Tricking binary trees: The (in)security of machine learning

SYSTEMS SECURITY GUEST LECTURE 19/10/18
JOE GARDINER (@THECYBERJOE)

Who am I?

- Research Associate in Bristol Cyber Security Group
  - Cyber physical systems security (ICS, IOT, IIOT etc)
- Final year PhD student at Lancaster University
- Previsouly Senior Teaching Associate at Lancaster University
  - Lecture Intro Security, Pentesting, Ethics
- Twitter: @TheCyberJoe

University of BRISTOL

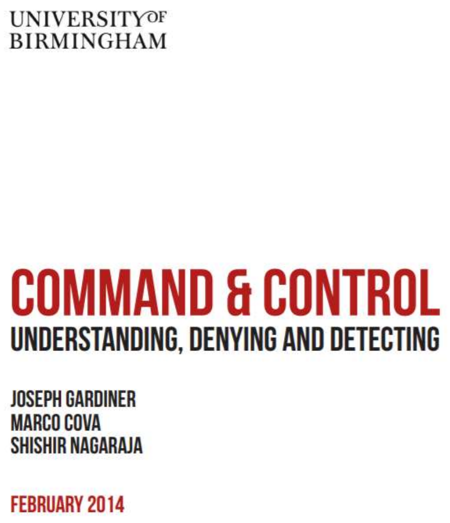
Bristol Cyber Security Group

Security Lancaster

Lancaster University

Background

- Centre for the Protection of National Infrastructure (CPNI) iData project
  - Report on malware command and control
  - Available at c2report.org
- Looked at a lot of botnet detection systems
- Most use machine learning in some way
  - And simple algorithms
- Got me thinking… Is this bad?



UNIVERSITY OF BIRMINGHAM

COMMAND & CONTROL

UNDERSTANDING, DENYING AND DETECTING

JOSEPH GARDINER
MARCO COVA
SHISHIR NAGARAJA

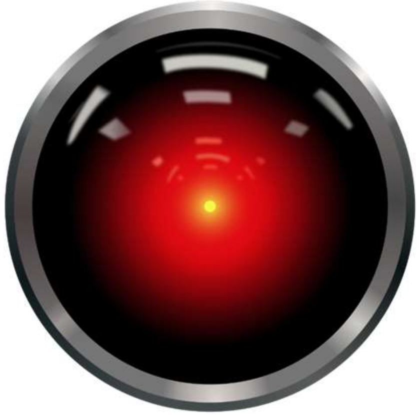
FEBRUARY 2014

Background

- Looked into attacks against machine learning
- Wrote a survey
- Published in ACM Computing Surveys
  - “On the Security of Machine Learning in Malware C&C Detection: A Survey”, J Gardiner and S Nagaraja 2016
- Talk previously presented at BSides Manchester 2018

Agenda

- Why do we use machine learning?
- What is machine learning?
- Attacker models
- The attacks
- Issues of attacks
- Defences
- Questions

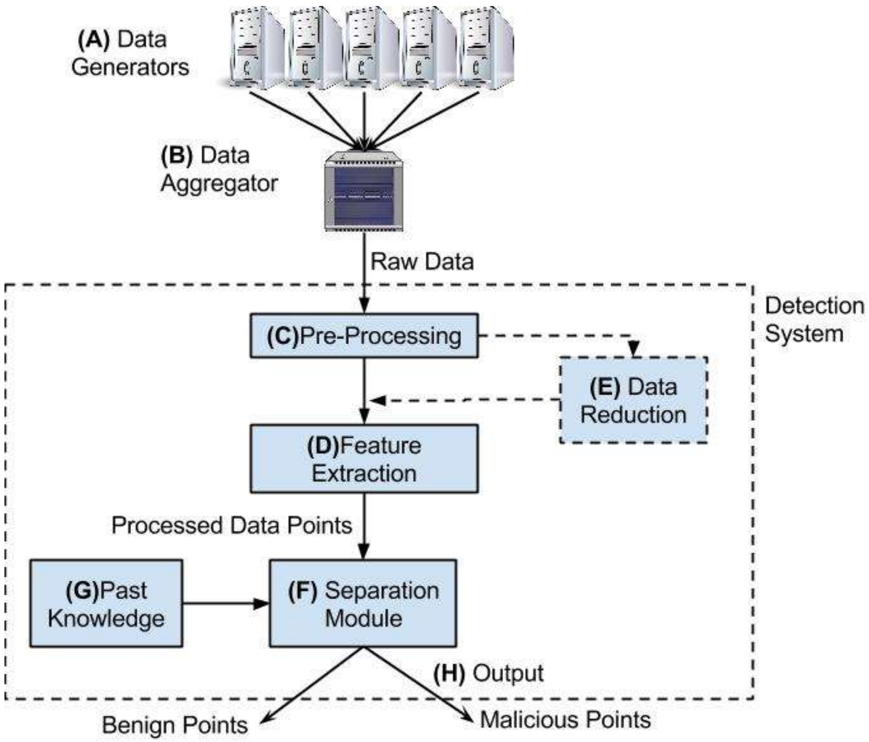


Why do we use machine learning?

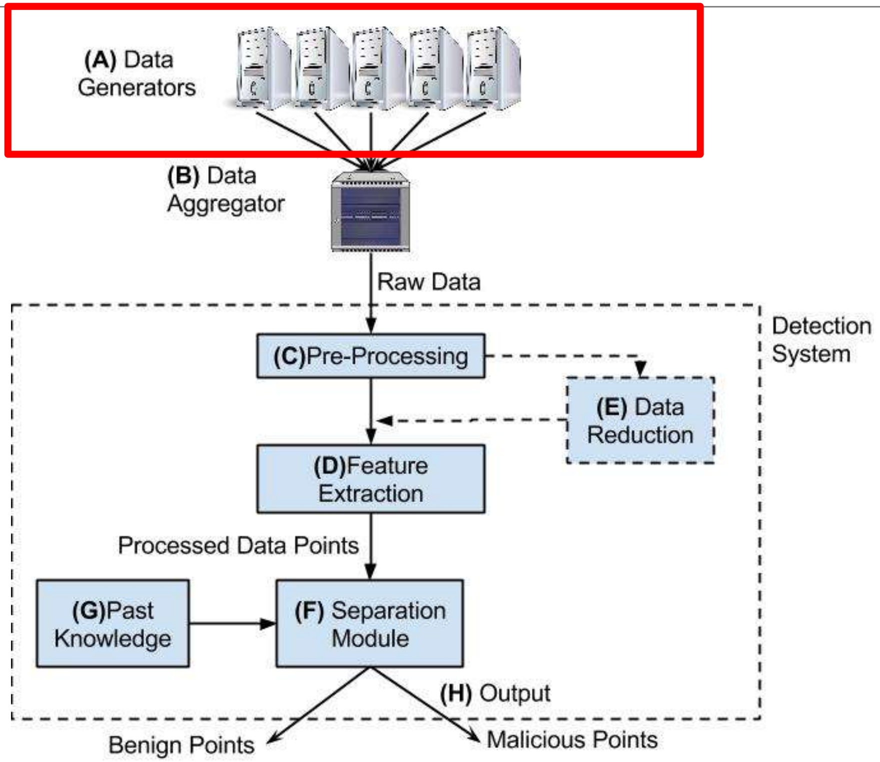
Why machine learning?

- Signature based detection methods are no longer sufficient
  - Thousands of new malware samples daily (polymorphism)
  - Signature databases to cover all samples would be too large
- Too much data for humans to investigate manually
- Machine learning can go some way to alleviate problem

