BlackBerry
Intelligent Security. Everywhere.

White Paper

Introduction to AI for Security Professionals

A Practical Guide for Solving Cybersecurity Challenges with Artificial Intelligence and Machine Learning Technologies

Introduction

Artificial intelligence (AI) has moved beyond the realms of academia and speculative fiction to enter the commercial mainstream, with innovative products that utilize AI transforming how information is accessed, leveraged, and secured.

In the United States, AI is recognized as strategically important to national defense. In October 2016, for example, the federal government's National Science and Technology Council Committee on Technology (NSTCCT) issued a report[1] that stated, "AI has important applications in cybersecurity, and is expected to play an increasing role for both defensive and offensive cyber measures." The NSTCCT subsequently issued a National Artificial Intelligence Research and Development Strategic Plan[2] to guide federally funded research and development.

It's important for security professionals to gain a practical understanding about what AI is, what it can do, and why it's becoming increasingly important to the ways real-world security problems are approached.

Types of Artificial Intelligence

The field of AI encompasses three distinct areas of research:

1. *Artificial Superintelligence* (ASI) envisions a likely fictional future in which computers outperform humans on every kind of cognitive task, thereby rendering humans obsolete.
2. *Artificial General Intelligence* (AGI) aims to design computers that are as intelligent and capable of learning and reasoning as humans.
3. *Artificial Narrow Intelligence* (ANI) is an approach to solving discrete problems that exploits a computer's superior ability to process and detect meaningful patterns and relationships within vast quantities of data. ANI systems are adept, for example, at classifying executable files as benign or malware, detecting and responding to anomalous behaviors that may signal a cyber attack, and much more.

In recent years, most of the fruitful research and advancements in applying AI to cybersecurity have come from the sub-discipline of ANI called machine learning (ML), which focuses on teaching machines to learn by applying algorithms to data. This white paper focuses exclusively on ANI methods that fall within the ML space.

[1] Preparing for the Future of Artificial Intelligence

[2] The National Artificial Intelligence Research and Development Strategic Plan

What Kinds of Problems Can Machine Learning Address?

Not all problems are candidates for an ML solution. First and foremost, the problem must be one that can be solved by analyzing a data set that is accurate, relevant, and collectible in sufficient quantities. It must also be possible to process this data within a reasonable timeframe. Tens of thousands of calculations must often be performed before a result is obtained, so computers with sufficient processing power must be available to do so expeditiously.

Fortunately, the security domain is well-suited to ML analysis. Huge quantities of network and telemetry data are routinely generated and collected, including data from logs, network sensors, endpoint agents, and distributed directory and other network services. Within that data are the contextual clues needed to identify malicious activity and ameliorate threats, but only if the tools capable of teasing them out are available.

Context is critical in security. A single data point may only be significant based on the context in which it appears and its correlation with other security events. By acquiring a broad understanding of the activity surrounding the assets under their control, ML systems make it possible for analysts to discern how events widely dispersed in time and across disparate hosts, users, and networks are related. Properly applied, ML can provide the context security professionals need to reduce the risks of a breach while significantly increasing the costs threat actors incur for launching their cyber attacks.

Preparing Data for ML Analysis

In a perfect world, security professionals might wish to analyze all data to ensure the results accurately reflect network and computing environments. In practice, however, BlackBerry data scientists apply statistical sampling techniques that allow them to work with representative, more manageable subsets. Next, it is determined which data elements within the samples should be extracted and subjected to analysis. In ML, these data elements are referred to as features, i.e., attributes or properties of the data that can be analyzed to produce useful insights.

The relevant features might include the percentage of ports that are open, closed, or filtered; the application running on each of these ports; and the application version numbers. If investigating the possibility of data exfiltration by employees, security professionals might want to include features for bandwidth utilization login times, user access permissions, and end-user behavioral attributes that could reveal anomalies in how sensitive data is being accessed and utilized.

Typically, there are many thousands of features to choose from. However, each feature added increases the load on the processor and the time it takes to complete the analysis. Therefore, it's good practice to exclude those features known to be irrelevant based on security domain expertise.

Most ML algorithms require data to be encoded or represented in a particular mathematical format. One common approach is to map each sample and its set of features to a grid of rows and columns. Once structured in this way, each sample is referred to as a vector. The entire set of rows and columns is referred to as a matrix. Once the data has been vectorized, a variety of ML approaches can be applied. This white paper considers three of them: clustering, classification, and deep learning.

Clustering

The purpose of cluster analysis is to segregate samples into discrete groups or clusters based on previously unknown similarities among their key features or attributes. The more similar the samples are, the more likely they are to share the same cluster.

