Rise of the machines: Machine Learning & its cyber security applications

Prepared by:
Matt Lewis, Technical Research Director

Table of contents

1. Overview

At NCC Group, we are researching Machine Learning (ML) and Artificial Intelligence (AI) from a number of different angles in order to fully understand the pros and cons of ML when applied to security problems. We're looking to understand where ML works well, where it doesn't, and more broadly whether ML/AI in cyber security is mere hyperbole or emerging salvation.

Some of the topics and questions that we are researching and addressing include:

- Good and bad applications of ML in cyber security
- Adversarial ML (or, Machines vs. Machine) – how might ML be used to defeat security controls, whether signature or ML-based?
- Learn-taint – how might adversaries subvert dynamic and evolving algorithms in ways that benefit the adversary and/or negatively impact the outputs of the underlying algorithm?
- The impact of General Data Protection Regulation (GDPR) and other regulations on automated decision making[1]
- Algorithmic morality – in applications such as autonomous vehicles, what are the implications of algorithms that may have to make life-threatening decisions?

An overarching aim of our research is to develop assessment methodologies and strategies for security products and systems that employ attack-based ML and AI, as well an understanding the applications in our defensive services provided by our Security Operations Centre (SOC) and Cyber Defence Operations teams.

This initial whitepaper is by no means intended to be exhaustive; we acknowledge that, as an industry, cyber security is still catching up on ML and AI topics that have been researched for decades within academia.

"By far the greatest danger of Artificial Intelligence is that people conclude too early that they understand it."

Eliezer Yudkowsky

2. AI, ML, Big Data & Deep Learning

In recent years, ML has gained prominence in business and technology mainstream vocabulary. However, much confusion has arisen around the difference between ML and AI, especially when other terms such as Big Data and Deep Learning are thrown in to the mix.

To help understand the differences in terminology we provide the following summary definitions:

AI

Broadly, AI is an overarching term for systems that employ computer intelligence. Examples might include systems that can play games, such as Chess or Go, against humans, or systems that can detect potential malicious behaviour within network traffic or audit logs.

ML

ML is a sub-field of AI and computer science. It provides computers with the ability to learn, without being explicitly programmed, when exposed to new data. This is done by the study and construction of algorithms which produce models from example 'training' data. These models are then used to make predictions on further data.

Big Data

Big Data is a generic term covering large volumes of data that are amassed by modern computer systems. Where Big Data exists, there is usually an appetite for mining that data in some way in order to derive value and insight from it. Examples include terabytes of network packet captures or audit logs, or telemetry from millions of different endpoint devices - such as laptops and sensors - being processed in different ways to look for evidence of host and/or network intrusion.

Much confusion has arisen around the difference between ML and AI, especially when other terms such as Big Data and Deep Learning are thrown in to the mix

As storage capacities have increased with an inversely proportional decrease in cost, capturing and storing Big Data has never been easier. The fact that ML typically requires training on large volumes of sample data and then works to classify, cluster or predict based on large working samples means Big Data plays a key role in ML.

Deep Learning

Deep Learning is a subset of ML. It involves the use of neural networks to simulate the way in which the human brain processes information through neurons and synapses. Deep Learning is primarily concerned with very large neural networks which are constructed to perform a computationally intensive AI function or to solve a complicated problem. Deep Learning relies on large subsets of data to be effective, meaning that, once again, Big Data plays a key role.

3. Types of ML problems

Problems (or tasks) in ML are usually classified into three main categories and relate to the data available or provisioned:

- Supervised Learning: Tasks where an algorithm is trained with labelled data. This could involve presenting an algorithm with input data, as well as the desired output data, and allowing it to produce a model to map each subsequent input to an output.
- Unsupervised Learning: Here, data has no labels and the ML algorithm is left alone to make its own decisions and inferences on the data.
- Reinforcement Learning: Here, data is presented as a dynamic environment.

Problems can be further categorised by their output or aim:

- Classification: Inputs are mapped to user-specified outputs, such as emails to 'malicious' or 'safe'.
- Clustering: Inputs are grouped into clusters. The definitions of clusters are not known beforehand, unlike classification.
- Regression: A technique from the field of statistics used to estimate or predict outputs from a continuous - rather than discrete - set
- Dimensionality reduction: The conversion of datasets with vast numbers of dimensions into datasets with fewer dimensions, resulting in more concisely-conveyed data for classification or regression tasks.

Supervised Learning

Supervised Learning deals with datasets that are labelled. That is, the dataset includes the features by which the event/object/thing is defined as well its desired output.

