ENDGAME.

Hunting for Malware with MachineLearning

INTRODUCTION

Traditional security tools, like antivirus, simply cannot keep pace with the ever-evolving threats facing most organizations. In addition, most enterprise networks focus on reactive defensive strategies, which are insufficient due to the sophistication of attacker tactics, techniques, and procedures (TTPs). Acknowledging that breaches are inevitable, organizations should think proactively to discover and remediate cyber threats in their enterprise. Machine Learning offers some unique advantages for detecting unknown threats in hunting for the advanced adversaries.

Machine Learning offers several unique advantages for detecting unknown threats in the hunt paradigm. Models can generalize to never-before-seen malicious content based on observations of past behaviors and characteristics that may be difficult for a human to describe concisely or discern given the vast amounts of data. The complexity of machine learning models can be closely controlled, depending on the task and environment at hand. Accuracy scales with data richness, quality and quantity, rather than with human labor. Appropriate applications of machine learning can dramatically reduce the data deluge to human analysts, allowing them to focus on applying deep domain knowledge in responding to a small subset of threats in a timely manner.

This paper describes a machine learning approach for automating the detection of unknown malware. Detecting malware is an important element of a multi-staged approach to detecting adversaries in enterprise networks. Our machine learning approach allows the hunter to identify initial exploits used to compromise a system as well as the adversary's persistence mechanisms. Unlike other successful applications of data science, malware detection provides unique challenges that require careful consideration and implementation. We first provide some context to the malware detection problem. Next, we provide a brief background of machine learning, then detail Endgame's unique application of machine learning for malware detection during the hunt.

LIMITS OF RULES, SIGNATURES, AND IOCS

The anti-virus (AV) community is by nature a reactive industry. Rules, signatures and IOCs are inherently derived from post-breach analysis, and then are deployed to prevent future but identical breaches. Indeed, this is useful in preventing the spread of known malware, since the characterization of malware is very human intensive. But these methods often miss polymorphic, obfuscated, and novel threats. Consequently, security teams are left playing catch-up to nefarious actors when pursuing an AV approach.

As part of malware forensics investigation, researchers try to discover the nature and purpose of a binary, the infection mechanism, how the malware interacts with the host system, if and how it communicates over the network, the method by which an attacker communicates with the malware, as well as determining the extent of infection and the sophistication of the adversary. Tools used by analysts fall broadly into static and dynamic analysis:

Features of malware derived from these methods are very valuable in characterizing malware. However, because of limited human resources under a flood of malware to examine, reverse engineers and malware experts are not able to thoroughly examine every file. Furthermore, this same resource limitation promotes efficient, targeted analysis in favor of exhaustive associative analysis. Rules, signatures and IOCs are ultimately derived from a small subset of the total information that could be available to an analyst. With more time, analysts might discover further evidence through hidden correlations that exist within the file, or in its relationships with other files.

Static Analysis	Dynamic Analysis
▪ Dissecting a binary without executing it using disassembly tools (i.e. IDAPro) to understanding malicious behavior.	▪ Sandboxed execution to identify behavior not observed during standard execution (e.g. registry manipulation, network activity, file creation)

A DATA SCIENCE SOLUTION

Data science has been hugely successful in recent years in healthcare, robotics, voice recognition, facial recognition, stocks and finance, sales and targeted advertising. Indeed, many of the same reasons for success in those industries is a key enabler to scaling and generalizing beyond the limits of AV signatures for malware detection and classification. Key advantages of employing data science for malware detection and classification include the following:

There are, of course, challenges with machine learning that pertain to the management of the lifecycle of data and metadata, model training, validation, deployment and maintenance. Indeed, the data and model management issue is a key distinction between machine learning (models and algorithms) and data science (which also includes data management).

The advantages of data science for malware detection and classification are not a secret in the industry. But, since “malware detection via data science” is auxiliary or even disruptive to most security businesses—the AV behemoths and even nimbler detection and prevention companies—many competitors are not well positioned to leverage data science as a core competency. And among the few that do focus on data science, it is typically in a passive detection rather than an active hunting mindset, or is simply a research project that fails to become operationalized in a product. At Endgame, data scienceinforms our multi-stage malware detection and many other elements of Endgame’s cyber operation platform to enable next generation threat hunting.

