Feature Extraction

Tales from the missing manual

Who Am I?

Ted Dunning

Apache Board Member (but not for this talk)
Early Drill Contributor, mentor to many projects
tdunning@apache.org

CTO @ MapR, HPE (how I got to talk to these users)
tdunning@mapr.com

Enthused techie
ted.dunning@gmail.com
@ted_dunning

Summary (in advance)

Accumulate data exhaust if possible

Accumulate features from history

Convert continuous values into symbols using distributions

Combine symbols with other symbols

Convert symbols to continuous values via frequency or rank or Luduan bags

Find cooccurrence with objective outcomes

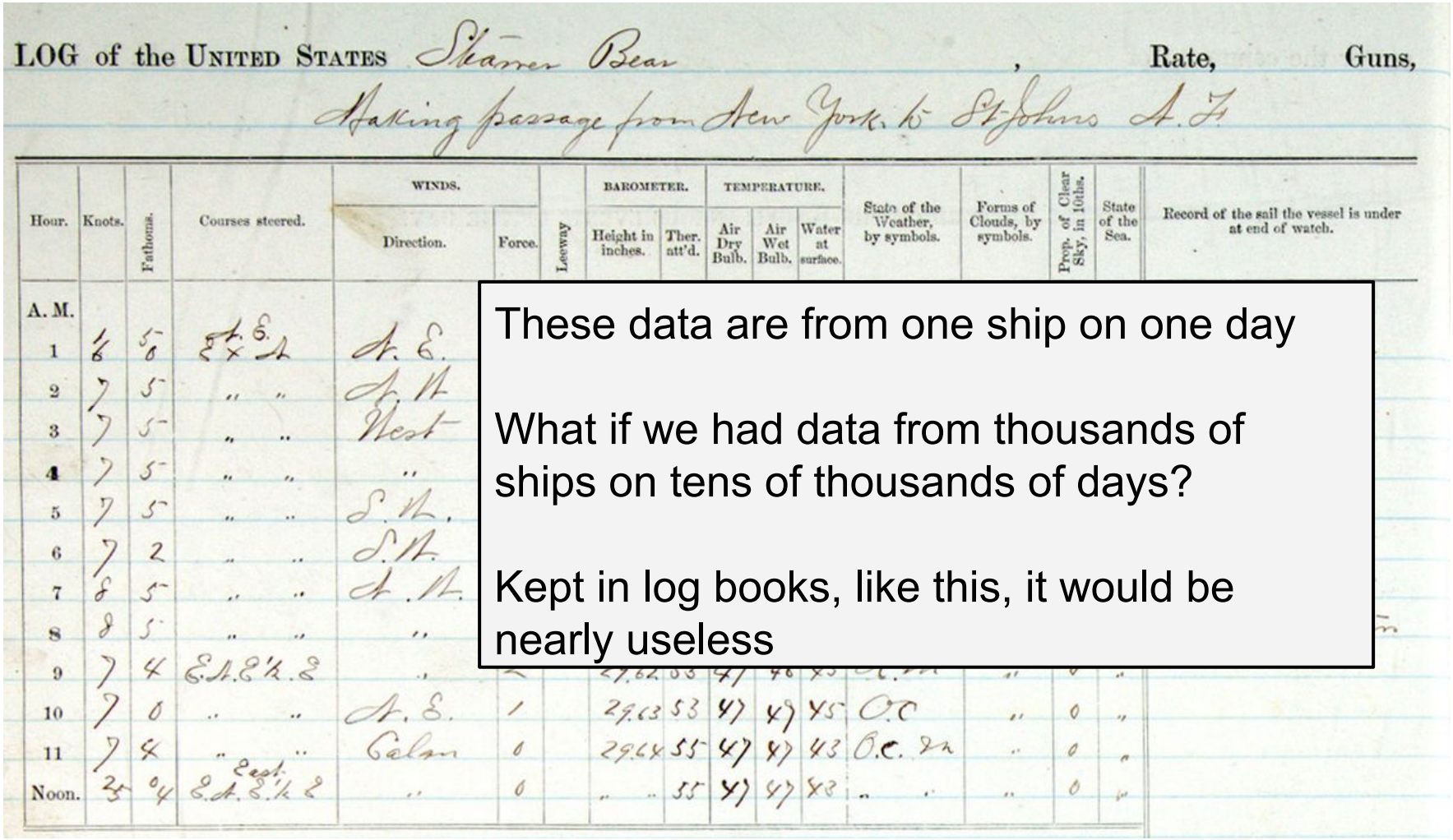
- Bag tainted objects together weighted by total frequency
- Convert symbolic values back to continuous values by accumulating taints

A True-life Data Story

LOG of the UNITED STATES Steamer Bear , Rate, Guns,

Making passage from New York to St Johns N.F.

Hour.	Knots.	Fathoms.	Courses steered.	WINDS. Direction.	WINDS. Force.	Leeway	BAROMETER. Height in inches.	BAROMETER. Ther. att'd.	TEMPERATURE. Air Dry Bulb.	TEMPERATURE. Air Wet Bulb.	TEMPERATURE. Water at surface.	State of the Weather, by symbols.	Forms of Clouds, by symbols.	Prop. of Clear Sky, in 10ths.	State of the Sea.	Record of the sail the vessel is under at end of watch.
A. M.																
1	6	5	E.S.E. E × N	N. E.	2		29.60	53	44	44	45	O.C.m	Nimb	0	S	Steam alone
2	7	5	" "	N. N. E	2		" "	54	45	45	44	" "	"	0	"	
3	7	5	" "	West	1		" "	"	"	"	"	" "	"	0	"	
4	7	5	" "	"	1		" "	"	"	"	"	O.C.f	"	0	"	
5	7	5	" "	S. W.	1		" "	"	45	46	41	" "	"	0	"	
6	7	2	" "	S. W.	1		29.62	52	45	46	41	" "	"	0	"	
7	8	5	" "	N. W.	1		29.61	51	45	45	43	" "	"	0	"	
8	8	5	" "	"	1		" "	51	49	46	45	" "	"	0	"	7.25 Sail + Steam
9	7	4	E.N.E ½ E	"	2		29.62	53	47	46	45	O.C.m	"	0	"	
10	7	0	" "	N. E.	1		29.63	53	47	47	45	O.C	"	0	"	
11	7	4	" "	Calm	0		29.64	55	47	47	43	O.C. ½	"	0	"	
Noon.	25	04	E.N.E ½ E (East)	"	0		" "	55	47	47	43	" "	"	0	"	



LOG of the UNITED STATES Steamer Bear , Rate, Guns,

Making passage from New York to St Johns N. F.

