Project Title

Analysis and detection of malicious events in Network traffic

Final Report

GitHub repo:
https://github.com/anushab97/Data-Mining-CSCI-5502-Project-Group-04.git

Video Link:
Link to Data mining project presentation video

Submitted by: Group #04

1. Tim Coleman (timothy.coleman@colorado.edu)
2. Anusha Basavaraja (Anusha.Basavaraja@colorado.edu)
3. Christopher Bisbee (Chris.Bisbee@colorado.edu)

Background, Motivation, Goals:

- As the number of IoT and network related devices keeps growing at an exponential rate, security is a major concern among network administrators and companies at large trying to secure the networks and devices from malicious attacks such as ransomware, spyware, DDoS, and other types of attacks.
- The goal of our project was to analyze and learn interesting patterns from network traffic that contains known malicious intent in an effort to try and prevent attacks from happening and mitigating loss.
- There are published data sets such as CTU-13 and IoT 23 which contain netflow and pcap network captures that are classified as benign or malicious. Part of the goal for this project would be to build a machine learning model in an attempt to detect malicious events and optimize the balance of false positives to false negative ratios and maximize the true positive detection rate.

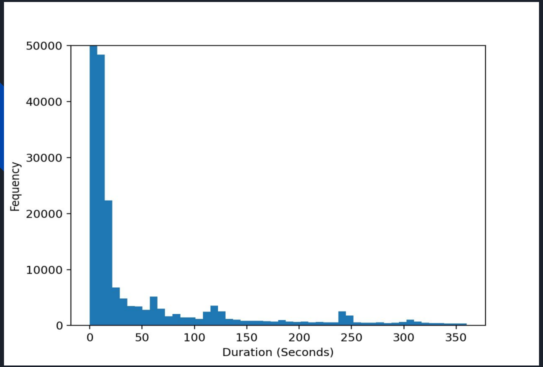
Questions:

- What are the preprocessing tasks that needs to be done to prep the dataset for Data mining task?
- What are the classification techniques used to classify the netflow data into benign and malicious classes?
- How to compare different supervised ML models to detect the botnet traffic?
- What future work can be performed?

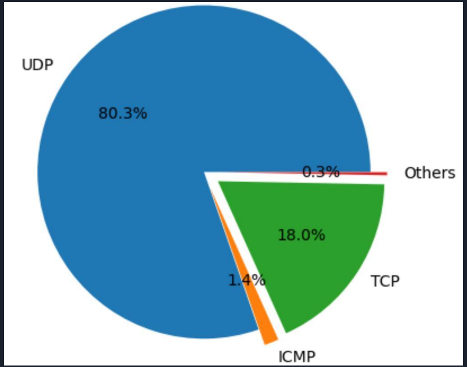
Dataset:

- Dataset: The CTU-13 Dataset (https://www.stratosphereips.org/datasets-ctu13)
- Data provider: *The Stratosphere Laboratory* - (https://www.stratosphereips.org/team)
- There are 13 scenarios in the entire dataset. Each scenario is different and comprises of different botnet captures.
- Each scenario had varying amount of Netflow data objects ranging from 107K records (5.2GB) in scenario-11 to 4.7M records (121GB) in scenario-3.
- The CTU-13 had total of 16 attributes in total present in all the scenarios. Few of them are - StartTime, Duration, Protocol, Source & Destination IP addresses, Source & Destination Port, Direction of flow, Total no. of packets, Total no. of bytes, Type of flow etc.

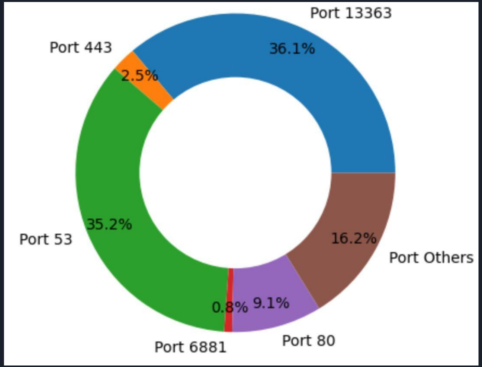
Data Visualization of CTU-13 Dataset:



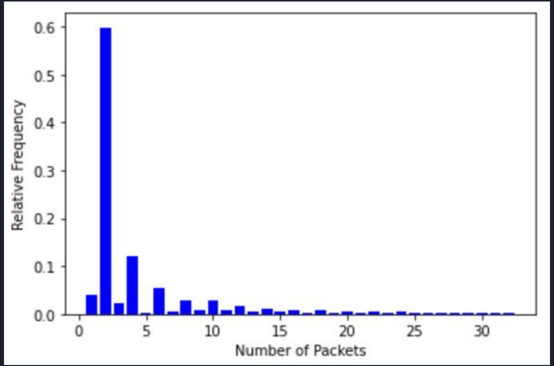
Duration of Communications
(Ranges from 0-3600 sec)



Distribution of Transport Protocols
(There are 15 protocols in total)



Distribution of Destination Ports
(There are a total of 73,786 ports)



Most frequent number of packets

Other Dataset Attributes:
StartTime: Initiation of packet transfer
SrcAddr: IP address of the source
Sport: Source port
Dir: Direction of the flow
DstAddr: IP address of the destination
State: Transaction state
sTos: Source TOS byte value
dTos: Destination TOS byte value
TotBytes: Total number of transaction Bytes
SrcBytes: Total number of transaction Bytes from the Source
Label: Label made of "flow=" followed by a short description of netflow type

Data Preprocessing:

- The dataset was readily available for our classification task.

Data transformation:

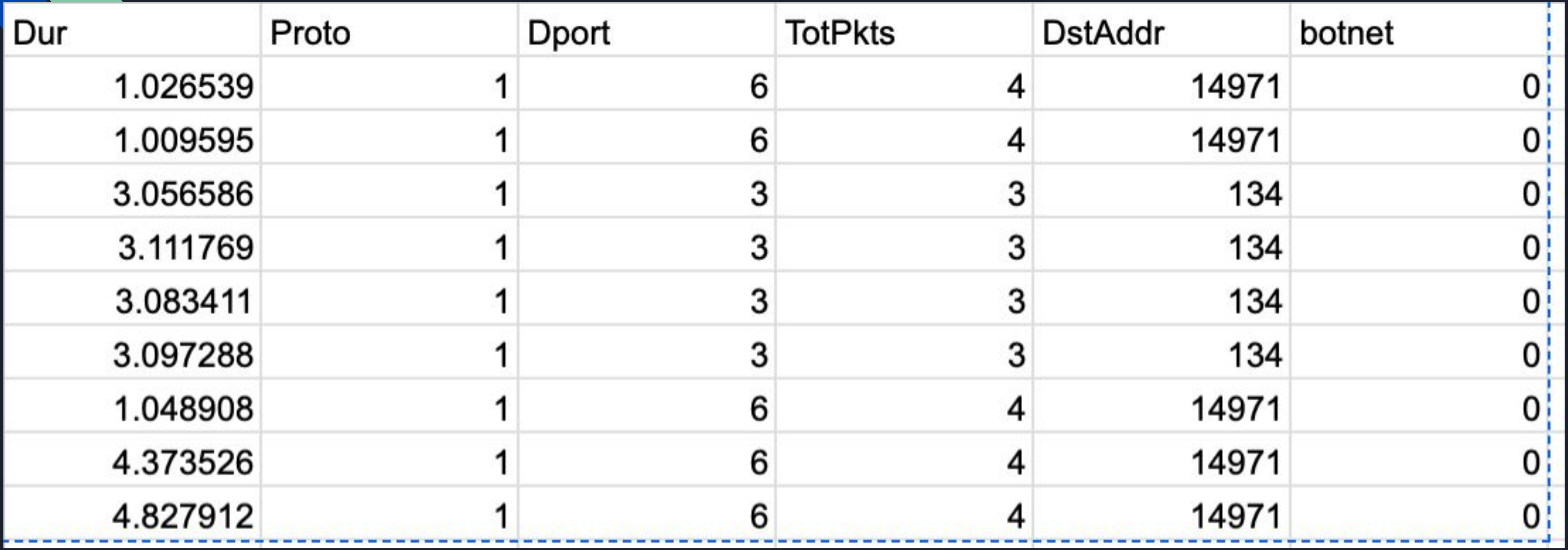
- Generation of label: Generated new class label - ‘botnet’
- ‘botnet’ had two class labels.
- ‘0’ (absence of botnet capture) & ‘1’ (presence of botnet capture).

Handle missing and unseen values:

- There were hexadecimal values in ‘Dport’ (eg: 0x8104, 0x20204ec4 in place of integer)
- There were no values or unseen values in‘DstAddr_octet’.
- The total no. of such records ~ 1%. So, we dropped them from our CTU-13 dataset.

