Loom: an open-source predictive database engine

Fritz Obermeyer, Beau Cronin

joint work with
Jonathan Glidden, Eric Jonas, Cap Petschulat

2014-11-11

http://fritzo.org/notes/2014/loom.pdf

Beau

PhD in computational neuroscience, MIT

Co-founder of Navia Systems and Prior Knowledge

Interests: Intelligence and perception

Fritz

PhD in pure and applied logic, CMU

Lead developer of Loom

Interests: Engineering probabilistic systems

Motivation

What is a predictive database?

What is Loom?

Case Study: Lending Club

Our Goals

Richer understanding of data

More robust, modular predictive toolchain

Principled handling of uncertainty

General Approach

Joint probability modeling

Nonparametric Bayesian priors

MCMC(ish) sampling

Loom is based on Cross-Categorization

Research

-Patrick Shafto, Charles Kemp, Vikash Mansinghka, Matthew Gordon, Josh Tenenbaum (2006)
-Vikash Mansinghka, Eric Jonas, Cap Petschulat, Beau Cronin, Patrick Shafto, Josh Tenenbaum (2009)
-Yue Guan, Jennifer Dy, Donglin Niu, Zoubin Ghahramani (2010)
-Patrick Shafto, Charles Kemp, Vikash Mansinghka, Josh Tenenbaum (2011)
-Fritz Obermeyer, Jonathan Glidden, Eric Jonas (2014)

Open Source implementations

BayesDB - http://probcomp.csail.mit.edu/bayesdb (2013)
Loom - https://github.com/priorknowledge/loom (2014)

Challenges

Probabilistic inference is (rightly) viewed as slow

Academic researchers have the wrong incentives to scale it up

Existing assumptions and conventional wisdom about predictive modeling

Mike Jordan's Reddit AMA

Q: "Why do you believe nonparametric models haven't taken off?"

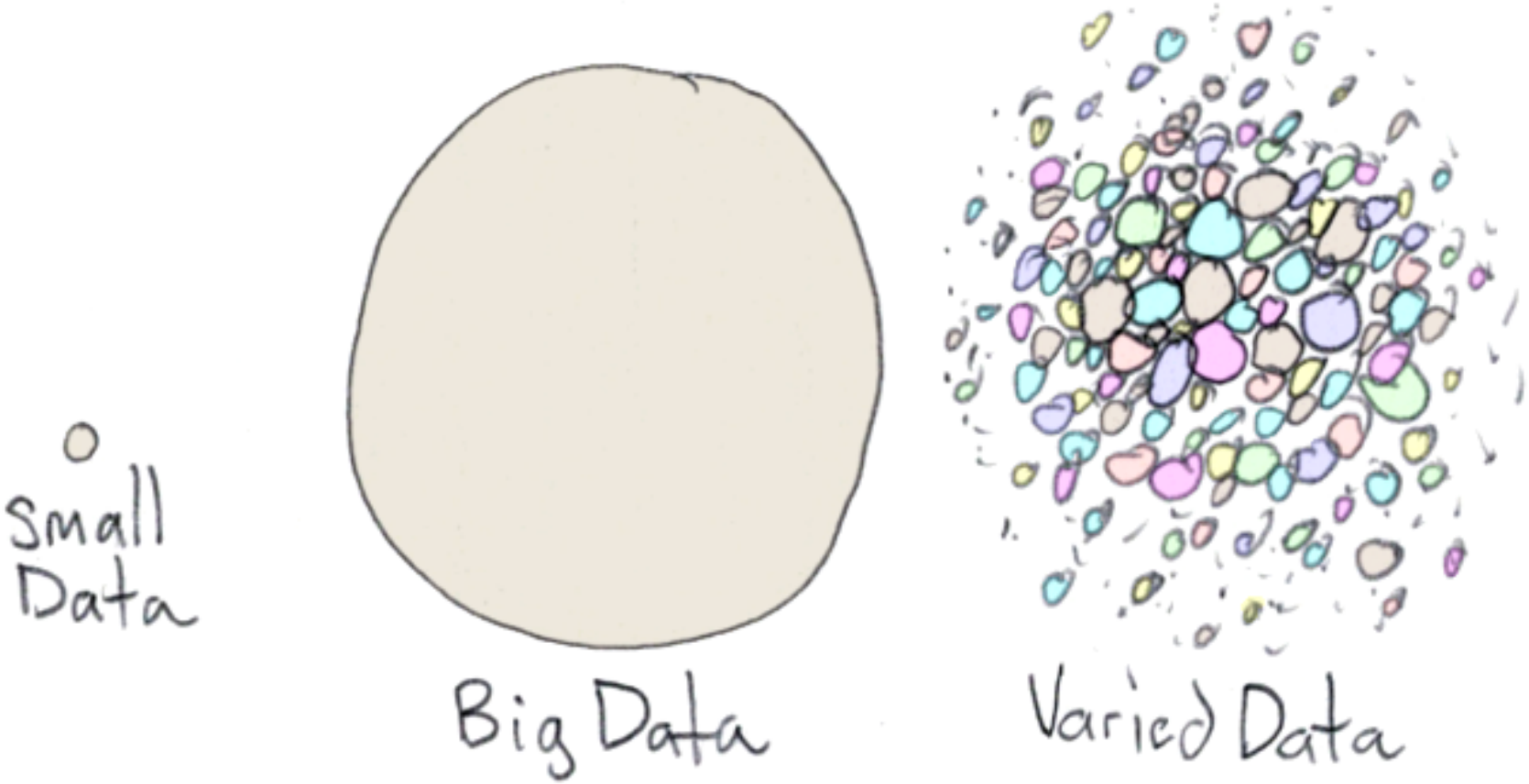
A: "I think that mainly they simply haven't been tried... I do think that Bayesian nonparametrics has just as bright a future in statistics/ML as classical nonparametrics has had and continues to have. Models that are able to continue to grow in complexity as data accrue seem very natural for our age"

What does scale mean?

