Why security data science matters and how it's different: pitfalls and promises of data science based breach detection and threat intelligence

Joshua Saxe, Invincea Labs

Work presented in this talk contains significant contributions from Alex Long, David Slater, Giacomo Bergamo, Konstantin Berlin, and Robert Gove

Presentation Overview

- Session 1:
  - What is data science and why does it matter for defending our networks?
  - Data science at a glance – machine learning and visualization
  - Unique problems in security data science
- Session 2:
  - Three case studies from Invincea Labs
    - Machine learning based malware detection – going beyond signatures
    - Machine learning based prediction of malware family "hockey stick" growth – predicting malware's future
    - Visualization and machine learning for threat intelligence – malware clustering and automated profiling

What is data science?

invincea™

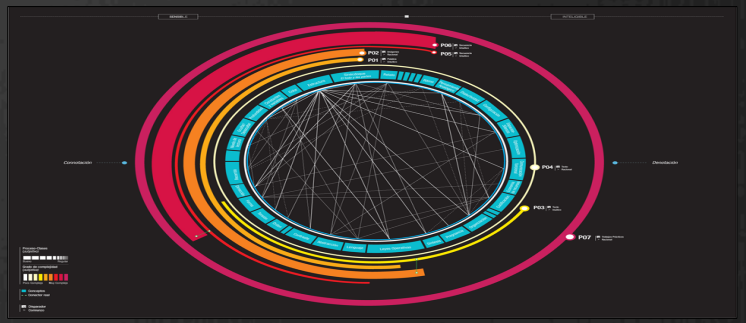
Can be a difficult question amidst all of the marketing hype …

… but at a high level, data science consists of three technical areas …

1. Scalable data storage technologies

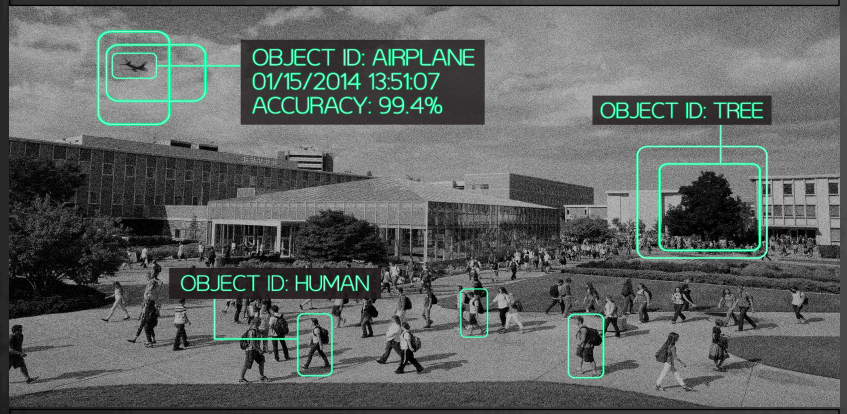
2. Machine learning / scalable analytics

3. Data visualization / decision support

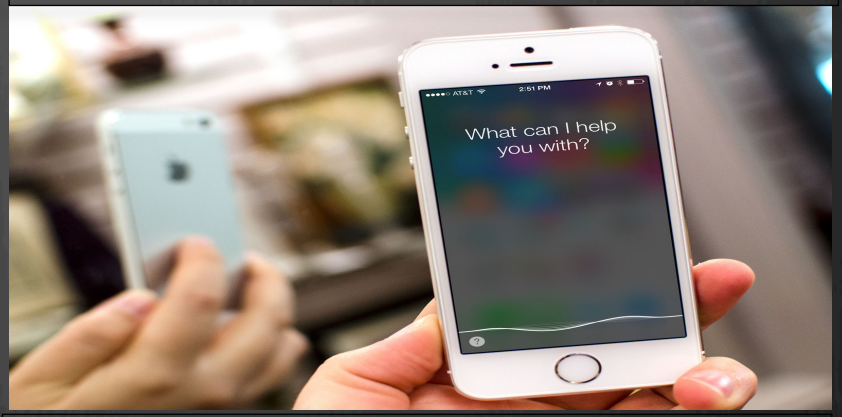


… joined with a growing number of application domains

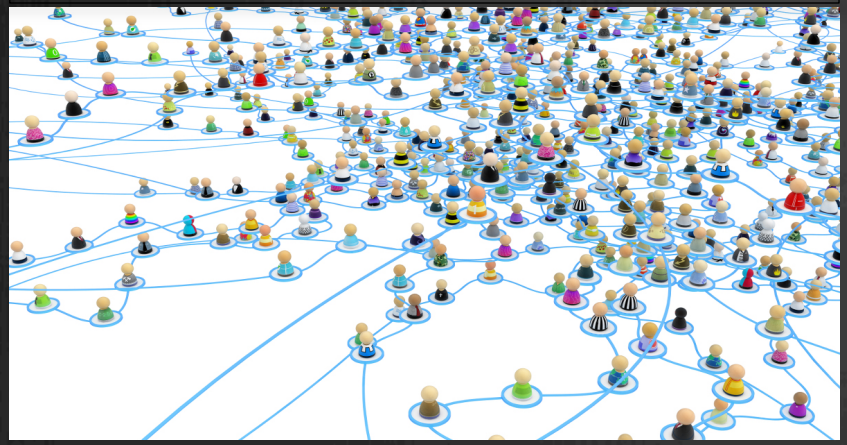
Computer vision



Voice recognition / natural language processing



Social network analysis



Robotics

…

invincea™

Data Science Area 1:

Scalable storage technologies

The new (and old) landscape of storage tech

Traditional / relational

New / non-relational

Document Database	Graph Databases
Couchbase, MarkLogic, mongoDB	Neo4j, InfiniteGraph (The Distributed Graph Database)
Wide Column Stores	**Key-Value Databases**
redis, amazon DynamoDB, AEROSPIKE, riak	accumulo, HYPERTABLE INC, Cassandra, APACHE HBASE, Amazon SimpleDB

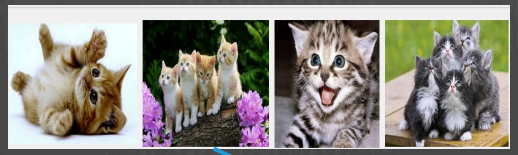
@cloudtxt http://www.aryannava.com

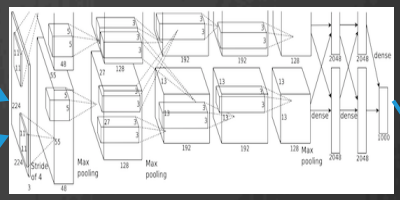
Data Science Area 2:
Machine learning

The machine learning workflow

1. Obtain training examples
2. Extract features from data
3. Train machine learning model
4. Use model to make predictions about new data

Young cats training data

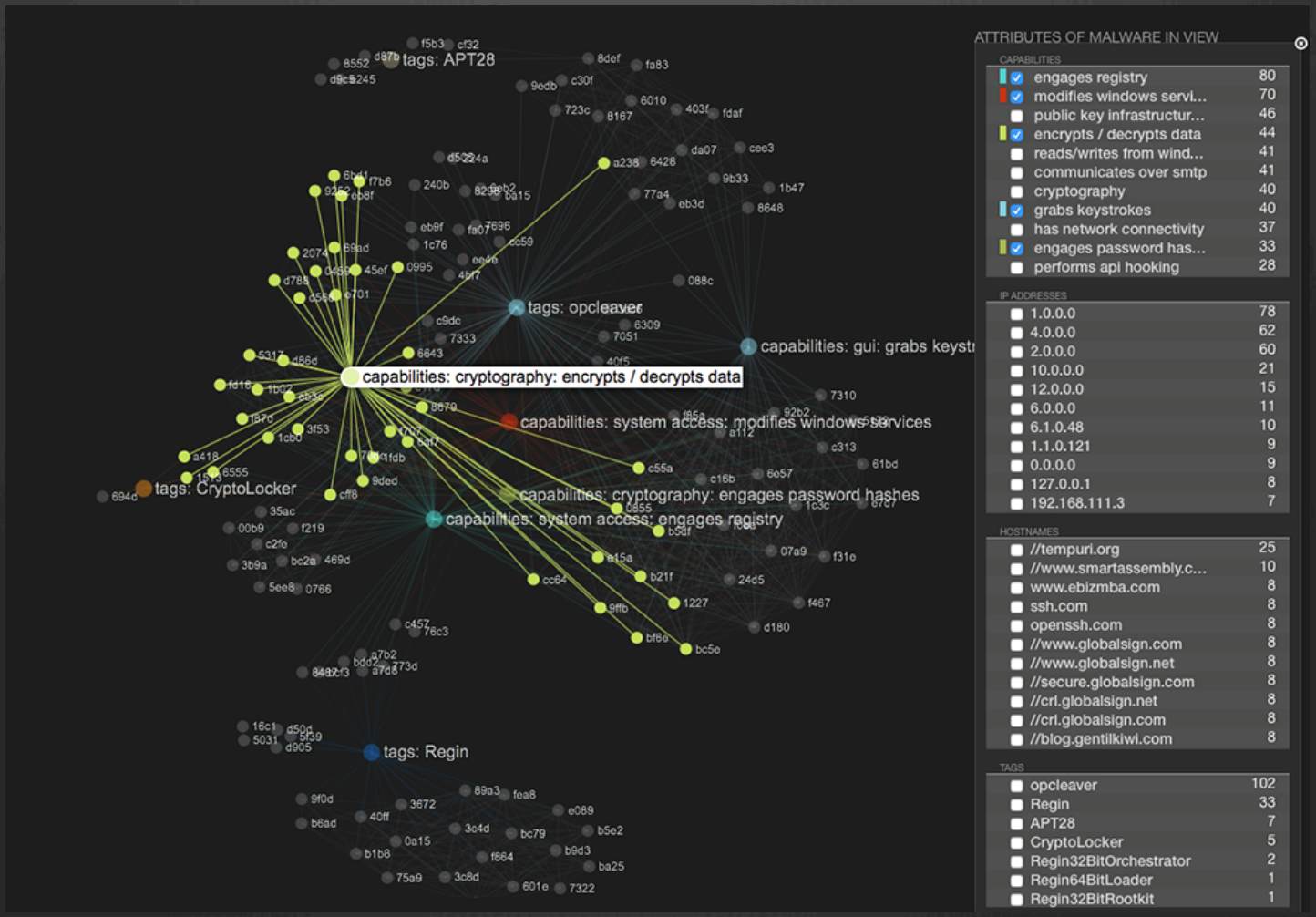




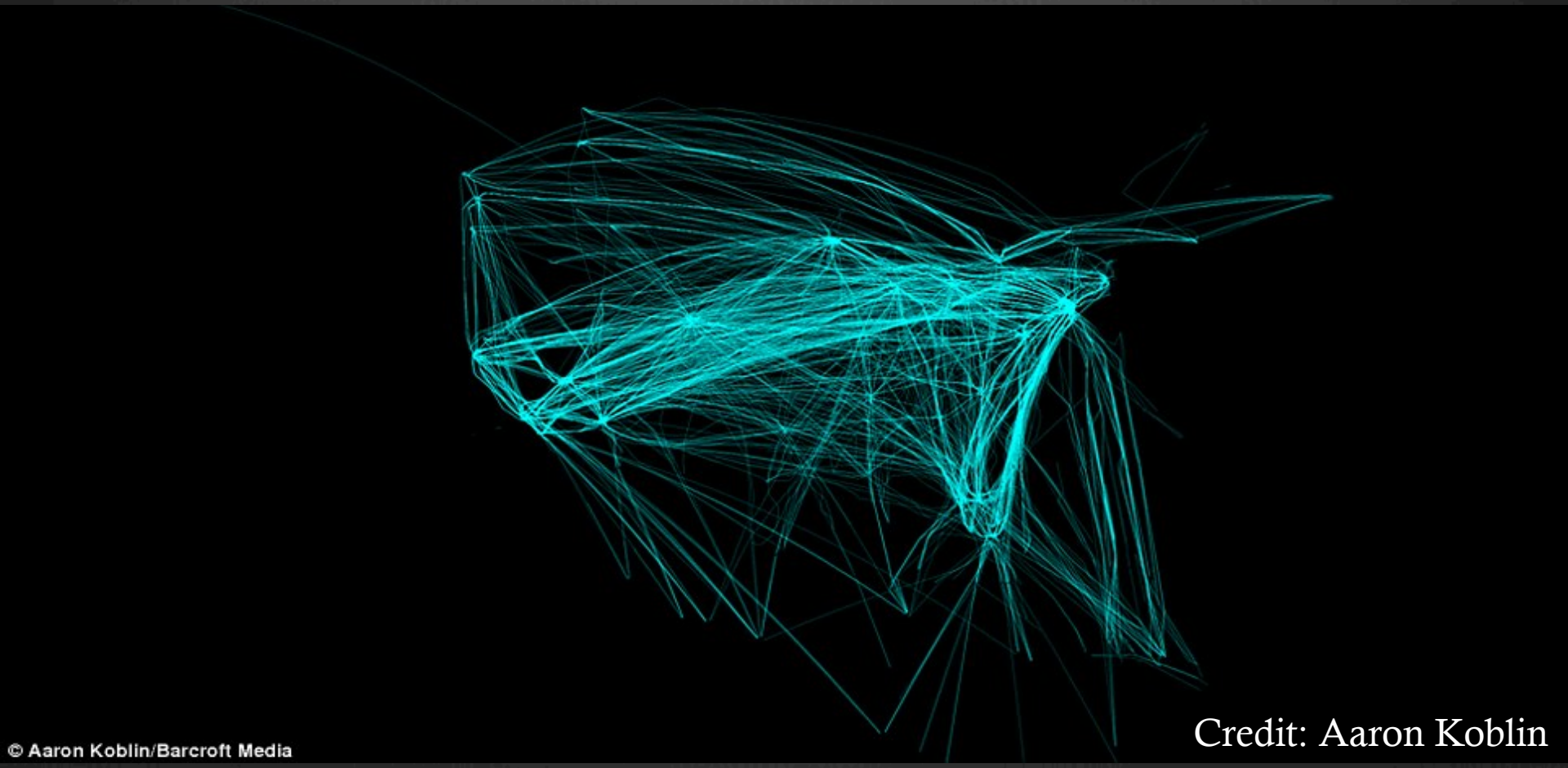
Old cat!