Key advantages of employing data science for malware detection and classification include the following:

Automation

Aggregate and analyze disparate data sources automatically.

Deep Insights

Learn from data what constitutes malicious content or behavior.

Scalability

Scales well with increases to data or computing resources rather than human labor.

Generalization

Generalize to never-before-seen samples or variants, based on behavioral relationships that are not obviously constructed by hand.

Transparency

Eliminate the black box problem if done well by providing context into the classification process, enabling insights into why samples are deemed malicious.

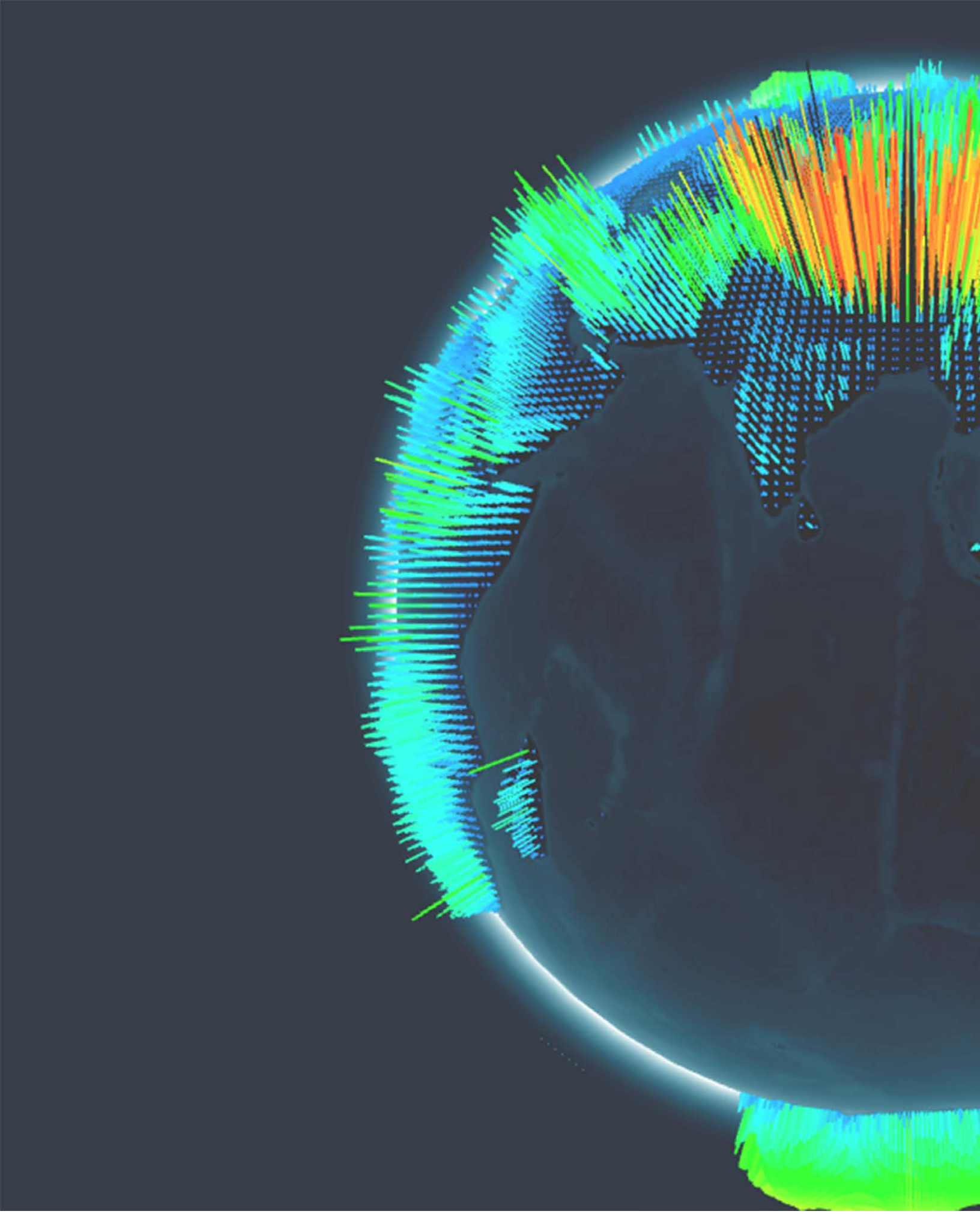
Machine Learning & Operationalizing Automated Detection