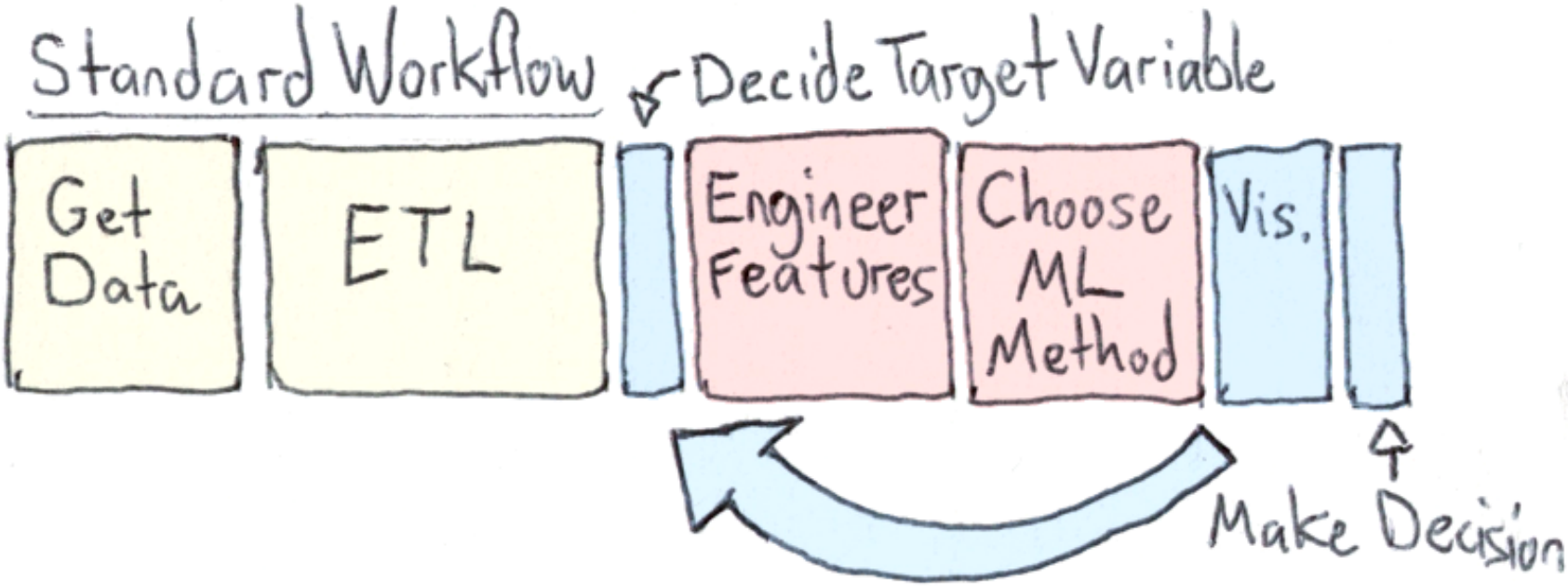
B2C: ∼10 data sources
high volume
high velocity

B2B: ∼10000 data sources, each with:
structured relational data
customizable database schema
small–medium volume

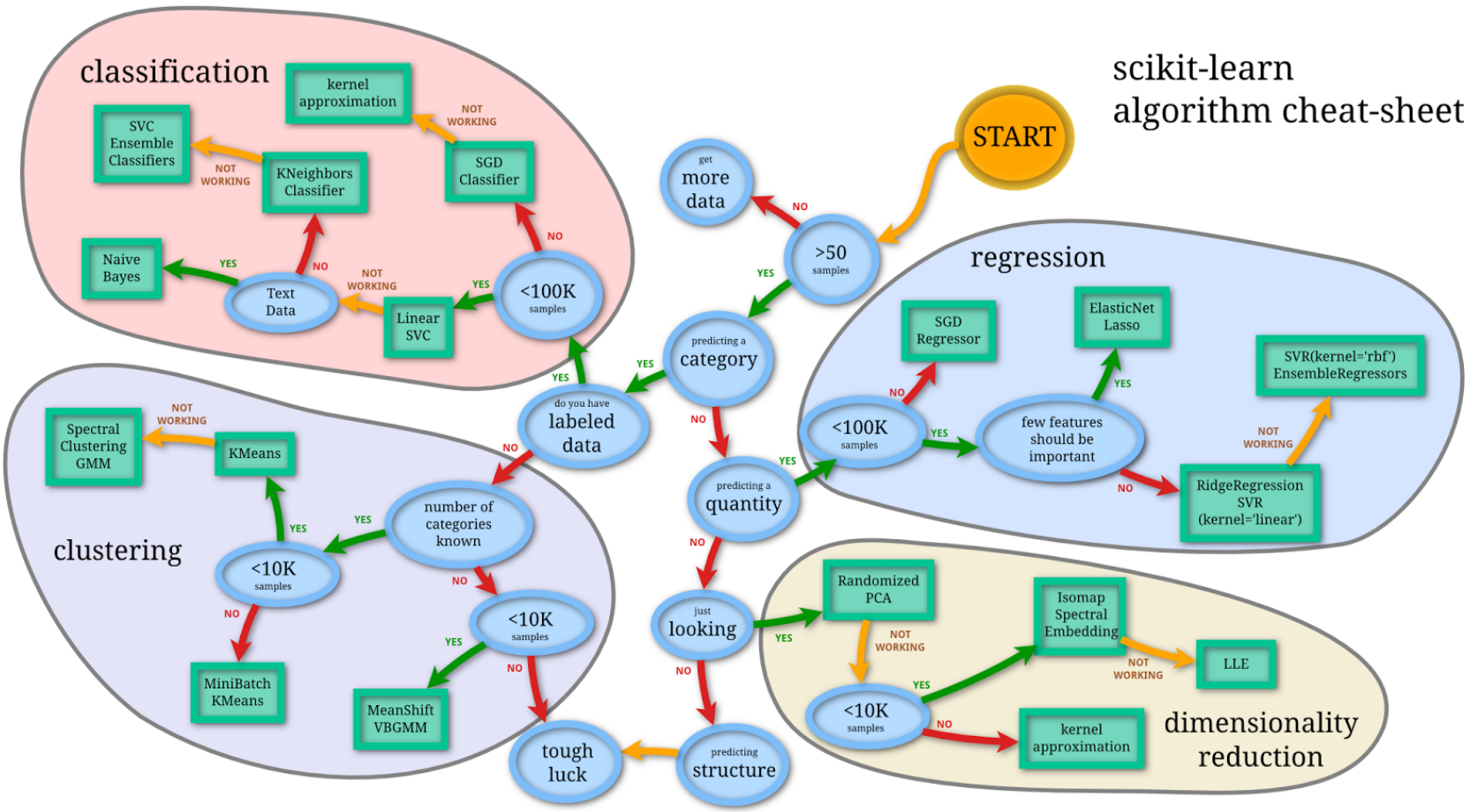
Problem: scale to 10000 datasets!