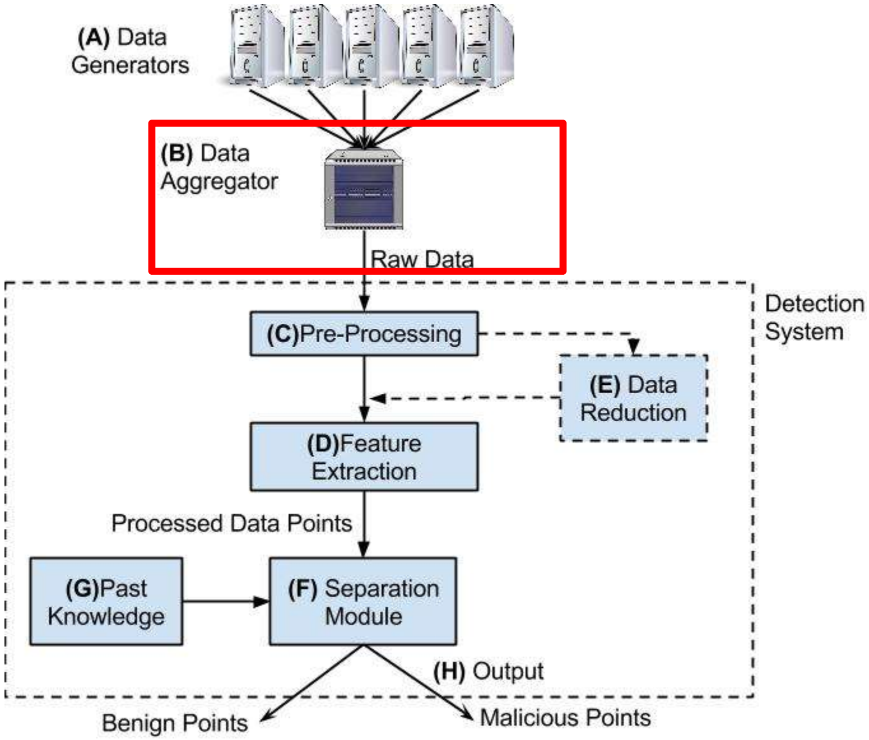
Typical detection system



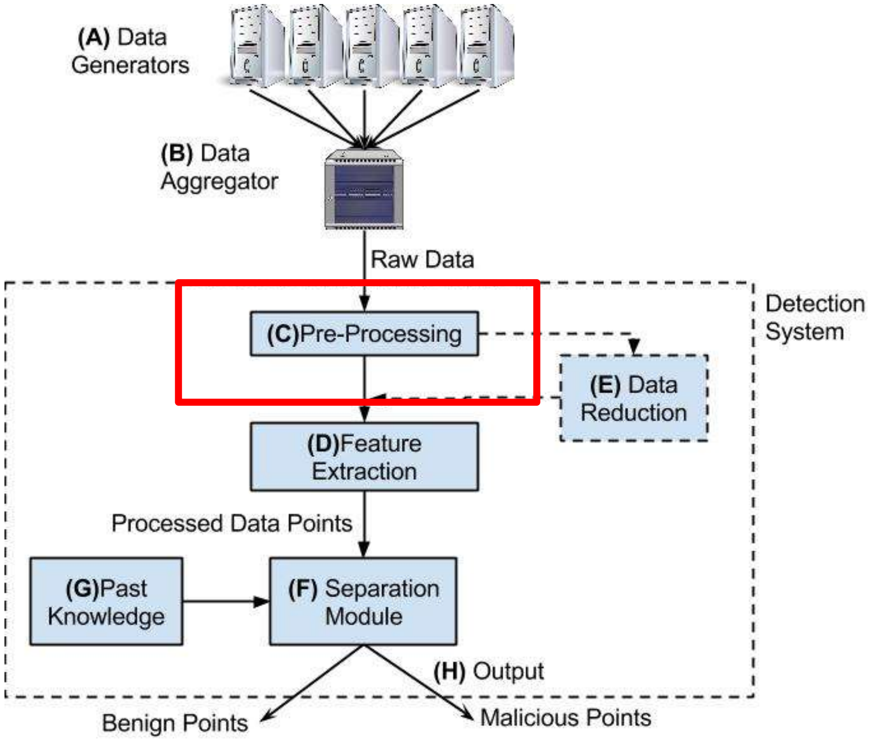
Typical detection system



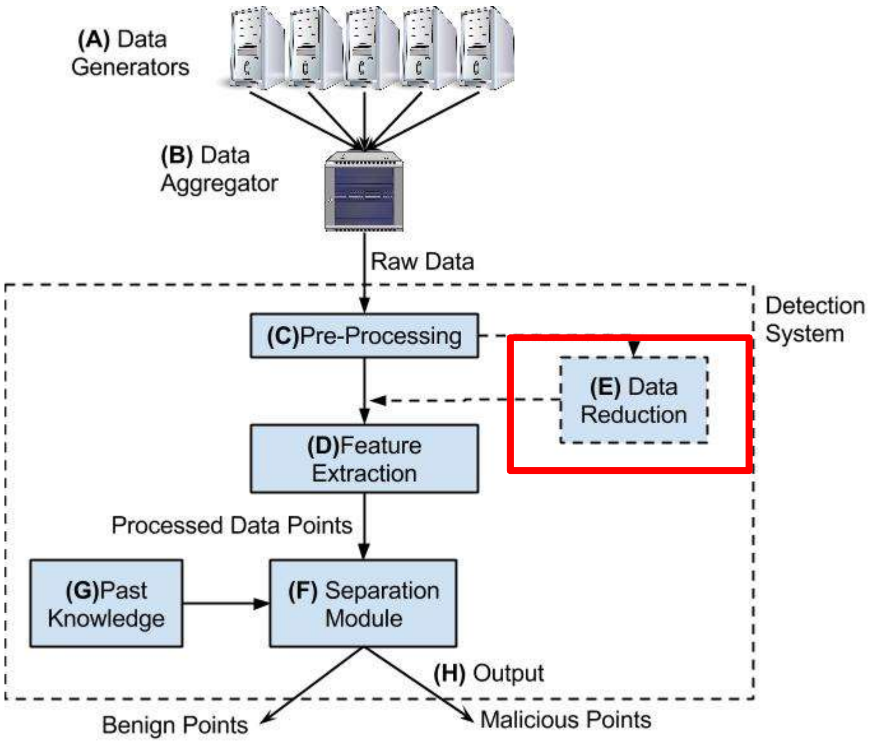
Typical detection system



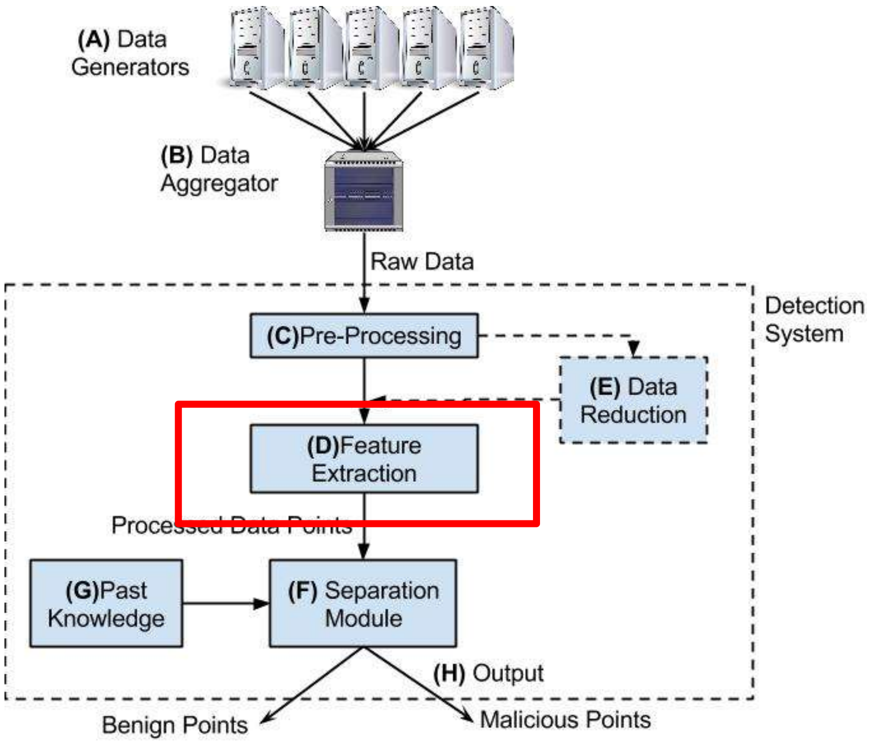
Typical detection system



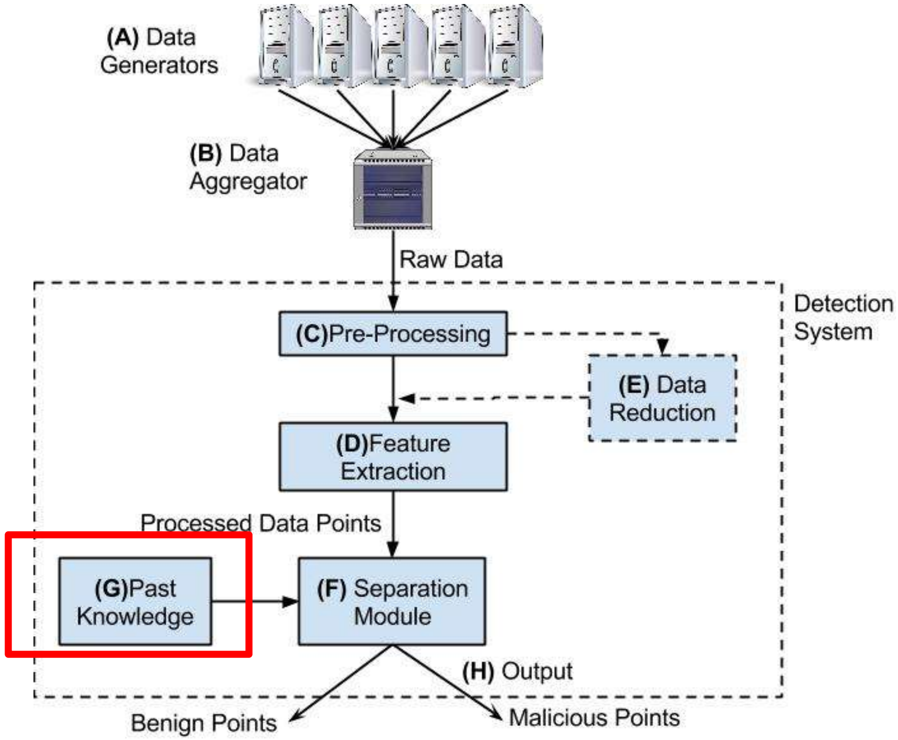
Typical detection system



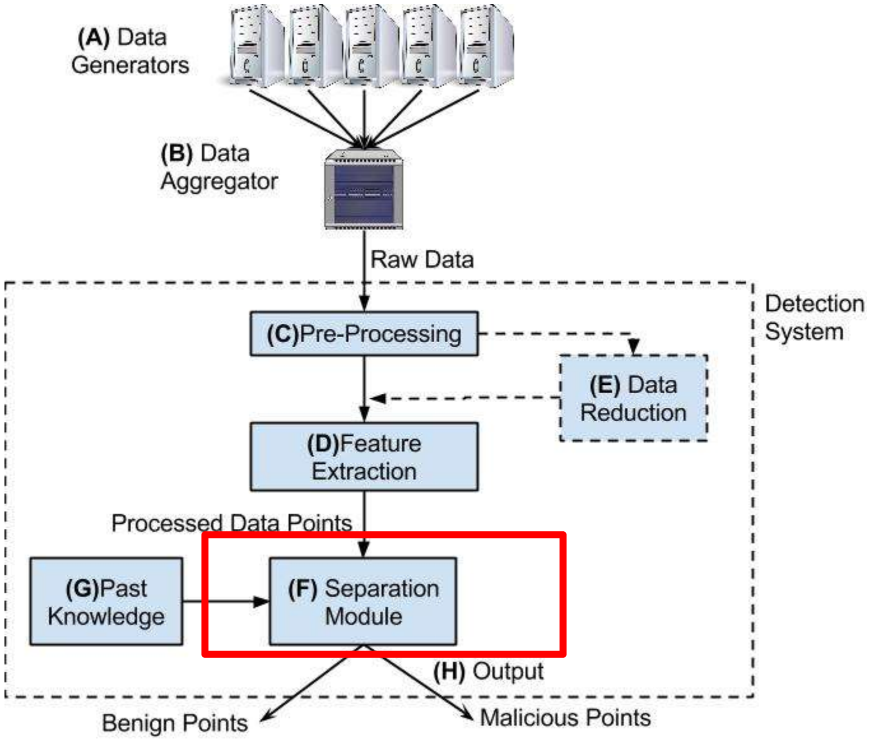
Typical detection system



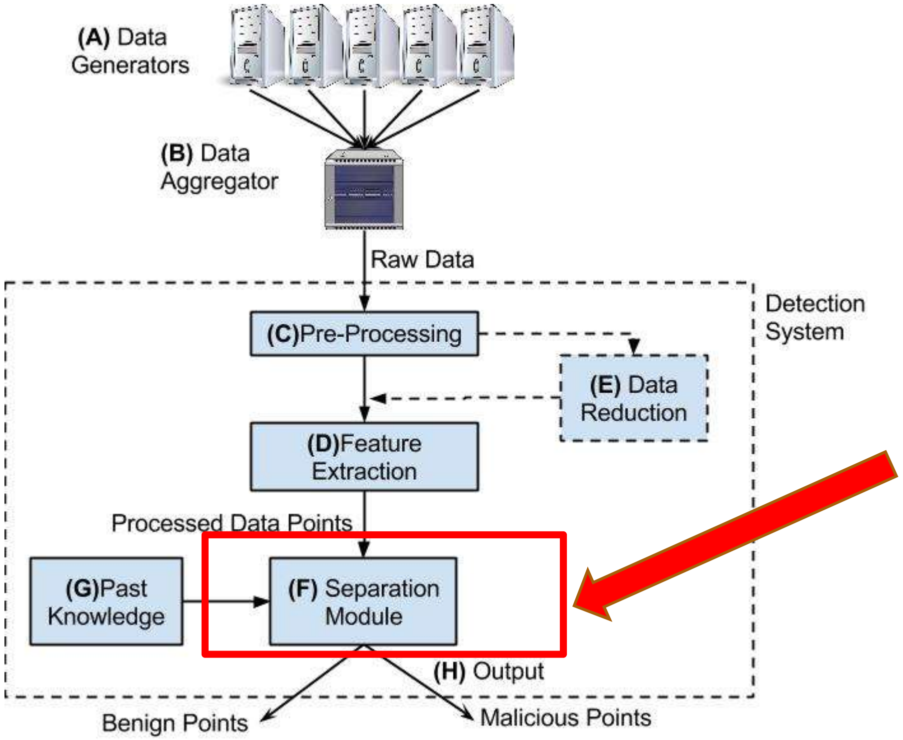
Typical detection system



Typical detection system



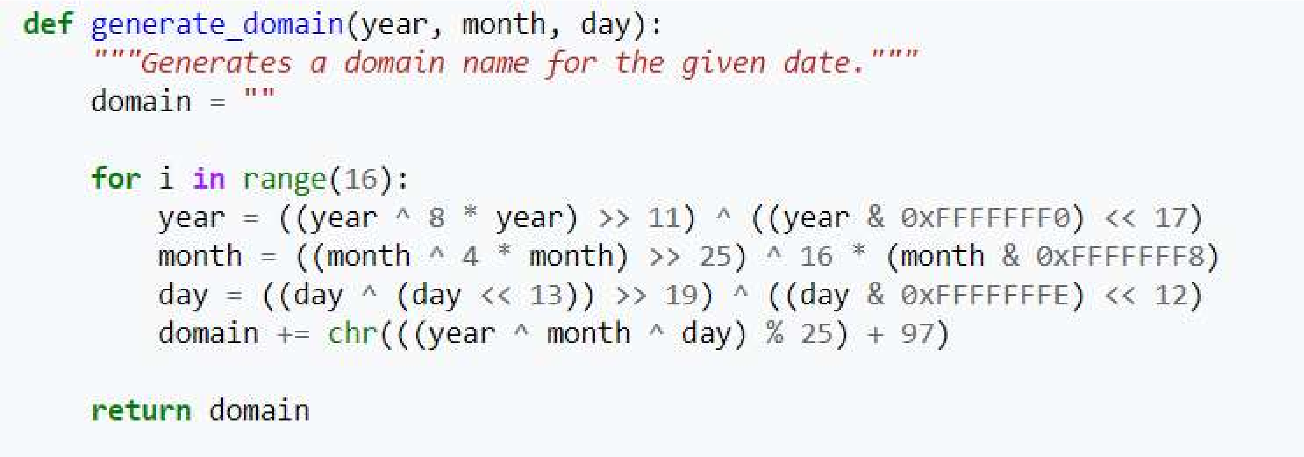
Typical detection system



ML goes here

Example

- Domain generation algorithm (DGA)
- Malware technique for computing domain names for contacting C&C server
  - Used in many famous malware variants, e.g .Conficker, Torpig
- Example:

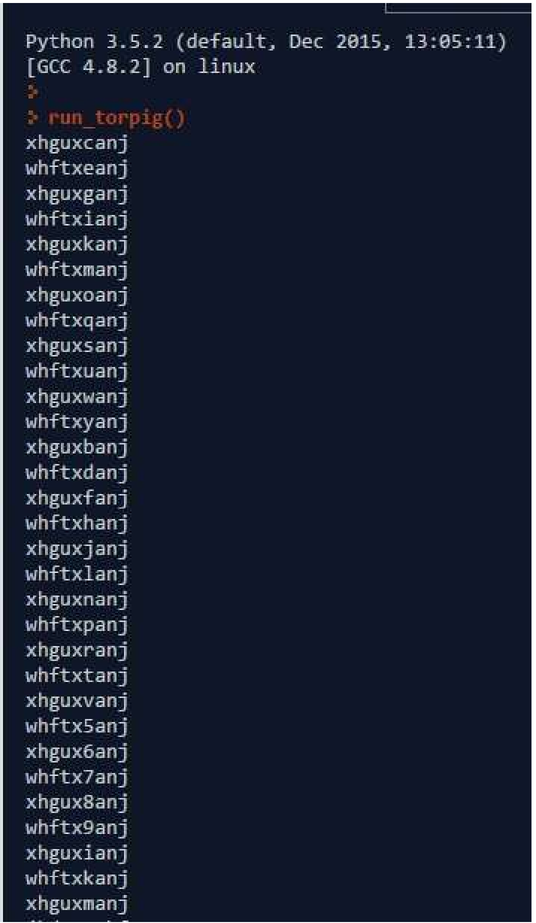


```
def generate_domain(year, month, day):
    """Generates a domain name for the given date."""
    domain = ""

    for i in range(16):
        year = ((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFF0) << 17)
        month = ((month ^ 4 * month) >> 25) ^ 16 * (month & 0xFFFFFFF8)
        day = ((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFE) << 12)
        domain += chr(((year ^ month ^ day) % 25) + 97)

    return domain
```

- Generates domains such as intgmxdeadnxuyla and axwscwsslmiagfah

Example

- DGA domains are usually structured, and easily recognisable
  - E.g Torpig on the right. Last 3 letters are current month, 2nd and 5th letters are h and x, length is always 9 characters.
- Relatively easy to build a signature to recognise domain
- A classifier could also learn how to identify these domains
- As domains are different to regular domains, they can be clustered together using a clustering algorithm