Old cats training data

Data Science Area 3:
Data visualization

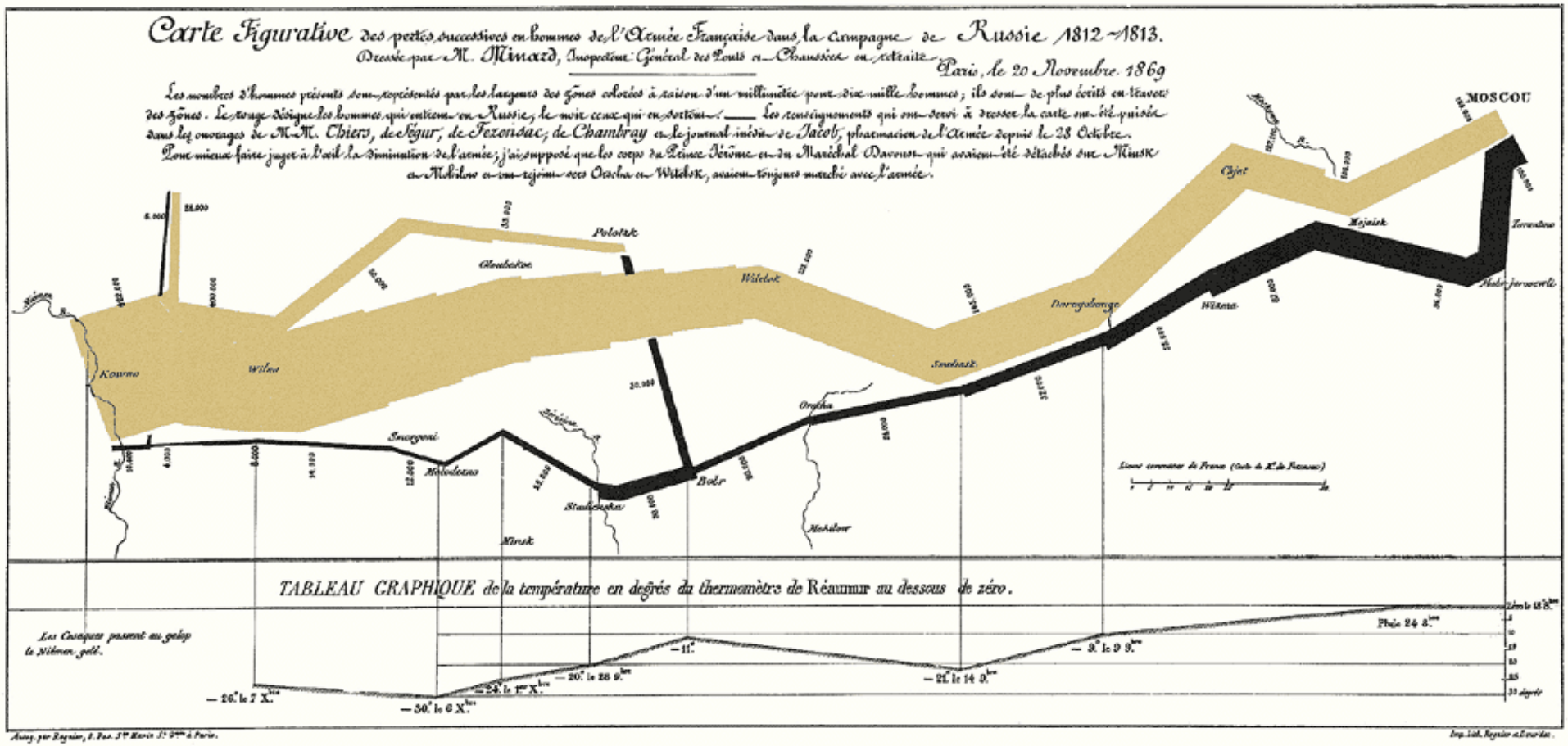


Some visualizations reveal “unknown unknowns” about data



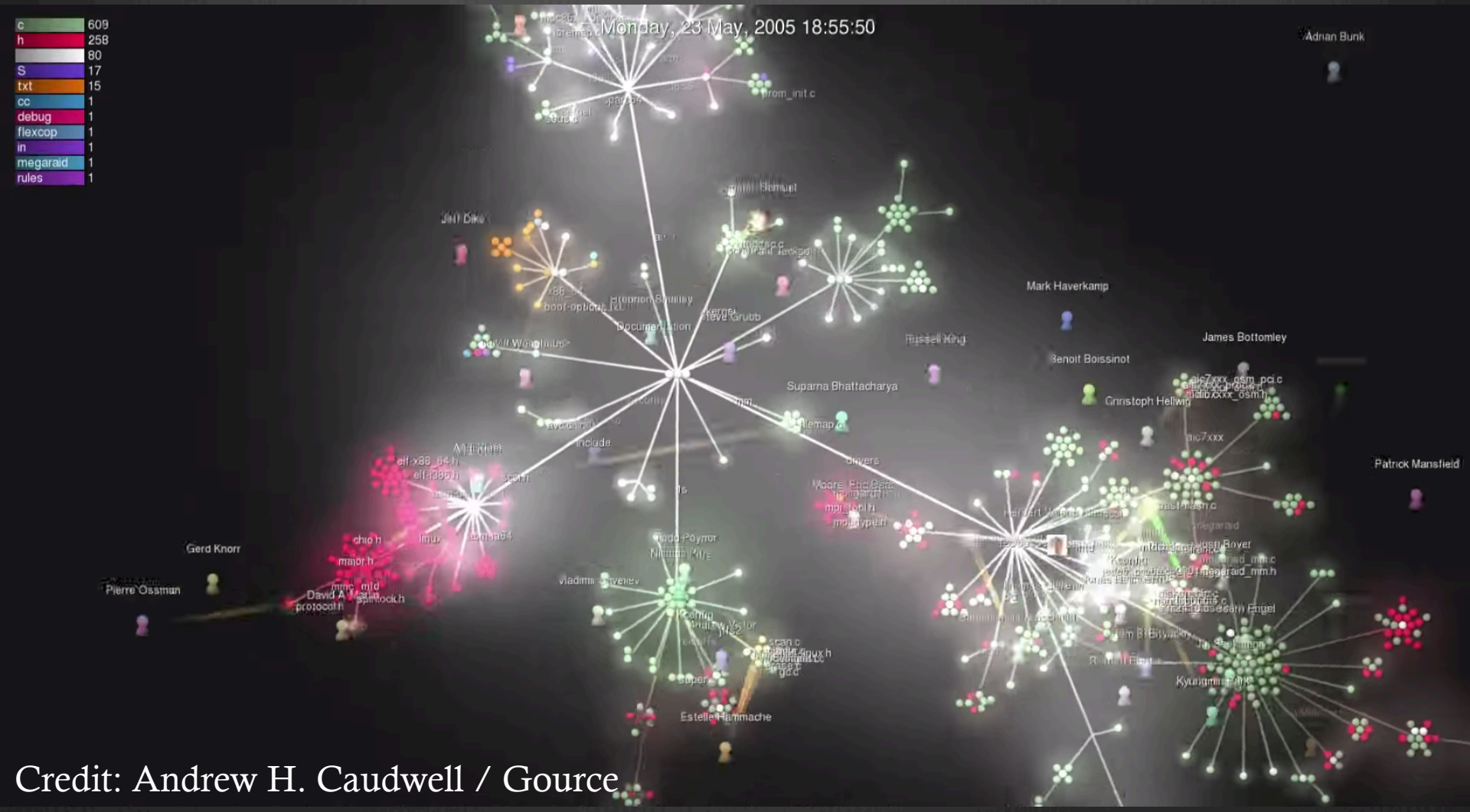
Credit: Aaron Koblin

Others answer hard questions in a visually intuitive way



Credit: Charles Joseph Minard

Visualizations are not necessarily static



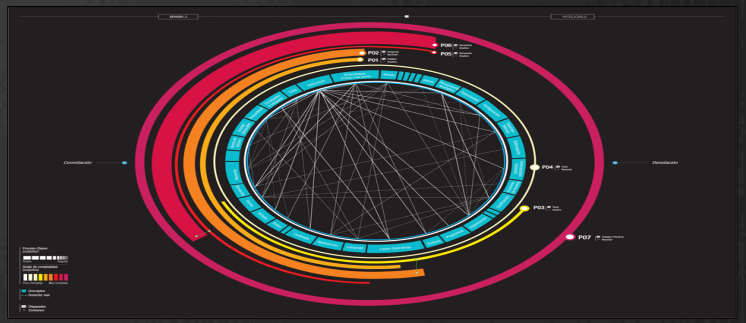
Credit: Andrew H. Caudwell / Gource

Recap: data science's three legs

1. Scalable data storage technologies

2. Machine learning / scalable analytics

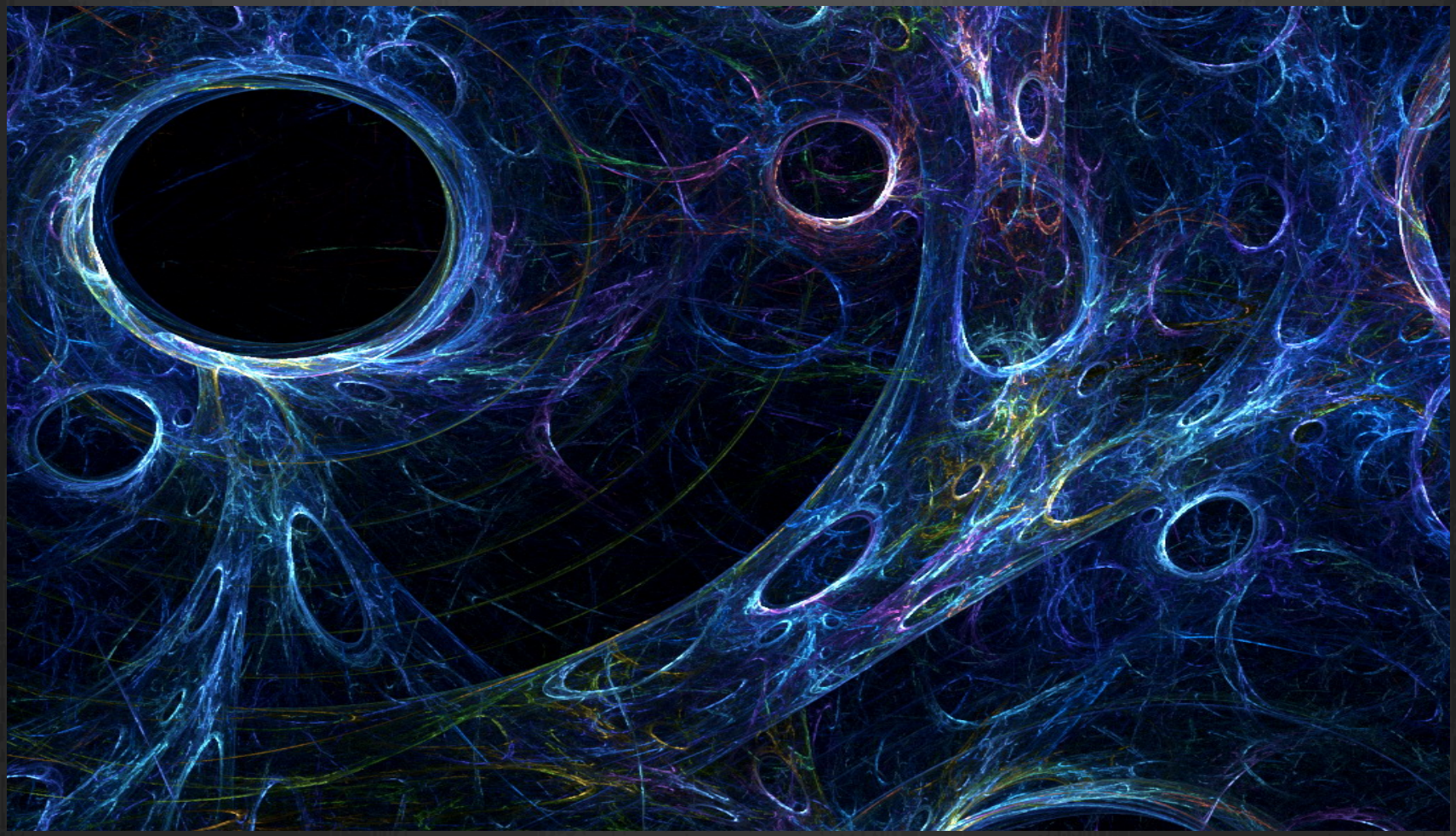
3. Data visualization / decision support



invincea™

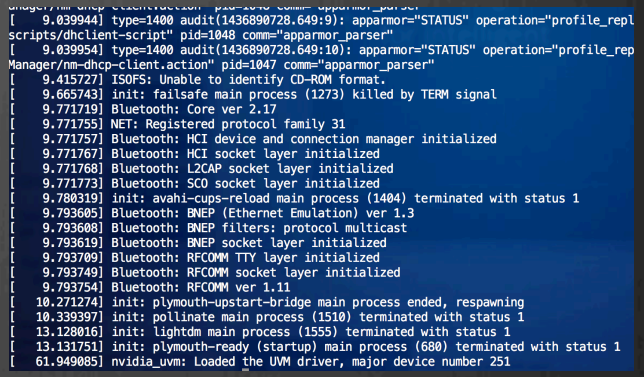
Why security needs data science

We have access to the signals needed to detect attackers, but they are currently the "dark matter" of our field

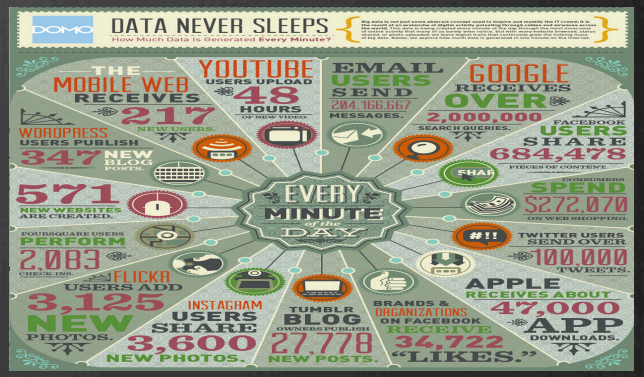


The “dark matter” of information security is where attackers hide

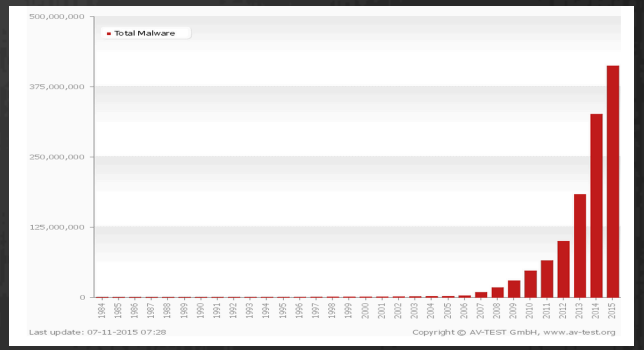
Terabytes of logs generated by enterprise networks that are never examined …



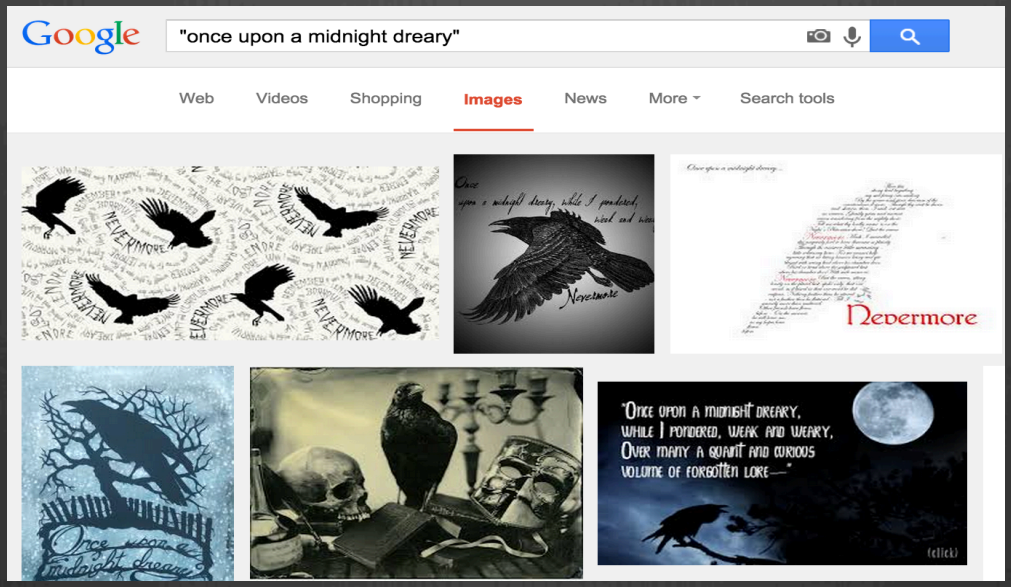
Reams of user behavioral data that could hold information about threats …



… the 400 million+ malware samples seen in the Internet, most of which have never been analyzed



Reason for hope: images and video were once also the "dark matter" of search – increasingly this is less and less the case, thanks to data science



Recent breakthroughs (most notably in neural networks) allow us now to identify objects in images with human-level accuracy, and transcribe audio with usable accuracy

A lightning overview of machine learning and visualization concepts

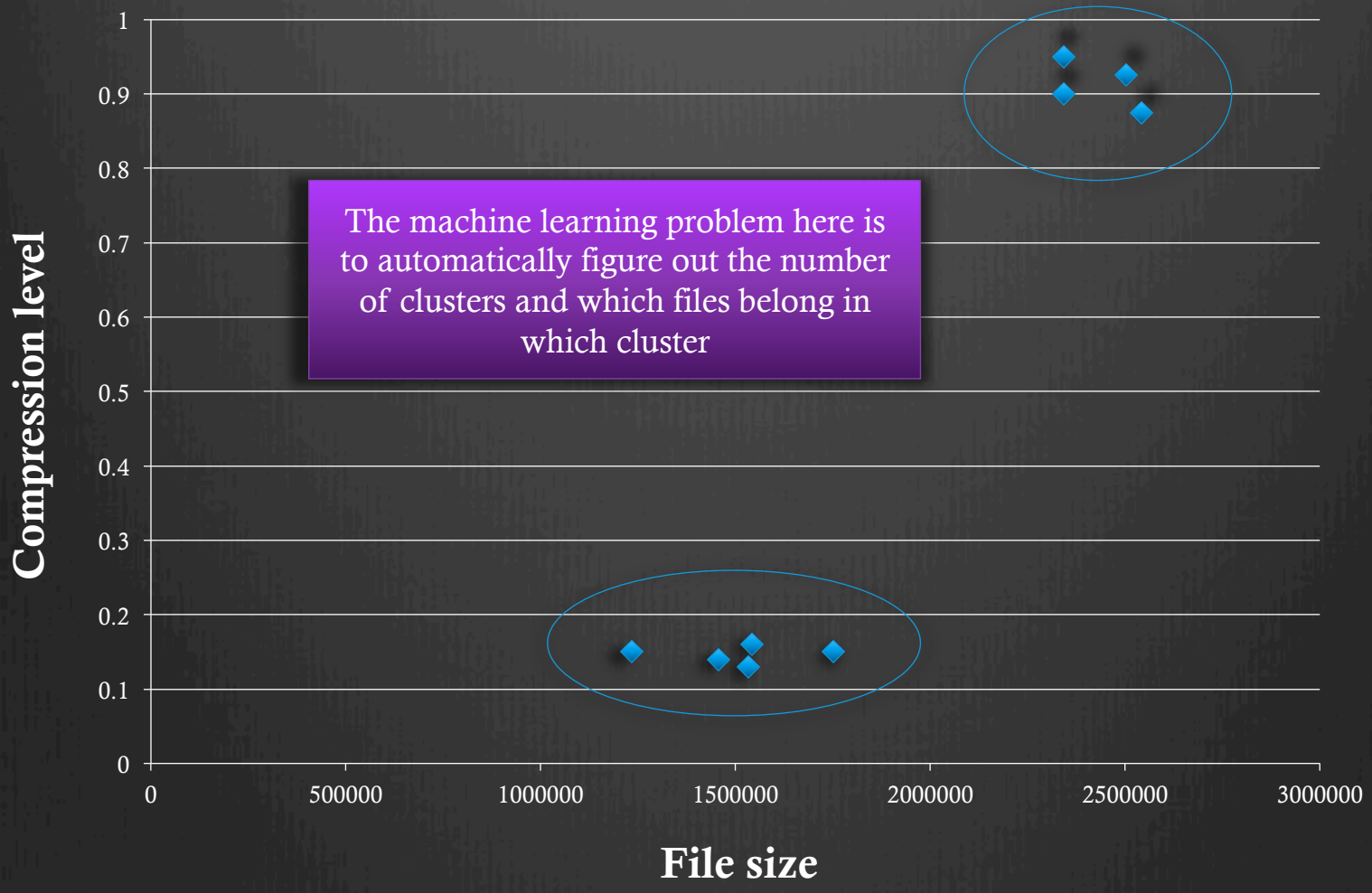
Machine learning in geometric terms

- In machine learning, data are thought of as vectors
- In programming terms, just think of these vectors as rows in a table

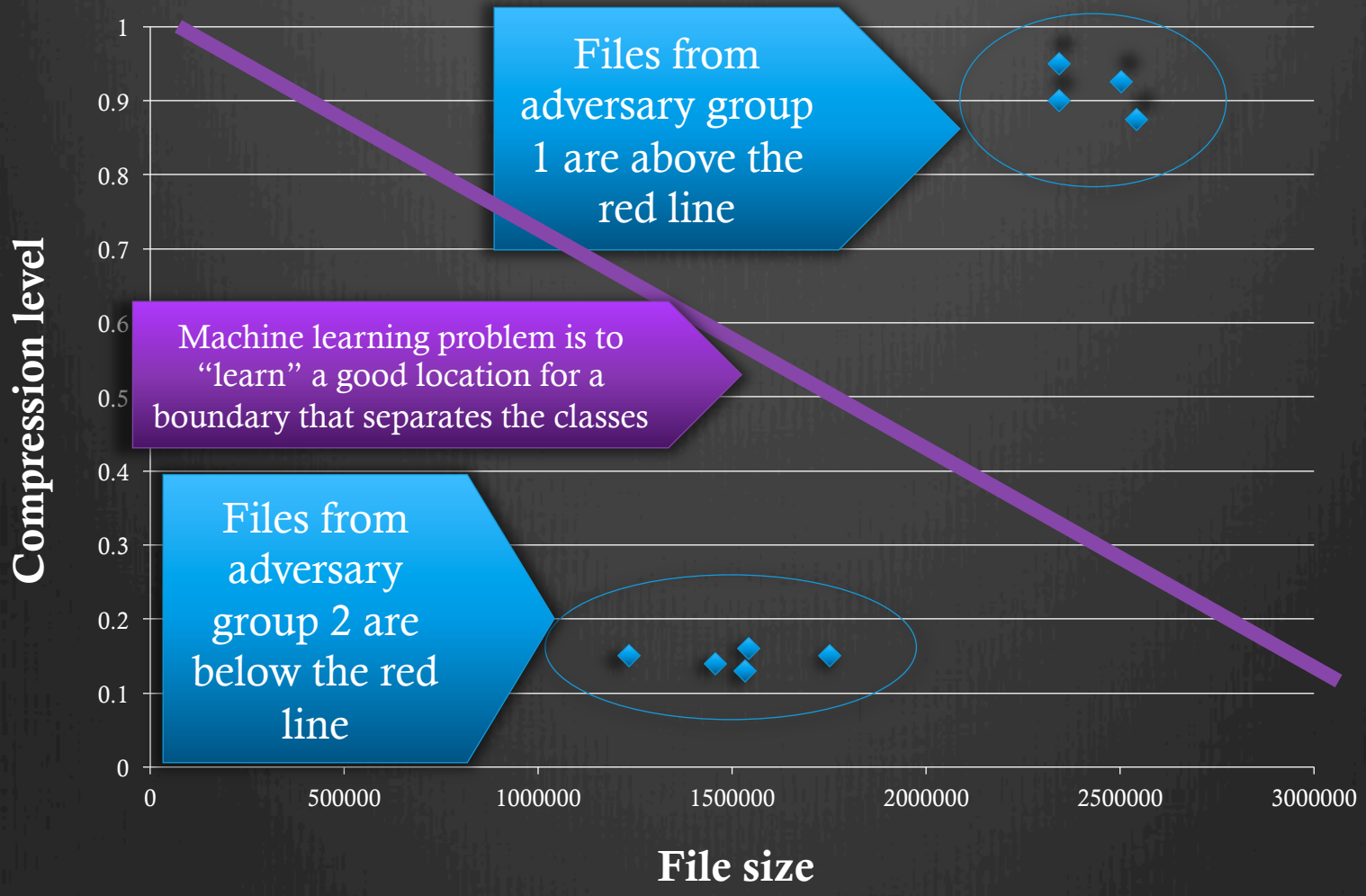
File size	File compression level
2342315	0.95
2343909	0.9
2504321	0.925
2541234	0.875
1234145	0.15
1534145	0.13
1542145	0.16
1751145	0.15
1456145	0.14