Data Science Workflows



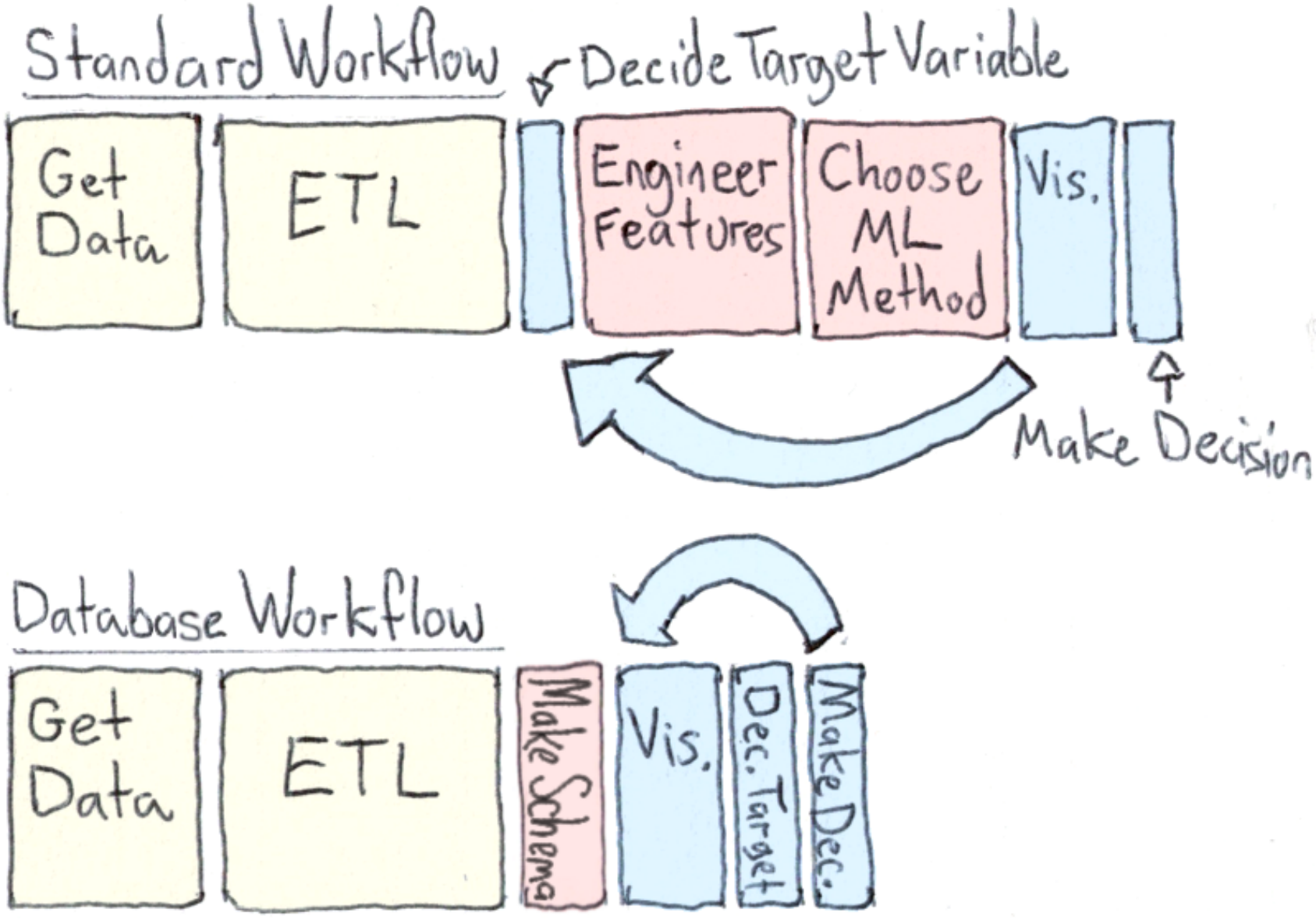
Which method to use?



