Data Exfiltration Detection over C2 Channels and Alternative Protocol using Classification Machine Learning Algorithms

Lim Yang Jun, Chong Wei Bing Nicholas, Chow Wen Jun, Xavier Lim Gui Ming, Ang Wei Herng, Tan Zhi Yu  
*Infocomm Technology Cluster*  
*Singapore Institute of Technology*Singapore  
{2000572, 2001898, 2000530, 2000952, 2000958, 2001228}@sit.singaporetech.edu.sg

***Abstract*—Confidential data networks belonging to organisations are common targets of data exfiltration. Exfiltration of data by way of covert channels has become increasingly popular among attackers. This has become a growing concern among enterprises nowadays. Most early detection methods relied on rule matching, whose accuracy is dependent on the rule-maker. Hence, developing an exfiltration detection tool that exclusively detects data exfiltration with high accuracy is of great importance. In this paper, we present a machine-learning analytics tool capable of detecting data exfiltration over a command-and-control channel [T1041] and an alternative protocol [T1048]. The tool applies 4 machine learning techniques in detecting data exfiltration 1) K-Nearest Neighbour (KNN), 2) Multinomial Naive Bayes, 3) Logistic Regression and 4) Random Forest. The results of our tests show that the proposed tool has an overall detection accuracy of 100%.**

***Keywords— data exfiltration, exfiltration detection, machine learning, accuracy, command-and-control, alternative protocol, K-Nearest Neighbour, Naïve Bayes, Logistic Regression, Random Forest***

# Introduction

Exfiltrated data is a highly valuable resource for hackers. Sensitive information, such as personal customer data or business financial details, might, for example, be used directly to conduct fraud or sold to other criminals. It is essential that companies are well equipped with an adequate number of technologies to prevent and detect this form of attack.

This paper describes the 4 applied machine learning techniques, how each technique performs, and which technique has the better score among the rest.

This paper is divided into the following sections. Section 2 describes the classification algorithms and the model that the team applied. Section 3 describes the related works similar to the team’s project. Section 4 discusses the proposed approach that the team will be taking such as the dataset and model used. Section 5 describes the results of each machine-learning technique and which technique rates the best.

# Classification Algorithms and Models

Classification models are based on supervised learning algorithms to classify the data. All classification models act based on a dependent variable y which is the binary prediction of an event outcome. The variable y could either be a value of one which represents data exfiltration has happened or a value of zero to represent data exfiltration will not happen. The independent variables x1, x2, …. x8 are the attributes that affect the outcome of variable y, the probability that a certain event is deemed as an act of data exfiltration from the server.

## Logistic Regression

Logistic Regression is a data analysis method using math to determine the associations between two data components. The value of one of those parameters is then predicted depending on the other using this connection. The outcome of the forecast often has a limited range, such as yes or no. Since logistic regression models need minimal computer resources (memory and processing power), they can handle huge amounts of data quickly. The model is popular and has been used in cybersecurity applications such as detecting spam emails and predicting the probability of a cyber incident based on vulnerabilities [1].

## Multinomial Naïve Bayes

Multinomial Naive Bayes is a probabilistic machine learning algorithm used for performing classification. These models provide class labels to problem cases, which are represented as vectors of feature values, and the class labels are chosen from a limited set. In order to train such classifiers, there is not just one technique, but rather a family of algorithms built on the premise that, given the class variable, the value of one feature is independent of the value of every other feature.

It is possible to operate with the Multinomial Naive Bayes model without embracing Bayesian probability or applying any Bayesian techniques since parameter estimation for Multinomial Naive Bayes models frequently uses the maximum likelihood method. Multinomial Naive Bayes has the benefit of only needing a small amount of training data to estimate the classification-related parameters [2].

## Random Forest

The Random Forest algorithm is a simple and adaptable machine-learning technique. The Random Forest algorithm boasts a high prediction accuracy that works well with datasets that contain both categorical and continuous variables. It employs ensemble learning and solves the problem of overfitting datasets. It can perform both regression and classification tasks and can effectively handle large datasets.