"I use mathematics to justify a conclusion after I have figured out what is going on by using physical intuition." – Geoff Hinton, Reddit AMA 2014

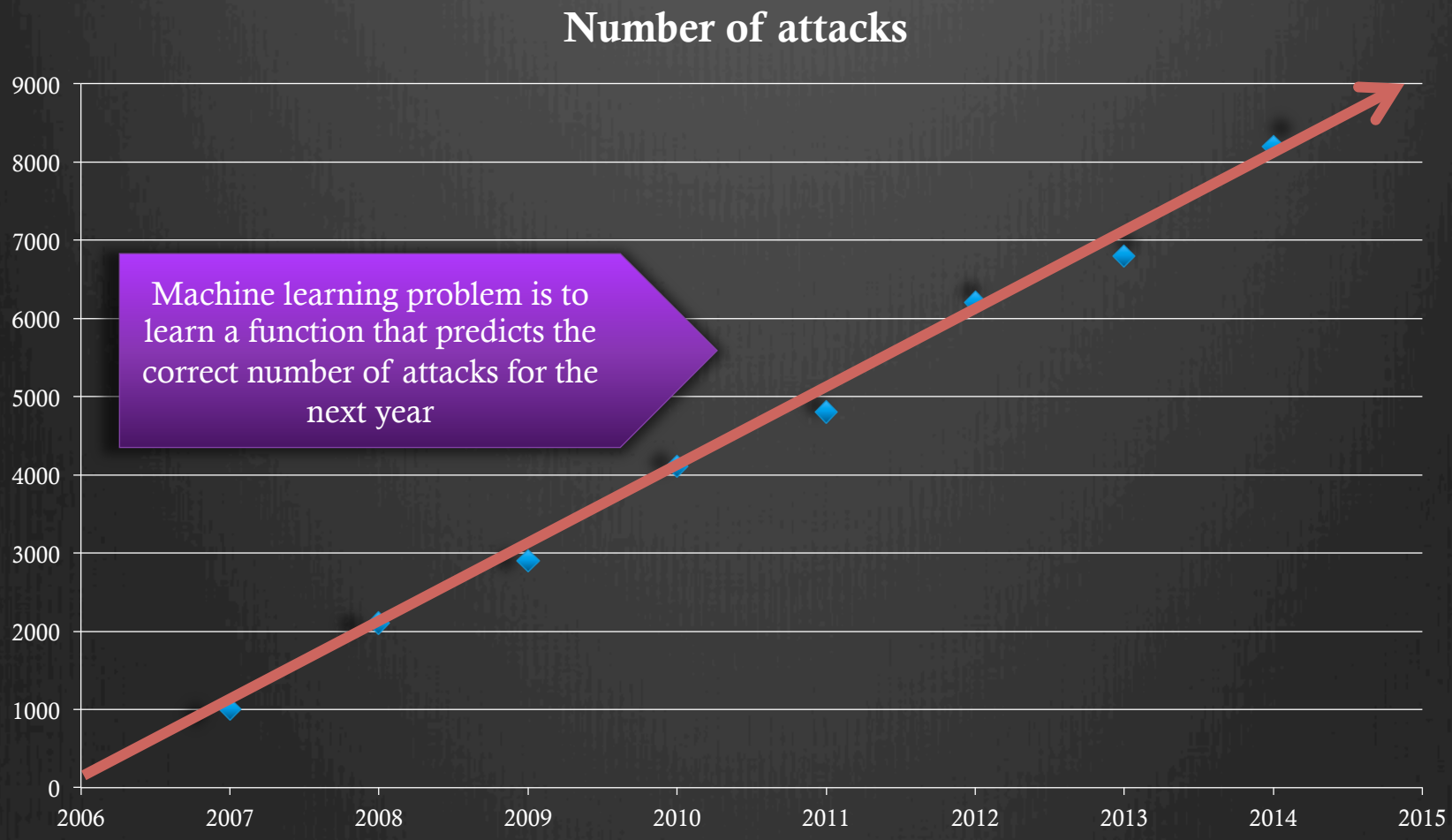
Clustering algorithms focus on identifying like-groups without any prior knowledge about the data



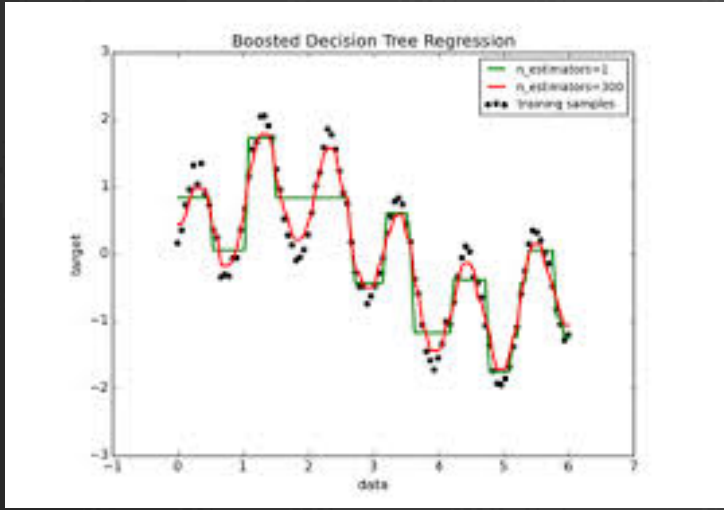
Classification algorithms assign data to known "classes"

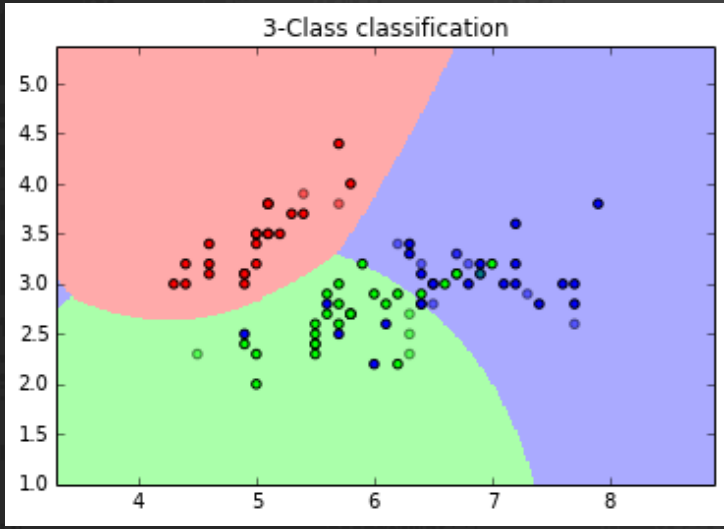


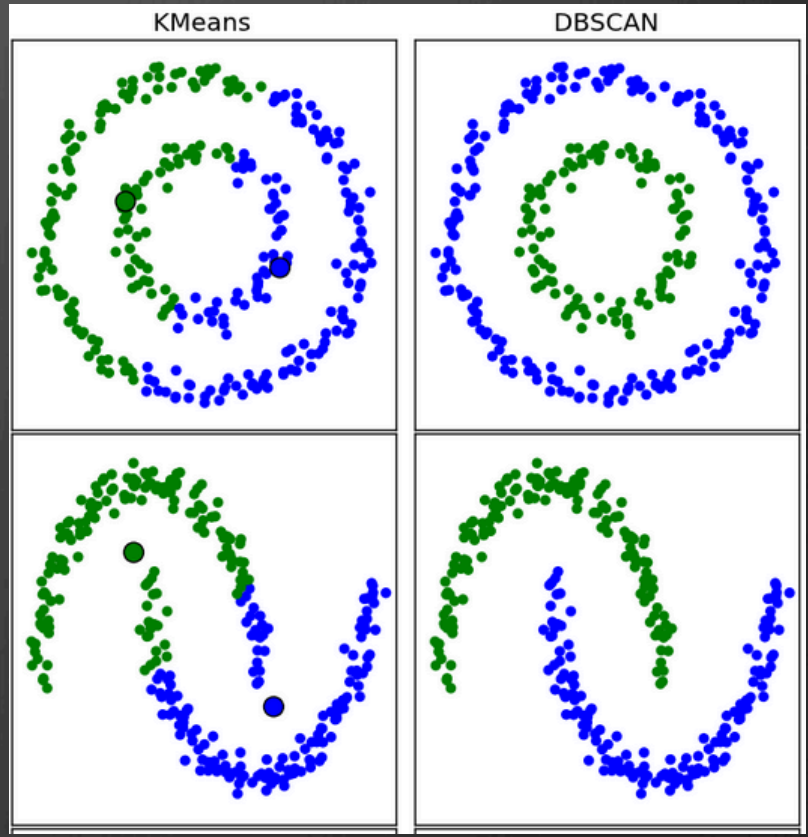
Regression predicts an unknown value based on some known value(s)



Machine learning problems are often non-linear: Example hard modeling problems in clustering, classification and regression



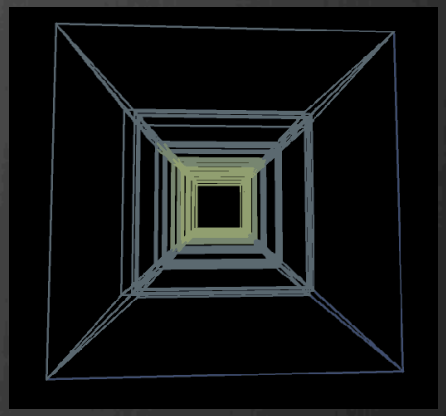




Credit for images: sklearn

Also, real-world data is almost never two dimensional

- In security data science, for most problems we care about, data have not 2-dimensions, but *thousands* or *millions* of dimensions



- High dimensional data makes physical intuition more difficult, but it nonetheless is how most data scientists reason about their models

Dimensionality is not that intimidating when we think about the data as a table

It's easy to think about performing clustering, classification and regression on data of arbitrary dimensionality in this format

Challenge is to maintain physical intuition about high dimensional spaces!

File size	File compression level	File epoch creation timestamp	Number of computers on which we've seen this file	…
2342315	0.95	1437360052	5	
2343909	0.9	1437360052	4	
2504321	0.925	1437360052	6	
2541234	0.875	1437360052	100	
1234145	0.15	1437360052	4	
1534145	0.13	1437360052	123	
1542145	0.16	1437360052	4	
1751145	0.15	1437360052	3	
1456145	0.14	1437360052	1	

Core ideas in data visualization

1. Design a visualization that supports human visual ***preattentive*** reasoning

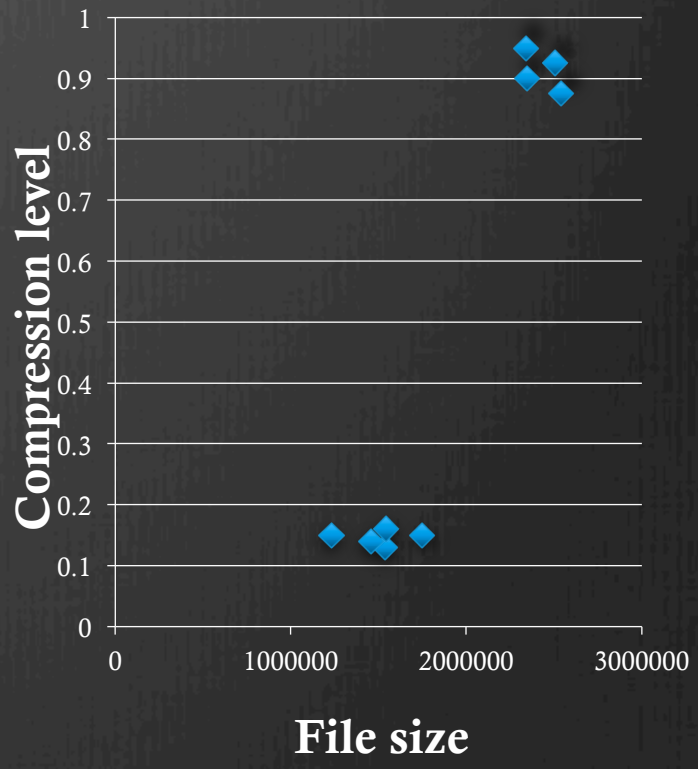
2. Map data variables to visual variables

Map to X axis

Map to Y axis

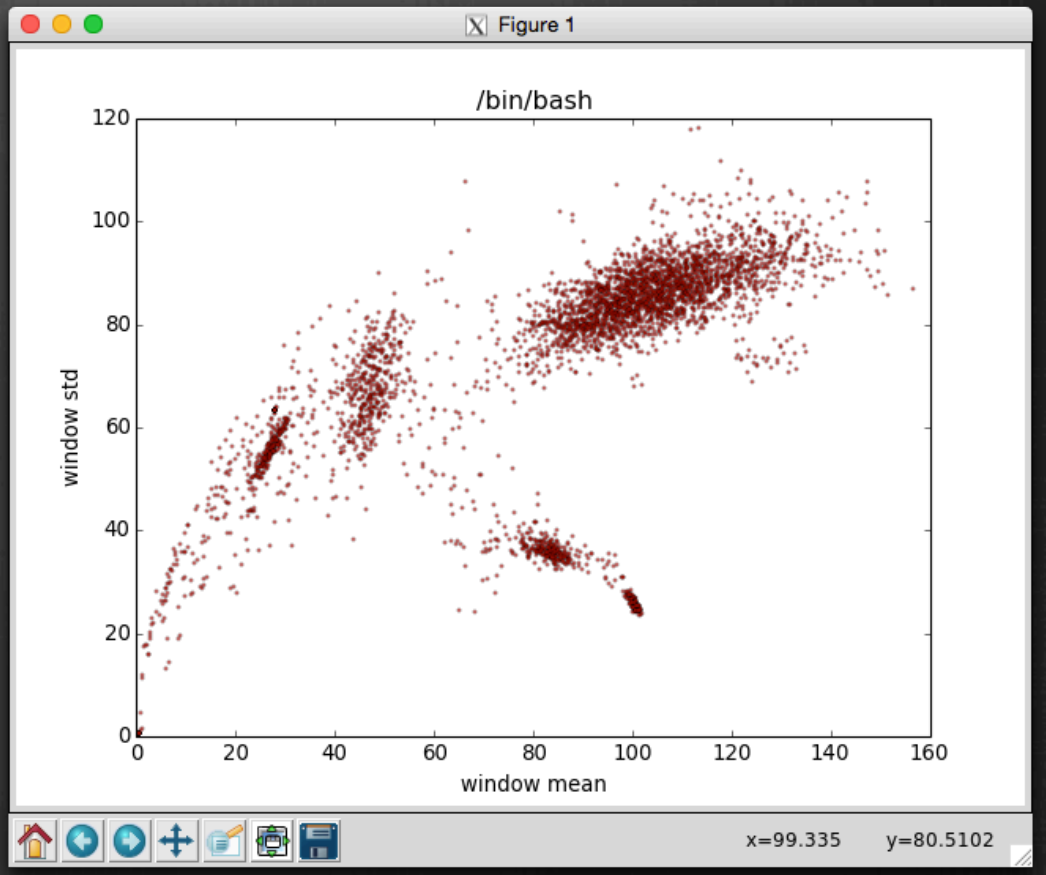
File size	File compression level
2342315	0.95
2343909	0.9
2504321	0.925
2541234	0.875

3. Render visual variables onto the screen



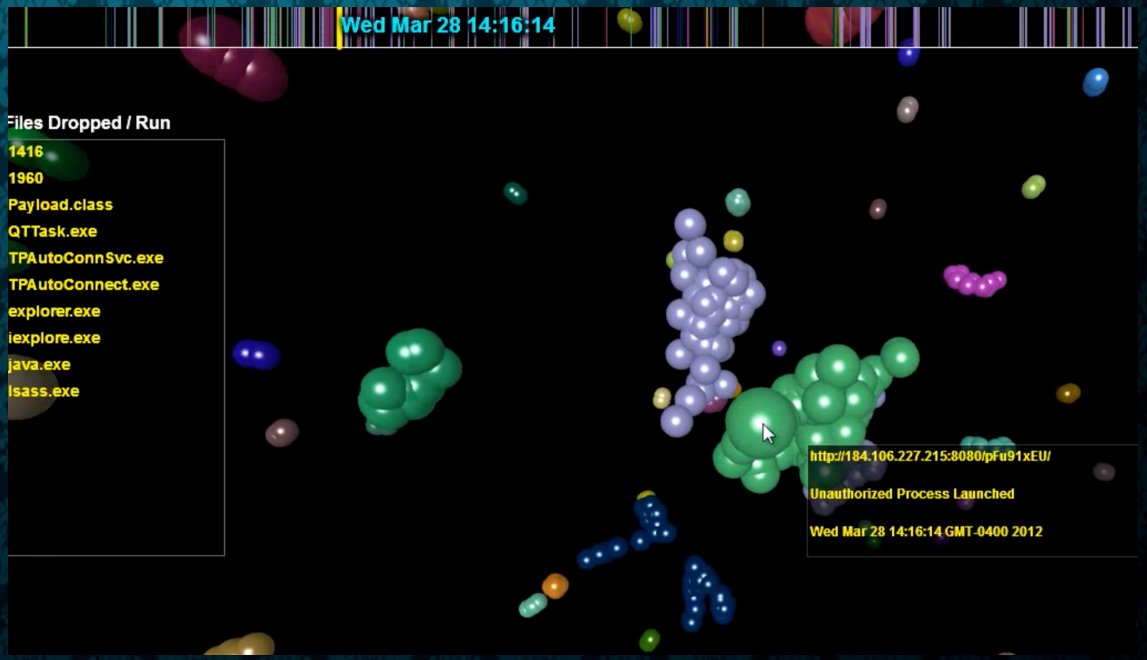
Core ideas in data visualization: Preattentive perception means the visual cortex effortlessly solves hard machine learning problems