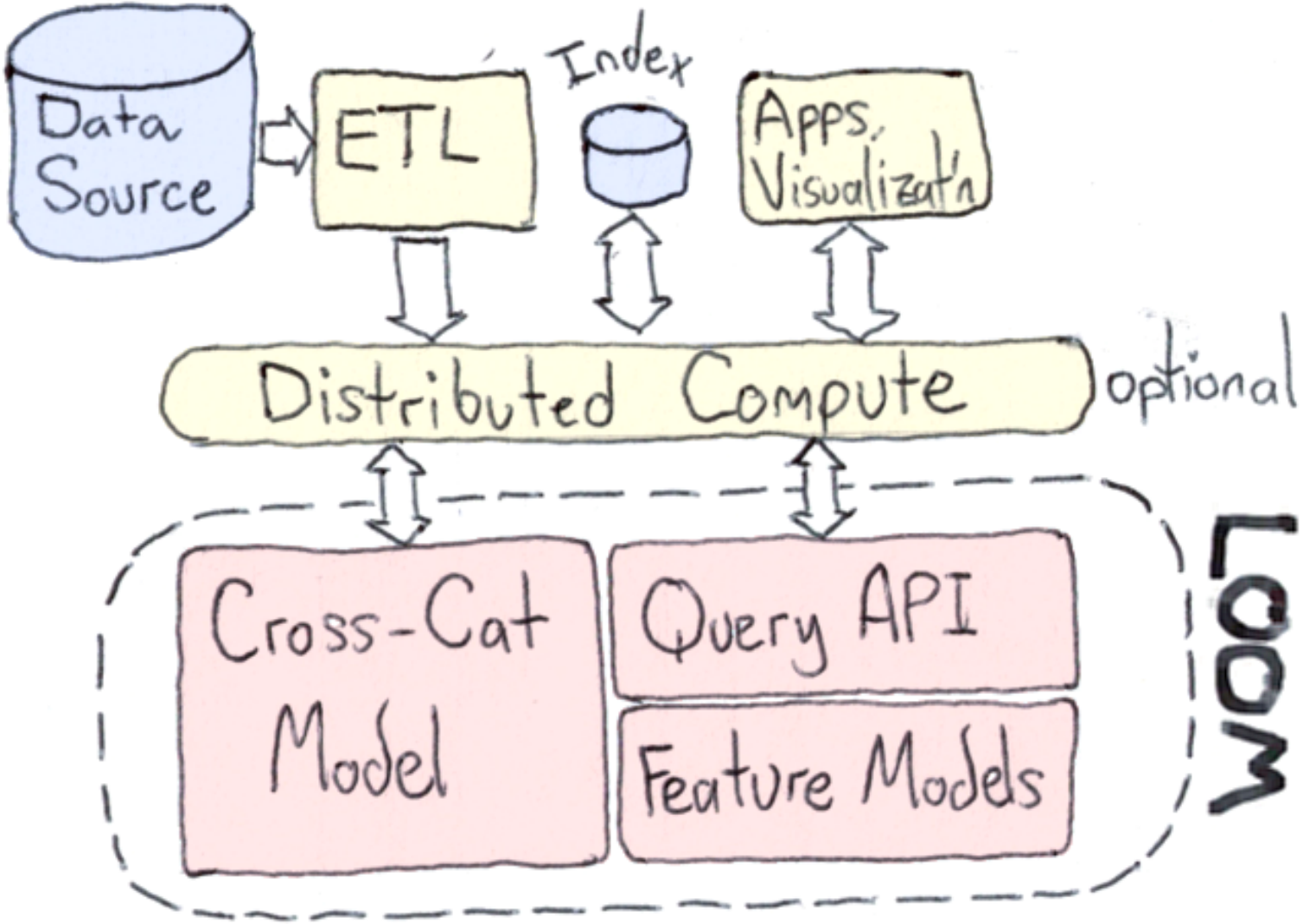
Possible solution: follow database workflow

1. Specify schema
2. Build a statistical index (expensive)
3. Make predictive queries (cheap)

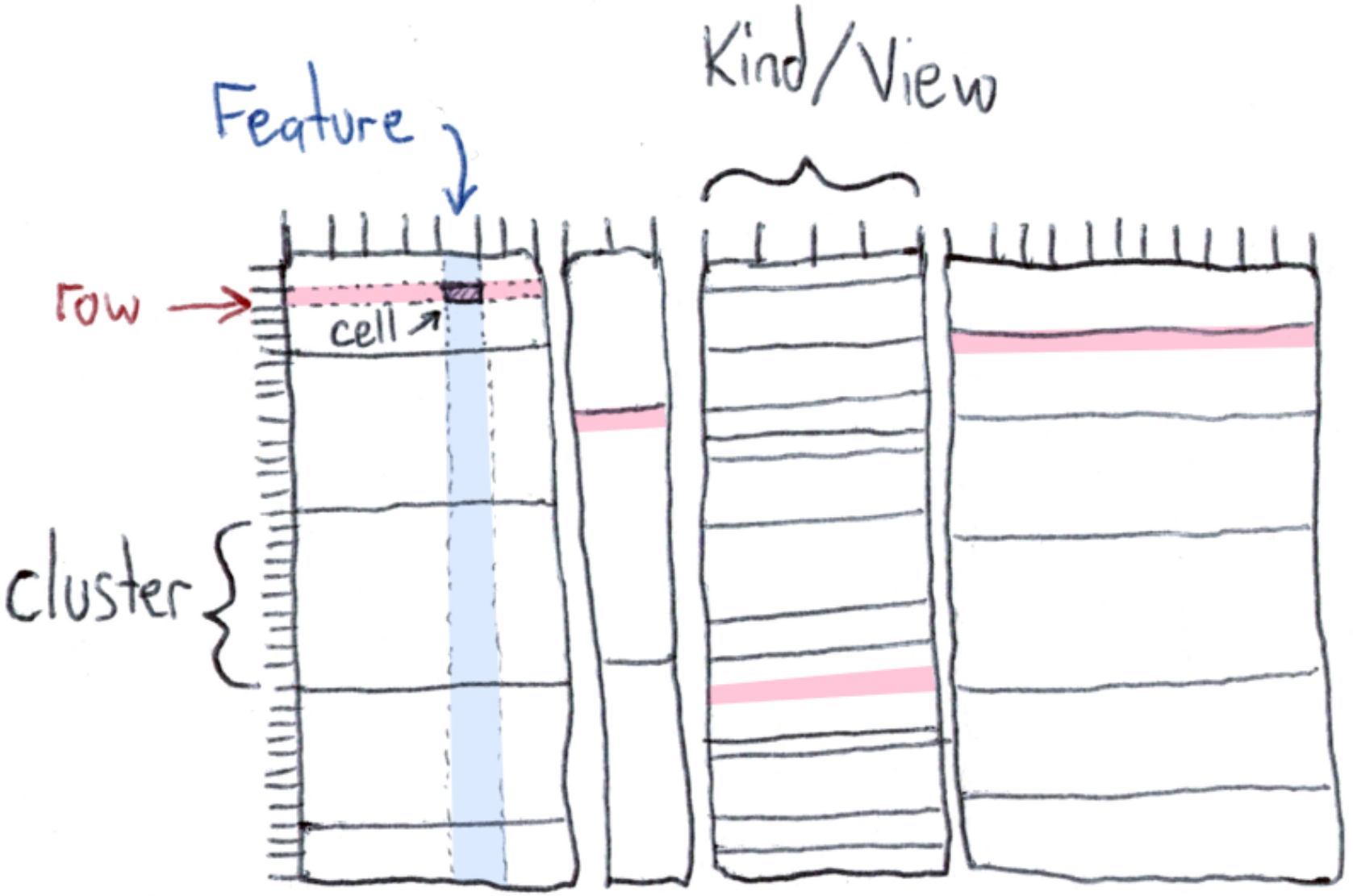
Data Science Workflows



Where Loom fits in



Cross-Categorization clusters features and rows



Loom API

```
# build index
transform(schema.csv, table.csv)
ingest(); infer()

# query
preql.relate() → relation matrix
preql.predict(known_values) → random samples
preql.group(feature) → row clustering
preql.refine(known_values) → relation matrix
preql.support(known_values) → relation matrix
preql.search(known_values) → ranked rows
PreQL.cluster(features, known_values) → ranked rows
```

