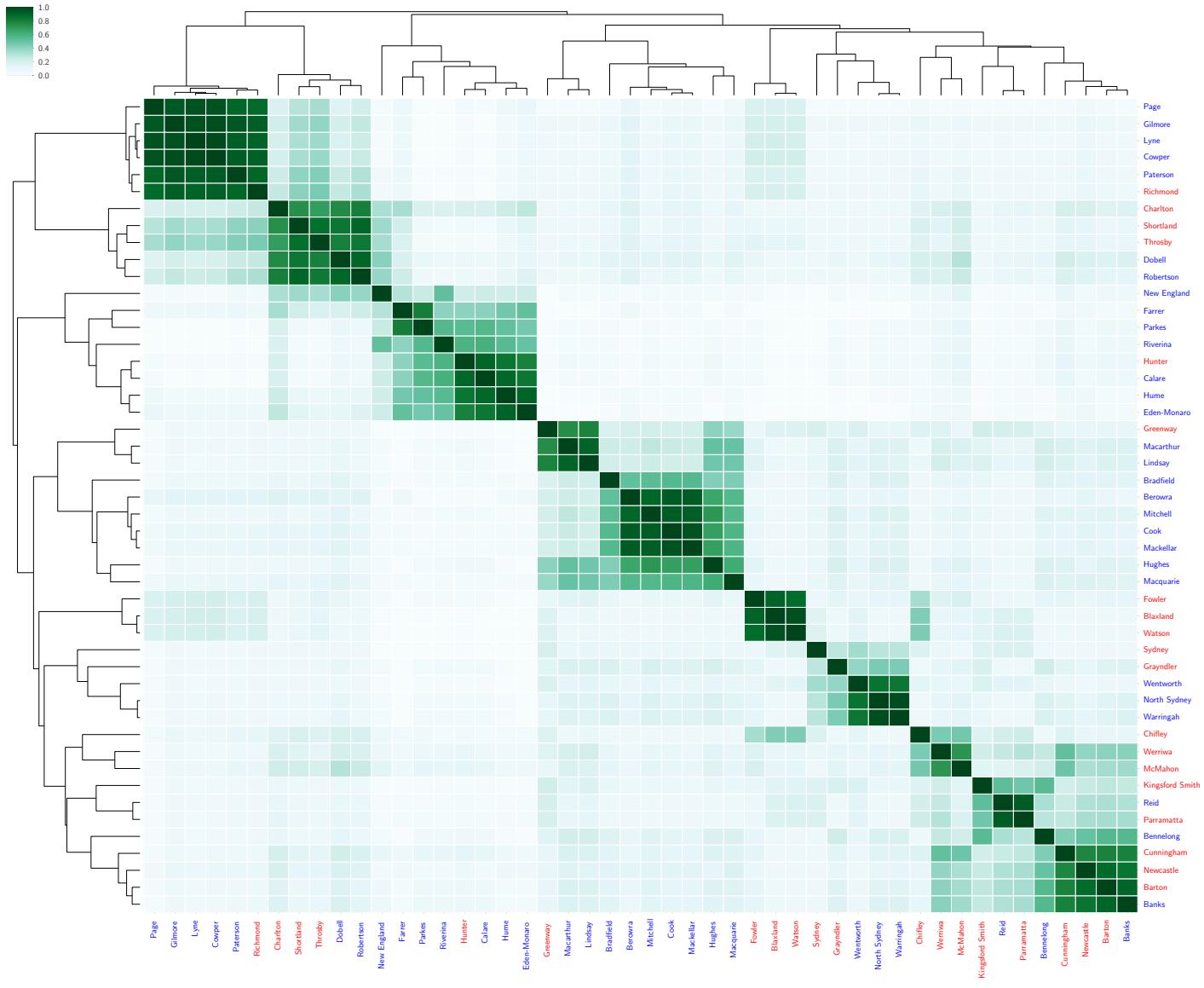
An alternative is dependence probability, which looks at the underlying CrossCat model.

```
votes.heatmap(
    votes.q('''
        ESTIMATE DEPENDENCE PROBABILITY
        FROM PAIRWISE COLUMNS OF votes_cc
     ''')
)
```


1.0
0.8
0.6
0.4
0.2
0.0
IND
LNP
KAP
CLP
GRN
PUP
sex_ratio
median_age
population
unemployment_
median_weekly_
mean_children
participation_rat
ALP
ALP_total
ALP_win
LP_total
area_size
LP
NP
IND
LNP
KAP
CLP
GRN
PUP
sex_ratio
median_age
population
ployment_rate
ekly_income
mean_children
ticipation_rate
ALP
ALP_total
ALP_win
LP_total
area_size
LP
NP