A good example of Supervised Learning is classification for malicious email detection. The classification algorithm would be supplied a large dataset of emails as well as the labels 'malicious' or 'safe'. This is known as a training set; a dataset used to train the algorithm.

The algorithm would produce a model which would correctly classify the training set. Any new, unlabelled emails are then passed against the model and mapped to 'malicious' or 'safe', based on the trained model.

Unsupervised Learning

Unsupervised Learning deals with unlabelled datasets. One of the most common applications is to find groups/subsets within datasets, with applications being many and varied.

An example of use in security is the BotMiner system which performs cluster analysis on network traffic for botnet detection[2]. By clustering the network flows, describing who is talking to who, and clustering the behaviours and actions of these machines, BotMiner is able to detect real-world botnets.

Reinforcement Learning

Reinforcement Learning is a reward-based learning system.

Most algorithms work in an 'environment' of states with a set of available actions and this can be modelled in a variety of ways with many uses being based on Markov[3] models. This is an environment with a set of states and a set of actions.

States and actions have scores and penalties associated with them and algorithms attempt to achieve certain states, learning which actions allow it to have the greatest score. This type of algorithm is closely linked to the field of intelligent agents and multi-agent systems, as well as technologies such as driverless cars and even AlphaGo (the AI which beat the world's number one Go player). The training step for the AlphaGo AI was it playing itself many times and building a model based on the rewards and penalties associated with the moves it made given its state.

Answering a question or solving a problem?

Although it seems a trivial question, when considering the use of ML within security applications it is necessary to know what we are planning to achieve:

- Do we have some malware samples and want to find malware families?
- Do we want to classify some network logs into malicious vs. benign?
- Do we want to make some predictions on network traffic?

As each of the above would require different approaches and different algorithms, setting a goal at the outset greatly improves the chance of a successful ML application.

4. Obtaining & dealing with data for ML

There are a number of things to be considered when collecting data for use in ML applications, as well as preparation that should be done before running it through any algorithms. Much of the data preparation can be reduced to three steps:

1. Data selection
2. Data pre-processing
3. Data transformation

Data selection

Data selection is one of the most important steps in the ML process. It is therefore vital to decide what is actually needed and if a dataset is of any use.

This can be done by highlighting key fields in the dataset which we determine would be relevant to what we are trying to achieve. For example, if we were attempting to classify suspicious and non-suspicious network access log entries of a system that operates 24/7, the time associated with the log entries might not be of much interest.

Another aspect to consider is availability and size of data. As ML algorithms, in theory, should become more accurate with larger training datasets, the more observations we have to do this the better.

Making sure we have a large enough dataset to produce a decent model to predict on or a method to obtain new data is therefore crucial before starting. This does not mean that observations require more features to describe them. Many algorithms in fact suffer from the curse of dimensionality, whereby the more features we have per observation, the less the algorithms work as intended.

> “Data selection is one of the most important steps in the ML process.”

Data pre-processing

Once we have a dataset to play with, we need to make sure that it can work with our algorithms.

This could include many possible tasks and could very well be dependent on the type of task and the algorithm to be used. In the example of classifying suspicious and non-suspicious network access logs with a dataset that includes time of day, we may want to remove this; we want to strip out all but the most necessary data elements.

At this point, we would also look at how to represent and store our pre-processed data. We may be extracting from PDF files or log files but it may be worth storing the resultant data in CSV or another structured format of choice. However, due to the row/column nature of CSV, it lends itself quite well to most ML feature vector needs.

This would also be the time to cleanse the data, while any missing data should be handled and tracked in some way. If dealing with sensitive information, this should be anonymised or removed.

Data transformation

This is the point at which we start making our data usable by the algorithm of choice.

There are some algorithms and problems that require us to use strings, such as document classification or malicious email detection, but very often algorithms will require mathematically-malleable data.

Scaling might also be useful for numerical data, with most ML algorithms working on real values between 0 and 1.

Where to find datasets?

Ideally, we want to work on datasets meaningful to our task. For the sake of practicing or learning the various algorithms, there are a few websites available which provide ‘toy’ or publicly available datasets[5]. Various cyber-related data repositories and curated lists are also available online[6], covering all manner of datasets, from web attack payloads to malware samples[7]. Many of these have been captured from online honeypots.

Other sites with potentially useful datasets for training[8], which are not necessarily security-related but still technological in terms of network logs and packets/headers[9], are also available.

