Artificial Intelligence for Security

Assignment: Analysing Network Traffic

(Mon 25/11/2019)

In this assignment we will analyse some network traffic data and build a system for detecting network attacks. An optional extension to the assignment will look at android application data and build a system to detect malware.

The assignment:

- must be completed in **groups**:
    - Each group should consist of a **minimum of 3** and a **maximum of 4** students.
- involves writing a **Jupyter notebook** to analyse the traffic data as described below.
    - Notebooks should be self-explanatory, with "mark-down" containing description of what analysis you are performing and discussion of the results.
- is due by **midnight Thursday the 19th of December** (via BEEP).
- requires each group to **present their notebook** during the **practical session** on Friday 20th of December at 2:15pm.

The datasets:

- The assignment makes use of 2 publicly available datasets.
- The first dataset contains network flow data with simulated attacks on a network.
    - It comes from: https://www.hs-coburg.de/forschung-kooperation/forschungsprojekte-oeffentlich/informationstechnologie/cidds-coburg-intrusion-detection-data-sets.html
    - The full dataset is quite large (each CSV file is over 1GB), so we have provided you with 10% samples of the data (see the folder description below).
    - If you'd like to download and use the entire dataset, you can, but beware that you will ***need considerable memory***, so first get things working on the sample we've provided!
        - (In the download, the folder we are using is: CIDDS-001/traffic/OpenStack)
- The second dataset contains features detailing the behaviour of android applications, some benign and some malicious.
    - The data was downloaded from: https://figshare.com/articles/Android_malware_dataset_for_machine_learning_2/5854653
    - The dataset is not particularly large, (although it does contain a large number of features per example), so we have not made any changes to it.

In the "data" directory you will find two folders:

- NetworkTraffic:
    - Use this data for part 1 of the assignment.
    - Note that there are 2 CSV files -- we'll use one for training and the other for testing.
    - There is also a PDF in the directory describing the data, which is a MUST READ.
- AndroidMalware:
    - Use this data for part 2 of the assignment.
    - Note that there are 2 CSV files, one contains the data and the other provides a mapping of features to categories.
    - A PDF file in the directory describes the (drebin) dataset and the experiments that that authors had performed on it (also a must read).

Part 1) Network Traffic Analysis

First load the dataset for week 1 into a python notebook.
- We will use data from week 1 for training a system and keep the week 2 data for testing purposes only.
- Once loaded, do some analysis of the data, such as SOME OF the following:

Describe the data:
- How many columns and how many rows are there?
- How many unique values does the 'class' attribute take?
- How many rows are there for each unique value of the class attribute?
- What unique values does the 'attackType' attribute take on?
- List the count of each of the possible attack types.

Have a look at the other columns:
- Which columns contain numeric values?
- Do they all correspond to "numeric variables" or are some of them categorical? Note that categorical values are often expressed as numbers! (For numeric types, comparisons with with “>” and “<” makes sense, for categorical variables they don’t.)
- List all numeric types and compute simple statistics (minimum, maximum, average, etc.) for each.
- Plot the distribution of the numeric variables using a histogram. Do any of the values look too large or too small?

Have a look at the times and dates that the system has come under attack:
- Convert the 'Date first seen' column to a datetime object (see code in Appendix).
- Create new columns for the day-of-week or time-of-day.
- Plot histograms of the day-of-week and of the time-of-day to see their distribution in the data.
- Select only the rows corresponding to attacks and replot the histograms.
- Does the day-of-week or time-of-day affect the probability of seeing an attack?

Now create some classifiers using just the numeric attributes and try to predict the ‘class’:
- Train a logistic regression classifier.
- How well does the classifier perform in terms of accuracy on the TRAINING data?
- Load data from the second week and use it as TEST data.
- How well does the classifier perform (in terms of accuracy) on the TEST data?
- Is that performance good? Print out confusion matrices to better understand the results.

Compare performance across different classifiers:
- Try a Decision Tree and a Random Forest. Does the performance improve? Why?
- How does GaussianNB perform on the numeric features?
- Beware running the SVM classifier: There is a lot of data here, so you would need to use a linear SVM rather than an RBF one, (and even then it may run out of memory without adjustments).
- Convert the class to a binary attribute (malicious vs benign) and compute precision and recall values. Anything of note?

Predicting another target:
- Rather than predicting the class variable, try to predict the “attackType” variable.
- Is that variable easier/harder to predict?
- The data is quite unbalanced. You could try to rebalance it by randomly down-sampling the majority class. Does it improve accuracy on the (not-down sampled) test data?

Introducing more features:
- Introduce other features to the classifiers to see if you can further improve performance …

- Try adding day-of-week and time-of-day features into the classifier. Do they improve the performance?
- Split the “Flags” column up into separate columns for each of the flags (see code below for how to do that) and then make use of the flags as features. Do they improve the performance?
- Source IP addresses can be one of four categories (code in the appendix can be used to map IP addresses to the categories). Are the categories useful features?

Creating counts

- When detecting a Denial of Service (DoS) or portScan attack, counts of the number of requests from the same Source IP Address within a given time window could be an important feature. (For a DDoS attack, the same may be true of Destination IP Addresses.)
- Try adding one or more count fields to the dataset using the code from the Appendix.
- Does the count field improve performance of the classifiers? What window or windows make sense?

Think about other analysis you could perform on the data:

- What other features might be interesting? Try to compute them by modifying the code in the Appendix.
- Try running some other analysis like clustering on the data.

Part 2) Android Malware detection (OPTIONAL → for higher marks)

Try doing some visualisation and analysis of the data.

- What types of features look most predictive of the class?
- Train some classifiers to identify the malware. How well do they perform?
- The second CSV file in the directory contains a mapping from features to categories (“API call signature”, “Intent”, “Manifest Permission”, etc.).
- It would be interesting to know which categories of features are most important for making good predictions. Use the mapping to select only features of a particular category and train the model on each category of features separately to understand which of the feature categories is most important.
- Try doing some feature selection to find the subset of features (say 10) that are most important.
- Try clustering the data to see if the malware instances cluster together.

Appendix:

Some code to help with some of the tasks in part 1 of the assignment:

- Converting column from string to datetime format:

```
df1['Date first seen'] = pd.to_datetime(df1['Date first seen'])
```

- Getting time-of-day and day-of-week features from datetime object:

```
df1['Date first seen'].dt.dayofweek
df1['Date first seen'].dt.hour
```

- Splitting the Flags field into separate fields:

```
df1["Flag_1"] = [x[0] for x in df1["Flags"]]
df1["Flag_2"] = [x[1] for x in df1["Flags"]]
df1["Flag_3"] = [x[2] for x in df1["Flags"]]
df1["Flag_4"] = [x[3] for x in df1["Flags"]]
df1["Flag_5"] = [x[4] for x in df1["Flags"]]
df1["Flag_6"] = [x[5] for x in df1["Flags"]]
```

- Code for assigning Source IPs to 4 categories:

```
def get_IP_category(ip_addr):
    if ip_addr == 'DNS':
        return 'DNS'
    elif ip_addr == 'EXT_SERVER':
        return 'EXT_SERVER'
    elif len(ip_addr.split('.')) == 4:
        return 'private'
    elif len(ip_addr.split('_')) == 2:
        return 'public'
    else:
        return '-'
```

- Create new column with count of requests from same 'Src IP Addr' over the last 10 seconds:

```
df1['Cum Count Src IP Addr (10seconds)'] = df1.groupby(['Src IP Addr', pd.Grouper(freq='10S',
key='Date first seen')]).cumcount()
```