Humans experience the world in three spatial dimensions. This allows for determining the distance between any two objects by measuring the length of the shortest straight line connecting them. This Euclidean distance is what is computed when the Pythagorean Theorem is utilized.

Clustering analysis introduces the concept of a feature space that can contain many thousands of dimensions, one each for every feature being assessed in a sample set. Clustering algorithms work by assigning samples to coordinates in this feature space and measuring the distance between them.

Once analysis is complete, security professionals are presented with a set of clusters with varying numbers of members. Since malicious behavior is relatively infrequent, the clusters containing the largest number of samples will generally be associated with benign activity. Clusters with few members may indicate anomalous, potentially malicious activity that requires further analysis.

Clustering provides a mathematically rigorous approach for detecting patterns and relationships among network, application, file, and user data that might be difficult or impossible to discern in any other way.

Classification

Humans employ a wide variety of cognitive strategies to make sense of the world. One of the most useful is the capacity to assign objects and ideas to discrete categories based on abstract relationships among their features and characteristics. In many cases, the categories used are binary ones. Certain foods are good to eat, others are not. Certain actions are morally right while others are morally wrong. Categories like these enable generalizations to be made about objects and actions already known so that accurate predictions can be made about the properties of objects and actions that are entirely new.

Presented with an oval object with a yellow skin, a soft interior, and a sweet and pungent smell, one might draw on past knowledge to predict that it belongs to the category of fruit. The accuracy of this prediction can be tested by bringing the object to a fruit store. If a bin full of similar objects labeled as mangos is discovered, it can be concluded that the prediction is correct. If so, one's general knowledge of fruit can be used to predict that the mango has a pleasant taste and offers sound nutritional benefits. This categorical knowledge can then be applied to decide whether to eat the mango.

In the security domain, classification enables prediction of whether a piece of email should be classified as spam, or if a network connection is benign or associated with a botnet. In ML, the diverse set of algorithms used for categorization are known as classifiers. There are numerous types, each with its own strengths and weaknesses.

To produce an accurate classification model, data scientists need a large quantity of labeled data that has been correctly sampled and categorized. The samples are then typically divided into two or three distinct sets for training, validation, and testing. As a rule of thumb, the larger the training set, the more likely the classifier is to produce an accurate model. After that, a classification session typically proceeds through four phases:

1. A training phase in which the analyst constructs a model by applying a classifier to the training data.
2. A validation phase in which the analyst applies the validation data to the model to assess its accuracy.
3. A testing phase to assess the model's accuracy with test data withheld from the training and validation processes.
4. A deployment phase in which the model predicts the class membership of new, unlabeled data.

In practice, analysts train, validate, and test multiple models using different algorithms and processing methods before choosing the model with the optimal combination of accuracy and processing efficiency.

Classification Example Using Decision Trees

Decision tree (DT) algorithms determine whether a sample belongs to one category or another by defining a sequence of if-then-else decision rules that terminate in a class prediction. DT algorithms are aptly named, since they are comprised of roots, branches, and leaves. A hypothetical decision tree for identifying malicious URLs is shown in Figure 1.

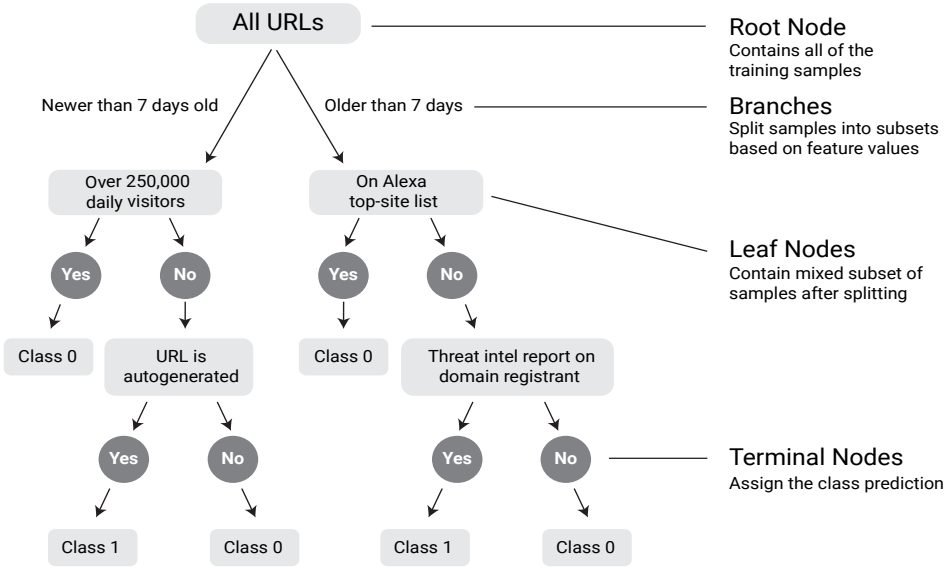

Figure 1: Hypothetical decision tree for identifying malicious URLs.