5. Clustering

Clustering is the process of grouping objects together in such a way that objects in the same group are more similar to each other than they are to objects in other groups.

Clustering is the top-level term for this process but a number of different approaches and algorithms exist within ML to achieve clustering.

K-means clustering

K-means is one of the simpler clustering algorithms. We need to supply it with the number of clusters we want it to output. It works by randomly creating K (the number of clusters we want) cluster centroids, assigning each observation to its nearest centroid, recalculating the centroid by taking an average of its members positions, recalculating membership and repeating until the centroids no longer change.

K-means clustering is a widely-used algorithm due to it being generally easier to understand and implement than other algorithms. As a result, it is commonly available in ML packages with support in a number of different programming languages. For example, for Python, it is available in the packages SciPy, NumPy and Sci-Kit, among others.

Hierarchical clustering

There are two types of hierarchical clustering: Divisive and Agglomerative (though Divisive is rarely used).

Agglomerative takes each observation as its own cluster, calculates similarities/distances between themselves and then fuses the two clusters that are most similar. The similarities/distances are recomputed and the process repeats until we have one cluster remaining. The idea is that the algorithm should run, produce the tree and we decide at which point to ‘cut’ the tree and therefore choose how many clusters we have after the algorithm runs. Many packages force specifying a number of clusters to be returned.

A major advantage of the hierarchical method is the fact that no pre-specified information about the number of clusters to be produced is required. This lack of pre-run bias allows for greater exploratory analysis and a set of results which could return information that was previously unknown.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

DBSCAN aims to cluster with an idea of neighbourhoods. Clusters are formed by points in the same neighbourhood and the user specifies criteria for this. There is no need for users to input the number of clusters to return and if a point doesn't fit in a neighbourhood it is classed as noise and doesn't affect the cluster.

The algorithm deals well with arbitrary-shaped clusters. This allows for greater exploratory analysis and the discovery of even more intricate subsets in data that K-means and hierarchical methods would struggle with. It is quite a fast algorithm but doesn't work well with high dimensionality/sparse datasets.

6. Classification

Classification within ML involves identifying which categories or groupings a new observation belongs to. The classification decision occurs as a result of a prior training stage where a system will have first learned about different types of data before using this prior knowledge to classify new observations.

K Nearest Neighbours (KNN)

This is possibly one of the simplest algorithms. All computation is done at the classification step and essentially classification is done by vote.

We specify a value for K and take that many of the nearest points to the data to be classified. The class of our data is whatever the majority of the points are labelled as; if we have K = 1, the prediction of class of our unlabelled data point will simply be that of the nearest point.

We could use a variety of distance metrics to define 'nearest, such as Euclidean, Manhattan, Chebyshev etc.

Support Vector Machines (SVM)

After being given a training set of examples which are labelled into one of two categories, an SVM will build a model that assigns new examples to one of the categories.

The model is built around the idea of a hyperplane. Given an n-dimensional feature space, a hyperplane is a subspace of n-1 dimensions dividing the space. For example, when provided with a two-dimensional feature space (x,y), a hyperplane would be a single line, such as the X-axis.

The SVM aims to produce a hyperplane that would separate the features according to their respective categories.

Decision Tree

The ML version of a Decision Tree is simply one that creates its own rules to produce a model that fits the pairings of training data with their outputs.

Rules are usually represented as true false statements - be it for categorical or discrete values – with some comparisons. There are quite a few different Decision Tree algorithms, however, the concepts are the same across most of these algorithms.

7. Regression

Regression analysis aims to outline the relationships among variables. Once a model on the data has been established, we can then predict dependent variables given some independent variables. For example, with a dataset of connection speed, download speed and file size, we could produce a model outlining the relationships among the data. Then, given a new sample containing only connection speed and file size, we should be able to predict the download speed (the dependent variable).

In regression, we predict with probability the value of the dependent given the independent variables.

A simple linear regression is essentially the plotting of a line of best fit through data. Defining this line is done by calculating the sum of least squares, like minimising the vertical distances from the point to the line.

8. Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) can be used for a variety of tasks, most often for classification and regression.

Modelled on the human central nervous system, an ANN consists of a number of input and output nodes, and a collection of connected nodes and edges named neurons and synapses, respectively. Each synapse is assigned an adaptive weight and its role is to multiply data passing through it by that weight.

The neuron's role is to apply a function on the collection of values received by its inputs (the various synapses connected to it). By passing through the 'hidden' network, the data is processed and outputs a value for classification or regression. While the neuron functions remain the same, the synapse weights change throughout the training steps – weight change can be done through a variety of algorithms, most notably with the back propagation algorithm.