This also works row-wise, to find electorates that are similar:

```
ESTIMATE SIMILARITY FROM PAIRWISE demograpics_cc
```


1.0
0.8
0.6
0.4
0.2
0.0
Page
Gilmore
Lyne
Cowper
Paterson
Richmond
Charlton
Shortland
Throsby
Dobell
Robertson
New England
Farrer
Parkes
Riverina
Hunter
Calare
Hume
Eden-Monaro
Greenway
Macarthur
Lindsay
Bradfield
Berowra
Mitchell
Cook
Macellar
Hughes
Macquarie
Fowler
Blaxland
Watson
Sydney
Grayndler
Wentworth
North Sydney
Warringah
Chifley
Werriwa
McMahon
Kingsford Smith
Reid
Parramatta
Bennelong
Cunningham
Newcastle
Barton
Banks

Classification/Imputation example

To test the classification accuracy, I created a third table, `votes_held-out_cc`, and randomly withheld half of the data.

```
INFER ALP_win FROM votes_heldout_cc
```

This manages to reach 73% accuracy, a little under logistic regression.
This seems to heavily depend on which part of the data we hold out.

Regression examples

What would happen if... everyone was younger or older (the median age is 38 or so)?

```
SIMULATE ALP, LP, NP FROM votes_cc
given median_age = 35 LIMIT 10000
```

```
SIMULATE ALP, LP, NP FROM votes_cc
given median_age = 40 LIMIT 10000
```

If everyone was 35:

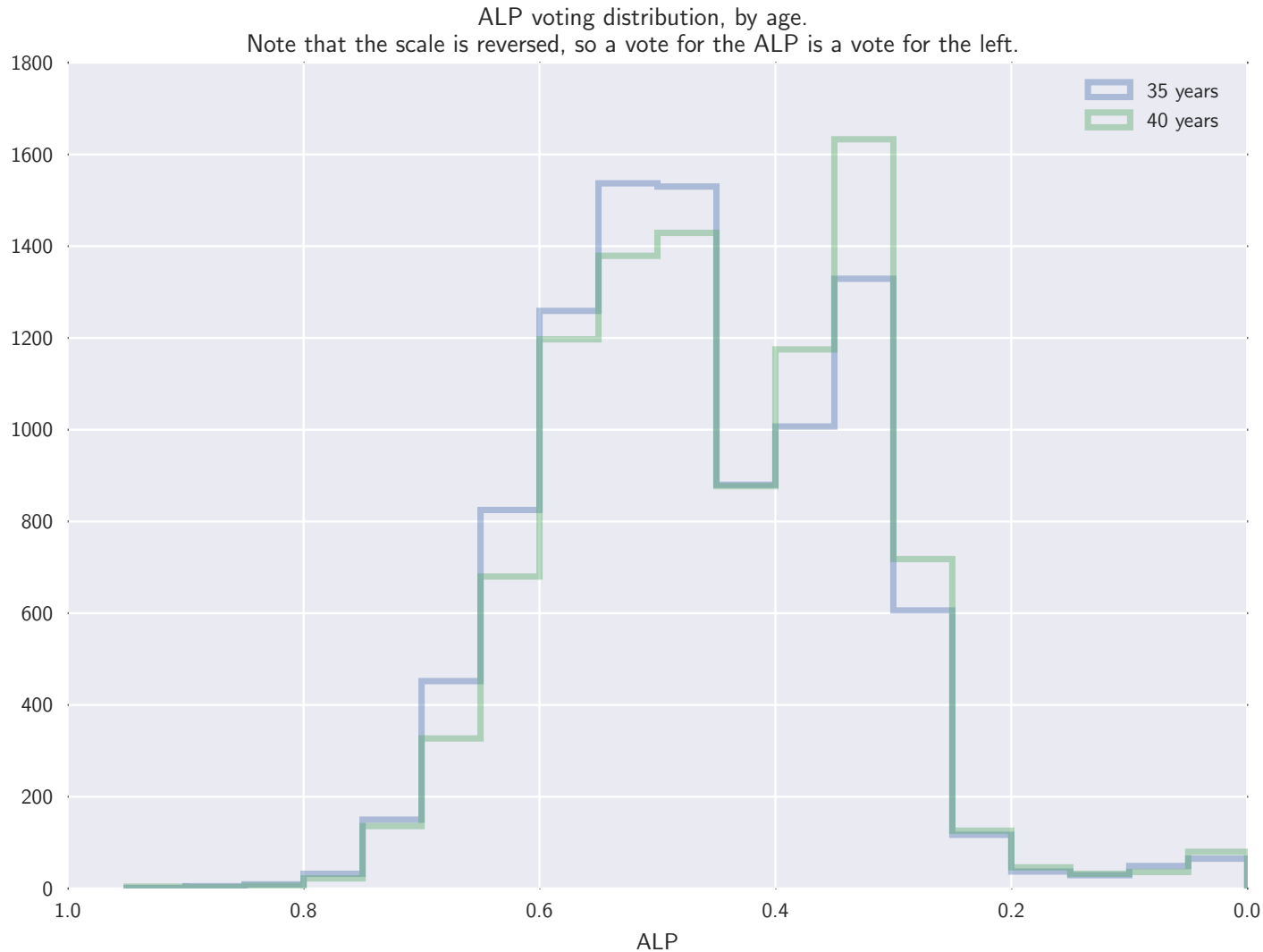
ALP	47%
LP	44%
NP	8.3%

And now, if everyone was 40:

ALP	44%
LP	42%
NP	12%

That sounds plausible...

We can also produce histograms, to understand what's going on:



BayesDB is based around a powerful, flexible, generative model for the rows and columns of a table.

- Simulation: draw from the model, subject to any conditions we want
- Classification/regression: draw from the model, subject to conditions, then take the mean (regression) or mode (classification)
- Filling in missing values: classification of the missing values, given all known values.
- Column dependence: check if two columns are dependent in the underlying model.

CrossCat - Hierarchical modeling of tables

CrossCat is a general purpose, Bayesian method for analyzing high-dimensional data tables

Because of its flexibility, CrossCat is a good "default choice"

CrossCat - Hierarchical modeling of tables

CrossCat is a general purpose, Bayesian method for analyzing high-dimensional data tables

Because of its flexibility, CrossCat is a good "default choice"

Data type	Distribution	Parameter Prior
Numerical	Normal	Normal inverse gamma
Binary	Binomial	Dirichlet multinomial
Categorical	Multinomial	Dirichlet multinomial

So how does it work

- It's a *generative* model.
- This is what lets us do so much with it.
- It just needs to be powerful enough to generate things well.

Slicing and dicing

CrossCat lets us slice up the rows as much as necessary, to find a good model.
Even to the point of slicing up columns into clusters.

To generate a table:

1. Assign dimensions to "views" with a Chinese restaurant process (rich get richer). Now you've set up your columns.

To generate a table:

1. Assign dimensions to "views" with a Chinese restaurant process (rich get richer). Now you've set up your columns.
2. Within each view, pick a category for each row, based on a CRP. Now you've got columns and latent sources, but no rows.

To generate a table:

1. Assign dimensions to "views" with a Chinese restaurant process (rich get richer). Now you've set up your columns.
2. Within each view, pick a category for each row, based on a CRP. Now you've got columns and latent sources, but no rows.
3. Pick parameters for the categories.

To generate a table:

1. Assign dimensions to "views" with a Chinese restaurant process (rich get richer). Now you've set up your columns.
2. Within each view, pick a category for each row, based on a CRP. Now you've got columns and latent sources, but no rows.
3. Pick parameters for the categories.
4. Generate row entries for each view, using the parameters for each category.

Of course, there's a few things I glossed over...

Baysian inference and Monte Carlo simulation

How do we take a way to *generate* data from parameters, to a way to estimate the parameters from the data?
The answer lies in Bayes' Rule, and that, perhaps, explains the name of the database.

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$

Bayes' casino

There's a family techniques, called *Markov Chain Monte Carlo* methods, that solve the problem.

- Pick a direction.
- Take a look.
- If the grass is greener (higher probability), go for it.
- If the grass isn't greener, take a probabilistic step.
- Then take a sample, wherever you are.

This lets you take samples from the distribution, without need to fully understand it.

References

For a more intuitive explanation: *A probabilistic model of cross-categorization* Shafto, Kemp, Mansinghka, Tenenbaum, 2011, in *Cognition*

If you want all the maths: *CrossCat: a fully Bayesian, nonparametric method for analyzing heterogeneous, high-dimensional data* Mansinghka, Shafto, Jonas, Petschulat, Gasner, and Tenenbaum

- Why was that CrossCat paper in a journal called *Cognition*?
- It seems to match how *humans* categorize things.

- Why was that CrossCat paper in a journal called *Cognition*?
- It seems to match how *humans* categorize things.
- But, do we want to do more of what humans do?