This is a simple visualization of the data in /bin/bash based on byte window statistics – identifying clusters is a simple visual task for humans, but is non-trivial for computers



Visualization design principles

- Overview first, details on demand
- Visualizations should answer questions, or should be explicitly exploratory
- Design with a user in mind, and a user workflow in mind
- Don't overload the user with too many visual channels

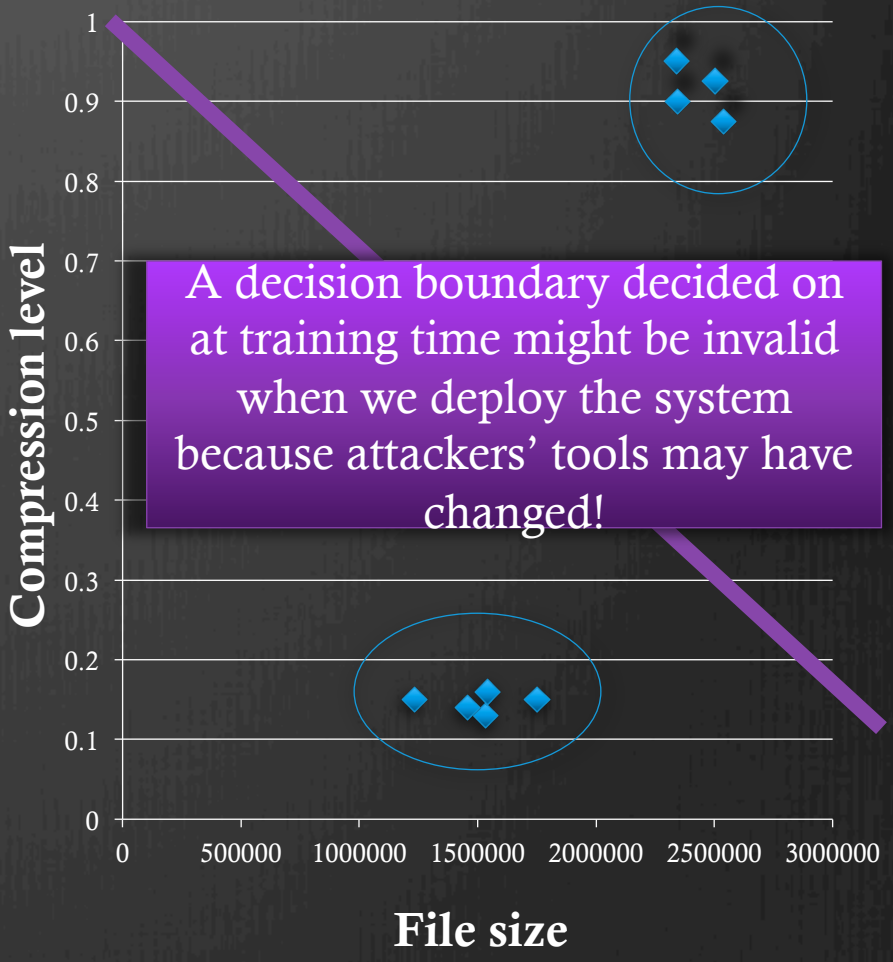


invincea™

What makes security data science different?

A technologically advanced adversary

- Attackers are deliberately trying to evade detection systems
  - This makes applying classification, regression, and clustering a far different problem than, for example, classifying news articles into categories
- Attackers techniques are constantly changing
  - This makes applying data science methods to security different from, say, voice recognition



Addressing the "evolving adversary" problem: Adversarial evolution should be addressed by modeling attacker behavior in the way we evaluate our data science tools

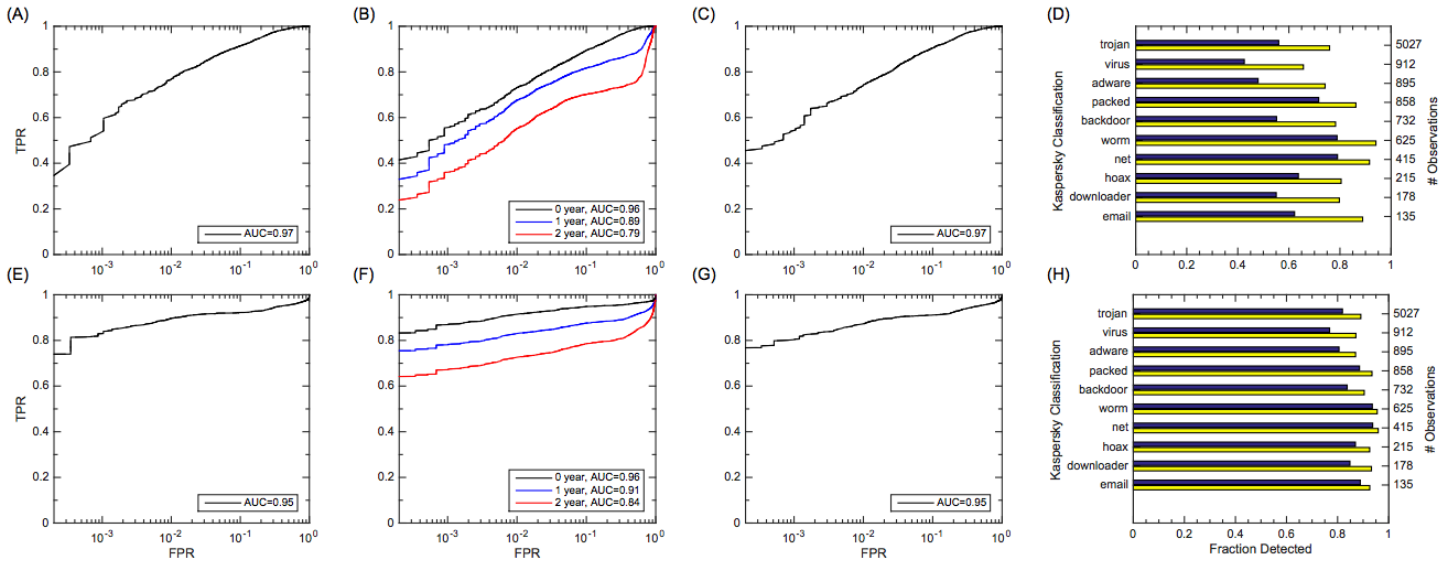
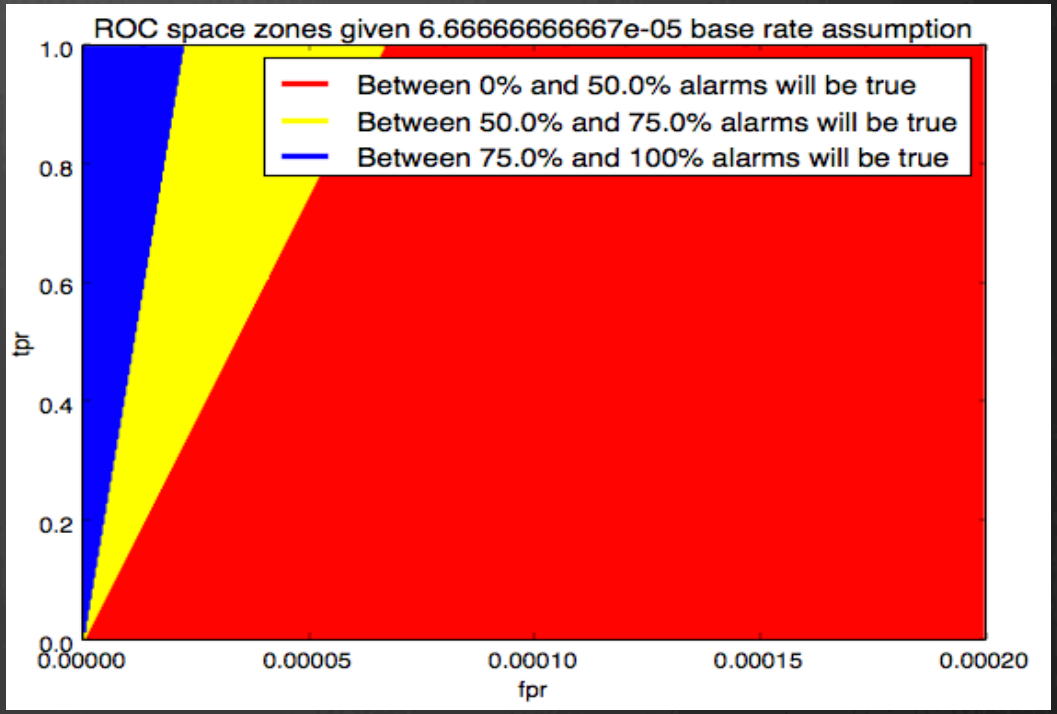


Figure 2: 10-fold validation results for our ℓ_1-regularized LR classifier. The top panels are validation results computed only on CuckooBox collected audit log data, while the bottom panels show validation results done only on synthetic malware and pure benign enterprise audit logs. Each column shares the same classifier between the top and bottom panel. The first column shows the ROC validation curves for randomly split data. The second column shows ROC validation curves based on compile time splits, where the training malware data is at least one (middle line) or two (bottom line) years older than validation malware data. The third column show ROC validation curved based on malware families splits, where we train on one set of malware families, and validate on the rest of the families. The fourth column shows results from the first column, at two deployment-relevant false positive rates (10^{-2} top and blue, and 10^{-3} bottom and yellow), broken down by specific malware types, with the number of total observations of each malware type is shown on the right axis.

Figure from work by Konstantin Berlin, David Slater, Joshua Saxe

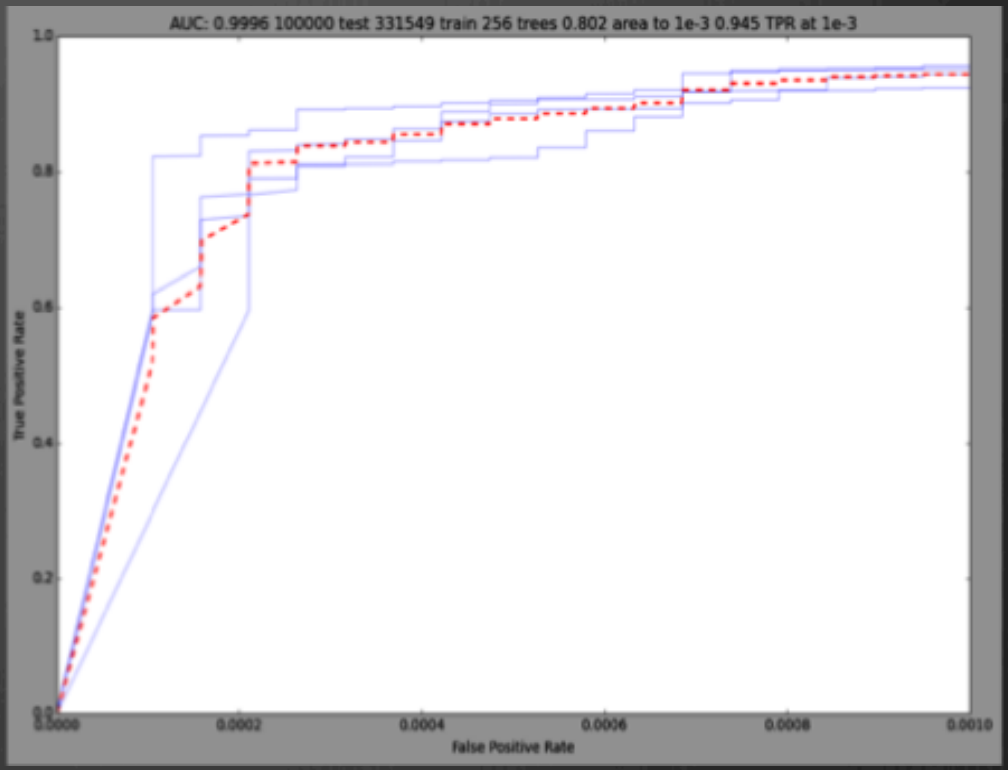
The false positive problem

- Many of the things we look for in security are extremely *rare*
  - Exploitable vulnerabilities are rare
  - On an enterprise network, malware is rare relative to benignware
  - Insider threat behavior is rare
  - False positives *rates* (the percentage of false cases mislabeled as true) have to be *extremely* low in security for approaches to be useful



Addressing the false positive problem

- Focus on measurement of system detection rate in the ultra-low false positive region
- Emphasize collection of huge volumes of benign examples so the false positive rate can be accurately measured in low-FPR region
- Pick models that predict accurate probabilities – calibrate these models
- Learn the dark art of designing systems that operate well at low false positive rates!



The need for interpretability

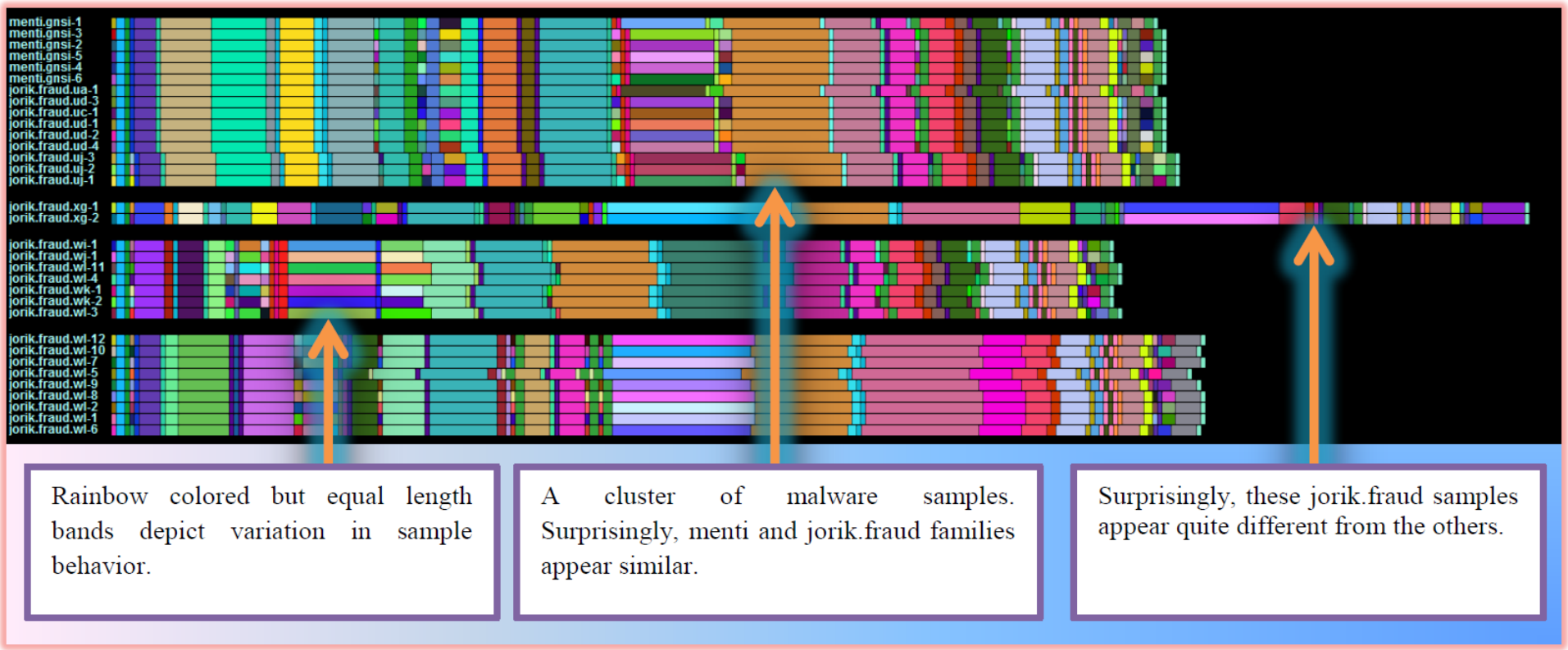
- In other areas of data science, all we care about are correct detections
- In security correct detections are only part of the story, we also need explanations (if a data science tool tells me I've been compromised, I want to see all of the evidence it is using to make that claim so I can follow up)

gui — captures mouse movement — 2

TOKEN	PROBABILITY OF CAPABILITY GIVEN TOKEN	SOURCE POST TITLE
SetCursorPos	0.69	Move mouse with c#
GetMessagePos	0.65	Mouse state winapi
DrawIcon	0.60	Draw mouse pointer icon?
TrackMouseEvent	0.53	Mouse Move Capture (Mouse leave & Mouse Enter)
MousePos	0.47	TListView and mouse wheel scrolling
SetCapture	0.46	Activate windows under mouse through mouse hook
WindowFromPoint	0.43	Activate windows under mouse through mouse hook
WheelDelta	0.40	Trigger 'dummy' mouse wheel event
OnMouseMove	0.33	mouse movements in wpf
OnMouseEnter	0.32	C# mouse picking XNA
SetCursor	0.32	draw mouse cursor in win32
GetCursorPos	0.31	Mouse wheel event
ReleaseCapture	0.29	Activate windows under mouse through mouse hook
GetIconInfo	0.26	Mouse cursor bitmap

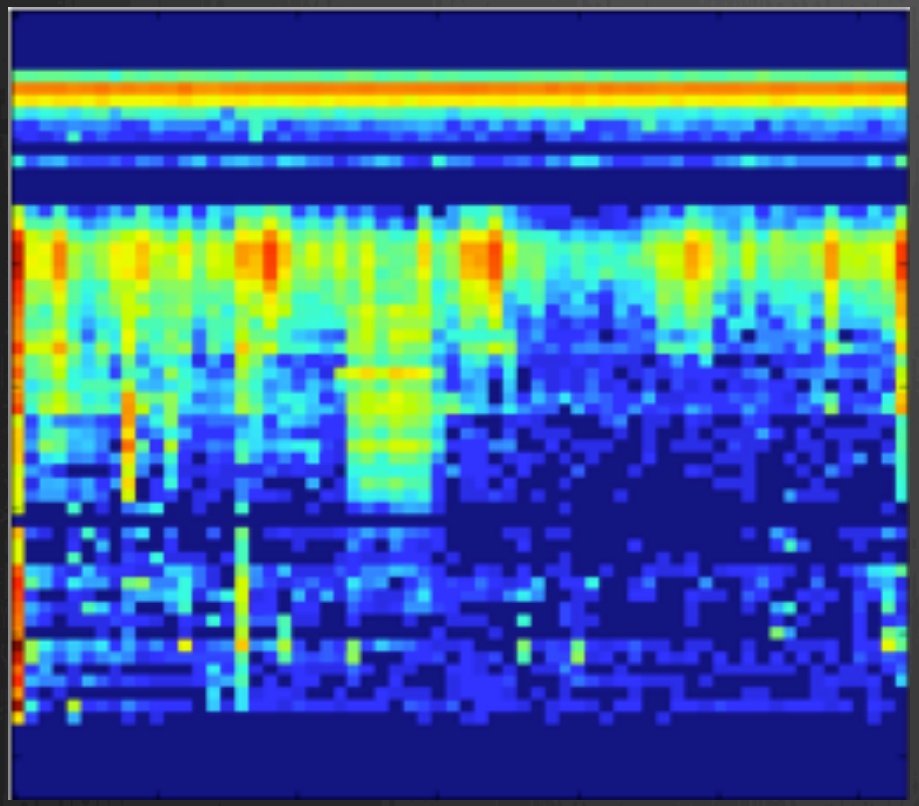
Building interpretability into the data science workflow