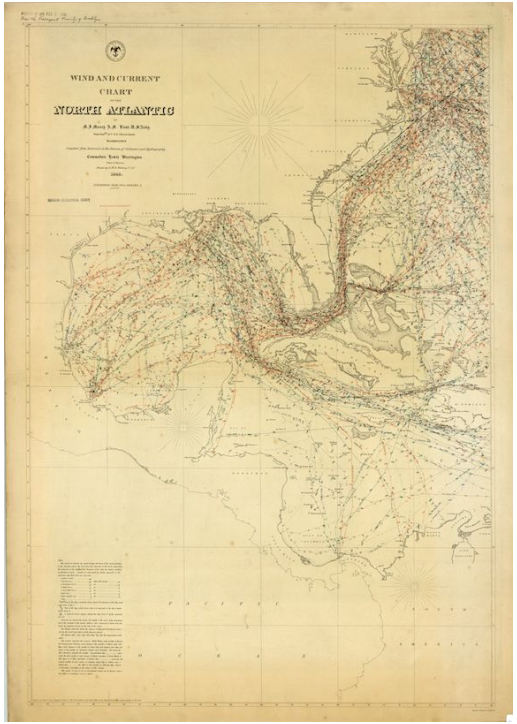
Hour.	Knots.	Fathoms.	Courses steered.	WINDS. Direction.	WINDS. Force.	Leeway	BAROMETER. Height in inches.	BAROMETER. Ther. att'd.	TEMPERATURE. Air Dry Bulb.	TEMPERATURE. Air Wet Bulb.	TEMPERATURE. Water at surface.	State of the Weather, by symbols.	Forms of Clouds, by symbols.	Prop. of Clear Sky, in 10ths.	State of the Sea.	Record of the sail the vessel is under at end of watch.
A. M.																
1	6	50	E.S.E. E × N	N. E.												
2	7	5	" "	N. N.												
3	7	5	" "	West												
4	7	5	" "	"												
5	7	5	" "	S. W.												
6	7	2	" "	S. W.												
7	8	5	" "	N. W.												
8	8	5	" "	"												
9	7	4	E.N.E ½ E	"			29.62	53	47	46	43	O.C.	"	0	"	
10	7	0	" "	N. E.	1		29.63	53	47	47	45	O.C	"	0	"	
11	7	4	" "	Calm	0		29.64	55	47	47	43	O.C. ½	"	0	"	
Noon.	25	04	E.S.E. ½ E (East)	"	0		"	55	47	47	43	"	"	0	"	

These data are from one ship on one day

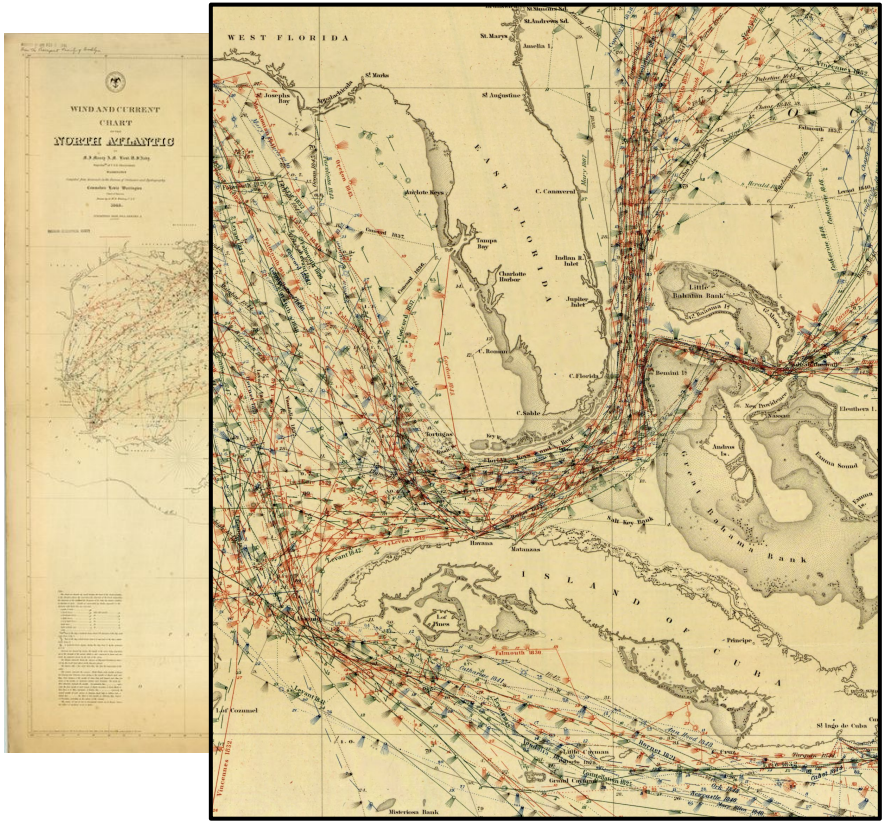
What if we had data from thousands of ships on tens of thousands of days?