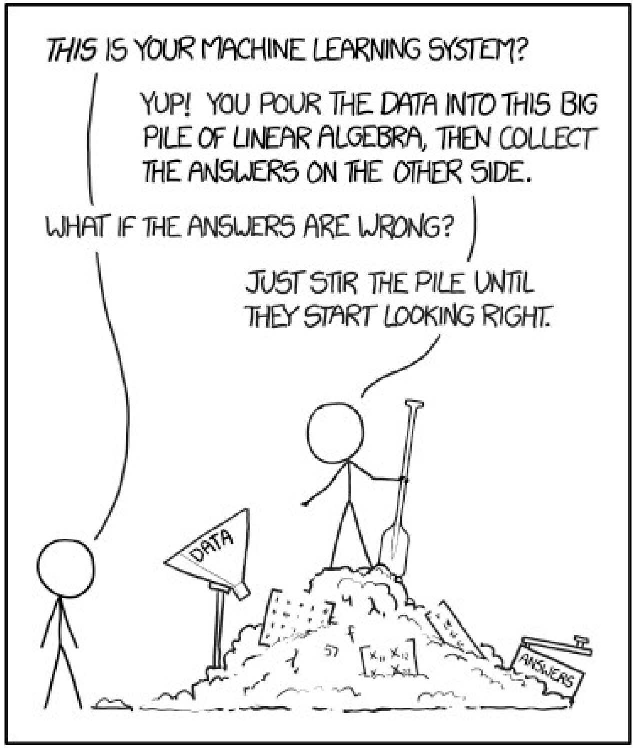
```
Python 3.5.2 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
>
> run_torpig()
xhguxcanj
whftxeanj
xhguxganj
whftxianj
xhguxkanj
whftxmanj
xhguxoanj
whftxqanj
xhguxsanj
whftxuanj
xhguxwanj
whftxyanj
xhguxbanj
whftxdanj
xhguxfanj
whftxhanj
xhguxjanj
whftxlanj
xhguxnanj
whftxpanj
xhguxranj
whftxtanj
xhguxvanj
whftx5anj
xhgux6anj
whftx7anj
xhgux8anj
whftx9anj
xhguxianj
whftxkanj
xhguxmanj
```

What is machine learning anyway?

IT'S JUST IF STATEMENTS RIGHT?

What is it?

- Artificial intelligence
- "Learn" about data, in order to make decisions about new data
- Split into two types:
  - Supervised
  - Unsupervised


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

Features

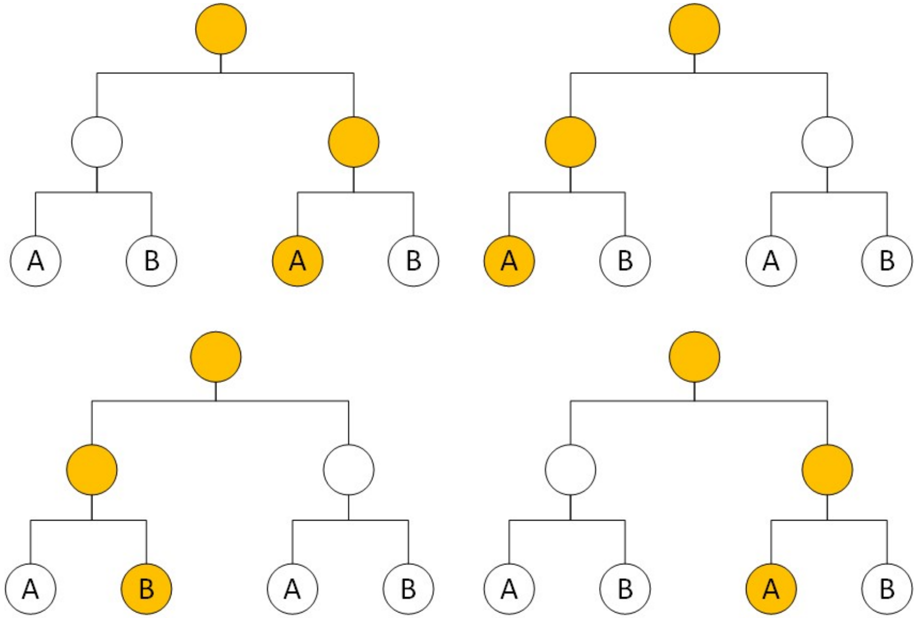
- Individual property of thing being observed
- Collection of features used by algorithm is “feature set”
- For example, a network packet could be represented as
  - Src IP
  - Dst IP
  - Protocol
  - Length
  - Contents tokens
- Could only have a few, potentially thousands

Types of machine learning - supervised

- Have labelled “training data”
- Train system to match input data points to output labels
  - Often referred to as “classification”
- Example algorithms:
  - Decision trees
  - Linear regression
  - Bayes
  - Support Vector Machines (SVM)

Random forest classifier

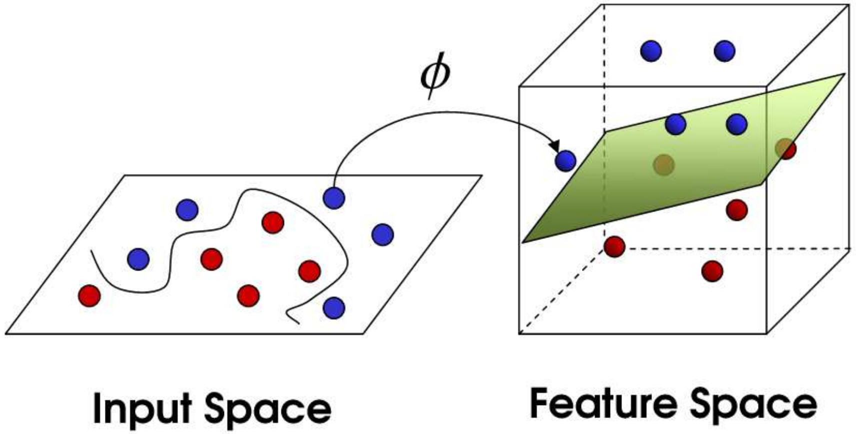
- Supervised learning
- Generate multiple decision trees
  - Each uses a subset of the features/training data
- Pass data point through all trees
- Majority vote to assign label



3 As 1 B -> Assign A

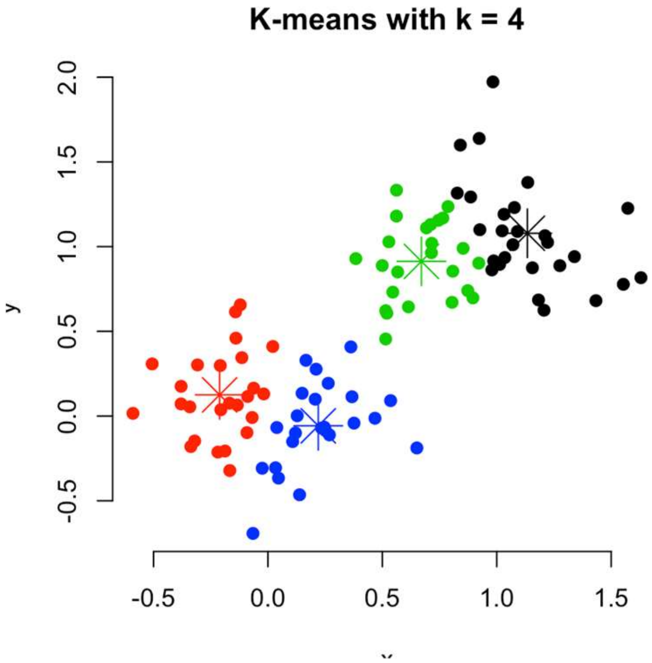
Support vector machines (SVM)

- Supervised learning
- Produces a hyperplane separating points of two classes
- New points are classified by seeing which side they fall of hyperplane



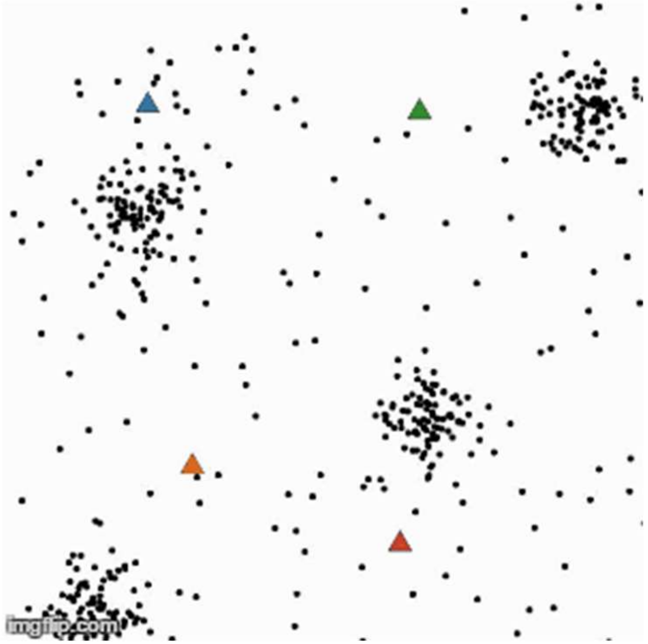
Types of machine learning - unsupervised

- Operates on unlabelled data, attempting to find structure
- Try and separate data points of different classes
- Primary example is clustering
  - Algorithms such as k-means, x-means, hierarchical etc
- Harder to evaluate
  - No labels!