- Throughout the data science workflow, think about how you're building interpretability into your data science product
- Presentation is often just as important as model accuracy, because it means people can actually *use* your results



Three security machine learning cases studies

Case study 1: Machine learning based malware detection



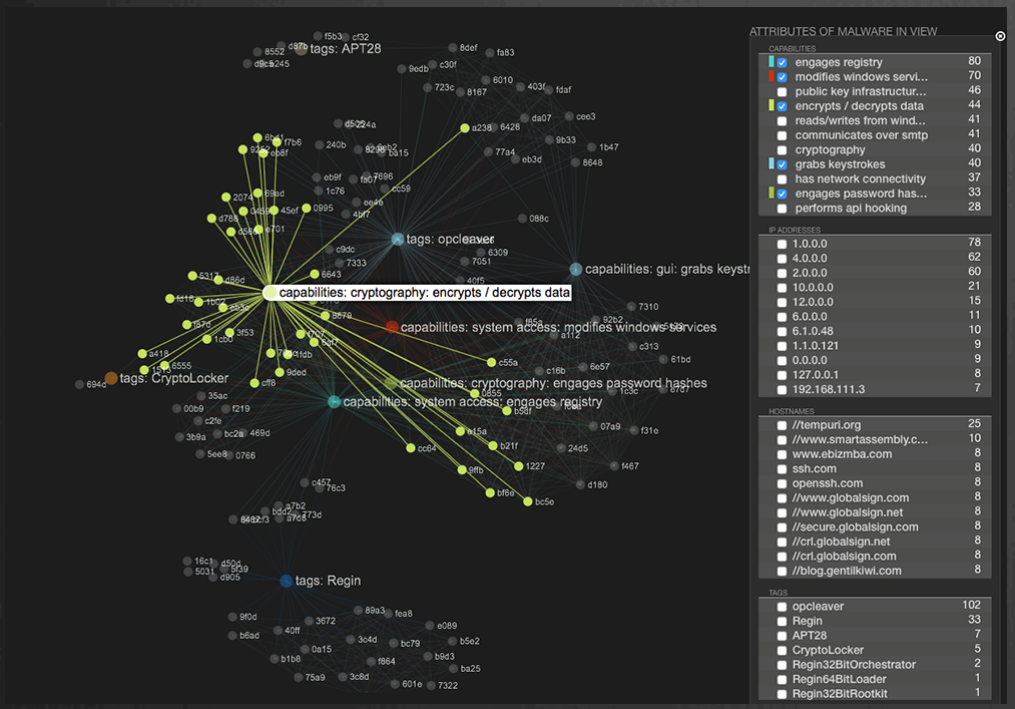
Project vision

Army of anti-virus company analysts discover new malware variants and generate signatures for these variants

Novel intelligent algorithms train on AV discovered variants, ***generalize*** from these training examples and discover ***new malware*** within millions of binaries acquired from customer sites

Modeling challenges

Exploit shared code relationships



*If we **train** on a malware sample with code component A, and then **test** on a sample with component A and previously unseen component B, and **component A occurs rarely or never in benignware**, we should classify the sample as malware*

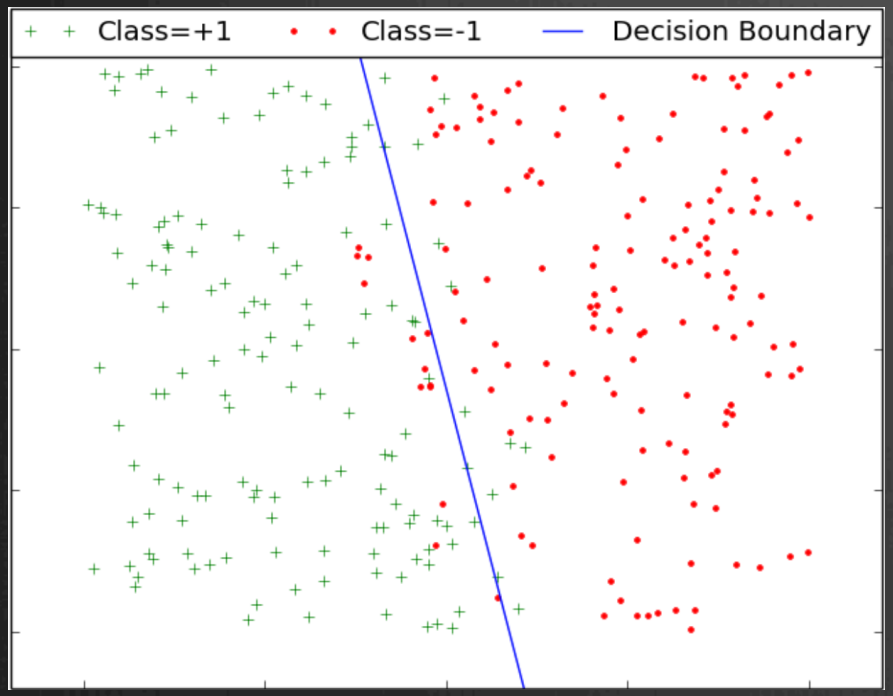
Modeling challenges
Exploit malware componentization

*If we **train** on a malware sample with data component A, and then **test** on a sample with data component A and previously unseen data component B, **and component A rarely or never occurs in benignware**, we should classify the test sample as malware*

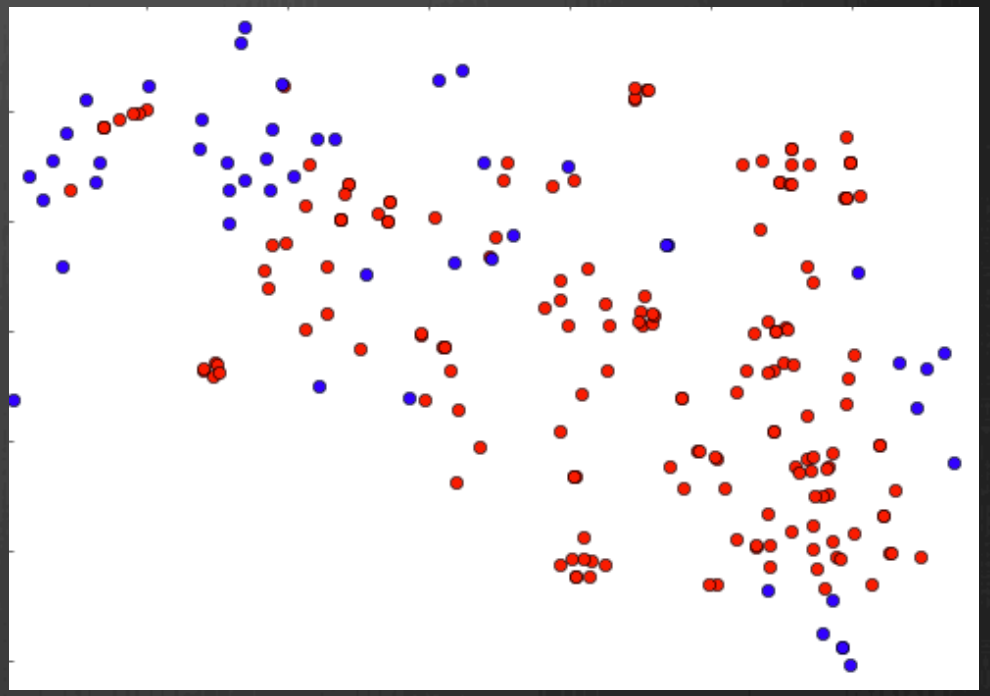
Modeling challenges

"Simple" machine learning models probably won't work

Idealized linearly separable data



Our actual malware / benignware feature space (projected onto 2 dimensions with t-SNE)



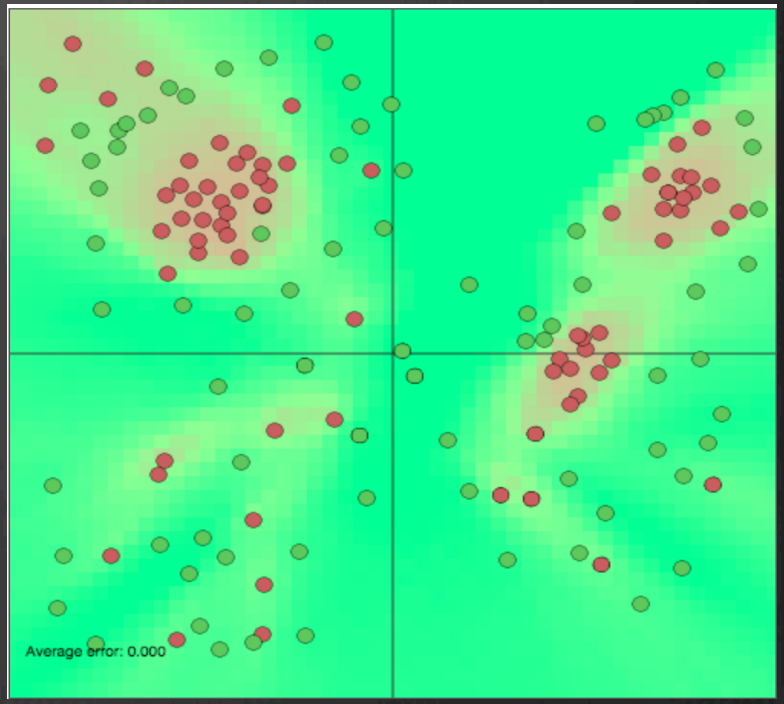
Modeling challenges

Model needs to rapidly learn from millions of training examples

A support vector machine, a commonly used machine learning model, trains in ***O(n_samples^2 * n_features)*** time — in practice, too slow to train on millions of training binaries without huge engineering effort

Many other models have similar computational complexity whose training time depends on the number of features

The challenge is to pick a model that scales to tens of millions of samples — we've found that to do this we also have to restrict the size of our feature space without losing accuracy



Approach architecture

Feature extraction (1-2 seconds) populates a 1024 ***fixed-dimensional*** feature space

- Contextual byte features
- Sliding window statistical features
- String 2d histogram features
- PE metadata features

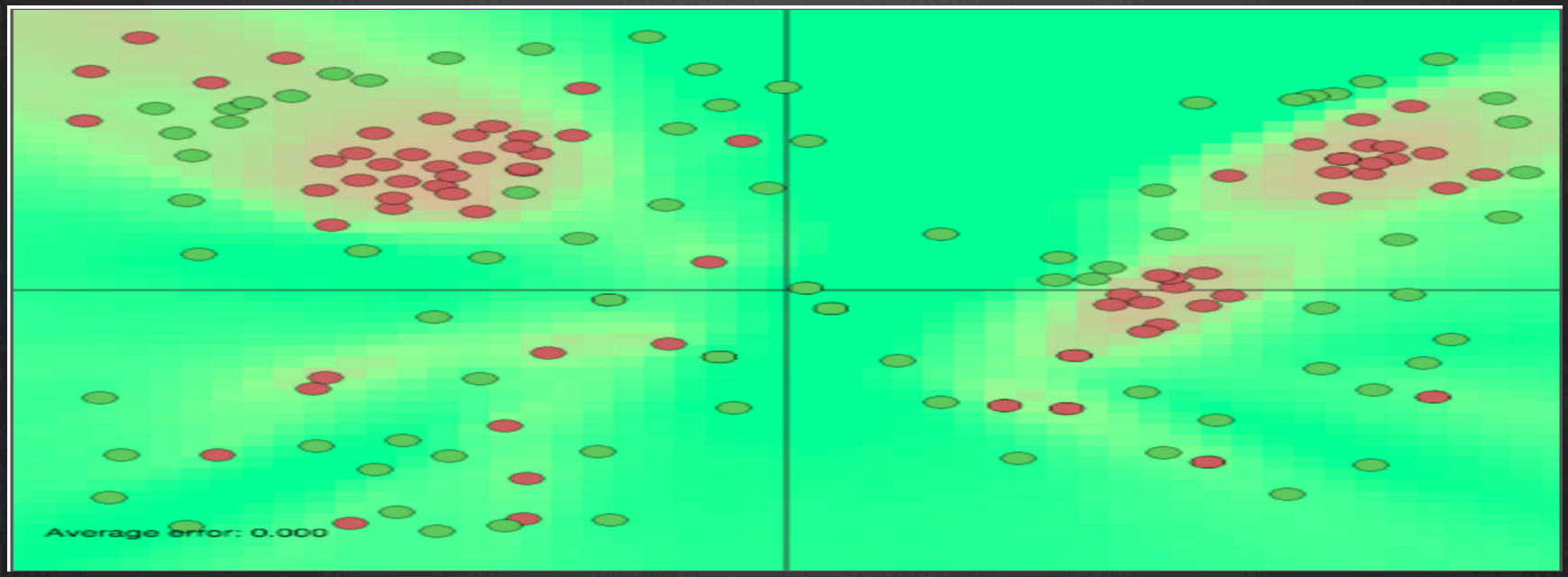
Random forest model (1-5 minutes to train on 400k files), using 1000 independent classification trees

Bayesian calibration model: convert random forest "threat scores" to principled *probability that a given file is malware*

Model operationalization: 80MB model data stored in memory representing 400k training samples, supporting streaming, parallel evaluation of files' P(malware)

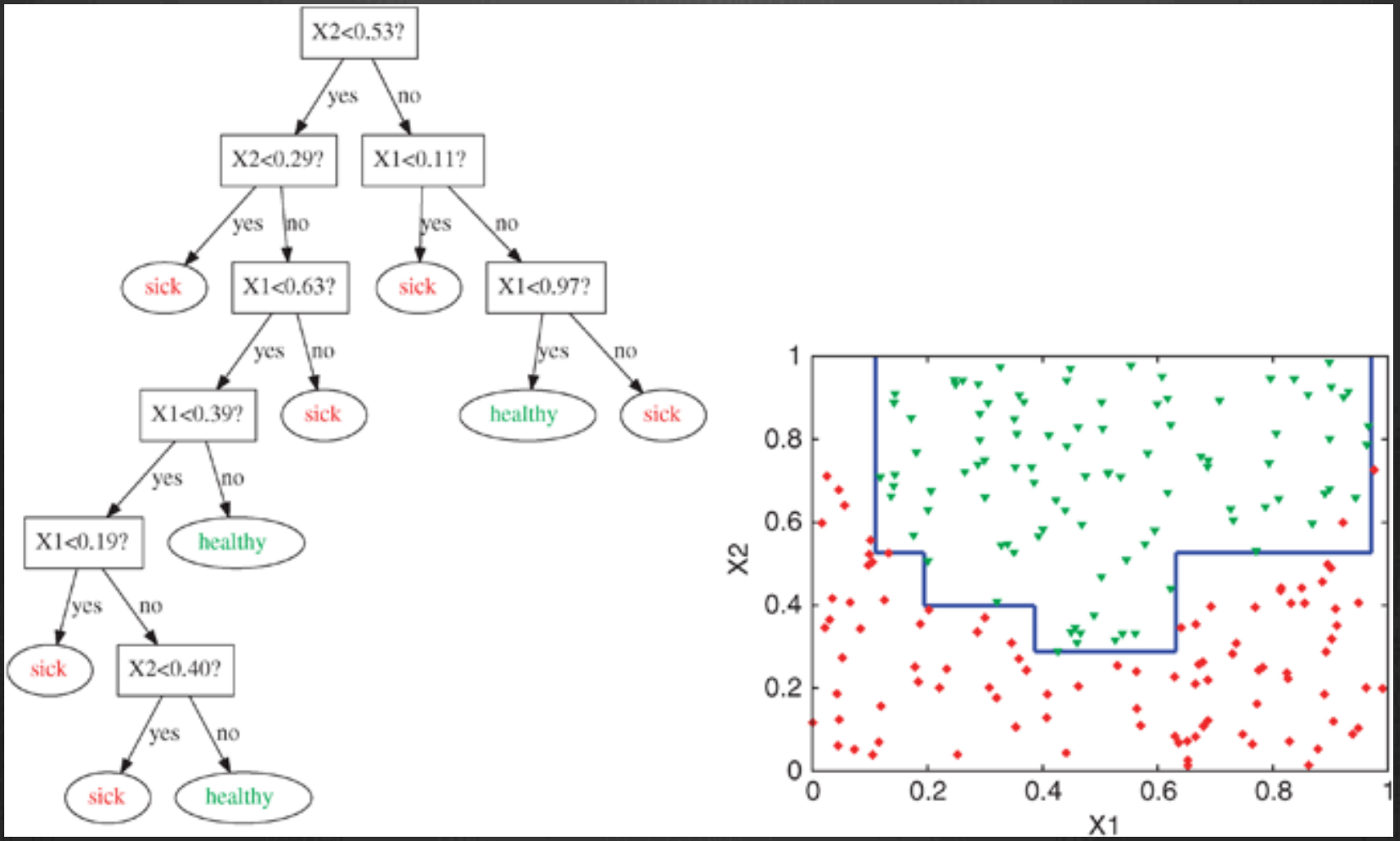
The random forest approach: train many decision trees, make sure they're *diverse* - then to decide if a sample is malicious or benign, average their output

Probabilistic characterization of the decision space given by random forest model