Kept in log books, like this, it would be nearly useless

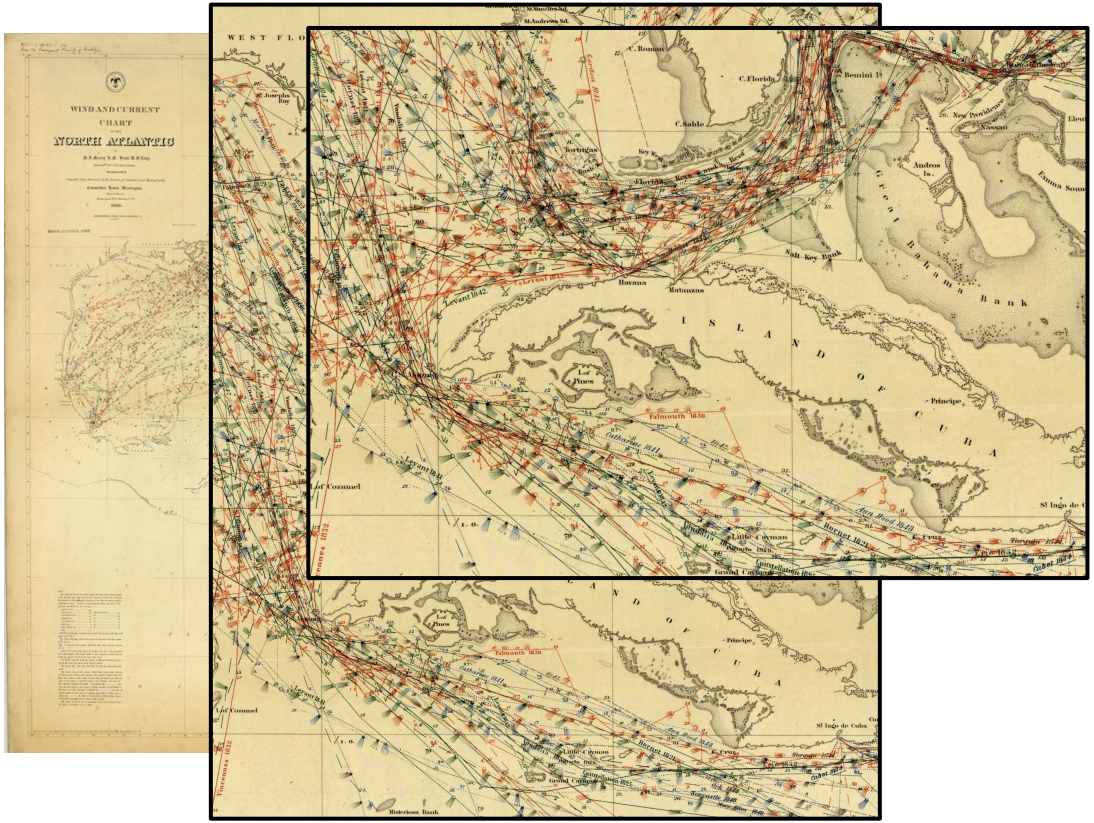
19th Century Big Data



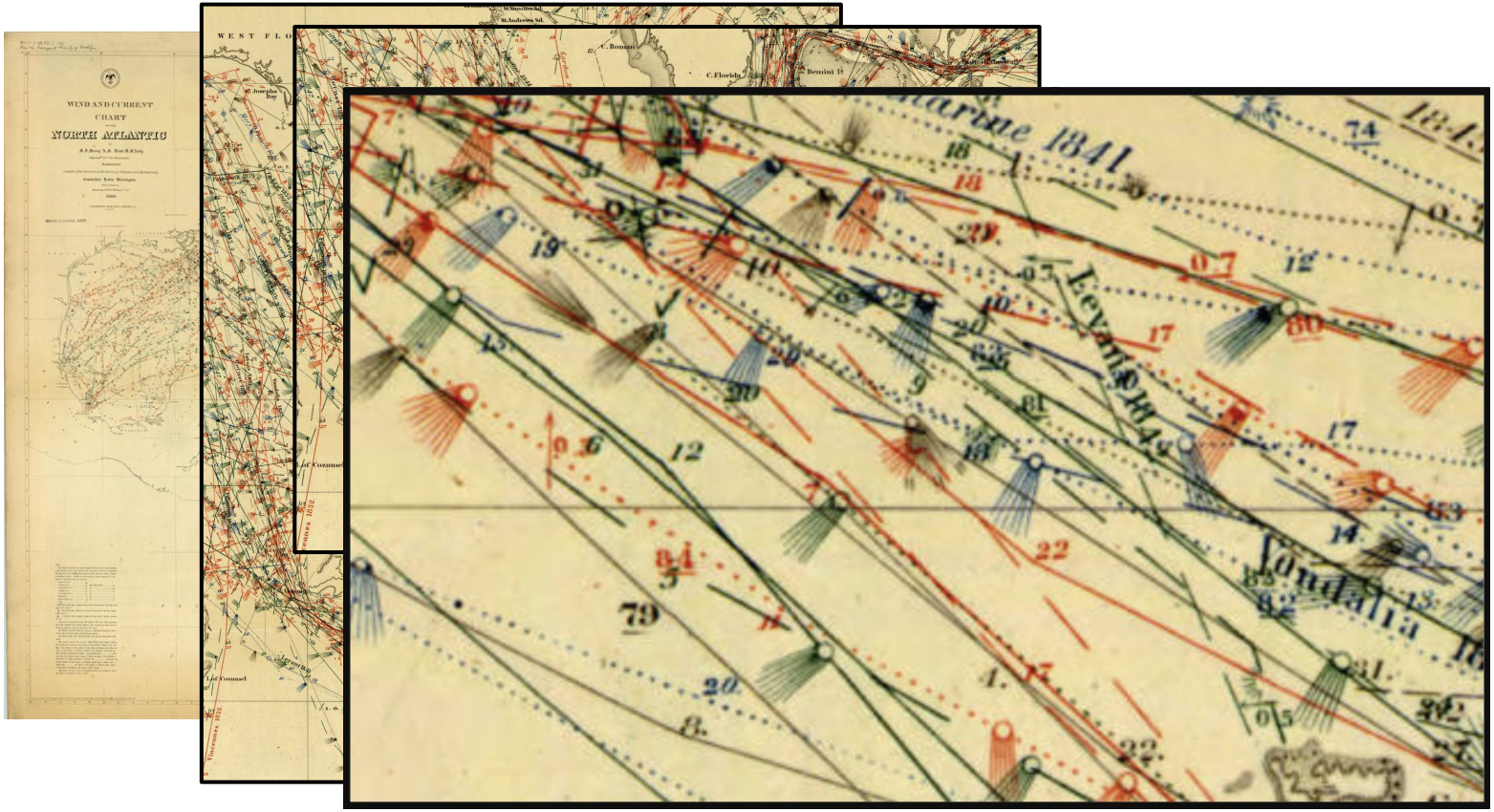
19th Century Big Data



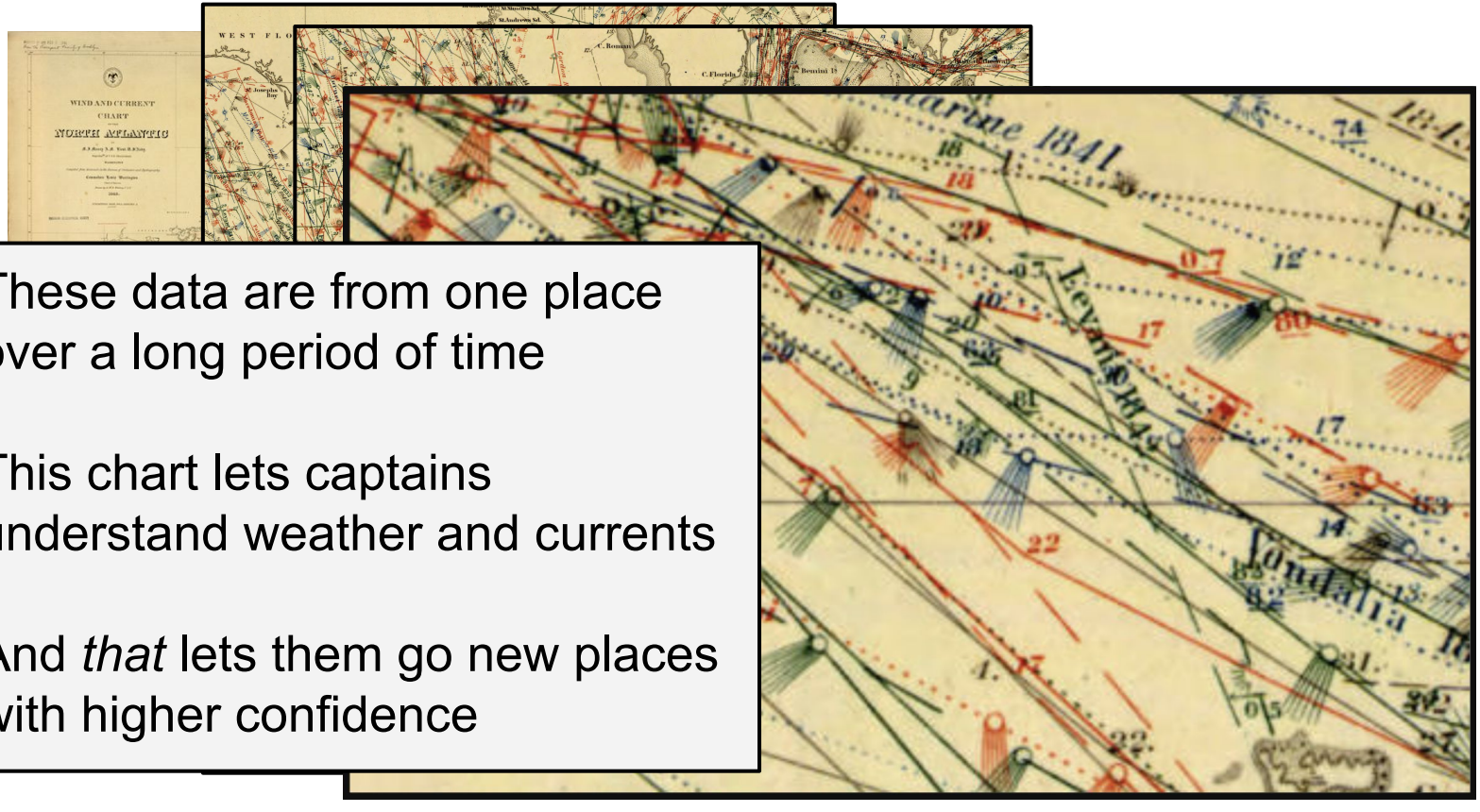
19th Century Big Data



19th Century Big Data



19th Century Big Data



These data are from one place over a long period of time