See https://github.com/priorknowledge/loom

Case Study: Lending Club

https://www.lendingclub.com/info/download-data.action

Lending Club published loan data 2007-2014:

- 376309 borrowers $\times$ 96 features
- Mixed types: quantities, dates, categoricals, text fields
- Some data is missing
- Some quantities are optional, e.g., `collection_recovery_fee`

Case Study: Lending Club

https://www.lendingclub.com/info/download-data.action

Lending Club published loan data 2007-2014:

- 376309 borrowers $\times$ 96 features
- Mixed types: quantities, dates, categoricals, text fields
- Some data is missing
- Some quantities are optional, e.g., `collection_recovery_fee`

Loom `transform()` extracts $\sim$1000 internal features (schema)

- text $\rightarrow$ word absence/presence
- date $\rightarrow$ absolute $\times$ relative $\times$ cyclic
- optional count $\rightarrow$ boolean $\times$ count
- sparse real $\rightarrow$ boolean $\times$ real

Case Study: Lending Club

https://www.lendingclub.com/info/download-data.action

Lending Club published loan data 2007-2014:

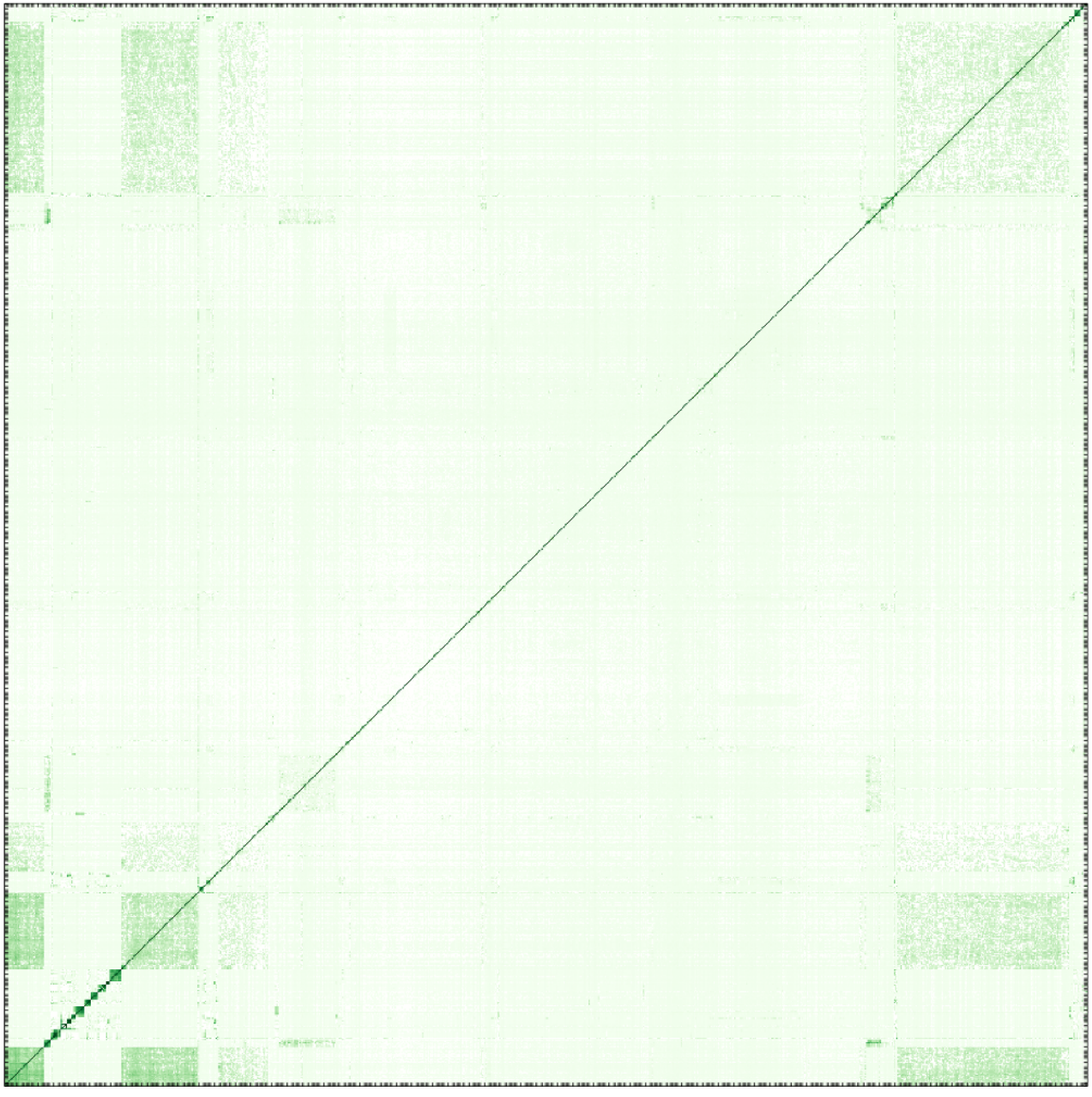
- 376309 borrowers $\times$ 96 features
- Mixed types: quantities, dates, categoricals, text fields
- Some data is missing
- Some quantities are optional, e.g., `collection_recovery_fee`

Loom `transform()` extracts $\sim$1000 internal features (schema)

- text $\rightarrow$ word absence/presence
- date $\rightarrow$ absolute $\times$ relative $\times$ cyclic
- optional count $\rightarrow$ boolean $\times$ count
- sparse real $\rightarrow$ boolean $\times$ real

Loom `infer()` takes $\sim$6 hours on a single big machine.

How are features related? `preql.relate()`



Which features most relate to `loan_status`?