Feature Extraction:

- Out of 16 total attributes in Netflow data, we chose mainly 5 attributes which are categorical and numeric attribute types.
- They are - Duration, Protocol, Destination Port, Total no. of packets, Destination IP address (count). Along with the class label ‘botnet’.
- The Destination IP address is in IPv4 format(32-bit) and we used frequency count.
- So, our “feature. csv” file had total of 6 fields including the ‘label’.
- Feature selection was based *point biserial*, *chi-squared* and Pearson *correlation* coefficient which are used to correlate a continuous variable vs a categorical variable, categorical vs categorical and continuous vs continuous variables respectively.



Dur	Proto	Dport	TotPkts	DstAddr	botnet
1.026539	1	6	4	14971	0
1.009595	1	6	4	14971	0
3.056586	1	3	3	134	0
3.111769	1	3	3	134	0
3.083411	1	3	3	134	0
3.097288	1	3	3	134	0
1.048908	1	6	4	14971	0
4.373526	1	6	4	14971	0
4.827912	1	6	4	14971	0

Sample of extracted features - *features.csv* file

Machine Learning Models:

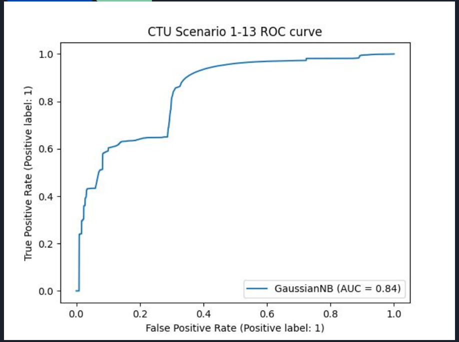
- We used three different supervised ML models for our classification task.
- *Logistic Regression* is a simple statistical Machine Learning model used for binary classification. The netflow data is represented in the form of a set of features after the feature extraction process. Then we score each netflow data based on the features using a *sigmoid function* also called *logistic function*.
- *Random Forest* is based on decision trees and the model builds many trees all of which act independent of each other but are used in the final consensus voting to determine the outcome.
- *Naive Bayesian* is based on Bayes' theorem and assumes each feature is independent of each other and uses a probability calculation based on statistics to score each attribute to determine the likelihood of an outcome for a given tuple.

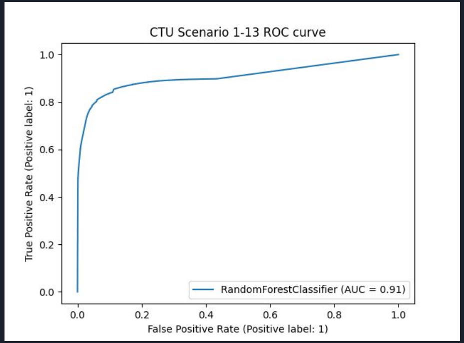
Classification Task:

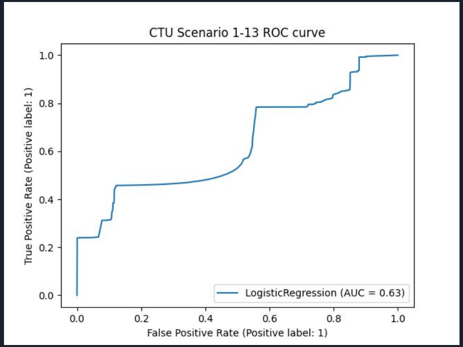
- Our goal was to predict the class labels - ‘0’ or ‘1’ (i.e, benign capture = 0 & botnet capture = 1).
- We have used multiple libraries to develop different ML models.
- In all these models, the first step was to split the dataset into Training and Test sets.
- Then we fit our model to Training set and train on it.
- Then we test the model on Test set to predict the class.
- We were also able to plot *ROC curves* and *Recall-Precision plots* on each scenarios.
- We then combined all the 13 scenarios in CTU-13 dataset into one and ran all the models on this aggregated data again.