This chart lets captains understand weather and currents

And *that* lets them go new places with higher confidence

Same data,
different perspective,
massive impact

But it isn't just prettier

A Fake Data Story

Perspective Can Be Key

Given:

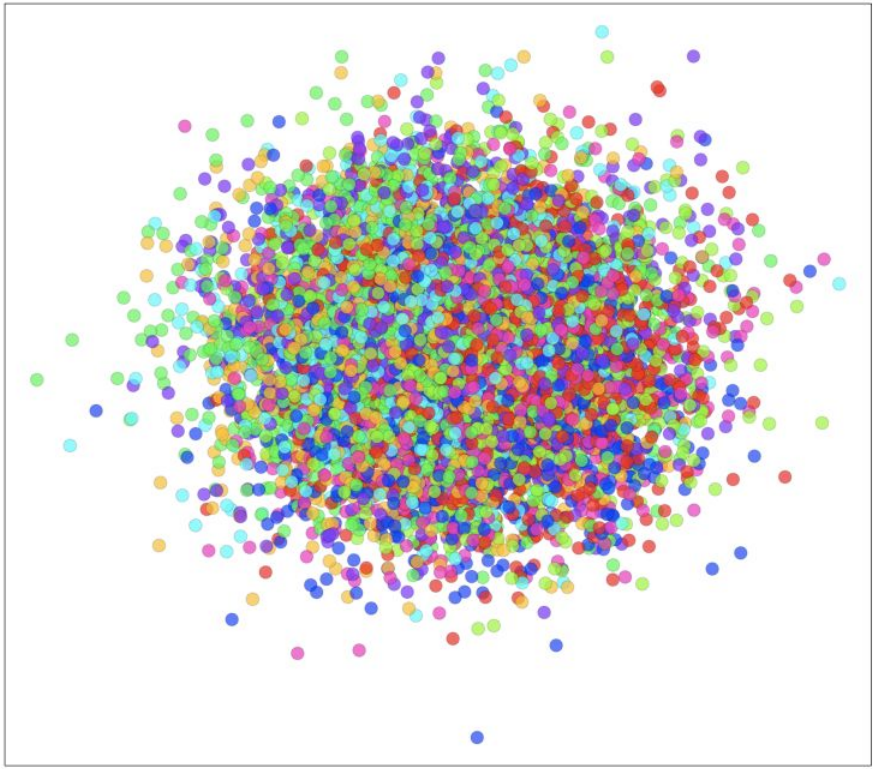
100 real-valued features on colored dots