`preql.relate(["loan_status"])`

.759	out_prncp_inv	.036	funded_amnt_inv
.758	out_prncp	.036	funded_amnt
.596	last_credit_pull_d	.033	percent_bc_gt_75
.517	last_pymnt_d	.032	title_nonzero
.212	last_pymnt_amnt	.032	policy_code
.124	recoveries_nonzero	.032	pub_rec_nonzero
.106	collection_recovery_fee_nonzero	.029	pub_rec_bankruptcies_nonzero
.072	total_rec_prncp	.029	pct_tl_nvr_dlq
.071	total_pymnt	.029	mo_sin_old_rev_tl_op
.070	list_d	.028	total_bal_ex_mort
.064	mths_since_recent_bc_nonzero	.026	mo_sin_old_il_acct
.059	total_pymnt_inv	.023	mths_since_recent_inq_nonzero
.053	int_rate	.021	total_rec_late_fee
.044	emp_title_nonzero	.021	term
.037	next_pymnt_d	.018	is_inc_v

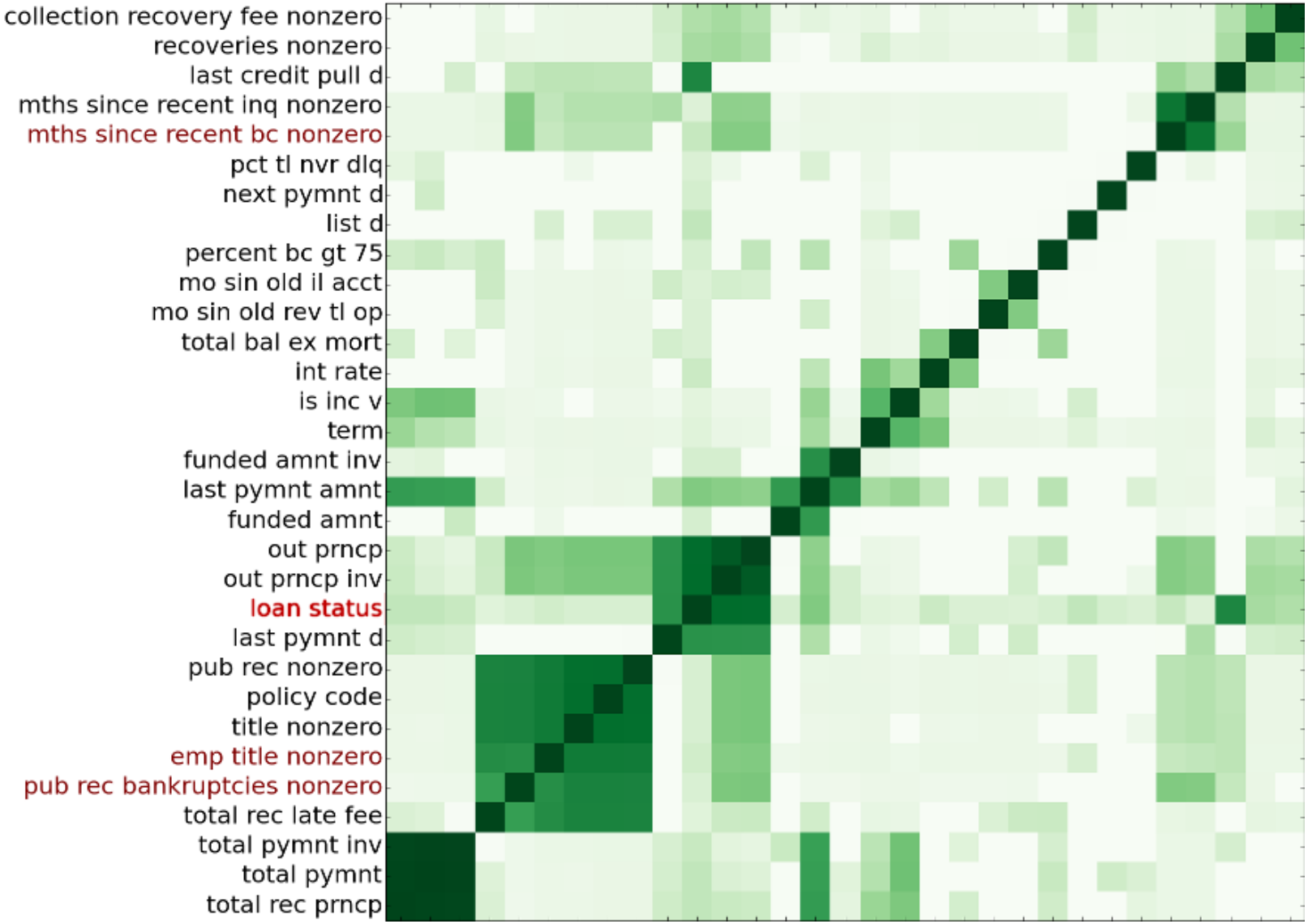
Which features most relate to `loan_status`?

```
preql.relate(["loan_status"])
```