Put simply, machine learning is a discipline that uses mathematics, statistics and optimization to give machines the ability to learn from data without being explicitly programmed how to do so. Machine learning is the process by which, given data, an algorithm produces a model that when presented with an input (e.g., an image, real estate specs, a binary file) it provides a corresponding output (e.g., there's a cat in the image, the estimated price of the property, the file is malicious). Machine learning is often broken down taxonomically into three major fields that are defined by the kinds of data available, and the algorithms used to train a model from the data. Broadly, they are as follows.

Supervised Learning	Unsupervised Learning	Reinforcement Learning
Using labeled training data—examples of input/output relationships that we'd like the model to learn—train a model to predict labels for new data that the model has not seen. The labeling requires domain expertise to label the data to help train the model	Given only unlabeled training data, find patterns or structure in the input data. For example, clustering is an unsupervised method to group together similar inputs. Anomaly detection is used to determine when new data is somehow different than data on which the unsupervised model was trained. This does not require domain expertise for data structuring, and instead focuses on finding patterns in the data without human input.	A model attempts to achieve some goal with only indirect feedback about the outcome of its trials. That is, the model learns only from an eventual reward or penalty associated with an outcome, without any direct feedback about its steps along the way. Google's AlphaGo is a good example of such a model.

Malware detection can be thought of as a binary classification (supervised learning) problem. In reality, only a fraction of the tens of millions of samples available to us is labeled as malicious or benign, and the remainder have no label at all. At Endgame, we leverage approaches in semi-supervised learning, in which both labeled and unlabeled data are used to train and condition a model. Nevertheless, for clarity we describe a purely supervised approach to training a model. In what follows, let's assume that we're providing a set of known malicious and benign samples.

A critical step which requires input from malware security subject matter experts (SMEs) is to determine what features from files are useful for discriminating between malicious and benign samples. In a departure from practices in the AV industry, it is not useful to choose as a feature a specific 4 byte sequence (0xdeadbeef) or string ("pwned") or other artifact that provides a fingerprint for a particular malware sample. Instead, we choose feature categories from which we hope a model can determine generally whether a group of samples is malicious or benign. For example, one might extract all 4-byte sequences from the .text section, or all human readable strings. The set of possible unique features across all files from just these two feature categories can become extremely, even unboundedly large. The machine learning model determines which features—actually, which combination of features when used together—are the most useful in discriminating malicious from benign. This theme of "leave it to the science" is incredibly useful, since there is not some small set of golden features (e.g., "the evil bit") that one can inspect to determine maliciousness. Combinations of features that correlate to maliciousness must be discovered by the modeling and feature selectio process.