Desired:

A model to predict colors for new dots based on the features

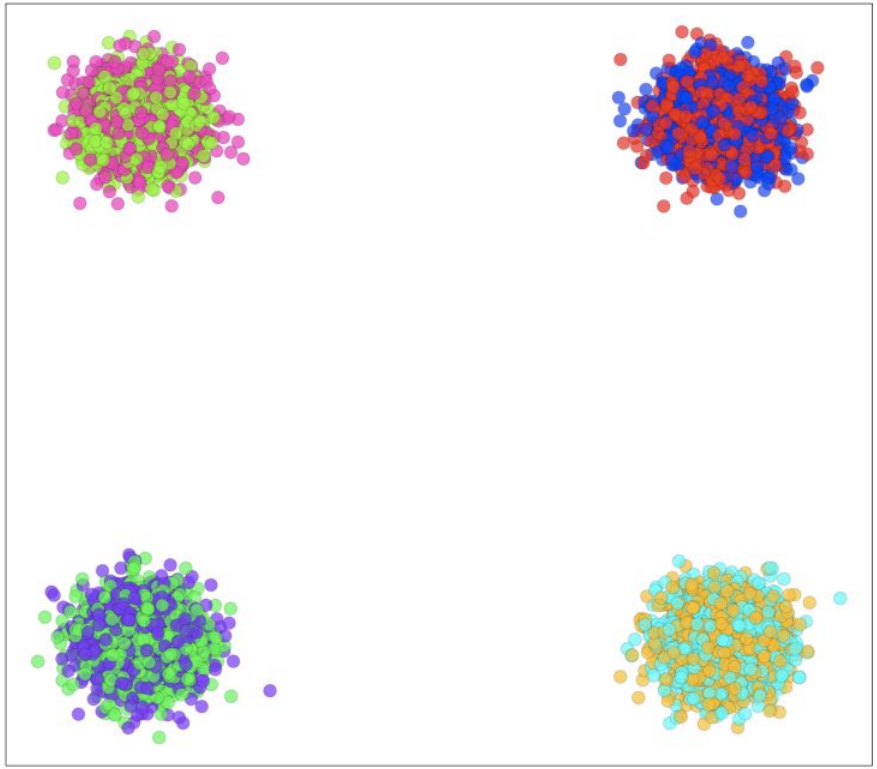
Evil assumption (for discussion):

No privileged frame of reference (commonly done in physics)



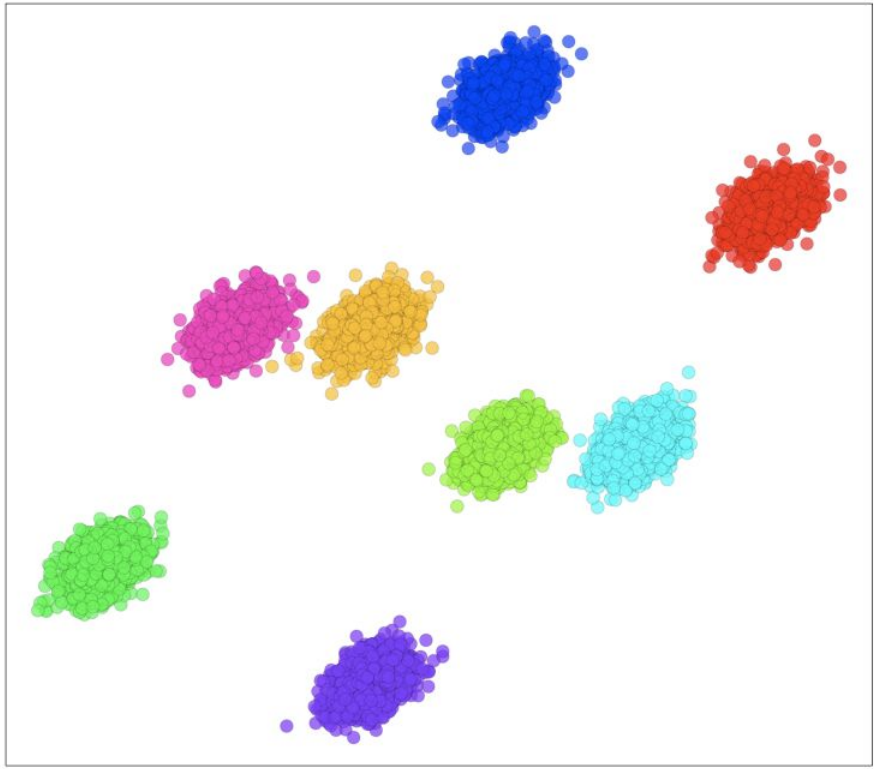
These data points appear jumbled

But this is largely due to our perspective



Taking just the first two coordinates, we see more order

But there is more to be had



Combining multiple coordinates completely separates the colors

How can we know to do this just based on the data?

