APPLIED MACHINE LEARNING FOR DATA EXFIL AND OTHER FUN TOPICS

Mike Wojnowicz, Data Scientist
Jay Luan, Data Scientist
GOALS OF THIS TALK:

APPLIED MACHINE LEARNING

• Identify suitable problems for ML approaches
• Demonstrate by example how to apply ML
• Help jumpstart additional research in the Security + ML space
WHO WE ARE/CYLANCE

• Endpoint security company built around the capabilities of artificial intelligence
• Protecting millions of enterprise endpoints
• Founded in 2012, $177 mm raised

Mike Wojnowicz
Data Scientist

Jay Luan
Data Scientist
TALK OVERVIEW

• Machine Learning Introduction
• NMAP Clustering
  • Feature Spaces
  • Distances
  • Clustering
• Botnet Panel Identification
  • Classification
  • Feature Reduction
• Obfuscating Data with Markov Chains
Sources:

- https://github.com/CylanceSPEAR/NMAP-Cluster.git
- https://github.com/CylanceSPEAR/MarkovObfuscate.git
- https://github.com/CylanceSPEAR/IDPanel.git
- https://github.com/bwall/kmeans_demo.git
- https://www.youtube.com/channel/UCJ7y5wQUIApOW-z0zmbD_Ew

- All links can be found at: http://pastebin.com/4JdtjEAk
MACHINE LEARNING OVERVIEW

• Machine learning techniques are data driven
• Available data should be able to solve the problem in a meaningful way
• Approaches exist for dealing with raw data, as well as labeled or annotated data
MACHINE LEARNING OVERVIEW

• Given some data, different types of machine learning can be applied
• **Clustering** is useful for finding similarity across dataset to uncover trends or other insight
• With **labeled** data, **classification** can be useful to build predictive models
MACHINE LEARNING OVERVIEW

• Often, raw data has to be transformed in some way to be used by machine learning algorithms
• Typical process is to extract **features** from data, and turn those **features** into **vectors**
• **Vectors** are then fed into ML algorithms for training or other purposes
MACHINE LEARNING OVERVIEW

• Recommended resources
  • Scikit-learn.org
  • Python

• Source code for all tools in this talk available on the Cylance public git repo
  • https://github.com/CylanceSPEAR

• Should be able to pull talk source and start modifying as needed to suit data driven problems in your own organization or research group.
TOOL – NMAP CLUSTERING

• NMAP is a popular port scanning tool
• Produces large amount of data per IP address
• Scans over large number of IPs can be difficult to make sense of
• NMAP Clustering is a tool which clusters (groups) IPs based on their open ports, services, etc
FEATURES

• Features are informative, discriminative information that can describe a sample/observation/phenomenon/etc.
• Feature extraction is pivotal to the machine learning pipeline
• Our features are based on NMAP output
• Each port is a feature, each service on each port is a feature, each version of each service on each port is a feature, etc
• Script output included in features (including website titles, public keys, etc)
VECTORS

- Numerical representation of a sample (IP in NMAP case)
- Array of values which represent all features from one sample
- Vectors can be thought of as points in high dimensional space
- Each feature is a dimension, the value of the feature in the vector is the position in that dimension
- If we have only two features, it is really easy to visualize
VECTORS – 2D

File size (kilobytes) vs. Filename length

- calc.exe: 918.528 bytes, 8 characters
- notepad.exe: 193.024 bytes, 11 characters
VECTORS – 3D

<table>
<thead>
<tr>
<th>File</th>
<th>Filename Length</th>
<th>Filesize (kB)</th>
<th>Size of headers (kB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>calc.exe</td>
<td>8</td>
<td>918.528</td>
<td>63</td>
</tr>
<tr>
<td>notepad.exe</td>
<td>11</td>
<td>193.024</td>
<td>45</td>
</tr>
<tr>
<td>malware.exe</td>
<td>11</td>
<td>193.024</td>
<td>10</td>
</tr>
</tbody>
</table>
DISTANCES

- **Distance:**
  Describe the discrepancy between two points.
- **Physical distance between two points:**
  **Pythagorean’s theorem:**
  \[ a^2 + b^2 = c^2 \]
<table>
<thead>
<tr>
<th>File</th>
<th>Filename Length</th>
<th>Filesize (kB)</th>
<th>Size of headers (kb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>calc.exe</td>
<td>8</td>
<td>918.528</td>
<td>63</td>
</tr>
<tr>
<td>notepad.exe</td>
<td>11</td>
<td>193.024</td>
<td>45</td>
</tr>
<tr>
<td>malware.exe</td>
<td>11</td>
<td>193.024</td>
<td>10</td>
</tr>
</tbody>
</table>

![2D Diagram](image2d.png)

![3D Diagram](image3d.png)

Applied Machine Learning • Wolff • Wallace • Zhao
DISTANCES

Multiple Distance Metrics –

As long as an operation satisfy certain mathematical criteria, it can be used as a distance metric

- Euclidean Distance: $\sqrt{a^2 + b^2}$
- Manhattan Distance: $|a| + |b|
- Other Distances

  - $L_p$ Norms: $(a^p + b^p)^{1/p}$
  - Cosine Distance: $\frac{a \cdot b}{\|a\| \|b\|}$
CLUSTERING

- With a way to measure distance, we can group items by how close they are, aka clustering
- Clusters are distinct groups of samples (IP) which have been grouped together
- Clustering is generally unsupervised learning
- Different algorithms with different configurations group these samples in different ways
k-Means

- Clustering algorithm
- User supplies $k$ (designated number of clusters)
- All samples are assigned to random clusters
- Center of each cluster is computed by taking mean (average) of all samples in cluster
- Samples are then assigned to the cluster whose center they are closest to
- Centers are recomputed, algorithm loops until no samples change clusters
k-Means
NMAP CLUSTERING – MANUAL/AUTOMATIC