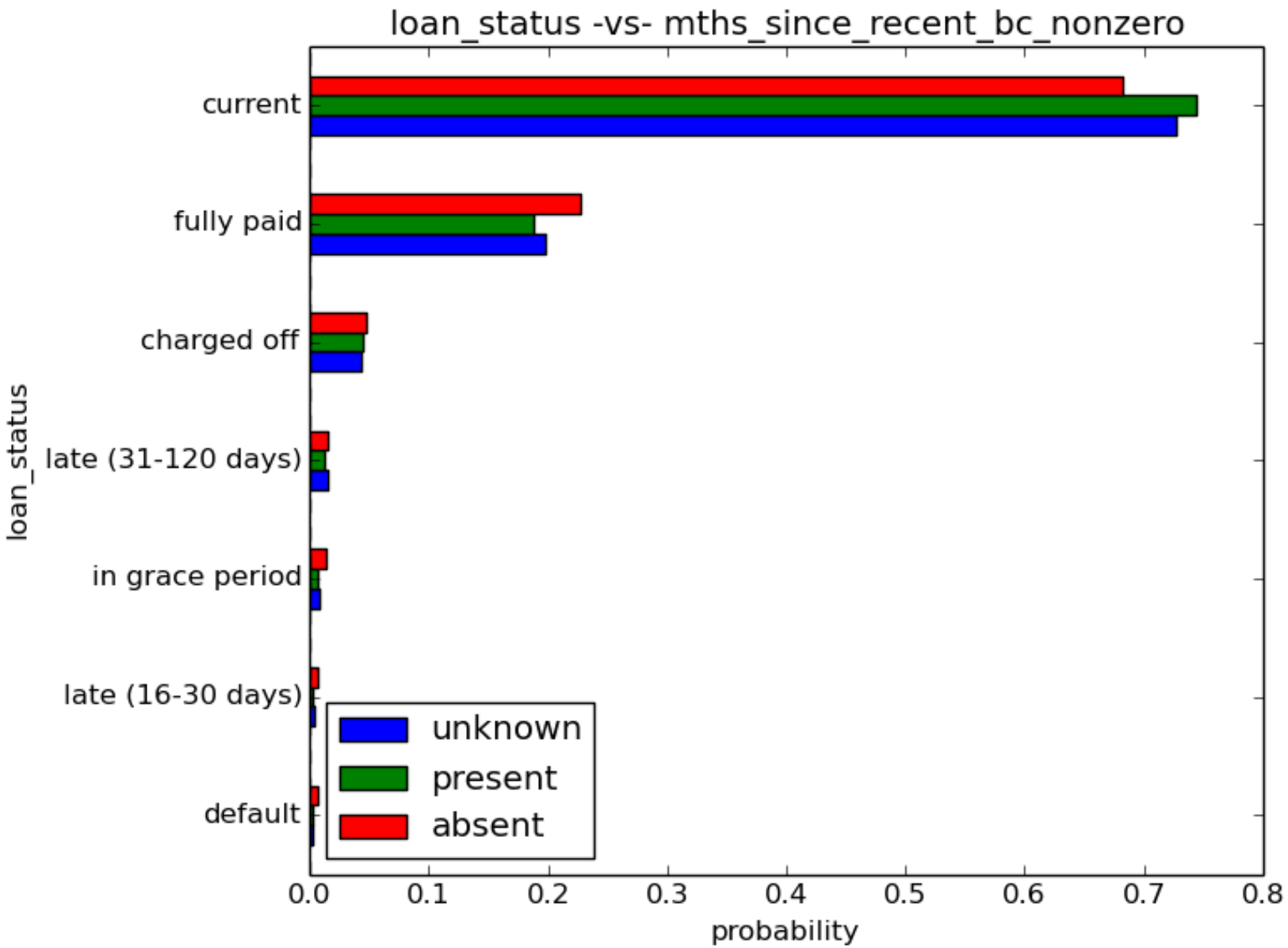
.759	out_prncp_inv	.036	funded_amnt_inv
.758	out_prncp	.036	funded_amnt
.596	last_credit_pull_d	.033	percent_bc_gt_75
.517	last_pymnt_d	.032	title_nonzero
.212	last_pymnt_amnt	.032	policy_code
.124	recoveries_nonzero	.032	pub_rec_nonzero
.106	collection_recovery_fee_nonzero	.029	pub_rec_bankruptcies_nonzero
.072	total_rec_prncp	.029	pct_tl_nvr_dlq
.071	total_pymnt	.029	mo_sin_old_rev_tl_op
.070	list_d	.028	total_bal_ex_mort
.064	mths_since_recent_bc_nonzero	.026	mo_sin_old_il_acct
.059	total_pymnt_inv	.023	mths_since_recent_inq_nonzero
.053	int_rate	.021	total_rec_late_fee
.044	emp_title_nonzero	.021	term
.037	next_pymnt_d	.018	is_inc_v

How are *those* features related?