K-means clustering

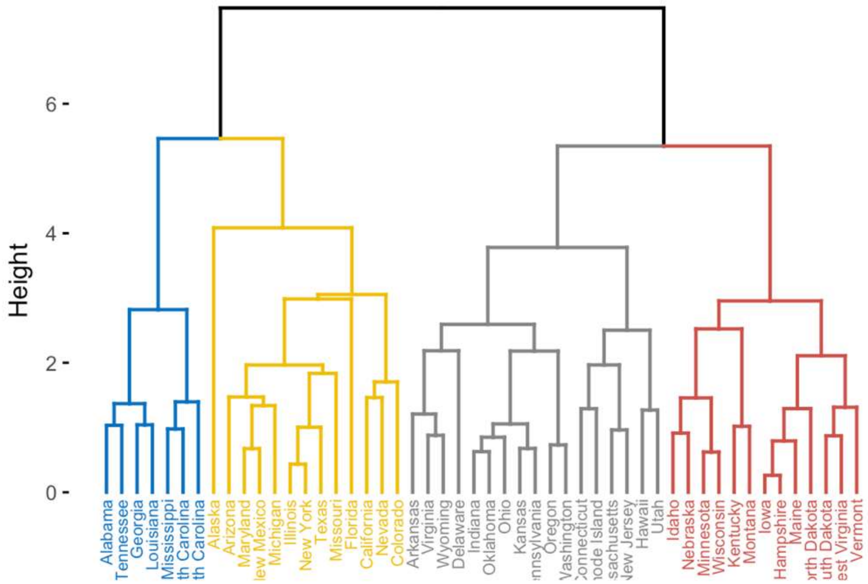
- Unsupervised learning
- Simple algorithm
    - Generate k random points (centroids), and assign all data points to nearest centroid
    - Move centroids to mean of assigned points
    - Repeat until centroid stop moving
- X-means variants also finds best value for k



Hierarchical clustering

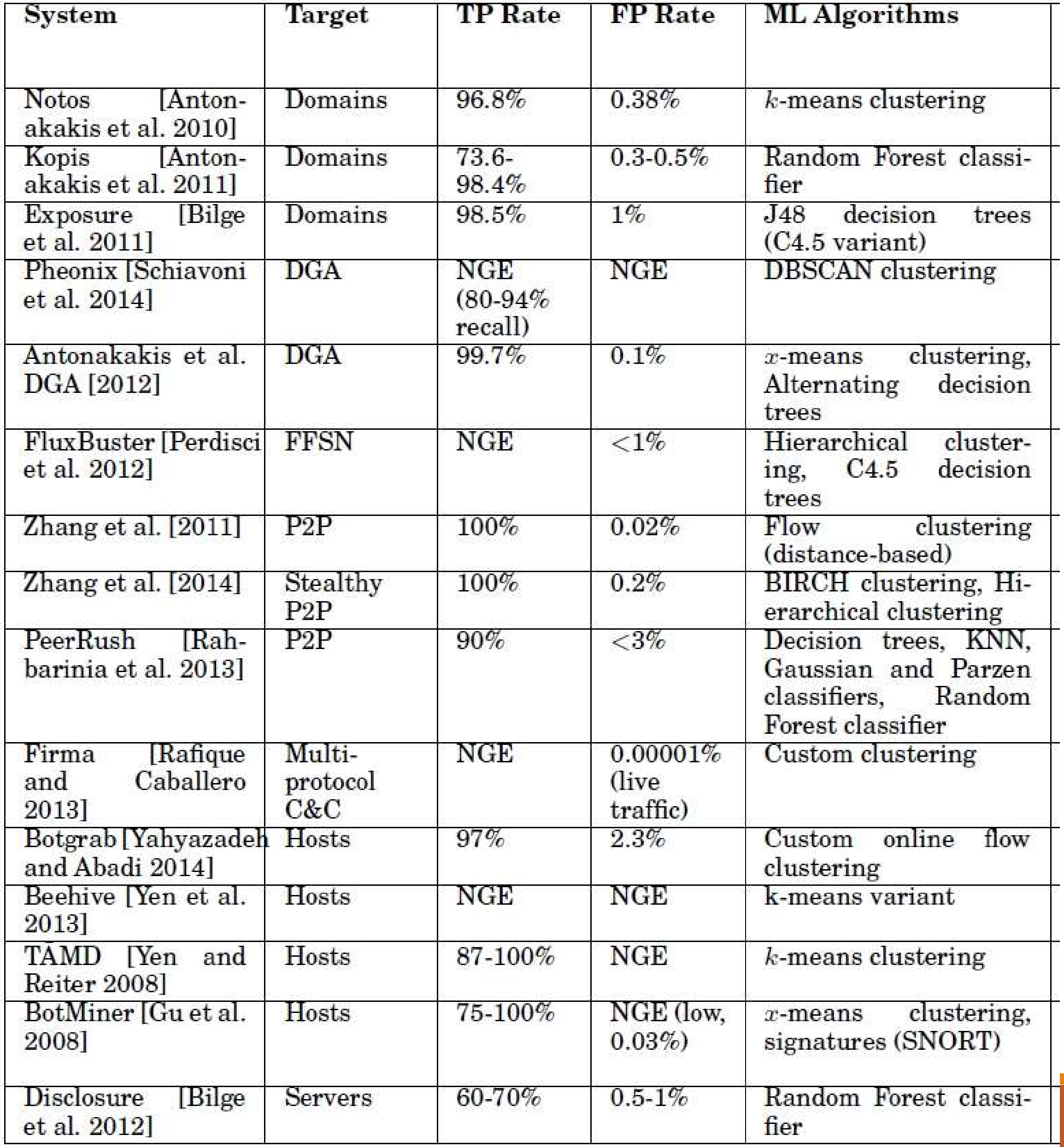
- Unsupervised learning
- Builds a hierarchy of cluster
- Usually represented as dendrogram
- Each data point starts as own cluster
- Each layer represents merging of two closest clusters from layer below
- Number of clusters is decided by which level you read at

Cluster Dendrogram



How do we measure performance

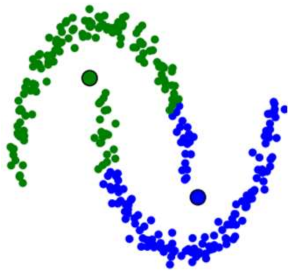
- True positive rate
  - Number of malicious points labelled as malicious (high is good)
- False positive rate
  - Number of benign (good) point labelled as malicious (low is good)
- True negative rate
  - Number of benign points labelled as benign (high is good)
- False negative rate
  - Number of malicious point labelled as benign (low is good)

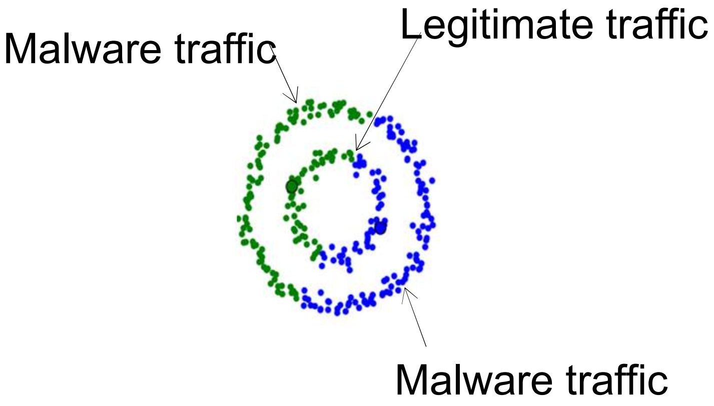


System	Target	TP Rate	FP Rate	ML Algorithms
Notos [Antonakakis et al. 2010]	Domains	96.8%	0.38%	k-means clustering
Kopis [Antonakakis et al. 2011]	Domains	73.6-98.4%	0.3-0.5%	Random Forest classifier
Exposure [Bilge et al. 2011]	Domains	98.5%	1%	J48 decision trees (C4.5 variant)
Pheonix [Schiavoni et al. 2014]	DGA	NGE (80-94% recall)	NGE	DBSCAN clustering
Antonakakis et al. DGA [2012]	DGA	99.7%	0.1%	x-means clustering, Alternating decision trees
FluxBuster [Perdisci et al. 2012]	FFSN	NGE	<1%	Hierarchical clustering, C4.5 decision trees
Zhang et al. [2011]	P2P	100%	0.02%	Flow clustering (distance-based)
Zhang et al. [2014]	Stealthy P2P	100%	0.2%	BIRCH clustering, Hierarchical clustering
PeerRush [Rahbarinia et al. 2013]	P2P	90%	<3%	Decision trees, KNN, Gaussian and Parzen classifiers, Random Forest classifier
Firma [Rafique and Caballero 2013]	Multi-protocol C&C	NGE	0.00001% (live traffic)	Custom clustering
Botgrab [Yahyazadeh and Abadi 2014]	Hosts	97%	2.3%	Custom online flow clustering
Beehive [Yen et al. 2013]	Hosts	NGE	NGE	k-means variant
TAMD [Yen and Reiter 2008]	Hosts	87-100%	NGE	k-means clustering
BotMiner [Gu et al. 2008]	Hosts	75-100%	NGE (low, 0.03%)	x-means clustering, signatures (SNORT)
Disclosure [Bilge et al. 2012]	Servers	60-70%	0.5-1%	Random Forest classifier

Assumptions

- Separation
  - There should be little to no overlap between malicious and legitimate traffic behavior.
  - Hierarchical clustering and Birch classification can easily deal with this
  - Linearity
    - Data points exist in linear space.