Limitations

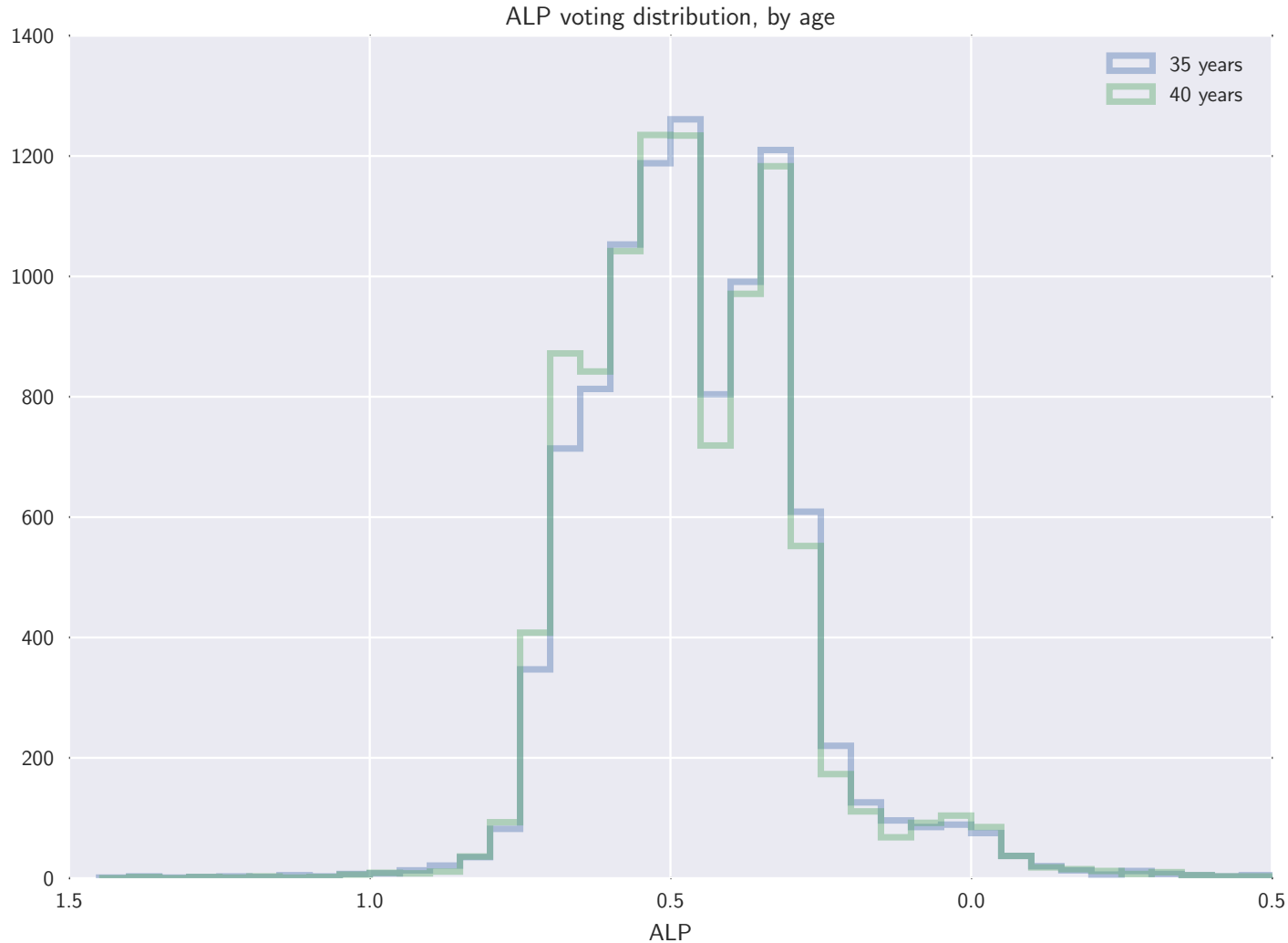
The small limitations:

- Speed - it's quite slow
- Not a random forest
- Not really *relational*

The big limitations:

- Model definition power
- Human prior - the CrossCat categorization method

If I expand what I had before to the left and the right...



The solution to the modeling issues

The MML and external models.

The solution to the modeling issues

The MML and external models.

DBA & Systems programmers	Web programmers & Designers
Expert modelers	Business domain experts

There can also be external models, full of forests.

So does it solve all our problems?

Well, no, but it's a big step in the right direction.

So does it solve all our problems?

Well, no, but it's a big step in the right direction.
Other things to look at:

- Madlib - data science tools for PostgreSQL. Much more mature, but less integrated.
- MsSQL and Oracle also have add-ons along these lines.
- Stan and PyMC3 - high quality probabilistic programming tools

Summary

- As data scientists spend a lot of our time moving data around.
- And we'd also like ways to make data science more accessible to others.
- BayesDB simplifies the modeling step, and makes it possible to separate it from the analysis step.
- Doing so will hopefully cut down on data munging and make DS more accessible.
- But there's a long way to go.

Questions?