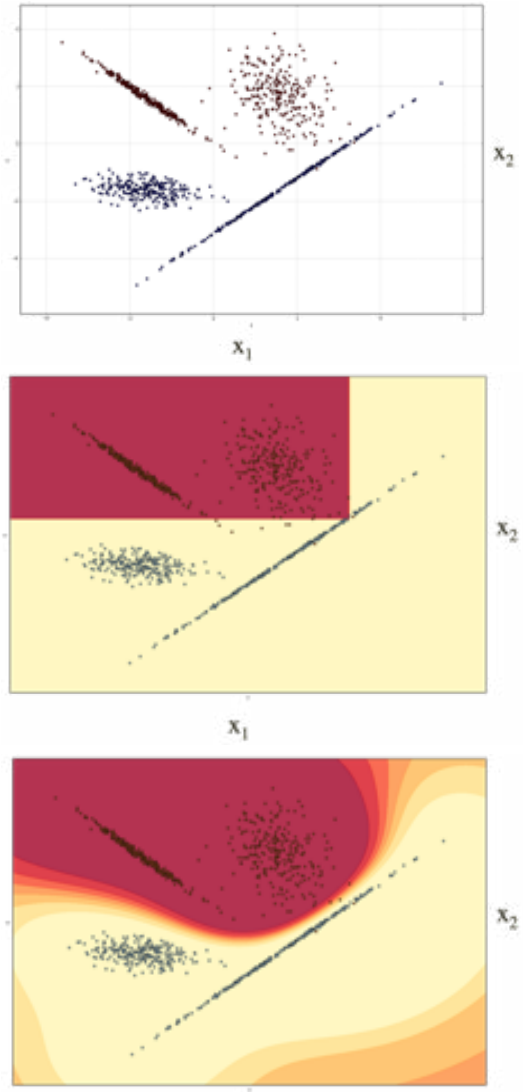
So, what really is a supervised learning model? Consider the following example that is contrived, but provides some useful intuition about machine learning models versus signatures and rules. Suppose that we are given a set of (fictitious) malicious and benign samples that we represent by two features (just two numbers for each sample), x1 and x2. For example, let x1 represent the file size in bytes and x2 represent the file entropy. In the first pane of Figure N, we'll represent each malicious file (red dot) and benign file (blue dot) by its (x1, x2) coordinates in this feature space. The goal of our supervised learning model is to readily distinguish between the malicious and benign samples.

Below is a simple model of the form (in Python):

```
label = 'malware' if x1 < a and x2 > b else 'benign'
```

This is depicted in the following graphs, where the malicious decision region of the model is depicted in the red shaded area. This kind of Boolean logic for determining threats is very similar to rule-based methods that are readily encoded in YARA, for example. It does represent a valid (albeit coarse) summary of the data, but is limited in its ability to generalize, and may be brittle to minor changes in a malware sample. On the other hand, in the right pane, a statistical learning model automatically partitions the feature space based on complex interactions of the features. In addition to forming a complex boundary, the statistical model may often be able to determine regions of its decision boundary where the model determination is less certain (the red-to-yellow gradient pictured, for example). The model's decision process may be too complex to enumerate concisely, but the model can provide valuable hints that sufficiently justify the reason for its determination. For example, "sample X is similar to samples A,B, and C" or the "sample X is classified as malicious because the following features are often found in malicious samples."

Statistical Learning in Pictures



Figures 1-3 — Overview of Machine Learning Problem

For an enterprise-relevant data science solution, one must be able to rely on a wholly automated process for acquiring samples (malicious, benign and unlabeled), generating labels for these samples, partitioning the data into training and validation sets (for feature and model selection), then updating or re-training a model as new data comes in. This may seem like a mundane point, but data lifecycle management and model versioning and management don't enjoy the standard processes and maturity that, say, software version management has come to rely on. Introducing the engineering processes into a machine learning solution narrows the chasm between an interesting one-off prototype and a bona fide production machine learning malware detection system. Once a model is trained and its performance on the holdout validation set is well characterized, the model is then automatically deployed to the cloud or pushed to a customer.

But the job doesn't stop there. In what constitutes quality assurance for data science, performance metrics of the deployed model are continuously gathered and checked against pre-deployment metrics. Is there an unusual spike in the number of detections? Or have the detections gone quiet? For a sampling of the files submitted to the model, can we discover the true label and compare them against the model's prediction? The answer to these questions is particularly important in information security, since malware samples are generated by a dynamic adversary. In effect, the thing we're trying to detect is a moving target: the malware (and benign!) samples we want to predict continue to evolve from the samples we trained on. This ugly fact violates a basic tenet of traditional statistical learning: that the training data and test data are independent and identically distributed (IID). In fact, neither independence (since software is released in related revisions) nor stationary distribution assumptions (since adversaries adapt) hold true in malware detection. Whether one acknowledges and addresses this issue head on is another issue that separates erroneous from sophisticated offerings. Clever use of unlabeled data, and strategies that proactively probe machine learning models against possible adversarial drift can be the difference between rapidly discovering a new campaign against your enterprise, or being "pwned".

MALWARE HUNTING AT ENDGAME

Endgame's application of data science to malware hunting is novel in the industry from several perspectives.