Attacker models

What does the attacker want to do?

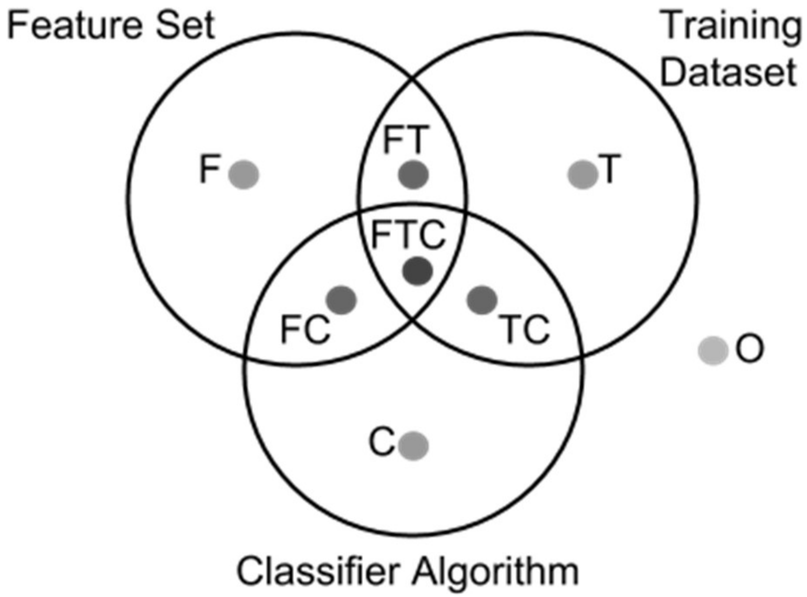
- Two main goals:
- Evade detection
  - Cause their attack point to be mislabelled as benign
  - Increase false negative rate
- Denial-of-service
  - Increase number of false positives to prevent system use
  - E.g. a DNS detector at a large organisation with 1 billion requests per day
  - FP rate of 0.01% = 100000 alerts per day
  - Admins will turn it off

Barreno model for classifying attacks

		Description
Influence	Causative	Alter training process through influence over training data
	Exploratory	Use probing or offline analysis to discover information
Specificity	Targeted	Focus on a particular set of points
	Indiscriminate	No specific target, flexible goal e.g. increase false positives
Security Violation	Integrity	Result in attack points labelled as normal (false negatives)
	Availability	Increase false positives and false negatives so system becomes unusable

Marco Barreno, Blaine Nelson, Russell Sears, Anthony D. Joseph, and J. D. Tygar. 2006. Can Machine Learning Be Secure? (2006)

Attacker knowledge



Nedim Srndic and Pavel Laskov. Practical Evasion of a Learning-Based Classifier: A Case Study. (2014)

Attacker capability

According to Biggio et al (for classifiers)

1. The attacker influence in terms of causative or exploratory.
2. Whether (and to what extent) the attack affects the class priors.
3. The amount of and which samples (training and testing) can be controlled in each class.
4. Which features, and to what extent, can be modified by the adversary

Also applicable to clustering (with the exemption of (2))

B. Biggio, G. Fumera, and F. Roli. Security Evaluation of Pattern Classifiers under Attack. IEEE Transactions on Knowledge and Data Engineering (2014).

Some terminology

- Learner
  - The target machine learning algorithm
- Production learner
  - Instance of learner in use by the target
- Surrogate learner
  - A local copy of the target learner, with the accuracy depending on the attacker knowledge. May not be exact same algorithm as target learner, and may use an estimated dataset for training/testing

The attacks

Mimicry attack

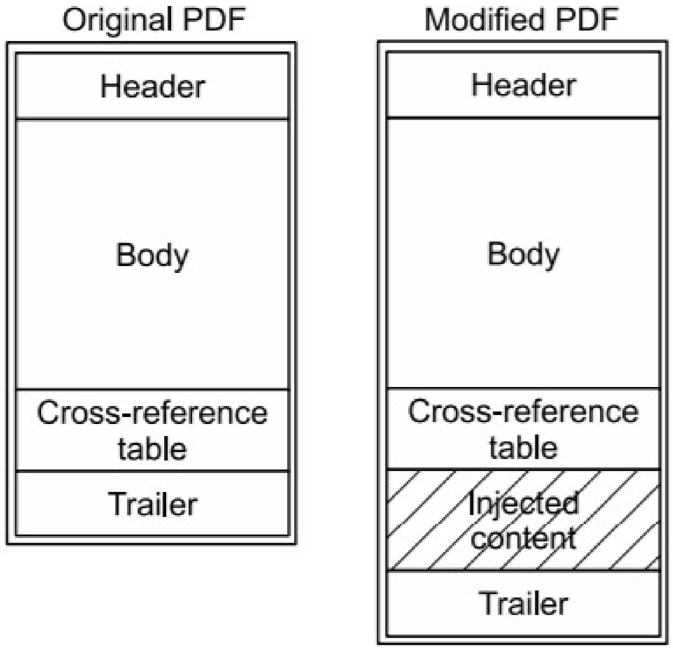
- Exploratory integrity attack
  - Targeted or indiscriminate
- Attempt to change attack point so that it resembles benign point
- Demonstrated against random forest, SVM, bayes, neural networks
- Theoretically applicable to most classifier variants
- Limited by attackers ability to modify feature values

PDFRate

- (Now defunct) website that analysed PDF files
- Uses random forest classifier to assign a score indicating maliciousness

Attacking PDFRate

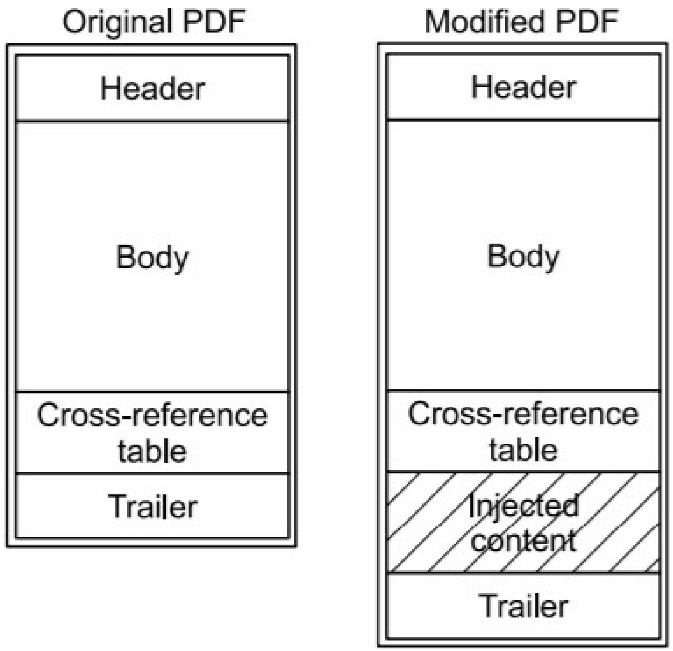
- Mimicry attack
  - Pick a legitimate PDF file and change features in malicious file to match
  - Goal is to reduce score outputted by PDFRate
- Main difficulty: PDF features are interlinked
  - Changing one feature may affect many others
- Attack files developed using offline surrogate learner



Img Src: Practical Evasion of a Learning Based Classifier: A Case Study, Srndic and Laskov 2014

Attacking PDFRate

- Content is injected into region between CRT and trailer
  - Area is read by PDFRate, but ignored by PDF viewers
- Can increment 33 features, and arbitrarily modify 35
- For example, if attack file has 5 `obj` keywords, and target 7, attack string "`obj obj`" is injected
  - `Count_obj` feature is now 7
- `Author` metadata field length can be reduced to 3 by adding "`/Author(abc)`"
  - PDFRate uses last seen metadata



Img Src: Practical Evasion of a Learning Based Classifier: A Case Study, Srndic and Laskov 2014

Attacking PDFRate

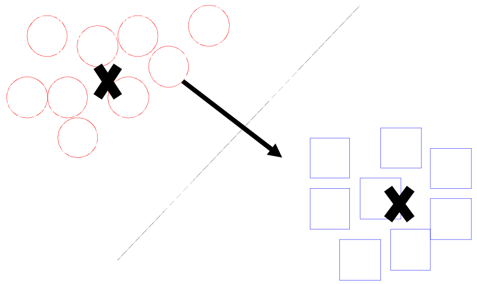
- Surrogate learner
  - Feature set is known (70% of features are described in original paper)
  - Benign and malicious PDF files taken from web
  - Tested with both random forest and SVM as surrogate learner
    - Measure effect of knowledge of target
- Attack points derived with random forest surrogate can reduce score by 28-42%
- Lower, but still significant, reduction when using SVM-based surrogate
- Available as a library
  - https://github.com/srndic/mimicus

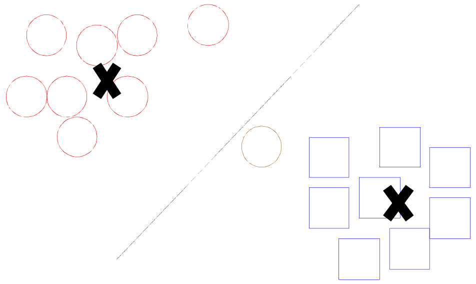
Mimicry – other examples

- Biggio et al (2013) test mimicry against svm and neural networks in the perfect knowledge (PK) and limited knowledge (LK) cases
  - Iterative approach with gradient descent component
  - Even in limited knowledge case can increase FN rate to 0.5, often higher
- Wright et al (2009) attack Bayes by changing traffic features to emulate benign traffic (identifying web pages by traffic volume)
  - Reduces accuracy from 98% to 4%, or 63% if classifier is trained with attack samples.