Feature extraction is how we encode domain expertise

A Story of Fake Data (that eventually turned into real data)

Background

Let's simulate a data skimming attack

Transactions at a particular vendor increase subsequent rate of fraud

Background rate of fraud is high

Fraud does not occur at increased rate at skimming locations

We want to find the skimming locations

More Details

Data is generated using a behavioral model for consumers

Transactions generated with various vendors at randomized times

Transactions are marked as fraud randomly at a baseline rate

Transacting with a skimmer increases fraud rate for a consumer to increase for some period of time

Modeling Approach

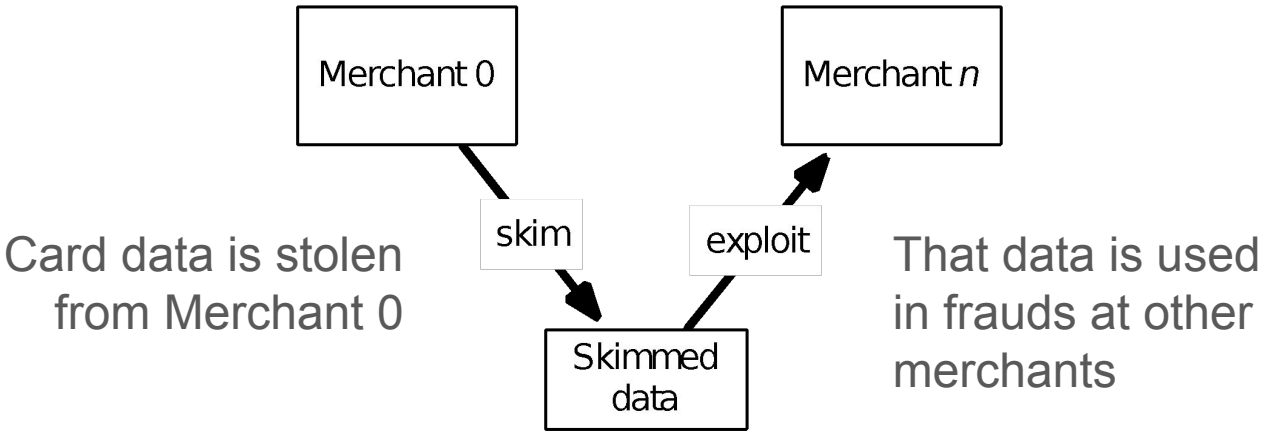
For all transactions

- If fraud, increment fraud counter for all merchants in 30 day history
- If non-fraud, increment non-fraud counter for all merchants in 30 day history

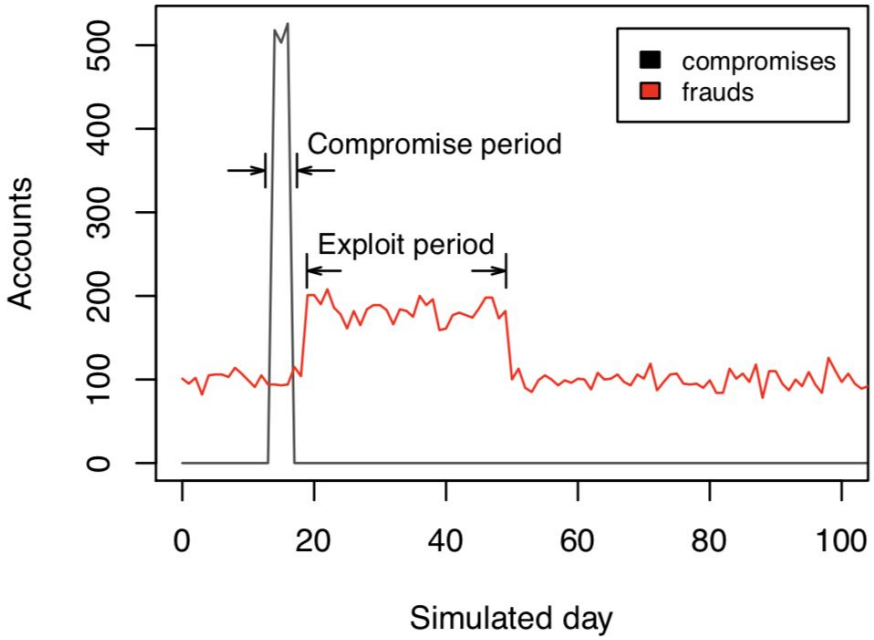
For all vendors

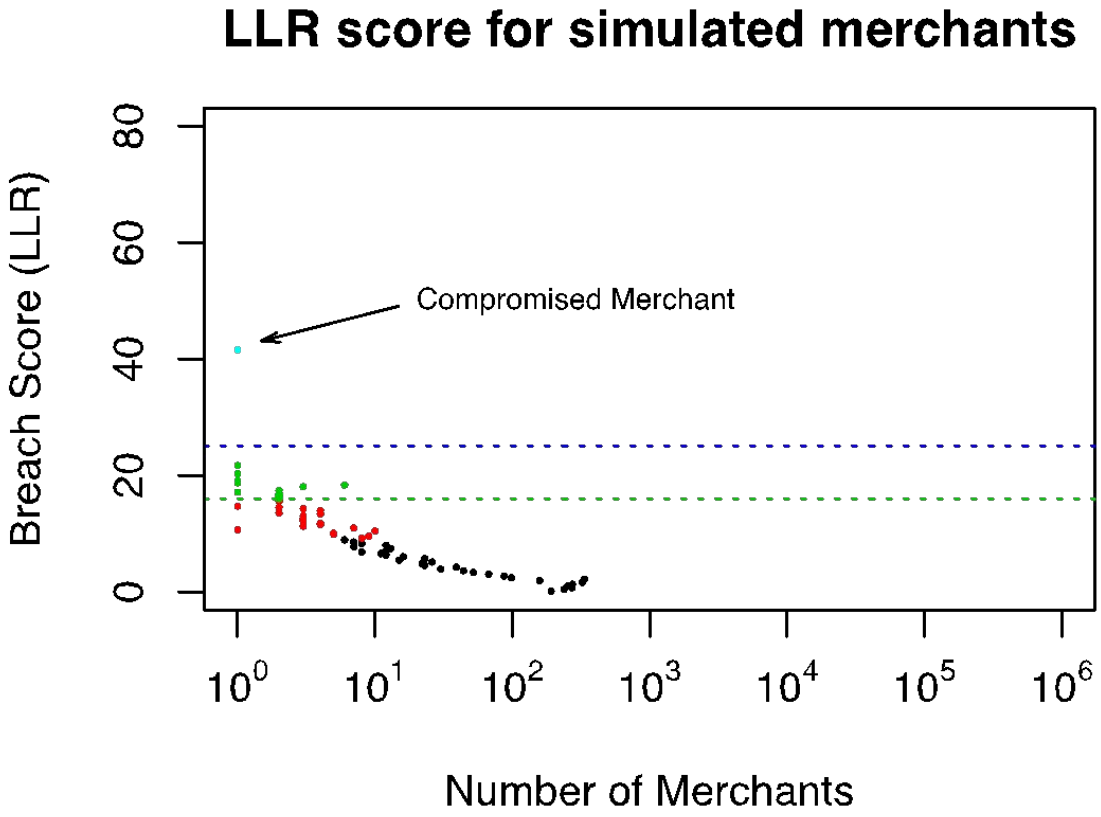
- Form contingency table, compute G-score