- Where others build machine learning models naively from well-curated and widespread data sources, Endgame's data science approach incorporates novel data feeds with the nuances of adversarial and semi-supervised learning. This enables Endgame to use the latest available data to protect against bleeding-edge attacks.

- Endgame's behind-the-scenes proprietary data and model management pipeline allows instantaneous updates to cloud-based models and frequent updates

- Endgame's machine learning models are specifically applied to proactively hunting rather than passively detecting.

The remaining discussion applies specifically to Endgame's robust, lightweight models that reside on the endpoint to support hunting as one part of the larger Endgame cyber operations platform. The task of this classifier is to determine the maliciousness of files that are:

(a) newly created,
(b) recently modified,
(c) deemed suspicious by automated hunting mechanisms, or
(d) specifically submitted to the system by the hunter.

The following discusses the details behind training such a model, and considerations that set Endgame apart as a provider of hunt technology.

Data Collection and Labeling

The development of a malware classifier requires a large amount of diverse training data. Endgame collects malicious, benign and unlabeled samples from a variety of public and proprietary sources. This includes Faraday, Endgame's geographically and technologically diverse network sensor platform, which yields novel malware samples actively used in campaigns worldwide. Targeted malware samples are also collected as a result of active hunting or from customer engagements. Public sources and data sharing partnerships provide for other commodity and targeted malware and benign samples.

We explicitly use unlabeled data in several ways. Unlabeled data is under-appreciated in the industry, but it is often the most important since it may represent bleeding-edge malware that is yet unknown to the industry. First, Endgame utilizes unlabeled data directly for model building via semi-supervised learning as will be explained shortly. Second, Endgame employs an adversarial active learning approach for malware triage. This process is critical for very judiciously selecting "important" samples to be inspected by an expert for reverse engineering and labeling. Samples are selected for triage that balance the greatest gain in data label discovery (e.g., for a cluster of samples, reverse only a single sample to gain a good understanding of the entire cluster), model impact (e.g., by labeling which sample will my model gain the most information?), feature space exploration (e.g., which samples is the model not really confident about?), adversarial drift (e.g., which samples might represent new attack trends?) and adversarial poisoning (e.g., which sample could confuse my model the most if an adversary nefariously implanted it?). All this while minimizing human resource cost. The adversarial considerations are particularly important for a hunting platform, in which it is assumed that an adversary is active in your enterprise and attempting to subvert the hunter.

Feature Engineering and Feature Reduction

Immediately upon submission to Endgame, a sample and its metadata are processed by Endgame's proprietary data management pipeline. In partnership with reverse-engineers, vulnerability researchers and red teams, we have identified–and continuously identify–features that can reveal the malicious or benign behavior of a sample. Since threat hunting happens on the endpoint, the current features are predominantly static in nature (no sandbox, but some runtime information may be available). They include byte-level features, statistics about file composition, information contained in the PE Header and its extensions, functionality revealed by imported or exported functions, and suggestive bit or text strings such as IP addresses, URLs, executables, certain folder names on the filesystem, or Windows registry locations. Some feature types are secondary dependencies on primary features, and Endgame's proprietary data management pipeline automatically walks the dependency tree, ensuring that feature sets are always up-to-date. Features for each sample are extracted in milliseconds and are available for instantaneous updates to Endgame's cloud model, or are queued for updates to on-premises models.

We perform a form of feature reduction that: (1) prevents overfitting and (2) is suitable for a lightweight model deployed to an endpoint. The resulting features represent a few thousand numbers per PE file, which in turn allows for a small memory footprint for a deployed model. Since this same feature extraction and feature reduction is performed on the endpoint for hunting, it is a convenient byproduct that our feature reduction technique anonymizes customer data that may be optionally shared with the cloud. This allows the platform to learn about malware, not your company!