The tree is constructed top-down beginning with the root, which contains a mix of malicious and benign URLs. The goal is to split these samples into increasingly pure subsets based on their features and feature values. The if/then processing continues in this way until terminal node leaves are produced that contain samples of one class only (benign or malicious).

Supervised vs. Unsupervised Learning

Classification is an example of *supervised* learning, in which an analyst builds a model with samples that have already been labeled with respect to the property under investigation. Here, the job of the classifier is to ascertain how the feature attributes of each class can be used to predict the class of new, unlabeled samples. In contrast, clustering is an example of unsupervised learning, in which the properties that distinguish one group of samples from another must be discovered by the algorithm. It's not uncommon for data scientists to use both unsupervised and supervised methods in combination.

Deep Learning and Neural Networks

Deep learning is based on a fundamentally different approach that incorporates layers of processing, with each layer performing a different kind of calculation. Samples are processed layer by layer in stepwise fashion, with the output of each layer providing the input for the next. At least one of these processing layers will be *hidden*. It is this multi-layered approach, employing hidden layers, that distinguishes deep learning from all other ML methods.

Deep learning encompasses a wide range of unsupervised, semi-supervised, and supervised learning methods that are primarily based on neural networks, a class of algorithms so named because they simulate the ways densely interconnected networks of neurons interact in the brain. Neural networks are extremely flexible, general-purpose algorithms that can solve a myriad of problems in a myriad of ways. Unlike other algorithms, for example, neural networks can have millions or even billions of parameters that can be applied to control how a model is defined. In the example below, how a generic neural network can solve a classification problem is considered.

Neural Network Processing

As shown in Figure 2, neural networks are composed of nodes contained within input, hidden, and output layers. Each layer plays a distinct role in computing a classification. In a fully connected network, like the one shown here, the output of every node in a layer is connected to the inputs of every node in the layer that follows. The example shown is also an example of a feed-forward neural network, in which information passes directly from one layer to the next, without backtracking, until it reaches the output layer, where the classification decision is assigned. However, this is only one of many possible configurations. Neural networks can also employ feedback loops between layers and utilize partially connected architectures that restrict the flow of information to certain nodes only. For simplicity, the processing performed by each layer of a fully connected feed-forward type is considered.

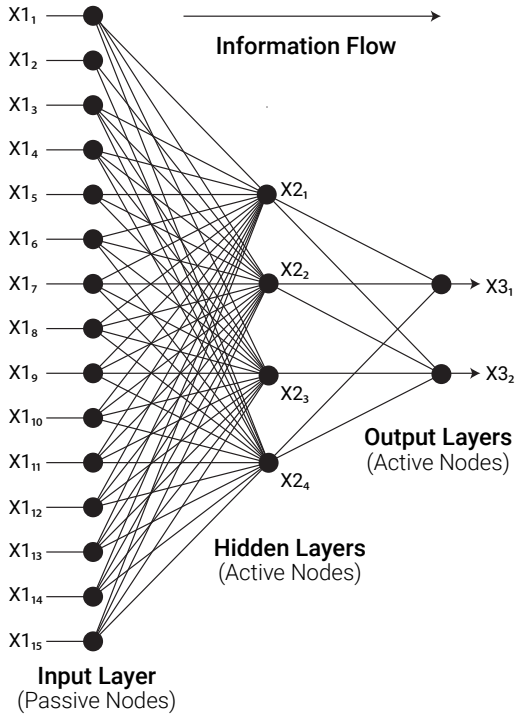

Figure 2: A generic neural network architecture.

Input Layer

The nodes in the input layer are passive. That is, they simply receive attribute values for a particular sample and then pass them on to be processed by all nodes in the first hidden layer. Consequently, the input layer must contain a node for every feature in the sample set. If categorizing pictures with a 64-pixel x 64-pixel resolution, for example, an input layer with 4,096 input nodes would be configured, one for each pixel. In solving a language processing problem, the relevant features might include the number of unique words in the sample being analyzed, the frequency with which each word appears, etc.

Hidden Layer(s)

Hidden layers are composed of nodes like the one in Figure 3 that perform the heavy lifting of the deep learning process. Since this is the first node in the first hidden layer, this processing proceeds as follows:

- **Receiving feature values from the input layer.** All attribute values for the first sample are received on the node's inputs x1-xm.
- **Applying weights.** Each attribute value is then multiplied by a corresponding weight value, e.g., the attribute on input x_1 is multiplied by weight w_1, the attribute on x_2 is multiplied by weight w_2, etc. If the magnitude of the weight value is greater than one, then the contribution of that feature to the eventual classification decision will incrementally increase. If the magnitude is less than one, its contribution will be decreased accordingly.