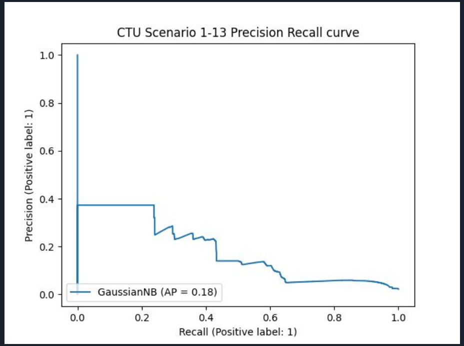
Results:

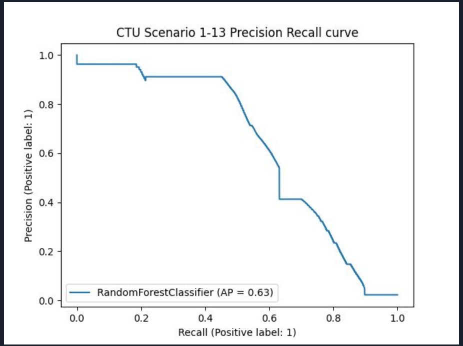
ROC curve & Recall-Precision plots:

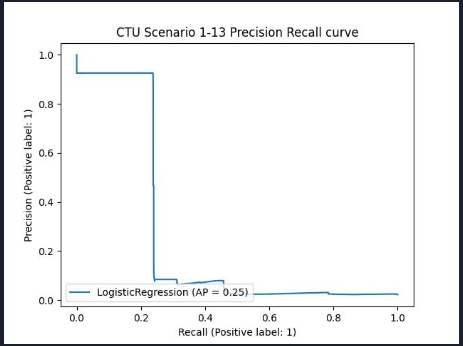












Gaussian Naive Bayes *Random Forest* *Logistic Regression*

Table 1: Comparing three models in terms of AUC scores

Table 2: Comparing three models in terms of AP scores

Area Under Curve (AUC) Score			
	Naive Bayes	Random Forest	Logistic Regression
CTU-1	0.83	0.94	0.72
CTU-2	0.89	0.94	0.94
CTU-3	1.00	1.00	0.96
CTU-4	0.96	0.96	0.91
CTU-5	0.86	0.92	0.87
CTU-6	0.99	0.99	0.97
CTU-7	0.74	0.70	0.67
CTU-8	0.93	0.97	0.91
CTU-9	0.75	0.93	0.56
CTU-10	1.00	1.00	1.00
CTU-11	1.00	1.00	1.00
CTU-12	0.74	0.75	0.70
CTU-13	0.87	0.94	0.81
1-13	0.79	0.84	0.50

Average Precision (AP) Score			
	Naive Bayes	Random Forest	Logistic Regression
CTU-1	0.58	0.87	0.57
CTU-2	0.06	0.81	0.14
CTU-3	0.97	1.00	0.91
CTU-4	0.12	0.48	0.05
CTU-5	0.04	0.73	0.04
CTU-6	0.49	0.87	0.50
CTU-7	0.00	0.24	0.00
CTU-8	0.04	0.44	0.14
CTU-9	0.27	0.69	0.28
CTU-10	0.99	1.00	0.99
CTU-11	0.99	1.00	1.00
CTU-12	0.02	0.80	0.02
CTU-13	0.17	0.83	0.50
1-13	0.18	0.63	0.25

Conclusion & Knowledge gained:

Conclusion:

- Better model - Random Forest
- Specific scenarios - Gaussian Naive Bayes, Random Forest and Logistic Regression
- Feature selection and extraction plays vital role on model performance.

Knowledge Gained and Applied:

- Using data mining tools such as Pandas, Scipy, Sklearn, Matplotlib
- Interpreting results such Precision, Recall, Sensitivity, Specificity, ROC, AP
- Feature extraction (e.g. correlations)

Future Work:

- Extract more features from Netflow data using different data transformation techniques and run the models.
- We can further extend our work to use *Random split*, *k-fold cross validation* and *Bootstrap aggregation* methods to improve the model performance in the real time.
- We can also expand our work to include more Machine Learning models. Then compare the results.

Tools Used:

- Mainly used python libraries to accomplish our tasks.

Python libraries:

- Sklearn: https://scikit-learn.org/stable/supervised_learning.html#supervised-learning
- Netflow: https://pypi.org/project/netflow/
- Pandas: https://pandas.pydata.org/
- Scikit-learn: https://scikit-learn.org/stable/
- Scapy - https://scapy.net/
- Data Visualization: https://matplotlib.org/

References:

- [1] Cyber Attack Detection thanks to Machine Learning Algorithms: https://arxiv.org/pdf/2001.06309.pdf
- [2] Machine Learning-Based Early Detection of IoT Botnets Using Network-Edge Traffic: https://arxiv.org/pdf/2010.11453.pd
- [3] D. Thomas. *Cybercrime Losses An Examination of U.S. Manufacturing and the Total Economy* NIST, 2020. https://doi.org/10.6028/NIST.AMS.100-3

Thank you!