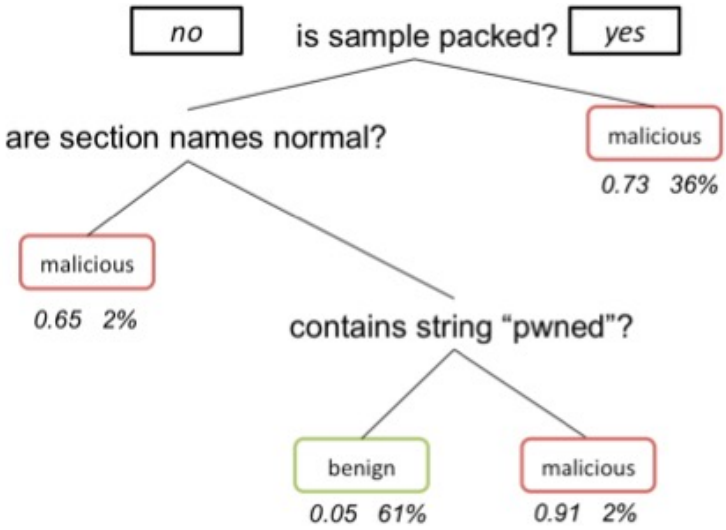
Once feature extraction and feature reduction have been performed automatically for a new sample, Endgame's proprietary data management pipeline alerts the model updating routine that a new sample is available for immediate use. Our cloud solution ingests the sample features immediately, and the on-premises model builder queues the sample and its metadata for a regular build.

Lightweight Models

The models deployed to a customer on-premises for malware hunting are engineered specifically to be lightweight but accurate. The stealth of Endgame's endpoint sensor is a key differentiator for the hunt, and machine learning models are engineered to respect that nimbleness. Endgame uses a variant of boosted decision trees in a WWsemi-supervised manner to achieve accuracy and interpretability in a very lightweight footprint.

A decision tree is a logical sequence of questions that in our case, aims to discriminate between malicious and benign. It's similar to playing a game of twenty questions on your feature set. Figure 4 highlights the process of asking a series of yes/no questions to determine the "branch" a sample falls in. When the series of questions finally falls to a "leaf", the result is a prediction based on the ratio of malicious and benign samples falling in that leaf. The tree is built by choosing appropriate questions that maximize the discrimination between malicious and benign. In boosted decision trees, we produce a weighted ensemble of trees where, for each tree, only a limited number of questions can be be asked. Each tree in the ensemble attempts to clean up the mistakes made by the previous trees. By limiting the depth of each tree (number of questions) and the number of trees, one can control the complexity of the model.

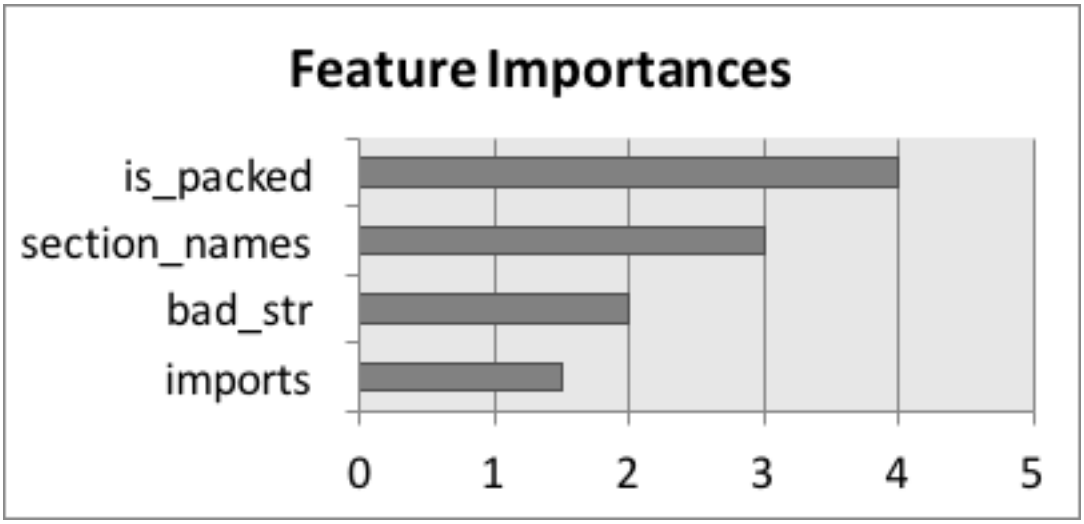
Figure 4. — Example of a Decision Tree. The figures under each leaf shows the probability of maliciousness and the percentage of training examples that fell in the leaf.