## K-Nearest Neighbour (KNN)

KNN is a classification method where all computation is postponed until after the function has been evaluated. Since this method depends on distance for classification, if the features reflect several physical units or have wildly different scales, normalizing the training data can significantly increase its accuracy [3][4].

# Related works

There exist a few papers that propose models or tools to address the problem of accurately detecting data exfiltration attacks. Rajamenakshi Ramachandran et. Al [5] presents a model to detect data exfiltration using the combination of two statistical processing techniques, which are Kernel Density Estimation (KDE) and correlation coefficient. The model is capable of detecting data exfiltration on generic TCP/IP networks but cannot detect data exfiltration through channels such as HTTP and DNS. Gan Ruiling et. Al [6] presents a DNS-based data exfiltration detection model constructed using the decision tree algorithm. The model was able to achieve a high accuracy rate but is limited to only DNS data exfiltration detection. Kseniya Trusova presented a paper about comparing different classification models of Avionics System for Health Analysis and the approach to performing analysis and visualisations on a dataset to give an understanding of what is required in producing research results [6].

Compared to the above approaches, our proposed tool is unique in its ability to detect data exfiltration on various channels with a high accuracy rate.

# The proposed approach

The research and exploration of classification machine learning algorithms were implemented using Python3 programming language and Google Collaboratory notebooks with built-in Python3 libraries. This section describes the dataset used, the technologies involved, the system diagram, the machine learning techniques that the team had adopted as well as the result and insights.

## Dataset Used

1. Dataset Chosen

| **MITRE ATT&CK Techniques** | **Groups** | **Dataset file** |
| --- | --- | --- |
| Exfiltration over C2 [T1041] | 5 | T1041\_packetbeat\_clean\_Gp5\_SuEnSiobhan.csv |
| 12 | T1567 \_ Packetbeat \_ clean \_ Grp12\_TseKinPing.csv |
| 20 | T1573\_packetbeat\_clean\_Gp20\_HarishBalamurugan.csv |
| 21 | T1020\_VSFTPD\_Clean\_Gp21\_JamesEscabas.csv |
| 22 | T1041\_ELASTICSEARCH(PACKETBEAT)\_clean\_Gp22\_YongHan.csv |
| Exfiltrate Over Alternative Protocol [T1048] | 8 | T1020\_T1030\_T1048\_packetbeat\_clean\_Gp8\_KevinPookYuanKai.csv |
| 10 | T1048\_T1053\_ELKexport(Packetbeat)\_clean\_Gp10\_LowYongLin\_TanChuQingAlicia.csv |
| 11 | T1048\_http\_raw\_Gp11\_Quah Kian Yang.csv |
| 17 | T1048\_Filebeat\_clean\_Gp17\_SuanHong |
| 18 | * T1048\_ELK\_clean\_Gp18\_KhoirunIlmanBinKamarudin * T1048\_ELK\_clean\_Gp18\_P.UdaiyaChandran.csv |

Out of the 30 groups’ datasets provided, these groups' datasets (Table I) were selected for our model training as they contain data exfiltration logs that is at least in relation to T1041 and T1048 MITRE ATT&CK Techniques.

## Technologies Involved

In this section, the technologies that were used in this solution will be discussed in detail. The team developed the solution with the use of the python programming language. Table II below indicates and explains the various important python libraries that the solution relies on.

1. Python Libraries Used

| **Python Libraries** | **Usage** |
| --- | --- |
| Explainerdashboard | This python library is used in the project for developing interactive dashboards to visualise the results of the machine learning algorithms. |
| Glob | This python library is used in the project to specify a certain type of filenames with wildcard characters. |
| NumPy | This python library is used in the project for data filtering and manipulation of the CSV datasets. |
| OS | This python library is used in the project to determine the necessary file paths. |
| Pandas | This python library is used in the project for data manipulation and analysis of the CSV datasets. |
| Re | This python library is used in the project for searching for specific characters in the filenames. |
| Sklearn | This python library is used in the project for developing machine learning algorithms. |

## System Diagram

A comprehensive diagram that depicts the various components of the project is shown in Figure 1 below.