Example 2 - Common Point of Compromise

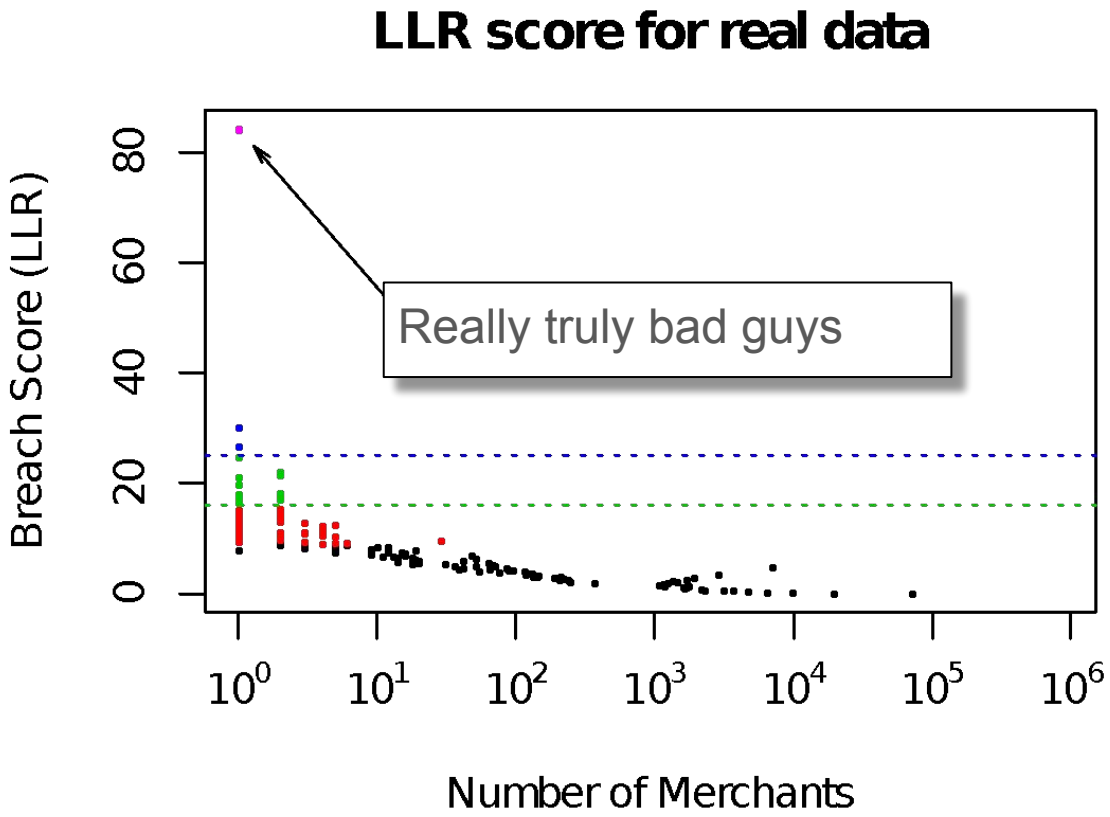


Simulation Setup




LLR score for simulated merchants
Breach Score (LLR)
0
20
40
60
80
Compromised Merchant
10^0
10^1
10^2
10^3
10^4
10^5
10^6
Number of Merchants

What about real data from real bad guys?


LLR score for real data
Really truly bad guys
Breach Score (LLR)
0
20
40
60
80
10^0
10^1
10^2
10^3
10^4
10^5
10^6
Number of Merchants

We can use cooccurrence to find bad actors

Cooccurrence also finds "indicators" to be combined as features

A True Story

Background

Credit card company wants to find payment kiting where a bill is paid, credit balance is cleared, and then the payment bounces

We have:

3 years of transaction histories + payment history + payment final status

We want:

A model that can predict whether a payment will bounce

More Details

A charge transaction includes:

Date, time, account #, charge amount, vendor id, industry code, location code

Account data includes:

Name, address, account number, account age, account type

Payment transaction includes:

Date, time, account #, type (payment, update), amount, flags

Non-monetary transaction includes:

Date, time, account #, type, flags, notes

Modeling Approach

Split data into first two years (training), last year (test)

For each payment, collect previous 90 days of transactions, ultimate status

Standard transaction features:

Number of transactions, amount of transactions, average transaction amount, recent bad payment, time since last transaction, overdue balance

Special features:

Flagged vendor score, flagged vendor amount score

Standard Features

For many features, we simply take the history of each account and accumulate features or reference values

Thus "current transaction / average transaction"

Or "distance to previous transaction location / time since previous transaction"

Some of these historical features could be learned if we gave the history to a magical learning algorithm

But suggesting these features is better when training data costs time and money

Special Features

We can also accumulate characteristics of vendors

In fact, our data is commonly composed of actions with a subject, verb and object

- The subjects are typically consumers and we focus on them
- But the objects are worth paying attention to as well

We can analyze the history of objects to generate associated features

- Frequency
- Distributions
- Cooccurrence taints

Symbol Frequency as a Feature

Consider an image that is part of your web page

What domains reference these images? (mostly yours, of course)

Any time you see a rare (aka new) domain, it is a thing

We don't know what kind of thing, but it is a thing

Tainted Symbol History as a Feature

We can mark those objects based on their presence in histories with other events

AKA cooccurrence with fraud | charge-off | machine failure | ad-click

Now we can accumulate a measure of how many such tainted objects are in a user history

Which cars are involved in accidents?

Which browser versions are used by fraudsters?