A key idea of ANNs is that the hidden network is allowed to make its own judgements on what is important and what should be extracted from the data during the training steps; we set the neuron functions and let the ANN do its thing. This allows for interesting findings by isolating individual neurons in the network to determine what is important or what the neuron is looking for.

As an example of ANN, images produced by Google's Deep Dream show use of software to help understand what a deep neural net is seeing when looking at an image – a deep neural net being an ANN with many hidden layers between input and output layers. These hidden layers include multiple neurons/functions that combine to produce an output from given inputs. Deep neural nets are large, complex and will involve intensive computation.

Figure 1 - Source image and output image from DeepDream illustrates neural net decision making

9. Natural Language Processing (NLP)

Natural Language Processing (NLP) is an area of study on computation around natural languages. It's a huge field with many different and specific sub categories of study and research such as speech recognition, language translation, determining the textual representation given sound clips of speech or machine translation and much more.

NLP can be a difficult task as it requires a deep knowledge of language through grammar, context and idioms to be able to determine parts of speech: verbs, adjectives and nouns.

A further use of NLP is sentiment analysis. This tries to analyse sentences and provide a score of its sentiment, or classify a sentence as positive or negative; for example, "NCC Group is 1337" should give a positive feedback score.

A further score associated with sentiment analysis is objectivity. NLP could be of value in OSINT and threat intelligence applications that need to read, digest and process volumes of written text, for example.

NLP could be of value in OSINT and threat intelligence applications that need to read, digest and process volumes of written text

10. Summary of algorithmic pros and cons

Each of the algorithms highlighted above have their own advantages and disadvantages. The table below summarises these and is presented to demonstrate that there is no 'killer solution' in the ML space – that is, the table serves as a reminder that we need to clearly understand the question or problem that we are trying to address and not only whether ML is a suitable means of achieving this, but which ML approach and algorithms are best fit (if any) for the specific problem area.

Algorithm	Advantages	Disadvantages
	Clustering	
K-means	• Simple • Fast (usually scales linearly)	• Need to specify (know in advance) a value for the number of clusters to return • Element of randomness in the centroid set up phase • Doesn't play well with non-linear or non-globular data
Hierarchical clustering	• Simple • Can produce a tree for visualisation	• Sensitive to noise • Slow
DBSCAN	• No need to specify clusters to return • Can find random shaped clusters • Quite fast at O(n x log(n)) • Allows for noise	• Need to know what you are doing to get good values for neighbourhood criteria • Doesn't like sparse or high dimensional data
	Classification	
K Nearest Neighbours (KNN)	• Simple • Linear in the size of training set • No training step	• Classification step is computationally expensive • Model cannot be 'described'. No real learned concepts
Support Vector Machines (SVM)	• Relatively simple concept • Good in high dimensional spaces	• Training phase can be slow and be memory resource hungry • Finding a suitable kernel/similarity function may be difficult

	• Good with datasets with clear separation between classes	
	Regression	
KNN Regression	• Very simple model • No training step	• Struggles with samples where independent variables fall out of the scope of the model • Struggles with high dimensional data • Prediction step is expensive especially with large datasets

11. Useful ML packages

Here, we present just a few useful ML packages that can be used for research and development tasks within the ML space. Most, if not all packages below are exclusive to, or available for, Python which lend themselves well to a cyber security sector already well-versed in Python.

Despite being freely available, be under no illusion as to the power and use of these packages - most of them are developed by and/or used in mainstream online services, such as Google's RankBrain based on TensorFlow for filtering page results, and Spotify's use of sci-kit learn for music recommendations.

Sci-kit learn

Sci-kit is probably one of the most popular ML packages available. It has implementations of most major ML algorithms and their various flavours as well as a variety of utilities for data pre-processing, validation and scoring, among others. It is very much considered a plug and play package, with most functions ready to go with little effort required around parameter modification.

It is widely used in the technology sector by many large companies such as Spotify and Evernote. It also comes with a number of useful 'toy' datasets - such as the Iris flower set, MNIST handwritten set and Boston house prices - which allow users to get used to the various algorithms available.

It uses both *scipy* and *numpy* for functions and data structures, with most functions that take data requiring them to be in the *ndarray* form provided by *numpy*.

Tensorflow

TensorFlow is an ML package from Google. It is mostly dedicated to producing artificial neural networks with a ground-up approach, but it can do a huge range of other tasks in ML and NLP.