Diagram

Description automatically generated

1. System Diagram

Each of the components will be discussed in further detail in the following sections below.

## Dataset Cleaning

The datasets were carefully curated to only include useful column headers that will be selected as features for the machine learning algorithms. Labelling was performed for each row of logs to indicate whether it was data exfiltration. Full stops (.) in column headers were replaced by underscores (\_) to allow the Explainerdashboard to read the dataset.

Data cleaning was performed on the datasets as there were multiple datasets such as group 22’s dataset that had missing values or entire columns of empty data. The datasets were fixed by filling these columns with the best fit value.

Once all datasets have been processed, they are consolidated into a single dataset named DataExfil.csv which will be utilised to train and test the machine learning models. The dataset processing mentioned above is processed with Python functions within the Google Collaboratory notebook datacleaning.ipynb.

## Dataset Pre-Processing

The DataExfil.csv will undergo one more pre-processing stage before using it for the machine learning models. The figure below illustrates the pre-processing stage.

Diagram

Description automatically generated

1. Stages of Pre-Processing

The source IP address feature ‘src\_ip’ is split into 4 features, ‘src\_ip-1’, ‘src\_ip-2’, ‘src\_ip-3’, and ‘src\_ip-4’. Each of these 4 features represents the respective octet in the original ‘src\_ip’ IPv4 value [7]. This method of splitting the IP address and one hot encoding is applicable for the dataset that is being used since the dimensions of the dataset are not high.

Next, the feature ‘filename’ is split into 2 features, ‘fn’, and ‘ext’. The feature ‘fn’ will store the name of the file extracted from the feature ‘filename’, while ‘ext’ will store the file extension value extracted from the feature ‘filename’.

The feature ‘bytes\_in’ will be modified based on the algorithm below:

As some of the entries in the dataset contain 0 bytes\_in and are labelled as data exfiltrated, having a numerical value of 0 for data exfiltrated will hinder the interpretation of the feature ‘bytes\_in’ in relation to data exfiltration.

When data is exfiltrated, the bytes will never be 0. Hence, the algorithm is executed to ensure that all entries labelled as data exfiltrated whose bytes are 0, will get the mean bytes of all the other data exfiltrated entries with the same protocol [8]. The encoding of the data will be further explained in Section F. Before encoding the data, duplicated rows are dropped, leaving only the first occurrence. This is an essential step since a tree-based algorithm such as the Random Forest is being used. If there are duplicated rows, there is a chance that the same row is split into both testing and training sets and the accuracy of the model will be high which may hinder our analysis of the model [9].

## Classification Model Features

The models utilise eight features as a set of independent variables Xtrain. The models utilise a dependent variable Ytrain that is labelled as a binomial variable represented as a one or zero in the datasets.

1. Description of features

| **Features** | **Data Type** | **Description** |
| --- | --- | --- |
| host\_name | String (Categorical) | Any string value |
| type | String (Categorical) | Eg. HTTP, SSH, FTP, DNS |
| network\_transport | String (Categorical) | Eg. TCP, UDP, ICMP |
| src\_ip-1 | Integer (Categorical) | First Octet of source’s IPv4 |
| src\_ip-2 | Integer (Categorical) | Second Octet of source’s IPv4 |
| src\_ip-3 | Integer (Categorical) | Third Octet of source’s IPv4 |
| src\_ip-4 | Integer (Categorical) | Fourth Octet of source’s IPv4 |
| src\_mac | String (Categorical) | Source’s MAC Address |
| dest\_port | Integer (Categorical) | Destination port |
| bytes\_in | Integer (Continuous) | Bytes transmitted |
| fn | String (Categorical) | Filename |
| ext | String (Categorical) | File extension |
| data\_exfil | String (Categorical) | Dependent feature Y that is labelled if the data input indicates data exfiltration or not. |

OneHotEncoder was used to encode the categorical variables less the feature ‘data\_exfil’. The feature ‘data\_exfil’ was encoded via a LabelEncoder.