Which OS versions of software crashes?

Key Winning Feature

For this model, the feature that was worth over $5 million to the customer was formed as a combination of distribution and cooccurrence

Start with a composite symbol

<merchant-id> / <location-code> / <transaction-size-decile>

Find symbols associated with kiting behavior using cooccurrence

These identified likely money laundering paths

Combined with young accounts, payment channel => over 90% catch rate

Combine techniques to find killer features

Combine techniques to find killer features

Killer features are the ones your competitors aren't using (yet)

Door Knockers

Background

You have a security system that is looking for attackers

It finds naive attempts at intrusion

But the attackers are using automated techniques to morph their attacks

They ***will*** evade your detector eventually

How can you stop them?

Modeling Approach

Failed attacks can be used as a taint on

- Source IP
- User identities
- User agent
- Browser versions
- Header signatures

If you can do cooccurrence in ***real-time*** you can build fast adapting features

The fast adaptation of the attacker becomes a weakness rather than a strength

High attack activity provides good surrogate target variables

Data Exhaust

Background

Everybody knows that it is important to turn off any logging on secondary images and scripts

The resulting data would be "too expensive" to store and analyze

This was true in 2004

Spot the Important Difference?

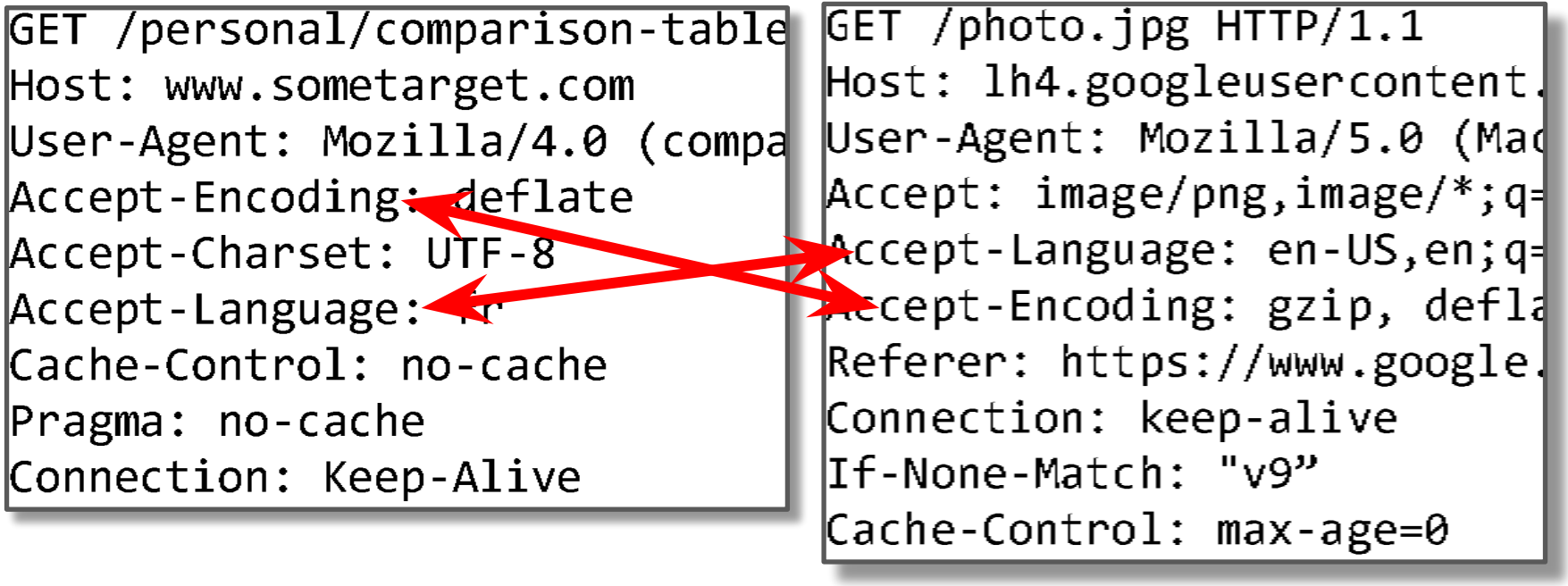
```
GET /personal/comparison-table
Host: www.sometarget.com
User-Agent: Mozilla/4.0 (compa
Accept-Encoding: deflate
Accept-Charset: UTF-8
Accept-Language: fr
Cache-Control: no-cache
Pragma: no-cache
Connection: Keep-Alive
```

Attacker request

```
GET /photo.jpg HTTP/1.1
Host: lh4.googleusercontent.
User-Agent: Mozilla/5.0 (Mac
Accept: image/png,image/*;q=
Accept-Language: en-US,en;q=
Accept-Encoding: gzip, defla
Referer: https://www.google.
Connection: keep-alive
If-None-Match: "v9”
Cache-Control: max-age=0
```

Real request

Spot the Important Difference?



Attacker request

Real request

Why are Experts Necessary

You could probably learn a whiz-bang LSTM neural network model for headers

That model might be surprised by change in order

It would *definitely* detect too few headers or lower case headers

But it would take a lot of effort, tuning and expertise to build

And your security dweeb will spot 15 things to look for in 10 minutes

You pick (I pick both)

Collecting data exhaust turns the tables on attackers

Summary

Accumulate data exhaust if possible

Accumulate features from history

Convert continuous values into symbols using distributions

Combine symbols with other symbols

Convert symbols to continuous values via frequency or rank or Luduan bags

Find cooccurrence with objective outcomes

- Bag tainted objects together weighted by total frequency
- Convert symbolic values back to continuous values by accumulating taints

The Story isn't Over

Let's work together on examples of this:

github.com/tdunning/feature-extraction

Several feature extraction techniques are already there, more are coming

You can help!

For data generation, see also

github.com/tdunning/log-synth

Who Am I?

Ted Dunning

Apache Board Member (but not for this talk)
Early Drill Contributor
tdunning@apache.org

CTO @ MapR, HPE (how I got to talk to these users)
tdunning@mapr.com

Enthused techie
ted.dunning@gmail.com
@ted_dunning

Book signing
HPE booth
at 3:30