Mimicry - clustering

- Almost identical approach to the classifier version
- Aims to reduce distance between attack and benign points to the point where they are clustered together
- Demonstrated against single-linkage hierarchical clustering
- Theoretically applicable to other common algorithms based on distance functions
- Limited by attacker knowledge of target clusters and ability to change feature values





Mimicry - clustering

- Biggio et al. (2013) attack single-linkage hierarchical clustering
- Draw line between target and attack point, and move attack point along this line until it is within a distance threshold to target
- Use both perfect knowledge (PK) and limited knowledge (LK) cases
- Tested on handwritten digits (represented as 28x28 greyscale images)
- Successfully merges attack samples into target clusters with limited modifications

Gradient descent attacks

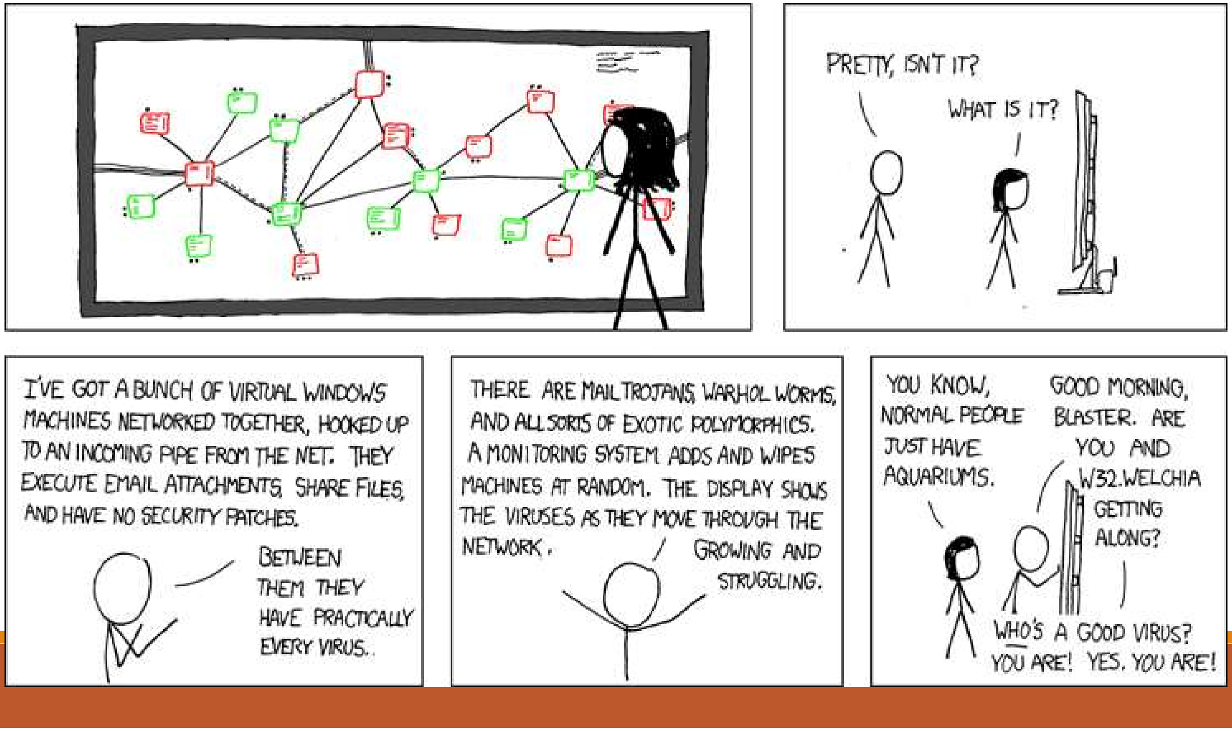
- Exploratory integrity attack
- Apply a gradient descent optimisation algorithm to find optimal attack point, changing features until point is misclassified
- Limited by the attacker knowledge and ability to change feature values
  - Requires a surrogate classifier and effectiveness is reduced as the accuracy of the surrogate dataset is reduced

Gradient descent attacks

- Srndic and Laskov (2014) test against PDF rate, with score reduced by 29-35%
- Biggio et al. (2013) test against SVM and neural networks
  - On SVM can increase FN rate to close to 1 with a limited number of iterations.
  - Neural networks more robust, achieving at best an FN rate of 0.3 in the perfect knowledge case
    - Possibly because attack point at local minimum is too far from decision boundary

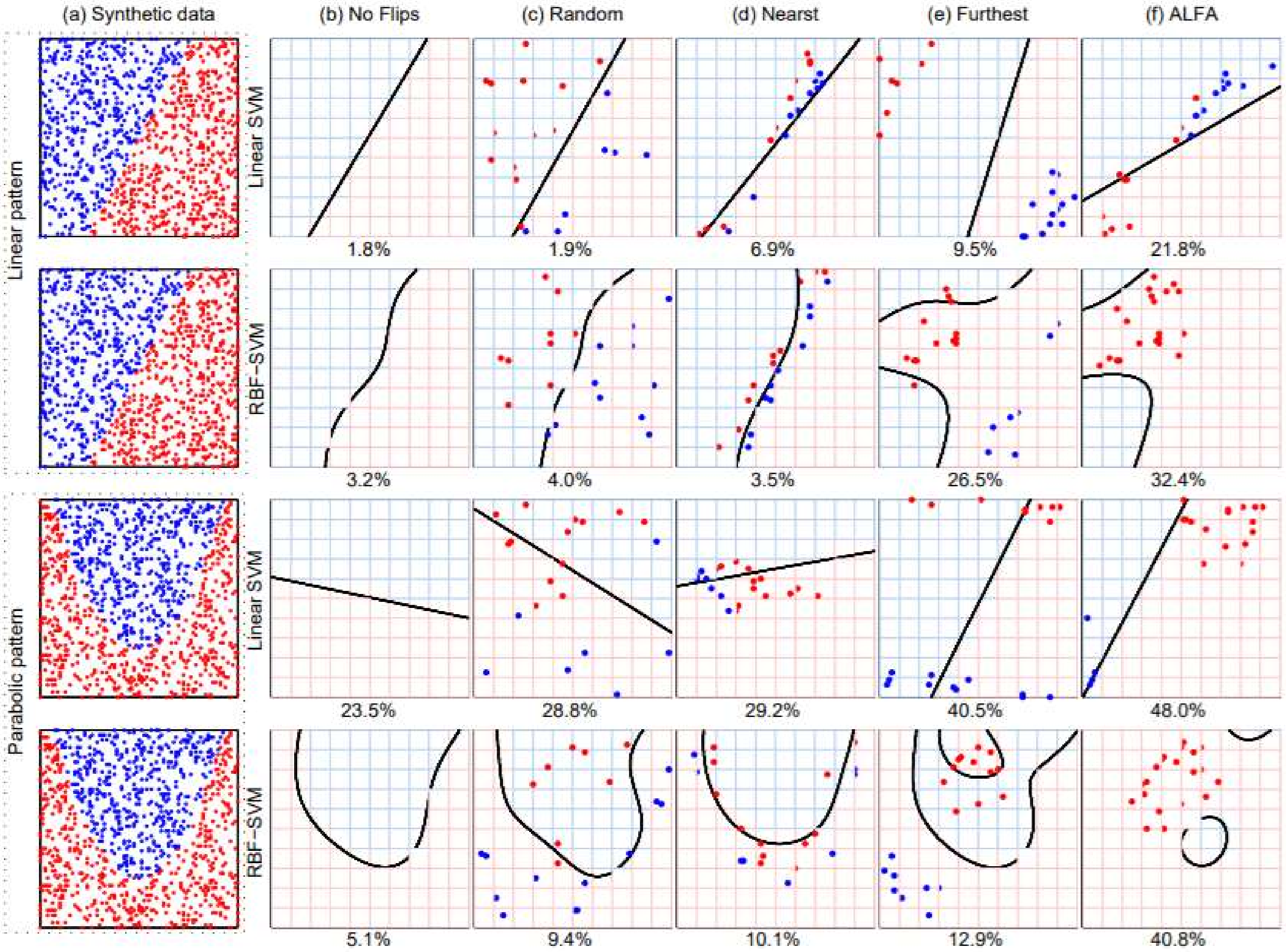
Poisoning attacks - classifiers

- Poisoning attacks aim to damage training data to cause misclassification
- Three types shown against classifiers
  - Label flipping
  - Gradient descent
  - Dictionary attacks


PRETTY, ISN'T IT?
WHAT IS IT?
I'VE GOT A BUNCH OF VIRTUAL WINDOWS MACHINES NETWORKED TOGETHER, HOOKED UP TO AN INCOMING PIPE FROM THE NET. THEY EXECUTE EMAIL ATTACHMENTS, SHARE FILES, AND HAVE NO SECURITY PATCHES.
BETWEEN THEM THEY HAVE PRACTICALLY EVERY VIRUS.
THERE ARE MAIL TROJANS, WARHOL WORMS, AND ALL SORTS OF EXOTIC POLYMORPHICS. A MONITORING SYSTEM ADDS AND WIPES MACHINES AT RANDOM. THE DISPLAY SHOWS THE VIRUSES AS THEY MOVE THROUGH THE NETWORK,
GROWING AND STRUGGLING.
YOU KNOW, NORMAL PEOPLE JUST HAVE AQUARIUMS.
GOOD MORNING, BLASTER. ARE YOU AND W32.WELCHIA GETTING ALONG?
WHO'S A GOOD VIRUS? YOU ARE! YES, YOU ARE!