## Splitting of Dataset

The dataset DataExfil.csv was split into an 80:20 ratio. 80% of the dataset goes into the training set and 20% of the dataset goes into the testing set. The dataset was shuffled so that the data rows were randomly split at each execution. The classification models were tested with the testing set after they were built.

## Data Analysis

* K-Nearest Neighbour (KNN)

The KNN algorithm is a supervised learning algorithm that performs the classification of a data point by comparing the feature vectors of the various other data points. The KNN algorithm is a supervised learning algorithm that performs the classification data point by comparing the feature vectors of the various other data points.

The rationale for using this algorithm was the fact that thirteen features are present to detect data exfiltration which is considered manageable for the KNN algorithm. In addition, the KNN algorithm is adaptable whenever training data are added; the algorithm maintains all training data in memory and as new data points are added, the algorithm adapts and changes to account for it.

Subsequently, only two hyperparameters are required in the KNN algorithm compared to other ML algorithms. The hyperparameters are as shown below:

* K value: the number of neighbours the algorithm will check to derive the classification of a data point.
* Distance Metrics: The distance between a data point and the other data points which aid in determining how close a specific data point is to the others.
* Multinomial Naïve Bayes

The multinomial naïve bayes model is a naïve bayes classifier variant that performs multinomially discrete feature data classification. This algorithm presupposes a multinomial distribution for each of the attributes. Of the different variants of naïve bayes, the multinomial naïve bayes model was chosen as an initial baseline classification.

The features in the dataset are similar to the components of a network log. Such a log is usually structured in a textual and descriptive manner. Multinomial Naïve Bayes is often used in text classification where the features are in relation to the word counts or occurrence.

Almost all the features are categorical which is advantageous for the Multinomial Naïve Bayes model. As more of the same feature is labelled as malicious, for example, a source IP address; the model could recognize if an unknown IP address is potentially malicious or not. In the dataset, the filename is split into 2 different features, one to represent the name of the file and the other the extension of the file. If many entries in the dataset containing the value ‘zip’ in the feature ‘ext’ (extension) are labelled as malicious, then any new data where the value ‘zip’ is in the feature ‘ext’ will have a higher probability of being classified as malicious.

* Logistic Regression

The Logistic Regression algorithm is proposed in this research as to exemplify its underwhelming performance when faced with nonlinear problems. The algorithm focuses on understanding the correlation of the independent variables to the outcome variable. The algorithm accuracy would be best on the precondition that the dataset is perfect and rid of missing or enriched values. The algorithm is modelled using a logistic function in which multicollinearity is a problem to the performance if the data could not be linearly separated. The Logistic Regression model had the lowest performance when testing the model with the same dataset as the features. Examples of real-world use cases of Logistic Regression models are predicting heart failure to identify spam emails [1]. Therefore, the research included the Logistic Regression algorithm to explore the performance of the algorithm.

The Logistic Regression model features. The model was selected to showcase that if the independent variables had collinearity with each other, the model would Examples of real-world application of Logistic Regression is predicting heart failure to identify spam emails.

* Random Forest

In this model, hyper-parameterization was used to ensure that the parameters fit the training data that the team have. These few parameters were tested:

* n\_estimator: The number of trees the team wants to construct before calculating maximum voting or prediction averages.
* criterion: The function for determining the quality of a split.
* max\_depth: The number of splits permitted by each decision tree. If the number of splits is too little, the model underfits the data; if it is too large, the model overfits the data.
* max\_features: Assists in determining the number of features to consider making the best split.
* min\_samples\_split: The smallest number of samples required to split an internal node.
* min\_samples\_leaf: The bare minimum of samples necessary at a leaf node.

After determining the parameters to use, GridSearchCV is applied to find the best parameter values in a grid for the given parameters. In GridSearchCV, a few parameters were used.

* oob\_score: The number of rows from the out-of-bag sample that was correctly predicted.
* n\_jobs: The number of jobs that will run concurrently for both fit and predict.
* param\_grid: To supply the parameters that were generated during hyper-parameterization.
* cv: Cross-validation. Divide the training set further into K subsets called folds. The model is then iteratively trained K times, each time training on K-1 of the folds and assessing on the Kth fold.