  Every input on every node in the hidden layers is initialized with different weight values, which are incrementally optimized after each processing cycle until the desired level of accuracy is achieved. Analysts can set these initial weights randomly, use functions to ballpark appropriate initial values, or set them based on their previous experience with similar problems and datasets.
- **Summing the products.** The products are then sent to a *Net Input* function that calculates the sum of the products and passes the result to an *activation function* for additional processing.
- **Applying the activation function.** The *activation function* performs the calculation specified for that layer. Analysts can choose from among a large set of *activation functions* based on the nature of the problem scenario and the sequence of computations required to produce a solution.
- **Outputting the result.** The result of the activation function is a numeric value that reflects the aggregate effects of that node's processing. Each of these values represents a different proportion of weights and combinations of feature attributes. By processing all these combinations and passing the results onward to additional hidden layers, neural networks determine step by step which combination of features and weights most accurately predict a sample's class assignment.

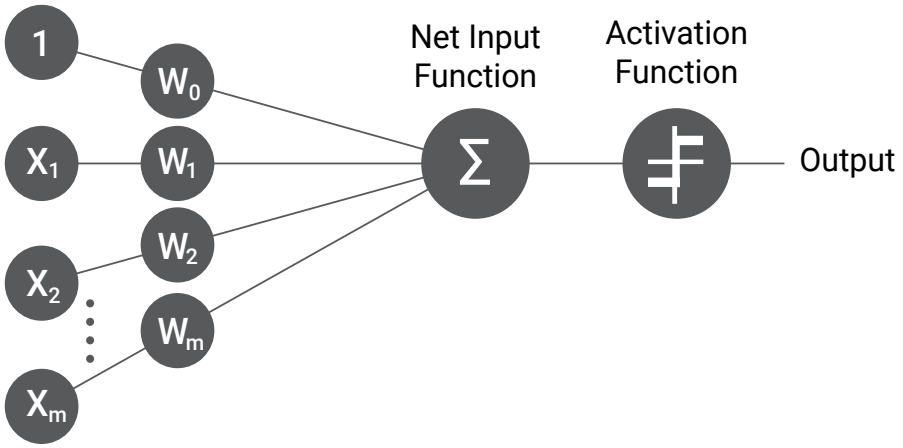

Figure 3: Node #1 in Hidden Layer #1.

Processing in Hidden Layers 2-n

Every node in hidden layer #2 receives the output values from *all* nodes in hidden layer #1. Once again, each of these values is multiplied by a particular weight value and the products are summed. The results are then subjected to an activation function to produce a new output for the next layer, where the process repeats. This continues until all the hidden layers have been traversed and the results of those calculations arrive at the output layer.

Output Layer

The output layer is the final layer in the neural network. Since this is a classification exercise, the output layer will incorporate a node for every possible class assignment. Like those in the hidden layers, the nodes here are active, meaning they too can incorporate activation functions. One common example is an activation function that converts the classification decision into a probability score. The node with the highest score will determine which class label is assigned.

After each training cycle, a loss function compares the classification decision to the class labels to determine how the weights in all hidden layers should be modified to produce a more accurate result. This process repeats as many times as required before a set of candidate models can proceed to the validation and testing phases.

Conclusion

This white paper provides an introduction to the kinds of tools and processes BlackBerry data scientists utilize to solve complex cybersecurity challenges. While the science is impressive, it's only a starting point. What sets BlackBerry apart is the sophistication and security expertise with which these tools are applied, and the huge, ever-expanding store of proprietary security data available to build BlackBerry® models. Collectively, these assets enable BlackBerry experts to frame security problems correctly and produce solutions that help organizations minimize their cyber-risk exposure and optimize their resilience.

To learn more, visit the website.

About BlackBerry

BlackBerry (NYSE: BB; TSX: BB) provides intelligent security software and services to enterprises and governments around the world. The company secures more than 500M endpoints including 175M cars on the road today. Based in Waterloo, Ontario, the company leverages AI and machine learning to deliver innovative solutions in the areas of cybersecurity, safety, and data privacy solutions, and is a leader in the areas of endpoint security management, encryption, and embedded systems. BlackBerry's vision is clear — to secure a connected future you can trust.

For more information, visit BlackBerry.com and follow @BlackBerry.

BlackBerry
Intelligent Security. Everywhere.

BB20-XXXX | 210623