Cyber Data Analytics
Assignment 3

Ahmet Güdek (4307445)
Takang Kajikaw Etta Tabe (4373758)
Group 40

June 24, 2018

All code used in this assignment can be found on `https://github.com/agudek/CDA/tree/master/data_stream`. Links to the datasets used in this assignment can also be found here.

Sampling

We implemented win-wise sampling and calculated the distribution of the top 10 IP address appearances in dataset 48 for different values of k. We then compared this to the real distribution of the full dataset. Only IP addresses communicating with the infected host with IP 147.32.84.165 were considered. The results can be found in table 1 below. These show that in order to obtain a fairly accurate distribution of the top 10, the reservoir size k needs to be very large (Close to the total number of values). However, the most occurring address can already be found at $k = 100$.

real (n=5877)		k=100		k=250		k=1000		k=5000	
IP	p(IP)	IP	p(IP)	IP	p(IP)	IP	p(IP)	IP	p(IP)
91.212.135.158	0.069	91.212.135.158	0.050	91.212.135.158	0.044	91.212.135.158	0.048	91.212.135.158	0.069
64.59.134.8	0.033	206.46.232.11	0.030	64.59.134.8	0.040	64.59.134.8	0.035	64.59.134.8	0.033
24.71.223.11	0.027	216.32.180.22	0.030	216.32.180.22	0.036	216.32.180.22	0.031	24.71.223.11	0.026
216.32.180.22	0.026	38.113.116.213	0.030	65.54.188.110	0.028	65.55.37.72	0.020	216.32.180.22	0.026
147.32.96.45	0.017	65.55.37.72	0.030	147.32.80.9	0.024	206.46.232.11	0.019	65.55.37.88	0.017
90.177.113.3	0.017	65.55.92.136	0.030	65.55.37.120	0.024	24.71.223.11	0.019	90.177.113.3	0.017
65.55.37.72	0.016	90.177.113.3	0.030	65.55.37.72	0.024	65.55.37.120	0.018	147.32.96.45	0.016
65.55.37.88	0.016	202.108.252.141	0.020	200.142.128.100	0.020	147.32.96.45	0.017	65.55.92.136	0.016
65.55.92.136	0.016	212.247.156.1	0.020	202.108.252.141	0.020	65.54.188.72	0.017	65.55.92.152	0.016
265.55.92.152	0.015	213.165.64.100	0.020	206.46.232.11	0.020	65.55.92.136	0.017	65.55.37.72	0.016

Table 1: Real vs. approximated distribution using min-wise sampling

Our implementation allows evaluation in one pass by storing necessary values in a linked list, thus having a space complexity of $O(k)$. Running takes between 9 to 9.4 seconds for increasing values of k with most time required for reading in the values from file.

Sketching

We implemented count-min sketching with Murmurhash hashing function and Kirsch-Mitzenmacher optimisation to speed up hashing operations when using the sketch with a large number of hashes. We then calculated the distribution of the top 10 IP address appearances in dataset 48 for different values of table size w and hash size d. Once again, only IP addresses communicating with the infected host with IP 147.32.84.165 were considered. Table 2 shows the obtained results and show a much closer resemblance to the real distribution found in table 1 above than min-wise sampling.

w=100, d=10		w=250, d=5		w=1000, d=5		w=500, d=3		w=250, d=3	
IP	p(IP)	IP	p(IP)	IP	p(IP)	IP	p(IP)	IP	p(IP)
91.212.135.158	0.071	91.212.135.158	0.070	91.212.135.158	0.069	91.212.135.158	0.069	91.212.135.158	0.070
64.59.134.8	0.035	64.59.134.8	0.034	64.59.134.8	0.033	64.59.134.8	0.034	64.59.134.8	0.035
24.71.223.11	0.029	24.71.223.11	0.028	24.71.223.11	0.027	24.71.223.11	0.028	24.71.223.11	0.028
216.32.180.22	0.029	216.32.180.22	0.027	216.32.180.22	0.026	216.32.180.22	0.026	216.32.180.22	0.028
147.32.96.45	0.019	65.55.37.88	0.018	147.32.96.45	0.017	147.32.96.45	0.017	65.54.188.72	0.018
65.55.92.136	0.018	65.55.37.72	0.018	90.177.113.3	0.017	65.55.37.72	0.017	65.55.37.72	0.018
65.55.92.168	0.018	147.32.96.45	0.017	65.55.37.72	0.016	90.177.113.3	0.017	65.55.37.88	0.018
65.55.37.72	0.018	90.177.113.3	0.017	65.55.92.136	0.016	65.55.92.136	0.017	147.32.96.45	0.017
65.54.188.72	0.018	65.55.92.136	0.017	65.55.37.88	0.016	65.55.37.88	0.016	90.177.113.3	0.017
202.108.252.141	0.018	65.54.188.72	0.016	65.54.188.72	0.015	65.54.188.72	0.015	65.55.92.136	0.017

Table 2: Distribution results using count-min sketch

To be able to compare with the performance of min-wise sampling, sketching was implemented in a similar way, allowing evaluation in one pass by storing the top 10 in a linked list in addition to the matrix required for the sketch. This gives the algorithm a space complexity of $O(wd)$. Running takes between 9.1 to 9.7 seconds for different values of w and d.