• Manual allows you to supply your own clustering parameters
• Automatic tries many different methods with theoretically-found optimal parameters and picks what it determines to be the best
• Demo with manual strategy
• Demo with automatic strategy
NMAP CLUSTERING - INTERACTIVE

• Incorporate the User’s decision into the clustering process.
• The Clustering result will be customized according to customer’s preference in this way
• Process (will show with a demo):
  1. User decide whether the cluster needs to be split or not:
  2. If yes, then split using divisive clustering
  3. If no, finalize this cluster
  4. Recursively split until users are satisfied with all the clusters
TOOL – ID PANEL

- Botnet panels (command and control websites) can be difficult to identify
  - Need previous knowledge of the botnet panels
  - Often modified to avoid detection or vanity
  - Many are based off others, so distinguishing can be difficult

- We can train a model to identify if we are looking at a bot panel and which one it is, with a small number of requests

- Minimizing the number of requests to classify improves stealth and rate of classification
CLASSIFICATION

• This is a classification problem
• Classification answers “Is it what we are looking for?”
• Classification is generally supervised learning
• Supervised learning requires training samples to have labels
• Classification methods range from simple to highly complex
ID PANEL FEATURES

- Botnet panels are similar to normal websites
- Contain various file types, often edited
- HTTP response codes + content comparison
- Encoding content as features difficult
- ssDeep provides a continuous value by comparing content
COLLECTING DATA

• Collection of known botnet panels
• Request all known paths for all known botnet panel types
• Store HTTP status code and ssDeep of content
• Collection of sites that are not botnet panels needed as well
DECISION TREES

- Decision trees are simple classifiers
- Splits the dataset one feature at a time until decision is confident
- Results in a tree of queries where the results produce a decision
- Simple to train
Decision Trees – Evaluating Splits

- Find best feature values to split on to get the most information gain
ENSEMBLE OF DECISION TREES

- A single decision tree may be over focused on training data
- Can alleviate this problem by building multiple Decision Trees for each label
- Combining the results allows each Decision Tree to vote
- Partial answers may still be of interest to the user
- Ensembles can obtain better predictive performance
ID PANEL DEMO – COMMAND LINE

• Quick way to check if a website directory contains a botnet panel
• Easy to batch searching of multiple websites/directories
• Easy to grep results
ID PANEL DEMO – CHROME EXTENSION

• Every website directory visited is tested
• Results are stored in browser
• ssDeep ported from C to Javascript
• https://github.com/kripken/emscripten
• Extension available in Chrome extension store (free, of course)
TOOL – MARKOV OBFUSCATE

• Data exfiltration from a network often requires avoiding an outbound firewall
• Deep packet inspection looks to block anything undesirable
• Easy to encrypt data, but it’s also easy to drop information that can’t be read
• We can make our data look like something else entirely
**MARKOV CHAIN**

- Simple machine learning method for characterizing sequence data
- Learns the transition pattern from a state to another based on how likely a state comes after another state in the training data
- We can create sequences with transition patterns that are learned from the data it was trained on

<table>
<thead>
<tr>
<th>Transition Matrix</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1 (25%)</td>
<td>3 (75%)</td>
</tr>
<tr>
<td>B</td>
<td>2 (100%)</td>
<td>0 (0%)</td>
</tr>
</tbody>
</table>

Training Data:

- AAB
- BAB
- ABA

**Transition Matrix**

\[
P(ABB) = 0 \\
P(ABA) = 0.75 \\
P(AAB) = 0.1875 \\
P(AAA) = 0.0625
\]
MARKOV CHAIN

• This and related ideas (hidden markov models, recurrent neural networks) can be used to accomplish all sorts of interesting goals:
  • Speech generation
  • Continuous speech recognition
  • Embedded shellcode detection

```python
markov.learn(["im on fire", "im going to die"])
```

<table>
<thead>
<tr>
<th></th>
<th>im</th>
<th>on</th>
<th>fire</th>
<th>going</th>
<th>to</th>
<th>die</th>
</tr>
</thead>
<tbody>
<tr>
<td>im</td>
<td>0</td>
<td>.5</td>
<td>0</td>
<td>.5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>on</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>fire</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>going</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>to</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>die</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Given our previous knowledge of how Markov chains work, we can potentially use this knowledge to create an obfuscated stream of data which appears like English.

```python
current_word = "apple"
sorted_list_of_words_following_apple = [
    "sucks",
    "over-charges",
    "sauce",
    "cider",
    ...
]
byte_to REPRESENT = 0x2
```

Instead of using the most likely word to follow our current word, we use the Nth most likely word to follow, where N represents a single byte we are trying to send.
MARKOV OBFUSCATE - ENCODING

• Train our model with a book
• Observing transitions from word to word
• Generate data based on transition probabilities
• Demo
MARKOV OBFUSCATE - WRAPPING

• Simple to transfer our data through a pipeline that looks like normal HTTP traffic
• Looks like a user posting to their blog
• Demo
MARKOV OBFUSCATE – HAVING FUN

• Train our models on Taylor Swift lyrics
• Train a Markov Model based on Taylor Swift songs
• Play the generate lyrics through festival with tones/beats learned from songs
• First live “Tylance Swift” concert, demo
WRAPPING UP

• Any problem where there is a significant amount of data generated could benefit from a machine learning approach
• Lots of great online resource to help anyone get started
• Having labeled or annotated data makes more ML approached viable compared to unlabeled data
QUESTIONS?

• Email: machinelearning@cylance.com
• Career opportunities: https://www.cylance.com/cylance-careers