Poisoning – label flipping

- Causative availability/integrity attack
- Aims to introduce label noise into training data
- Attacker causes benign samples to be labelled as malicious in training data, or vice versa
- Demonstrated against SVM
  - Xiao et al. (2012) cause error rate of 50% with 10% of labels flipped
  - Find RBF kernel is more effected than linear kernel
  - Biggio et al. (2011) show effectiveness against label noise robust SVM (LN-SVM)
- Limited by the degree to which the attacker has influence over the training set



Source: Adversarial Label Flips Attack on Support Vector Machines, Xiao et al 2012

Poisoning – gradient descent

- Causative availability attack
- Change benign points such that classifier becomes less accurate
  - Pick a benign point, flip label and change features
  - Could be done by causing malware in honeypot to send modified benign traffic that will be mislabelled.
- Demonstrated against SVM
  - Biggio and Laskov (2012) attack VSM in the case where attacker knows the training set used by the learner
  - On artificial dataset, achieves error rate of 0.06 for linear kernel and 0.035 for RBFD kernel.
  - On handwritten digits with linear kernel error rates of 0.1 to 0.3 with 200 iterations

Poisoning – dictionary attack

- Causative availability attack
  - Targeted or indiscriminate
- Specific to classifiers trained on token-based features
- Inserts malicious points into training data which include tokens found in benign data

Poisoning – dictionary attack

- Nelson et al. (2008) demonstrate against SpamBayes
  - SpamBayes labels emails as Spam, Unsure or “Ham” (benign)
  - Indiscriminate version send spam to target with words likely to appear in legitimate emails with a goal of causing large amounts of false positives. If the attacker can affect 1% of the training set, can cause 90% FP rate
  - Targeted approach assume knowledge of a specific email attacker wants misclassifed. Knowing 30% of the target email causes 60% FP rate
  - “Exploiting Machine Learning to Subvert Your Spam Filter”, Nelson et al. 2008

Poisoning attacks - clustering

- Two types
  - Bridging attacks
  - Gradient descent

Poisoning – bridging attacks

- Causative integrity/availability attack
- Introduce points in space between clusters to cause clusters to split and merge
  - In hierarchical clustering affects the inter-cluster distance.
  - Demonstrated against single and complete linkage hierarchical clustering
- Demonstrated against hierarchical clustering
- Should have some impact on any distance-based clustering algorithm
- For best results, requires perfect knowledge of the target classifier in order to find attack points. May be viable with a surrogate dataset, although not tested in the literature

Poisoning – bridging attacks

- Biggio et al. (2014) demonstrate against Malheur
  - Malhuer clusters MIST malware behaviour reports (flows of threads and processes)
  - Assumes perfect knowledge
  - Treated as optimisation problem maximising distance of clusters formed while under attack, to those while not under attack
  - Iteratively adds attack points until desired goal is reached
  - Can reduce number of clusters from 40 to 5 with 2% of the training data being injected
  - “Poisoning complete-linkage hierarchical clustering”, Biggio et al. 2014

Poisoning – gradient descent (clustering)

- Causative availability attack
- Similar to bridging approach, but using a gradient descent component to find optimal attack points
- Demonstrated against hierarchical clustering
  - Biggio et al. (2014) test in a perfect knowledge scenario against three datasets (PRTools dataset, simple C&C dataset and handwritten digits). Causes clusters to merge in all three cases
  - Also demonstrate an estimation-based approach to reduce number of iterations that is also effective

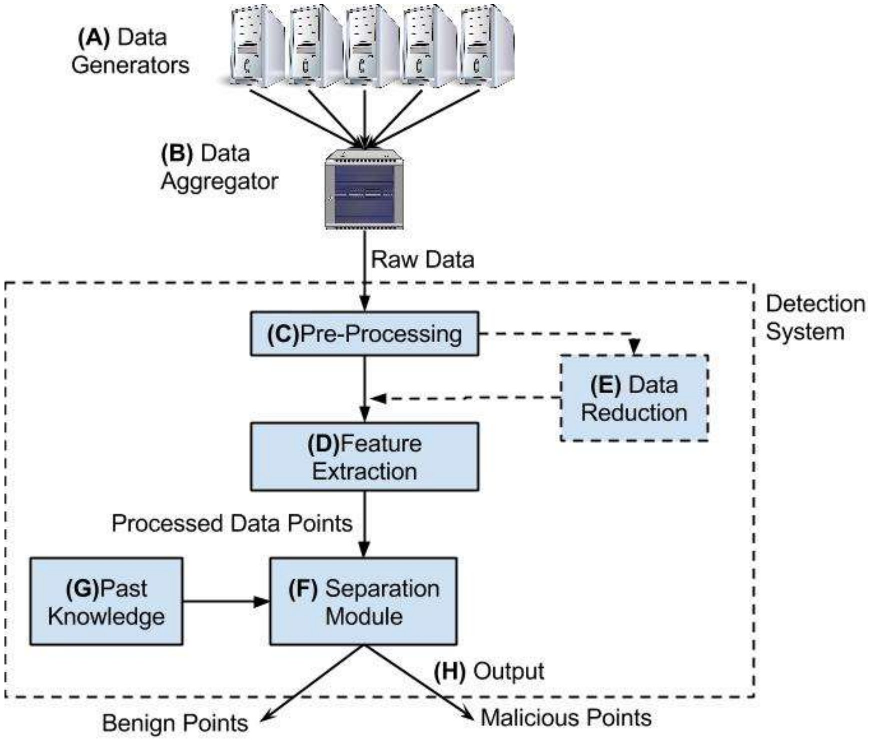
Issues

How difficult is it?

- Attacks discussed are mostly tested against simple examples
- C&C data can be far more complex
- Some features cannot be modified (certain network features such as IP addresses)
- Poisoning attacks could be carried out by targeting honeypots
- Attacker knowledge may be limited
  - For commercial systems training data and feature sets may be kept secret
  - Academic papers may leave out certain details (full feature sets)
  - Datasets may not be available
  - Knowledge could be gained through reverse engineering or social engineering

How effective are these attacks against full detection systems?

- ML component is usually just one part of a much larger system
  - Large amounts of pre and post processing can occur
  - Different algorithms can be applied in sequence (e.g. two rounds of clustering with different algorithms)
- As attacks are evaluated against simple systems (usually just feature extraction and the algorithm itself) unclear how attacks will perform



Is it happening?

- It’s hard to know
  - Evasion attack should only produce one attack point, which is misclassified so hidden
- Many talks have been given this year on the subject
  - Defcon, Bsides Vegas, RSA, Blackhat USA
  - Attackers know about it

Defences

Defences

- Two common approaches
  - Multi-classifier systems
    - Use multiple classifiers trained using different feature or datasets
  - Game-theoretical approaches
    - Incorporate attacker strategies into learning algorithms
- Defence approaches have limitations
  - Simple evaluation scenarios against single attacks
    - Spam emails are the most common test case using binary features
  - Game theoretical approaches rely on attack playing the game

Why is secure ML not in use?

- Lack of awareness
  - System designers do not follow ML literature and so are not aware
- Ease of access
  - Lack of existing implementations of secure algorithms or difficulties in implementing
- Reduced performance
  - Secure versions may have lower TP/FP rates in normal scenarios so are less attractive (TP/FP rates are selling points for papers)
- Lack of clear security metrics
  - While plenty of easy to follow metrics exists for measuring performance (TP./FP rates, precision/recall etc), no clear metrics for easily evaluating security performance

What should you take away?

- Machine learning is good…
- … if used properly
- Incorporate attacks against machine learning into your threat model
- Look to using secure variants of algorithms

Questions?

PRE-PRINT PAPER AVAILABLE FOR FREE AT HTTP://EPRINTS.LANCS.AC.UK/83888/1/PAPER_ACMSURVEYS_CHANGES.PDF

(OR EMAIL ME FOR PRINT VERSION JOE.GARDINER@BRISTOL.AC.UK)