Flow Discretisation

Before performing the discretisation, we first need to find out which attributes to discretise. For this reason we first filtered out flows labeled as background and then plotted the continuous attributes (bytes, packets and duration). Figure 2 in the appendix shows these attributes with a red background for flows that are malicious. From this we can see that the bytes and packets attributes show the most distinct changes during malicious activity, so these attributes are discretised.

For discretisation we used simple splitting boundaries chosen appropriately from figure 2 and discretised the atributes into 5 labels each. The results of the discretisation can be found in figure 1 below. The labels were then combined

into a single label by concatenating the labels of the discretised attributes and written to a separate csv file for further usage.

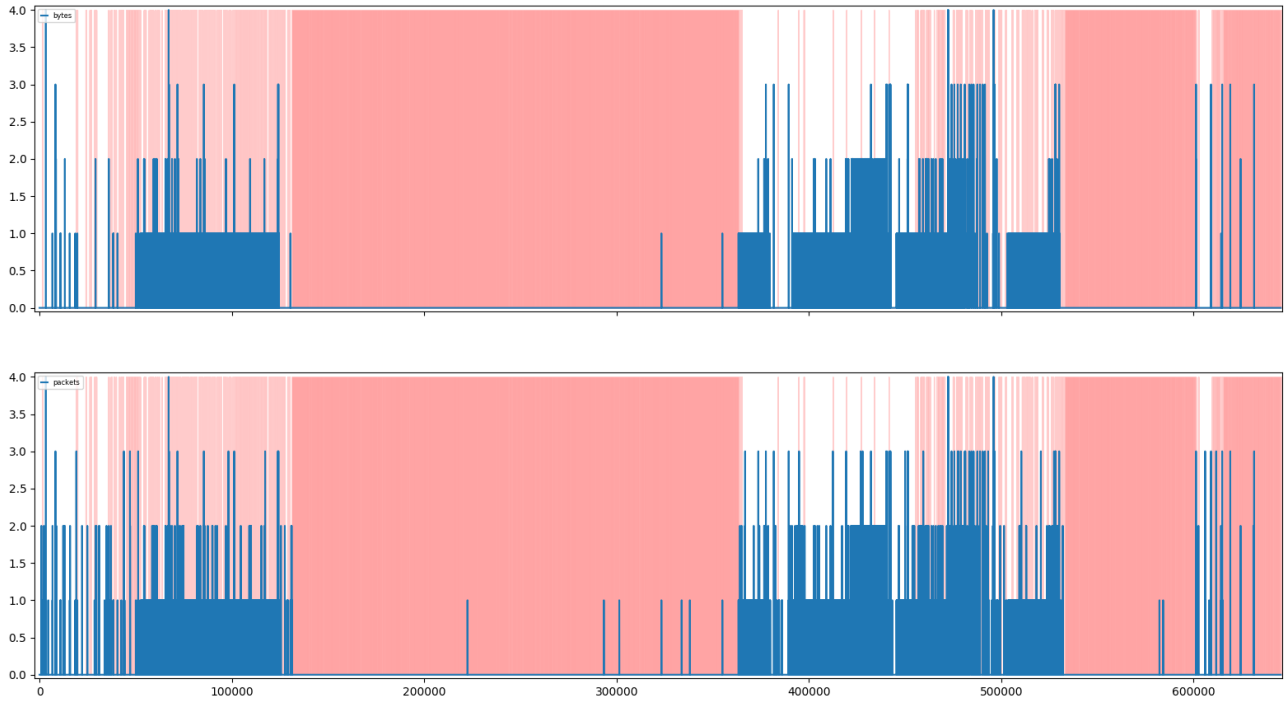

Figure 1: Visualisation of continuous variables packet length and packet count after discretisation. Malicious activity is plotted on a red background.

Botnet Profiling

In order to answer this question, the discretized data from the previous question is used. In order to obtain sequence data, we made use of sliding windows. In order to check the performance of the model, different windows are used as shown in Table 3. N-grams are used as probabilistic sequential model on infected host "147.32.84.165". For this, the n-grams had to be extracted from the aforementioned sequence data. In order to achieve this, we made use of the top 9 N-grams for the modeling of the fingerprints. The fingerprints for each host consists of pairs of N-grams and their corresponding frequency of appearance in the specified host. For the fingerprints of the host to be classified, we compare these fingerprints with those of a normal host's fingerprints and those of an infected host's fingerprint.