It generally requires a little more background theory on ML and the inner workings of neural networks but once understood has many useful applications. The backend is written in C++ so as to improve performance.

TensorFlow also has a feature called TensorBoard which runs as a web application. It points to the log directory of running code and can visualise a program's learning process and neural network structure, as well as provide graphs and statistics on runtimes and other associated features.

Natural Language Toolkit (NLTK)

The NLT is a popular open source NLP package. It comes with a huge range of functions and utilities for processing or analysing text, as well as a wide collection of written texts available for a variety of uses such as stop words, lemmas, movie reviews and news articles.

TextBlob

TextBlob is a simple plug and play NLP package. Once a TextBlob object is created on a string it offers simple utilities such as separating strings into words or sentences, tagging parts of speech, lemmatizing, and sentiment analysis.

The sentiment analysis function is handy as it gives the score in a range -1 to 1 and gives a score of objectivity 0 to 1. It does this without a need to train a classifier like in NLTK.

MATLAB & R

Two other packages - or frameworks - worth mentioning in this section are MATLAB and R.

Historically, these frameworks have been available for engineers, scientists and academics for mathematics and statistics computation, however, both have ever-increasing support and modular extensions for ML capabilities.

12. Conclusion

We hope that this whitepaper has been useful to those wanting to learn more about the preliminaries of ML. As our research on ML from various angles continues we will share our analyses and findings in further whitepapers.

At NCC Group, we have a number of experienced ML practitioners and consultants with ML-based academic backgrounds who form our ML Working Group. This includes consultants with PhDs in ML and data science, and many graduates in computer science and mathematics whose example theses have covered topics such as:

- Genetic programming to find potential crypto attacks
- ML with CUDA for increased performance in cyber security applications
- Android malware classification using ML
- Unsupervised learning to extract RAM from infected machines and understand if another machine is infected by the same malware or a variation
- ML for detecting potential intrusions from audit logs

In addition to our own internal ML-based research, we are increasingly engaging with academia on this topic. For example, some of the content in this paper was produced by a Royal Holloway graduate intern who has since returned to Royal Holloway to study for a PhD in cyber security and ML.

As a CyberInvest member, we have committed to a £500k funding stream over a five-year period and anticipate that some of this funding will go towards ML-based research in cyber security.

Additionally, we recently supported University College London's (UCL) successful bid to host the Science & Technology Facilities Council's first Centre for Doctoral Training in data intensive science (DIS). DIS encompasses a wide range of areas in the field of Big Data, including the collection, storage and analysis of large datasets, as well as the use of complex models, algorithms and ML techniques to interpret the data.

Our commitment to this centre will include studentship secondments to NCC Group which will see great collaboration between ourselves and academia on research around real-world ML-based cyber security problems and applications.

13. References

[1] https://ico.org.uk/media/for-organisations/documents/2013559/big-data-ai-ml-and-data-protection.pdf

[2] https://www.usenix.org/legacy/event/sec08/tech/full_papers/gu/gu.pdf

[3] https://en.wikipedia.org/wiki/Markov_model

[4] https://deepmind.com/research/alphago/

[5] http://archive.ics.uci.edu/ml/

[6] https://github.com/jivoi/awesome-ml-for-cybersecurity#-datasets

[7] http://www.secrepo.com/

[8] https://turi.com/

[9] http://httparchive.org/downloads.php

[10] http://deepdreamgenerator.com

[11] https://www.tensorflow.org/about/uses

[12] http://scikit-learn.org/stable/testimonials/testimonials.html

[13] http://scikit-learn.org/stable/

[14] https://www.tensorflow.org/

[15] http://www.nltk.org/

[16] https://textblob.readthedocs.io/en/dev/

[17] https://www.mathworks.com/solutions/machine-learning.html

[18] https://www.r-project.org/about.html

[19] https://www.ncsc.gov.uk/articles/cyberinvest-securing-our-future-through-research

[20] https://www.ucl.ac.uk/mssl/research-degrees/CDT_Data_Intensive_Science

14. About NCC Group

NCC Group is a global expert in cyber security and risk mitigation, working with businesses to protect their brand, value and reputation against the ever-evolving threat landscape.

With our knowledge, experience and global footprint, we are best placed to help businesses identify, assess, mitigate & respond to the risks they face.

We are passionate about making the Internet safer and revolutionising the way in which organisations think about cyber security.

Headquartered in Manchester, UK, with over 35 offices across the world, NCC Group employs more than 2,000 people and is a trusted advisor to 15,000 clients worldwide.