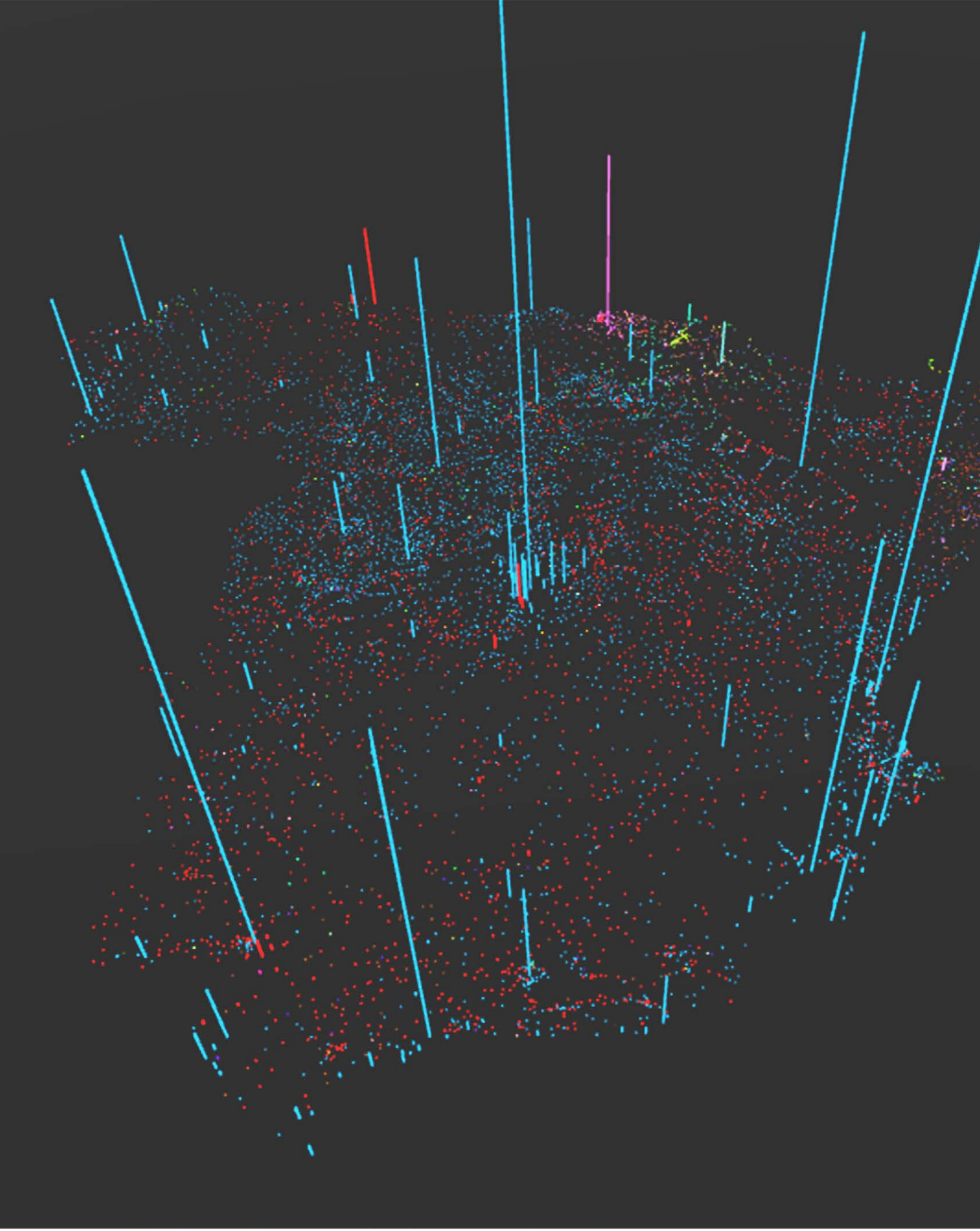


Decision trees are known to suffer from being overly sensitive in regions of the feature space where there are few labeled samples: it can be overconfident of its decision even where little evidence exists. Our novel use of boosted decision trees uses unlabeled data in addition to labeled data to reduce this effect, by instead training the model to be uncertain in regions of the feature space where the number of unlabeled samples dominates the number of labeled samples. The model can cue the hunter about both maliciousness and its confidence of maliciousness. This is a critical element in malware hunting, so that the hunter is not overwhelmed by false positives or succumbs to alert fatiguing, which inherently eliminates the utility of an alerting system.