Table 3 shows that the sliding window has minimal effect on the results for small time differences. The most important factor to look at is the N value (number of N-grams used). For N = 2, 1 botnet is misclassified as benign and

no normal host is classified as a botnet. For N = 6, we notice that 3 botnets are misclassified as normal, while 1 normal host is misclassified as a botnet. The fact that the model also considers some normal hosts as botnets portrays that it chooses to go for high recall over higher precision.

In order to evaluate the model, the single scenario simulation implemented in Pellegrino's paper is performed.

For the single scenario simulation, scenario 10 was chosen which consists of 10 infected hosts and 6 normal hosts as presented in `https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-51/`. From both the normal and infected host, one IP of both categories is used for the training data as $train\_normal$ and $train\_infected$ for which the fingerprint is modeled and used as a model of how normal and botnet network flows look like. We use the other 15 IPs as testing. Each of the 15 testing hosts are then classified based on if it's nearest neighbor is the train fingerprint of the normal or botnet host.

To conclude, the specific model is good at detecting botnets eventhough it sometimes misclassifies for normal hosts as botnets.

Table 3: N-grams model for different window size and N values

window	N-grams	TP	FP	TN	FN	Precision	Recall
90ms	6	9	1	1	3	90%	75%
60ms	6	9	1	1	3	90%	75%
90ms	2	9	0	4	1	100%	90%
60ms	2	9	0	4	1	100 %	90%

Appendix

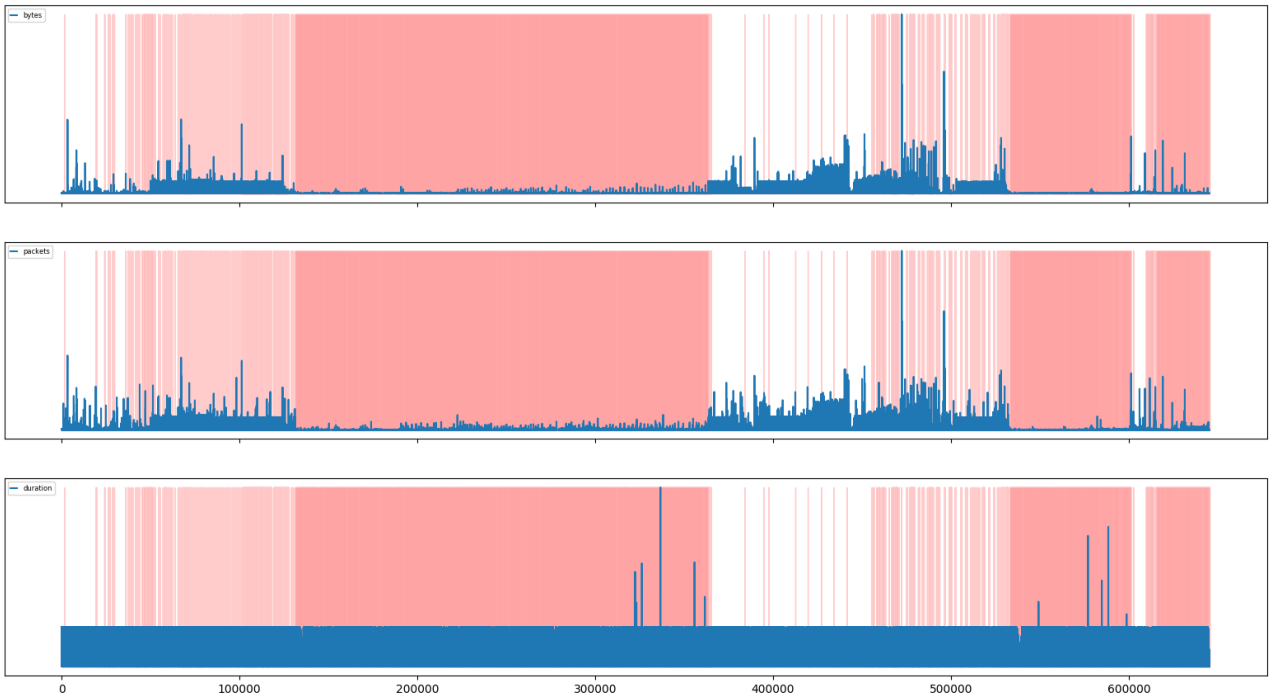

Figure 2: Visualisation of continuous variables packet length, packet count and duration (normalised). Malicious activity is plotted on a red background.