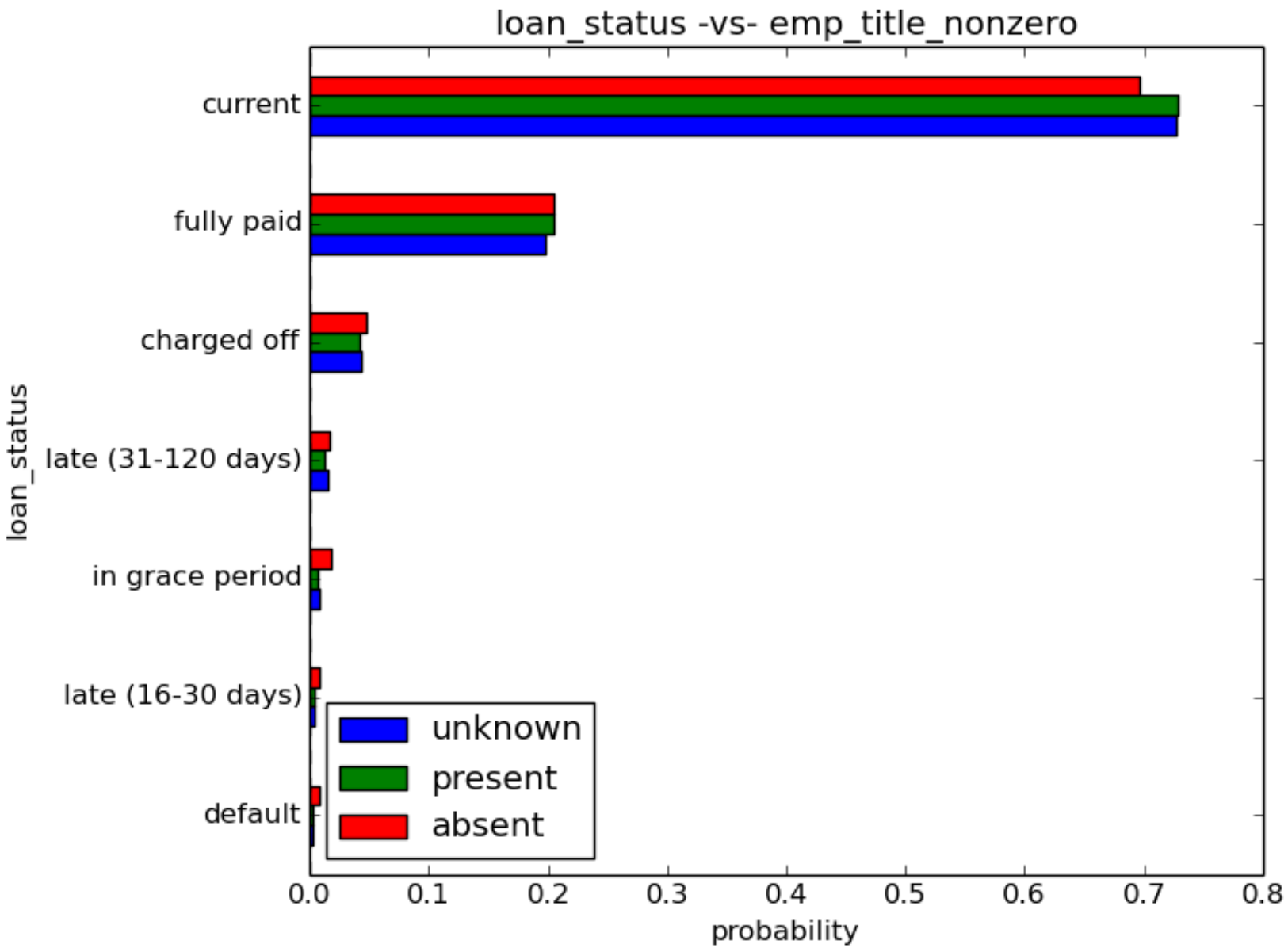
`preql.relate([...], [...])`



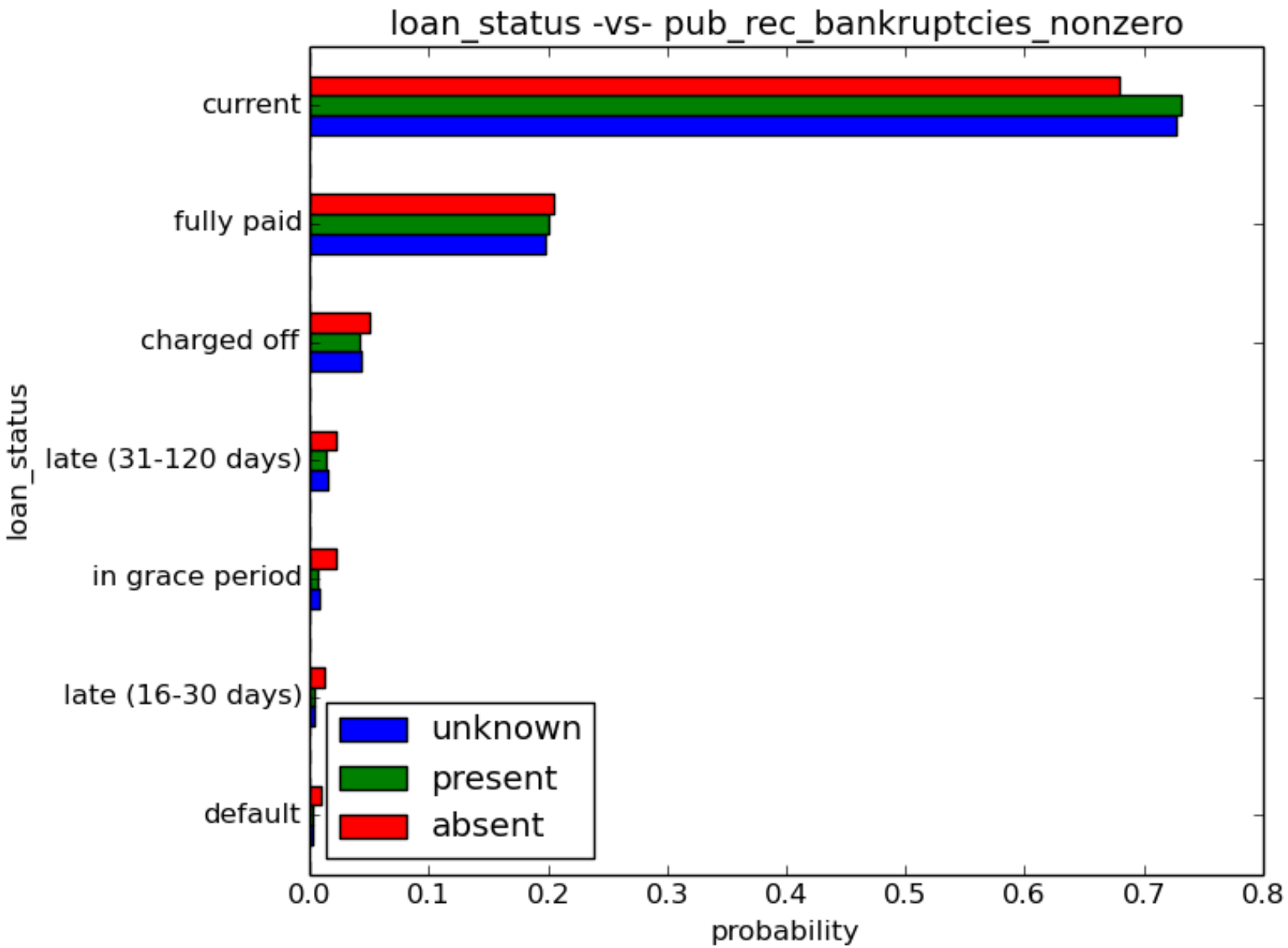
```
preql.predict("loan_status,mths_since_recent_bc_nonzero...")
```



preql.predict("loan_status,emp_title_nonzero...")



```
preql.predict("loan_status,pub_rec_bankruptcies_nonzero...")
```



What can I do with Loom?

Data Scientists:
- analyze your datasets
- contribute new examples

Developers:
- add new feature transforms
- integrate with distributed frameworks, e.g. Spark

Researchers:
- add new conjugate feature models
- extend to IRM for relational data
- hierarchical priors

Further information

Example applications:

https://github.com/priorknowledge/loom

Model: Cross Categorizaion

http://web.mit.edu/vkm/www/shaftokmt11_aprobabilisticmodelofcrosscategorization.pdf

Inference: Subsample Annealing

http://arxiv.org/pdf/1402.5473v1.pdf