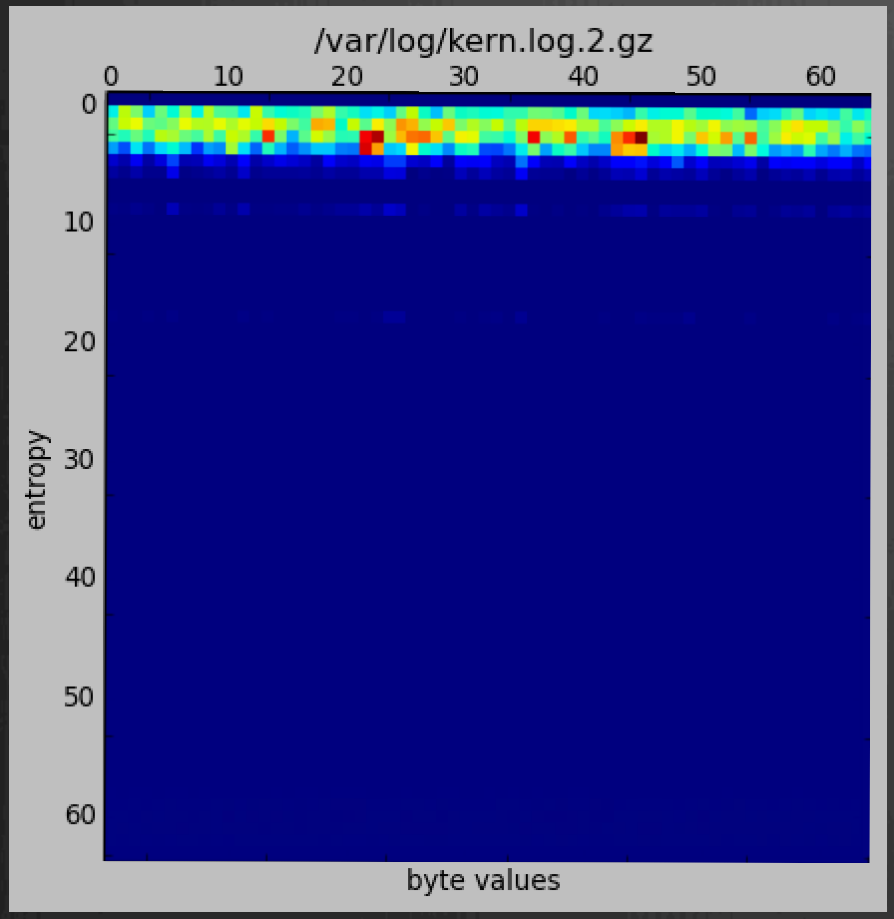


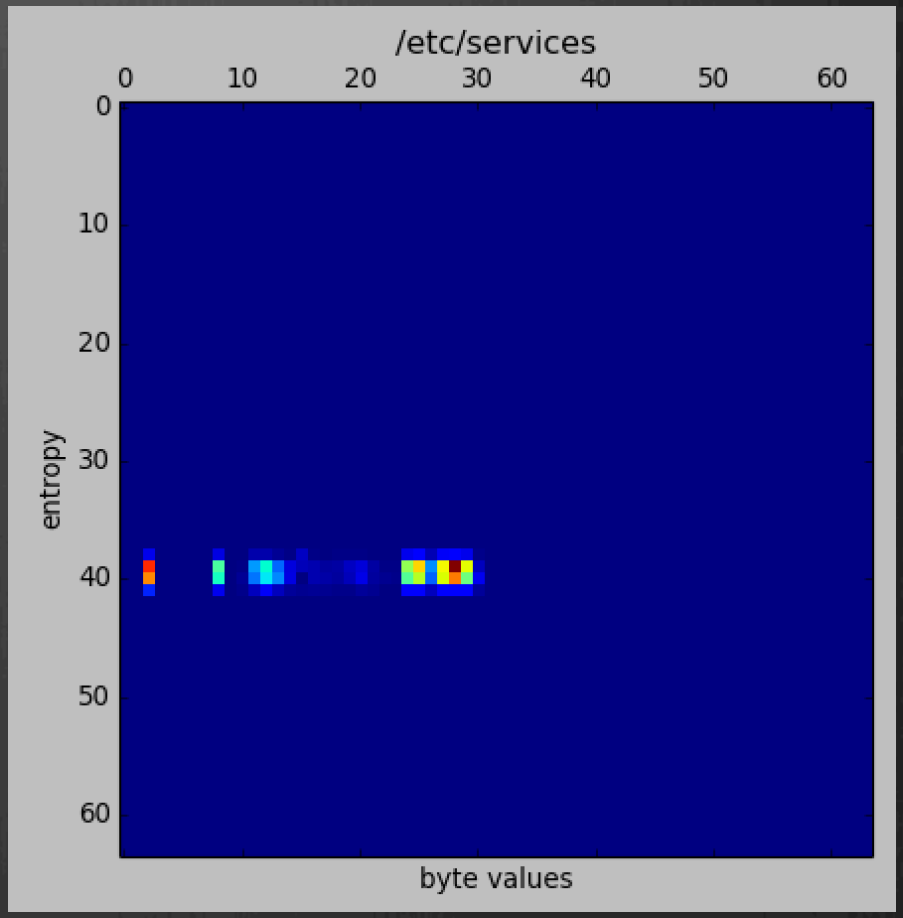
Credit: Andrej Karpathy

Decision trees: the trees that make up the random forest

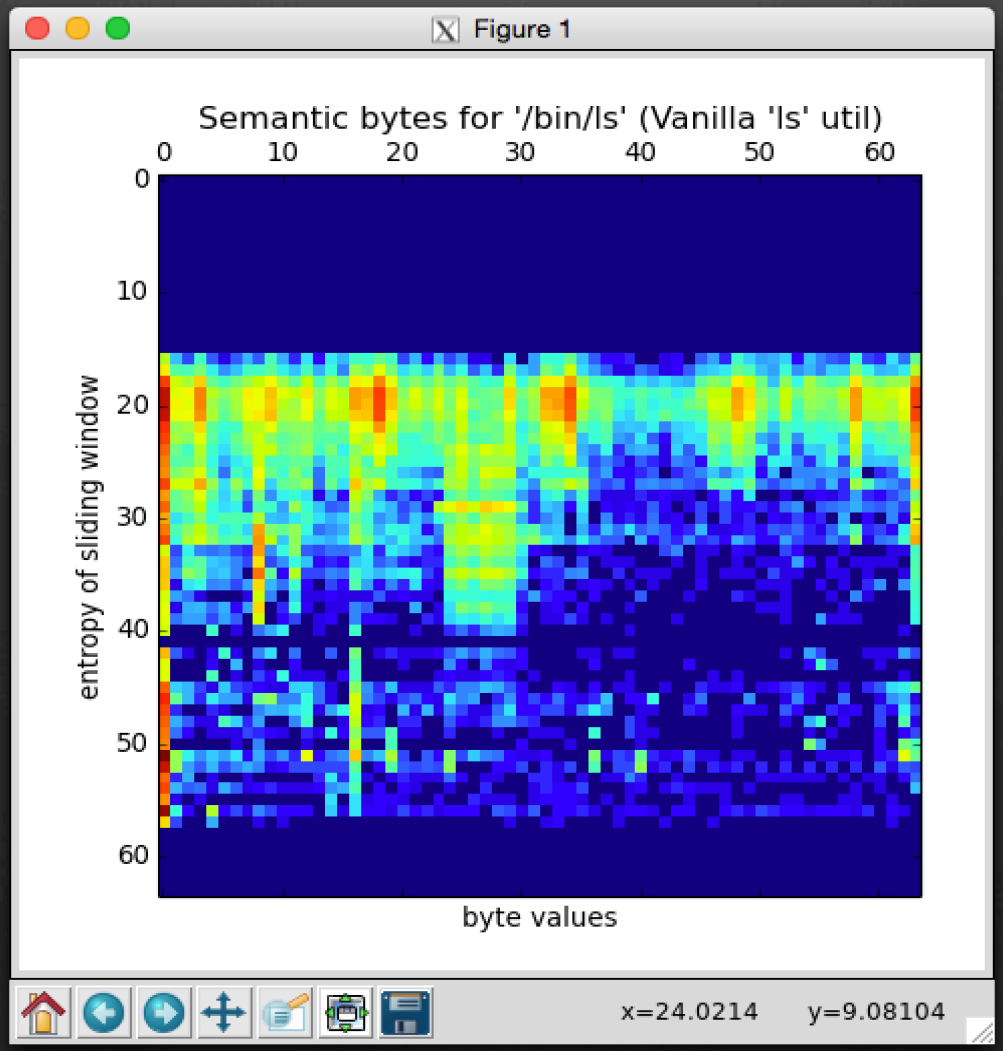


Byte/entropy histograms: a key component of our feature space

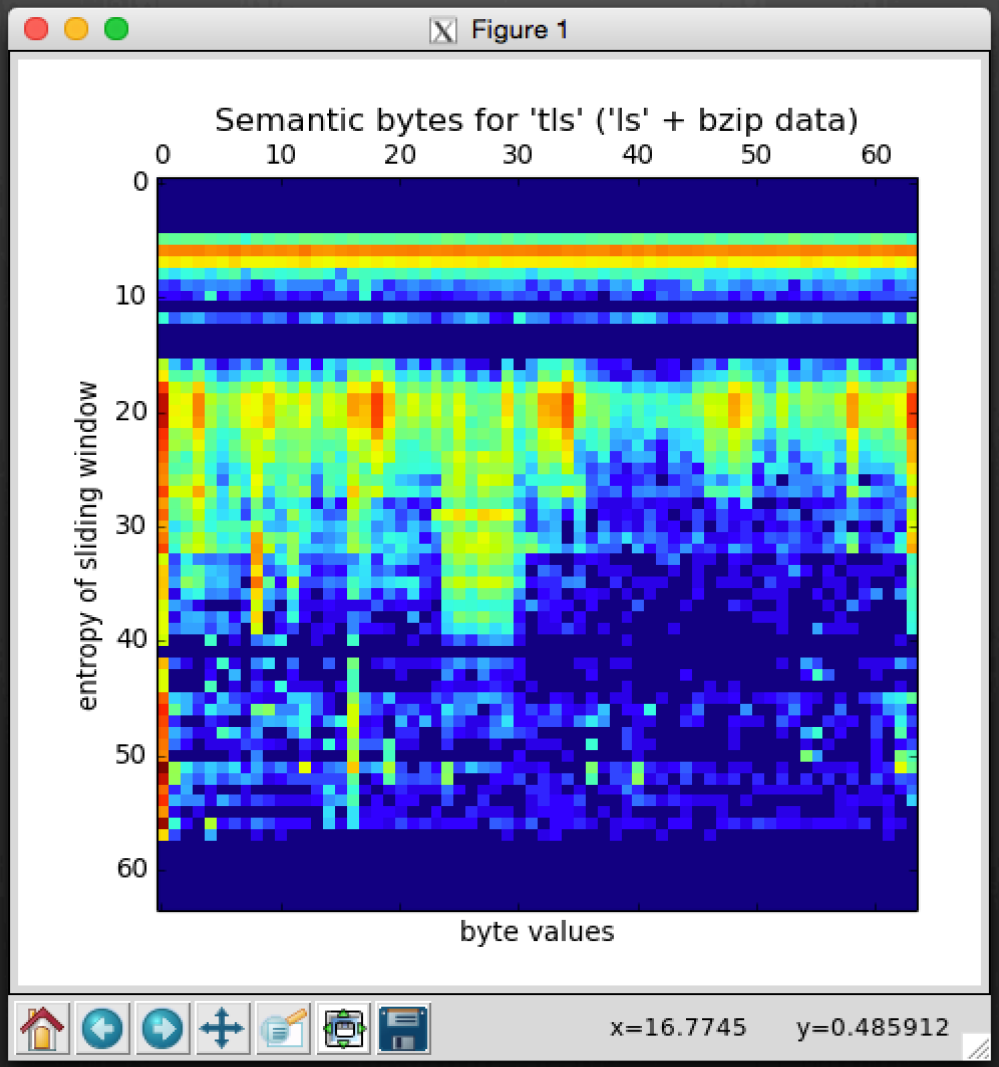




Byte/entropy representation of a binary file (benign in this example)

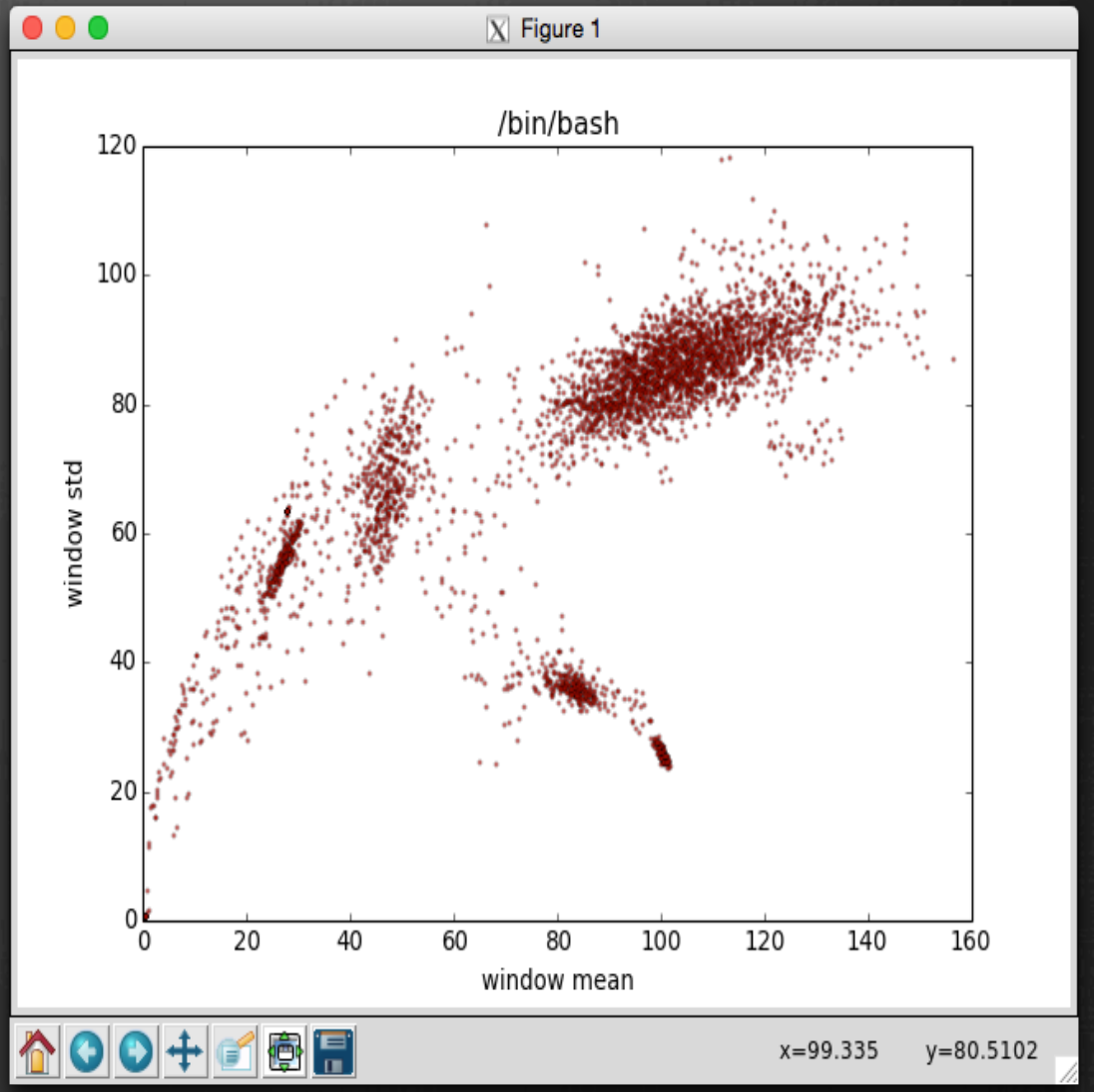


Byte/entropy representation of a binary file with a simulated component added



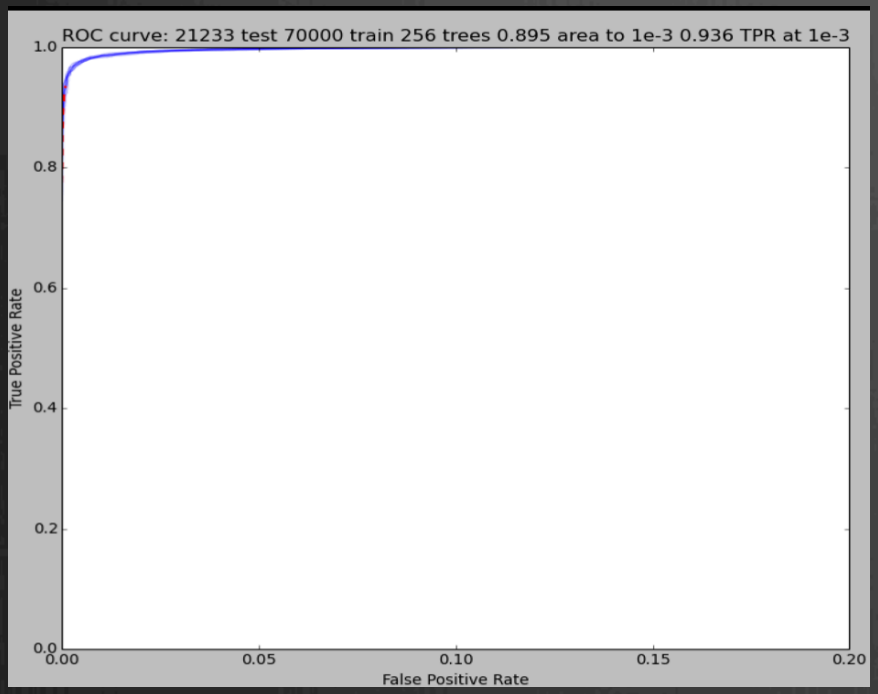
Another statistical feature set we used

- Slide a 256 byte wide window over the file
- For each window, capture standard deviation of byte values and mean of byte values
- Clear cluster structure emerges: *we're measuring how much of each "data type"* occurs in the file

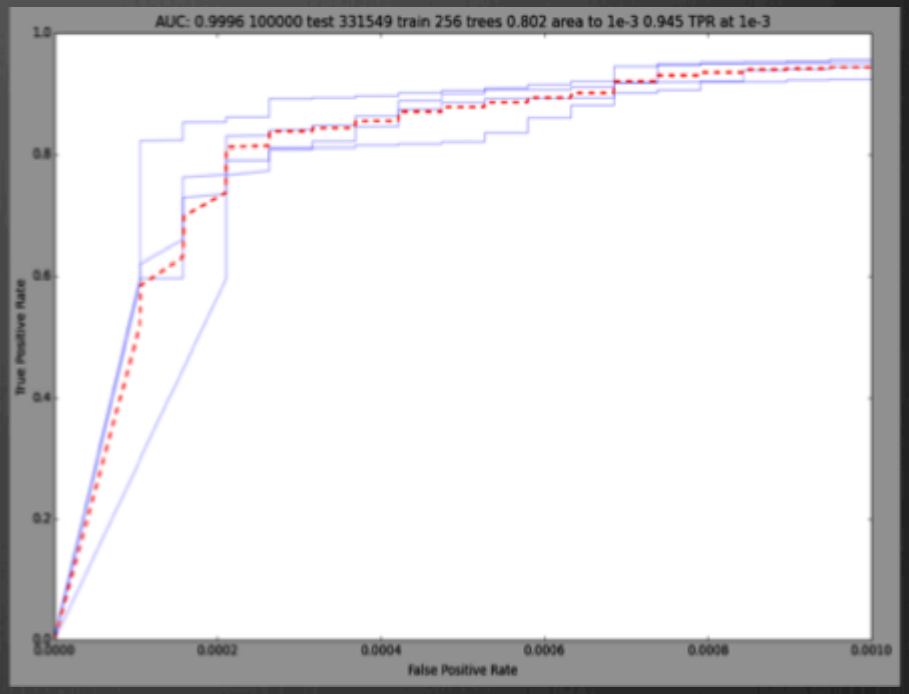


Accuracy evaluation

Overall ROC curve evaluation



ROC curve zoomed to low FPR range



Takeaway: these results model the zero-day malware detection problem (showing detection of previously unseen malware) and are the best publicized results we know of

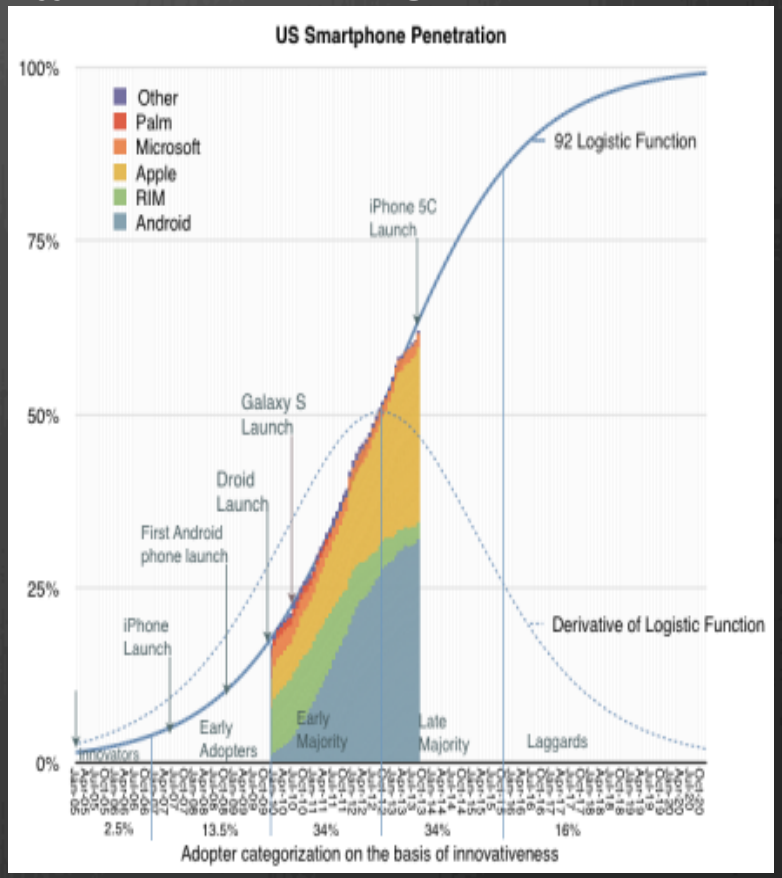
Case study 2. Predicting the future "success" of malware families

Project overview

The problem: predict which malware families will grow explosively

- New malware families (such as Koobface, Storm and Zeus) become targets for robust detection and disruption after they have achieved widespread criminal adoption
- Can we predict this *in advance*?
- Payoff:
  - Focus our reversing and countermeasure efforts on malware likely to achieve breakaway success

Our approach: adapt models of technology diffusion to predicting explosive malware



Problem statement and experimental setup

Research problem statement

Given timestamped observations of new malware binaries (new SHA1 hashes) grouped into family groupings:

- Can we correctly select the families on the cusp of exploding?
- Can we correctly detect when families have stopped growing explosively?