A key benefit to using decision trees and their variants is the opportunity to gain context into why the model came to a certain conclusion. Endgame's proprietary boosted decision tree analysis provides per-sample descriptions for each prediction the model makes. Our use of an introspective machine learning model eliminate the black-box stigma commonly associated with other statistical learning techniques (Fig 5).

Figure 5. — Example of features importance that might be extracted for a sample. The lengths denote the number of times the corresponding feature was queried when making a determination of maliciousness for the sample in question.





Choosing just the right parameters of the model can be done safely (to protect against overfitting) through a slight twist on the common technique of parameter search via cross-validation. This is a try-and-compare process in which several variants of the model are trained on a subset of the training data, and cross-validated against the remaining holdout data. The model variant that, on average, performs best during cross-validation is selected as the winner. As eluded to previously, a nuance of malware data in an adversarial setting is that the "shape, size and color" of malware drifts over time which violates a basic tenet of vanilla statistical learning: that training and test data are drawn from the same statistical distribution. Thus, we depart from common cross-validation techniques that take random subsets to form the training and holdout sets. Instead, we perform progressive cross-validation, in which models are trained on the older samples, and cross-validated against progressively newer samples. This process more closely resembles model building and testing in a setting with adversarial drift. We capture performance metrics about the selected model in a similar way–on a holdout set of "most recent" samples.

In reality, malware comes in varying degrees of maliciousness, and sensitivity to various categories of malware varies from customer to customer. For example, adware and junkware (e.g., preloaded or bundled software) are often treated as less malicious for hunting operations, but tolerance for riskware (advanced tools that may or may not be used for malicious purposes) depends greatly on context. Endgame produces models that allows operators to prioritize alerts based on their risk aversion to these finer-grained categories.

Machine Learning and The Hunt

Endgame's use of machine learning for malware detection is a critical component of automating the hunt. Sophisticated adversaries lurking in enterprise networks constantly evolve their TTPs to remain undetected and subvert the hunter. Endgame's malware hunting solution automates the discovery of their tools in a covert fashion, in line with the stealth features within Endgame's cyber operations platform which was developed for elite US Department of Defense cyber protection teams. We've detailed only one layer of Endgame's tiered threat detection strategy: the endpoint. As alluded to, complementary models exist in the cloud that can provide additional information about threats, including malware, to the hunter as part of the Endgame cyber operations platform.

Endgame is bringing the latest research in machine learning and practices in data science to revolutionize information security. Although beyond the scope of this white paper, machine learning models—and data science more broadly—are equally applicable at the other stages of the hunt cycle—survey, secure, and respond. We've described the machine learning aspect of malware hunting, specifically the ability to to identify persistence and never-before-seen malware during the "detect" phase of the hunt. Given the breadth of challenges in the threat and data environment, automated malware classification can greatly enhance an organization's ability to detect malicious behavior within enterprise networks.

ABOUT ENDGAME

Endgame automates the hunt for the most sophisticated adversaries in enterprise networks. Endgame's technology and techniques are proven to detect and respond rapidly to cyber threats in the most extreme environments - from defending US national security interests to protecting the world's critical infrastructure.

The Endgame Cyber Operations Platform, developed for elite US DOD cyber protection teams, enables enterprises to automate the entire hunt mission, detecting and blocking adversaries at every phase of the cyber kill chain. Endgame's world-class R&D team extends our advantage with novel stealth technologies, vulnerability and threat analysis, and unique detection and prevention technology. At Endgame, we help our customers move from being the hunted to being the hunter.

Endgame was founded in 2008 and has offices in Washington, DC, San Francisco, CA, San Antonio, TX and Melbourne, FL. For more information, visit www.endgame.com and follow us on Twitter @EndgameInc.