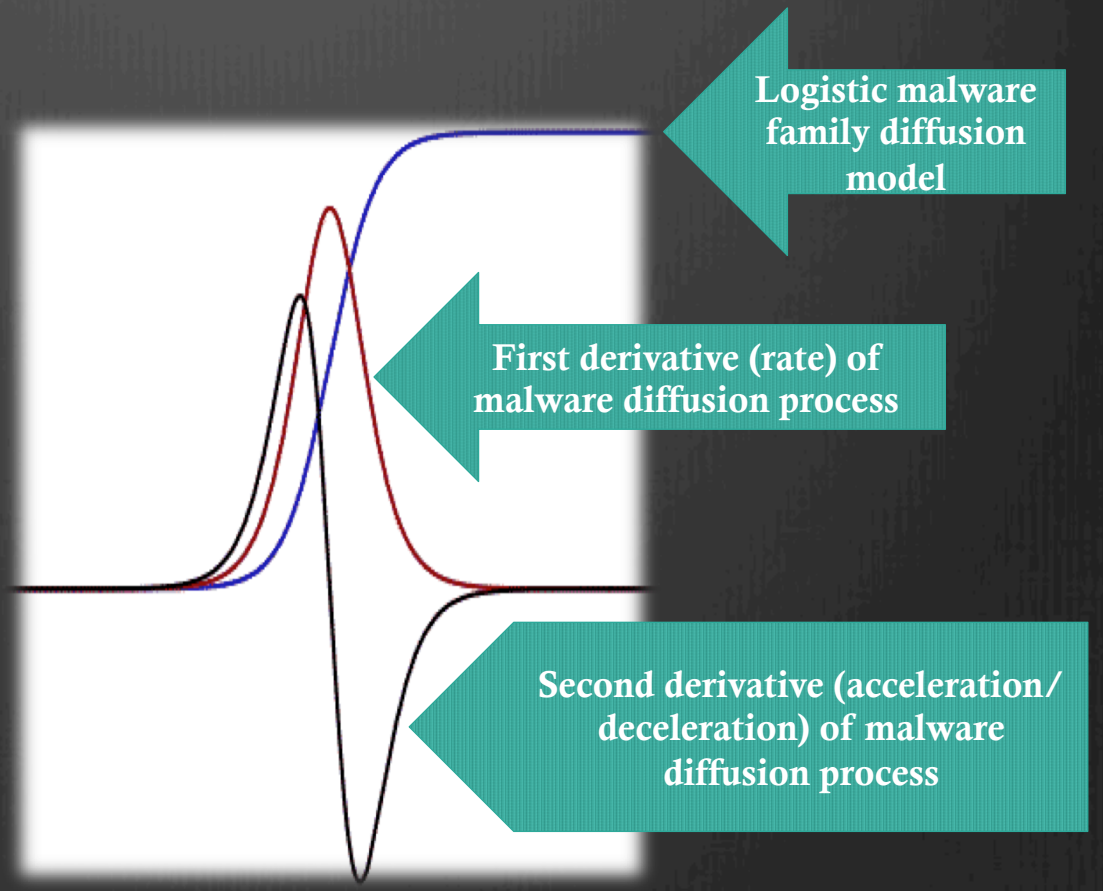
Research dataset

- Used 20 malware families from our 2 million malware sample research dataset
- For each unique MD5 in each family, noted 'first seen' VirusTotal timestamp
- Because we pursue a time series regression approach in this work, this is all the research data that we needed

Predictive modeling approach

Key modeling points:

- We want to observe how many new variants we're observing for each family in each discrete time bin (say, per-month)
- Use linear regression to calculate month by month speed of new variant production
- Second order regression measures acceleration / deceleration in this process
- *Threshold on acceleration determines if takeoff has been entered*

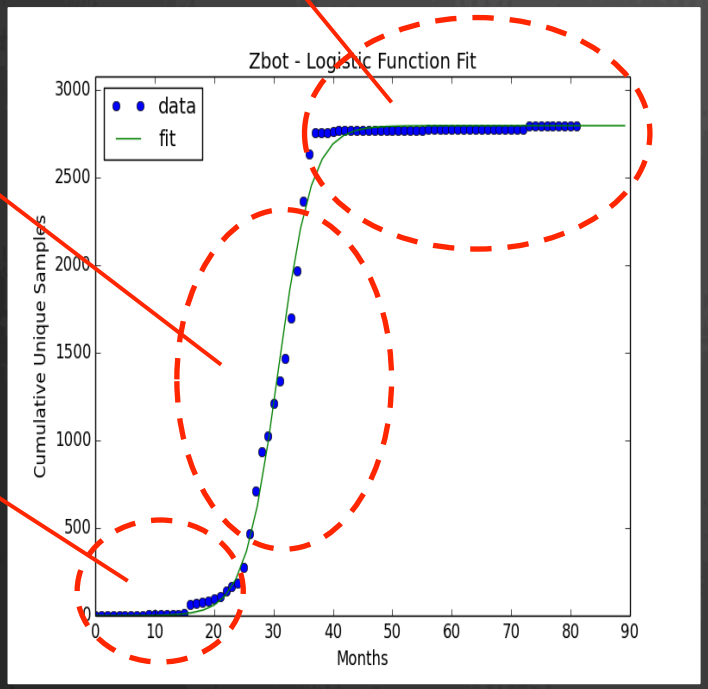


Exemplar malware family: Zbot / Zeus

Late adopters

Rapid growth

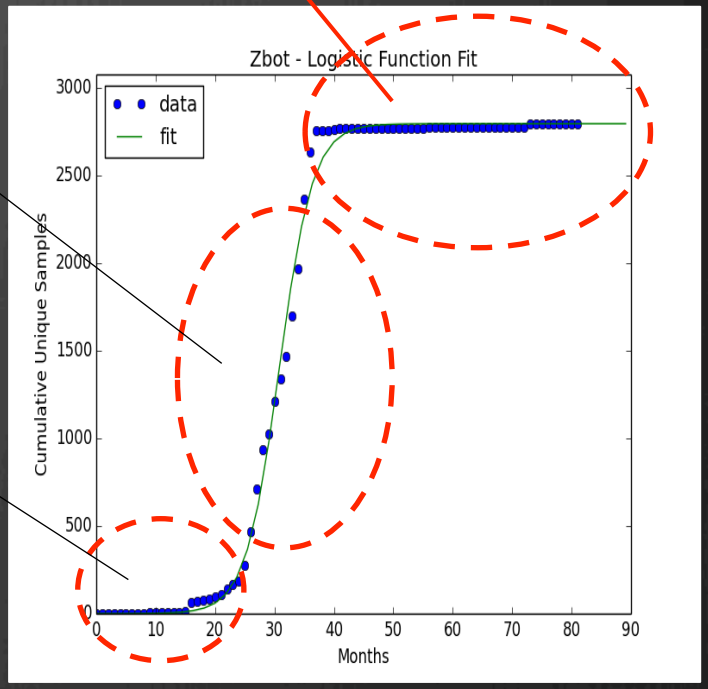
R&D



Creator "retires": Late 2010

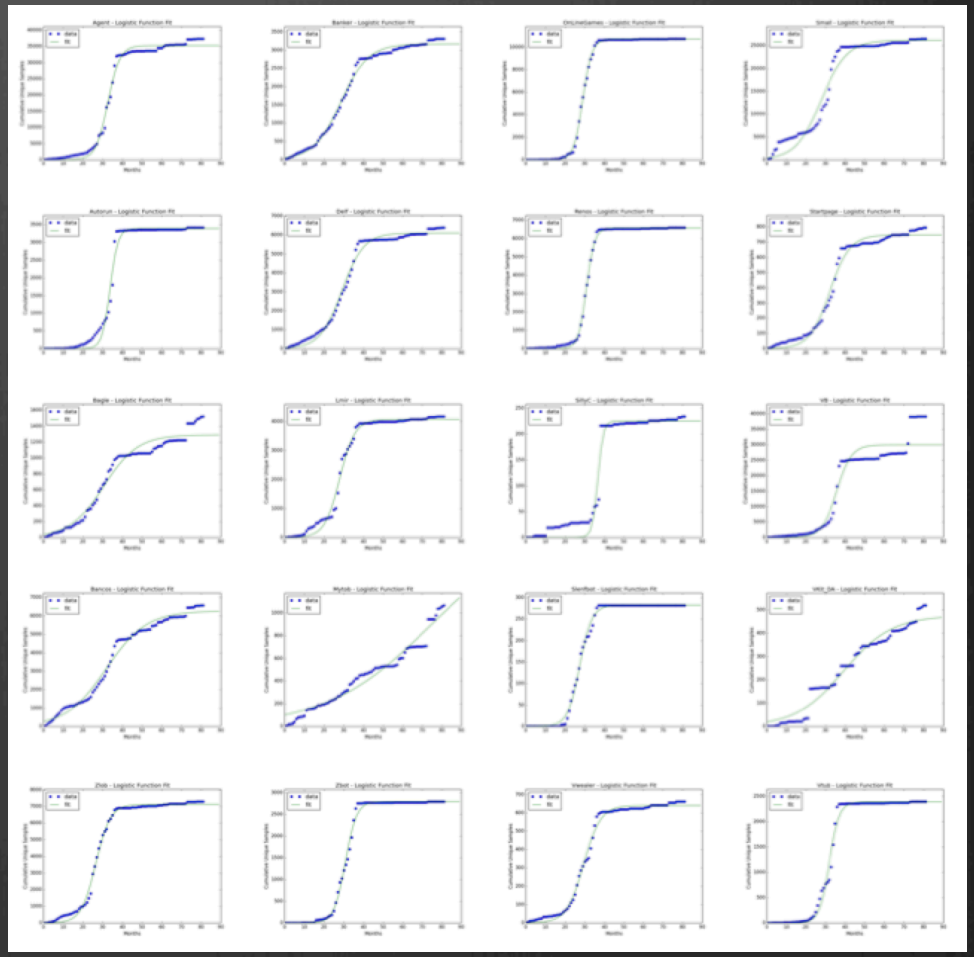
Media notoriety, 2009

First identified July 2007

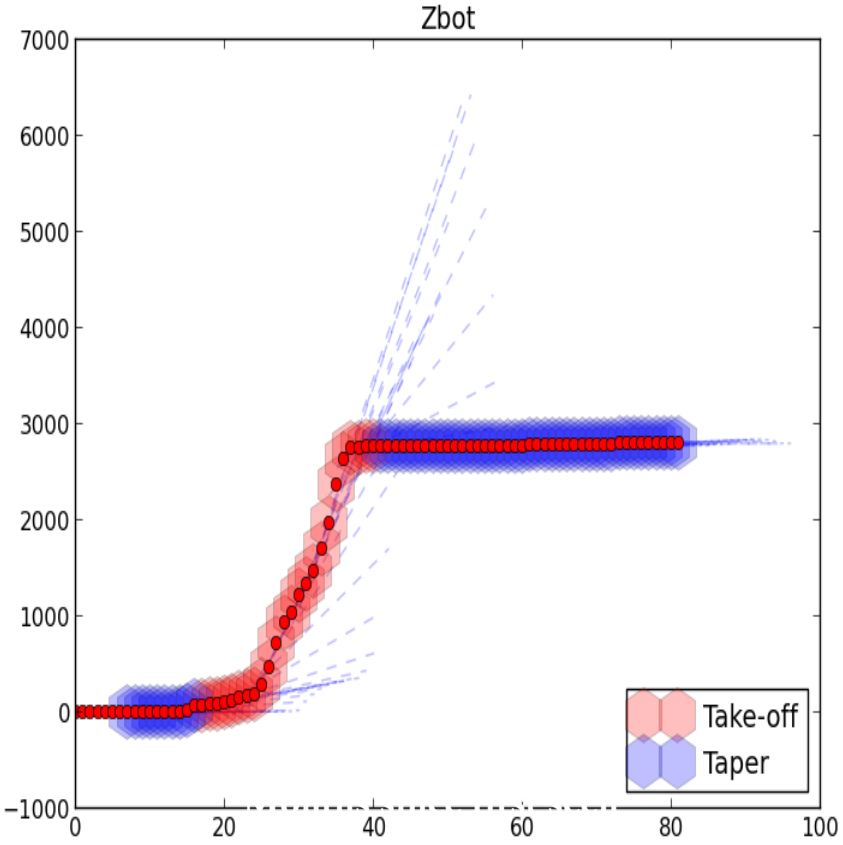


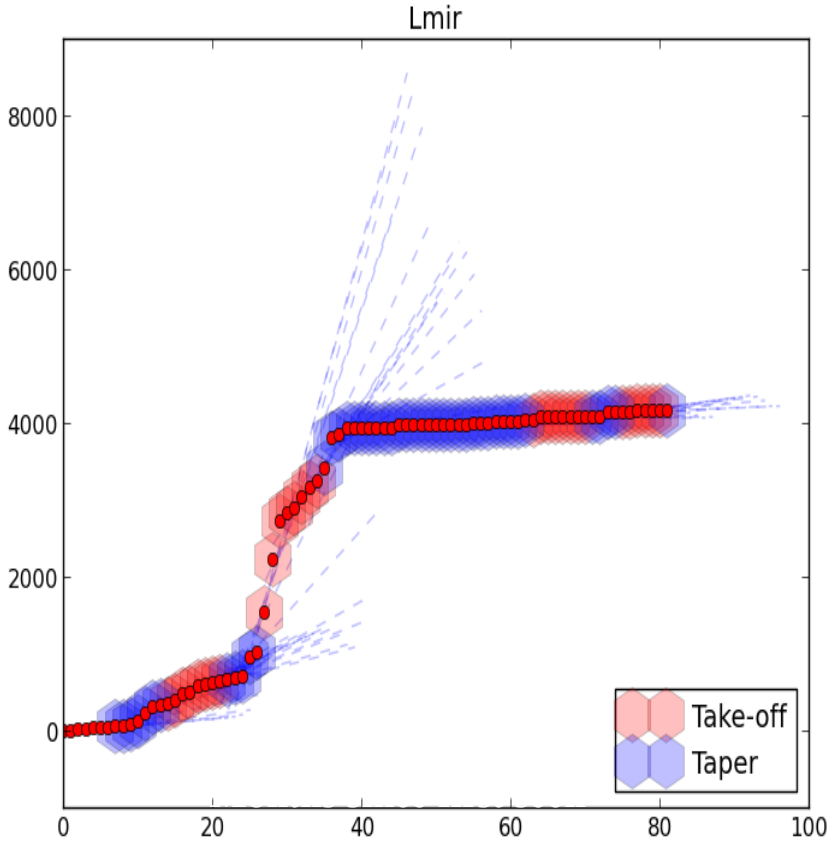
Bird's eye view of 20 malware families 'goodness of fit'

- Considerable variation in shape of data
- Still tends to fit a parametric 'S-shape'
- Given this variation how well can we predict explosion and taper?

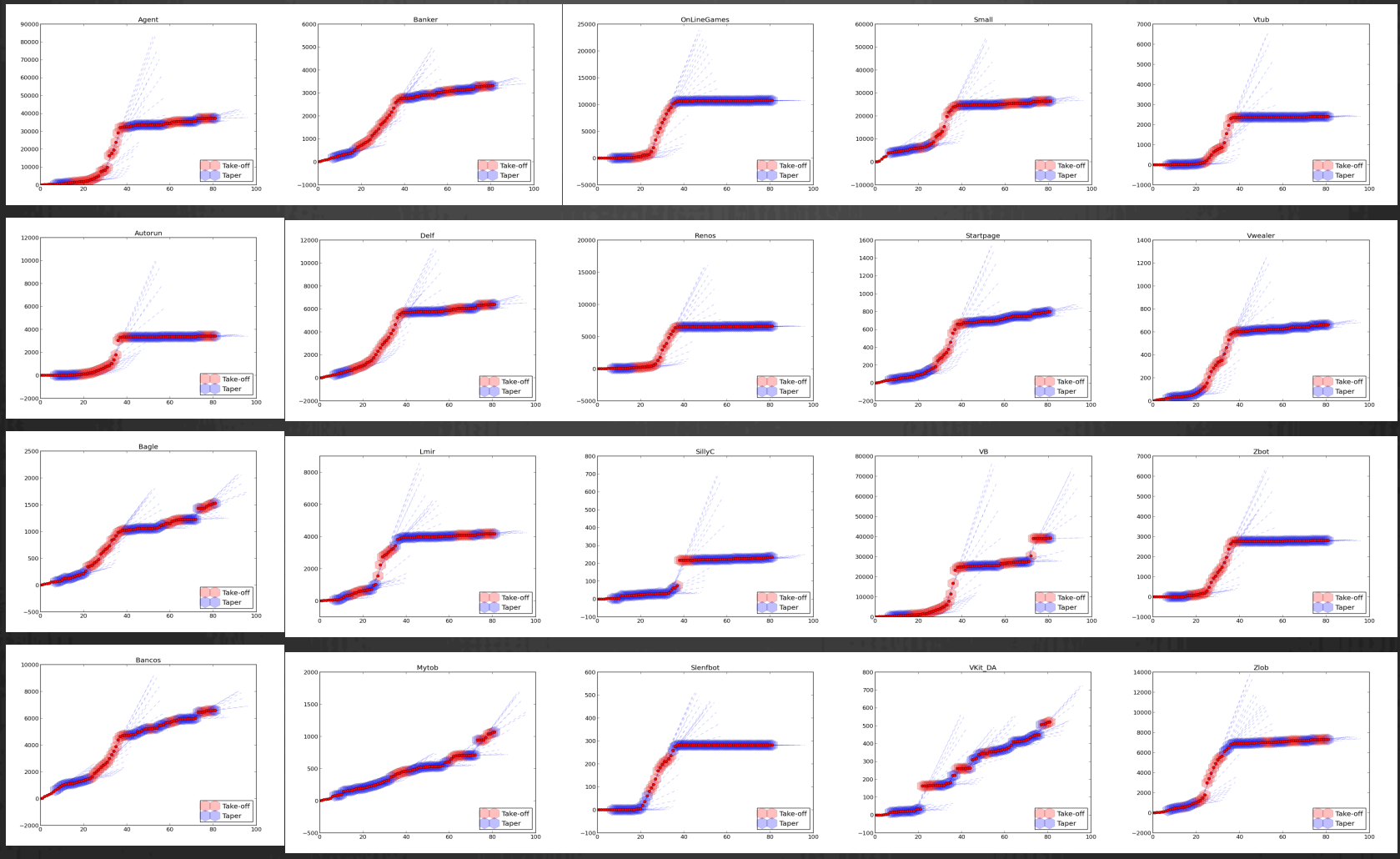


A good case and a bad case: Lmir and Zbot



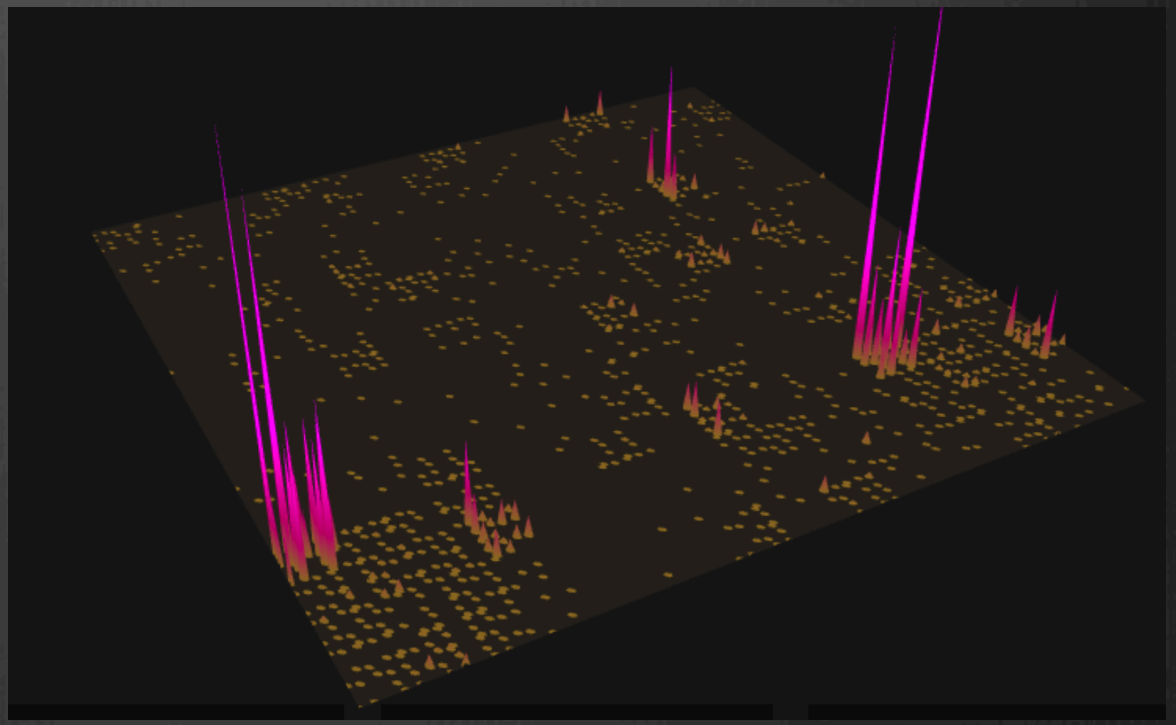


Results on all 20 of our malware families



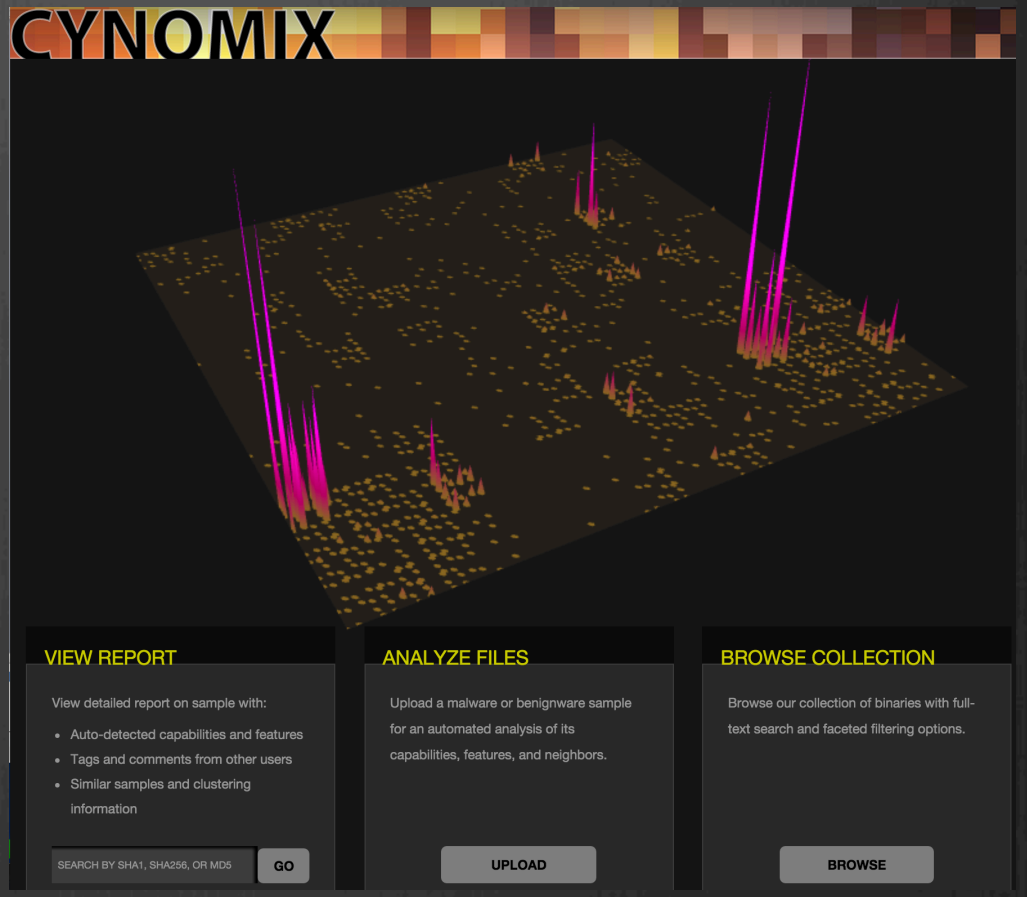
invincea™

Case study 3: A machine learning-aided malware analysis workbench



Capabilities of our research prototype ("Cynomix")

- Performs automatic capability recognition by "learning" from StackOverflow
- Builds a "social network" of malware samples based on shared code relationships

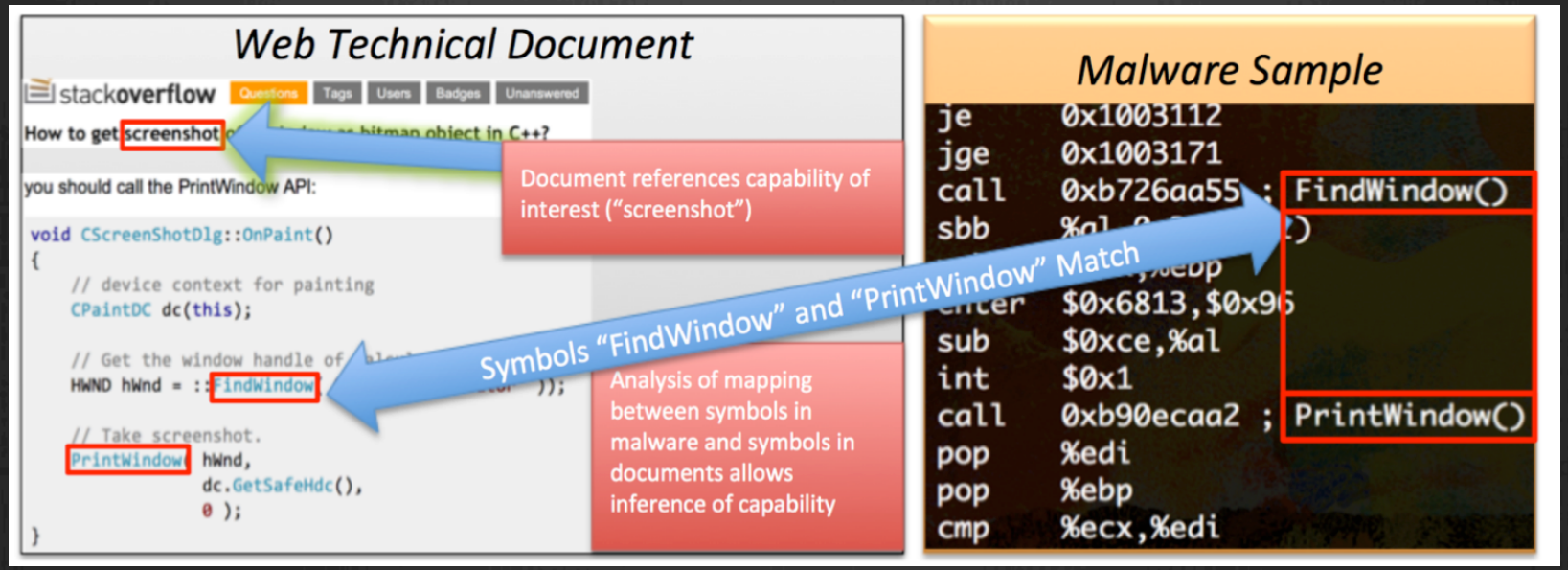


Profiling malware capabilities

Lots of expert knowledge required to reverse engineer malware

Our key insight: this knowledge is expressed on the web

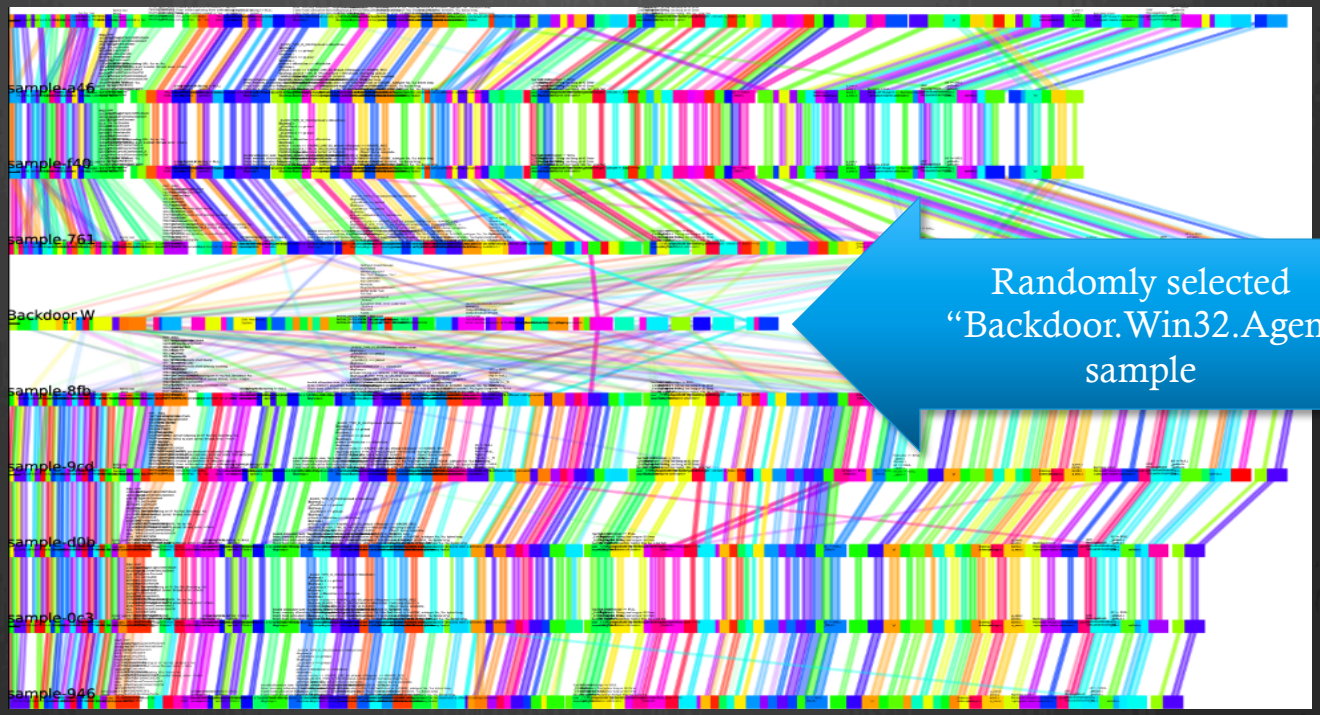
Research and development challenge: computational linguistics and machine learning to extract knowledge, inference model to profile malware samples



Building the malware "social network": identifying shared code between malware samples

Members of the Kbot malware family

Members of the Kbot malware family



Randomly selected "Backdoor.Win32.Agent sample

The approaches we have developed discover genealogical relationships between millions of malware samples based on identification of shared attributes

invincea™

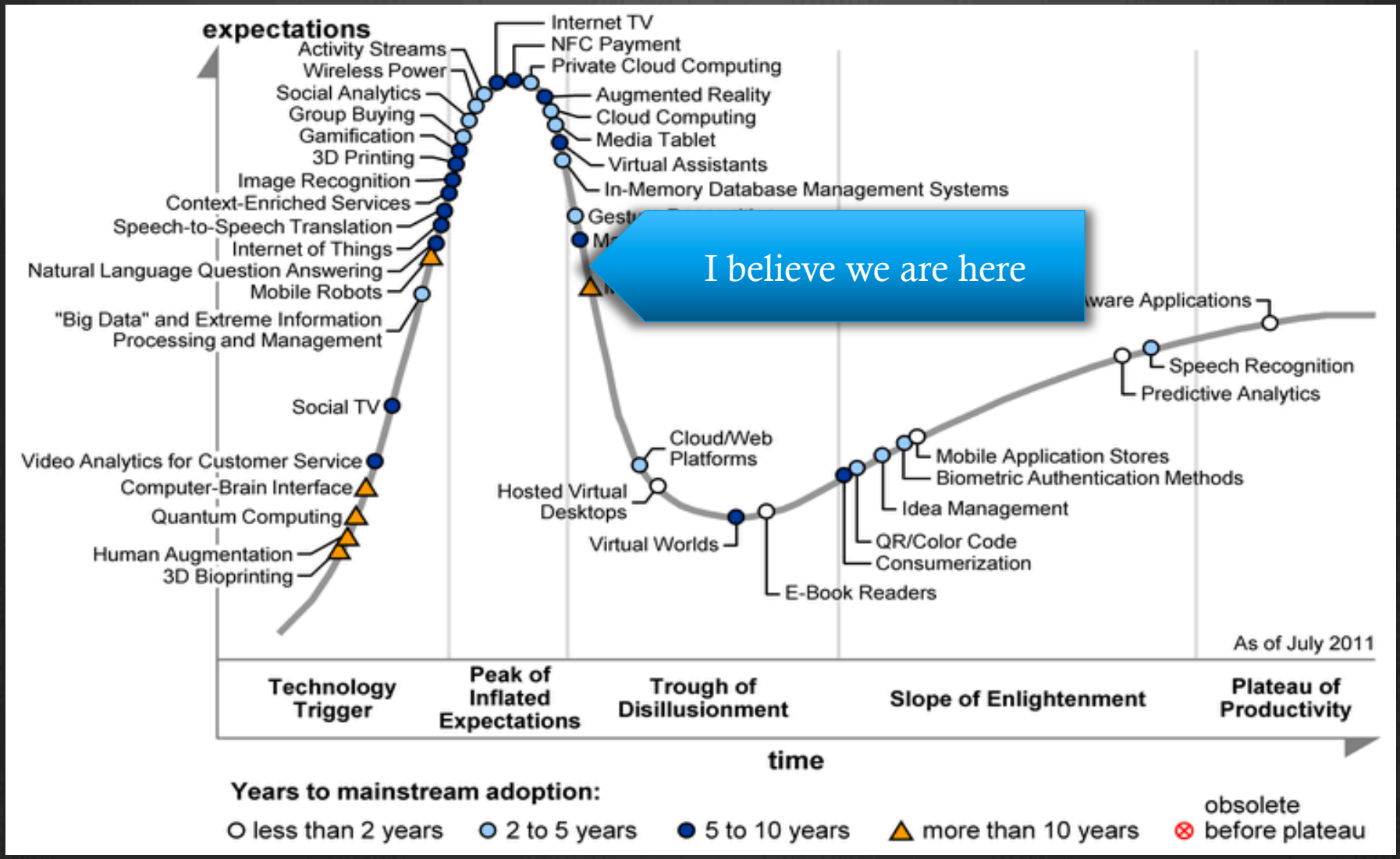
Demonstration / guided tour of malware “social network” visualization and malware capability recognition interface

Conclusion

Takeaways / Sound Bytes

- Security data science holds the promise of shining a spotlight into the reams of "dark matter" security data that we're currently not exploiting, thereby changing the game in network defense
- Security data science is different: it requires ultra-low false positive rates, interpretability, and adversarial modeling
- As data science advances, security will not be unaffected – there is every reason to think that data science will disrupt security just as it has advertising, human computer interface design, and computer vision

Security data science: Where we are in the innovation cycle



Credit: Gartner

Get involved!

- Security data science is a new field in which security professionals of all backgrounds can make an impact
- If you're new to data science, pick up scikit-learn, networkx, numpy, theano, R, or any of the other open source tools, along with a good data mining textbook, and start hacking
- If you're new to security, work with security professionals to find data science applications that make sense