Once the results of the best parameter values are given, the parameters will be supplied to the model and trained with the training dataset that was split above using the fit() function call.

# Analysis results

## Evaluation of machine learning algorithms

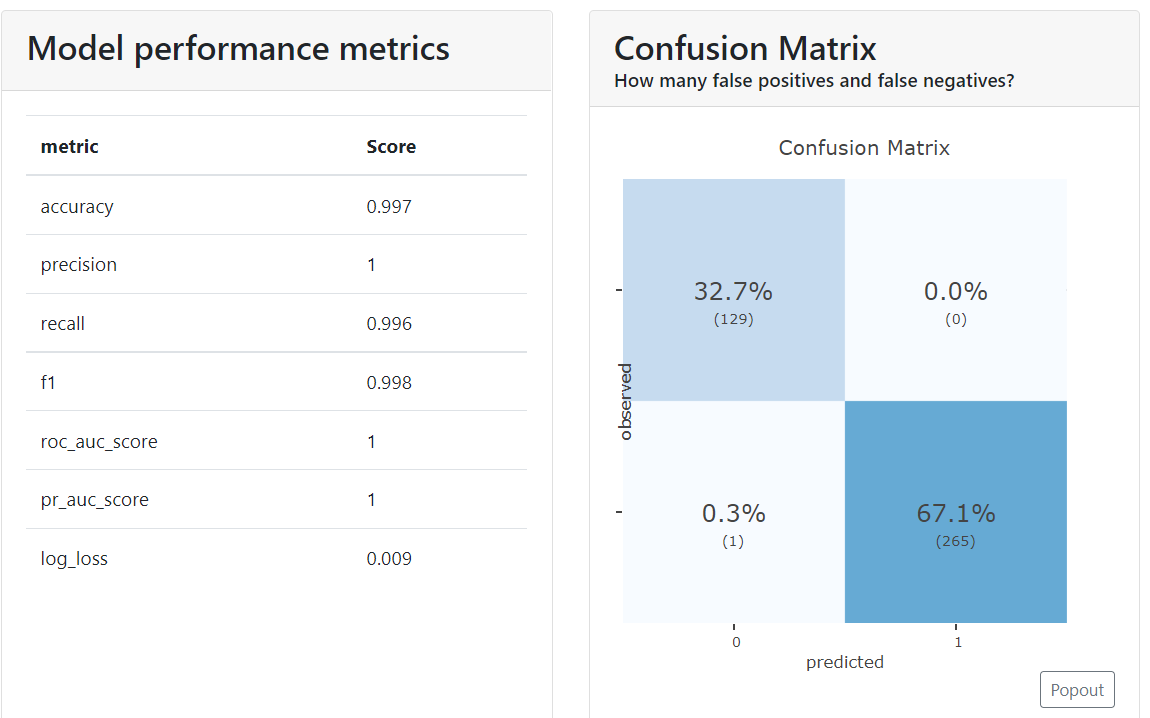
The machine learning classification models were evaluated through a few metrics. Firstly, the classification accuracy was measured across the different models with the same testing set. The F1 score combines two opposing variables — accuracy and recall — to calculate a model's prediction performance. Precision is defined as the proportion of correctly predicted positive observations to all predicted positive observations. The recall score is the ratio of correctly predicted positive observations to all observations in the actual class.

1. Accuracy of Classification

| **Model** | **Metrics** | | | |
| --- | --- | --- | --- | --- |
| ***Accuracy Score*** | ***F1*** | ***Precision Score*** | ***Recall Score*** |
| Multinomial Naïve Bayes | 89.1 | 86.3 | 92.5 | 83.5 |
| Random Forest Classification | 99.7 | 99.7 | 99.6 | 99.8 |
| Logistic Regression | 67.8 | 67.7 | 73.8 | 75.3 |
| K-Nearest Neighbor | 97.0 | 96.5 | 96.3 | 96.7 |

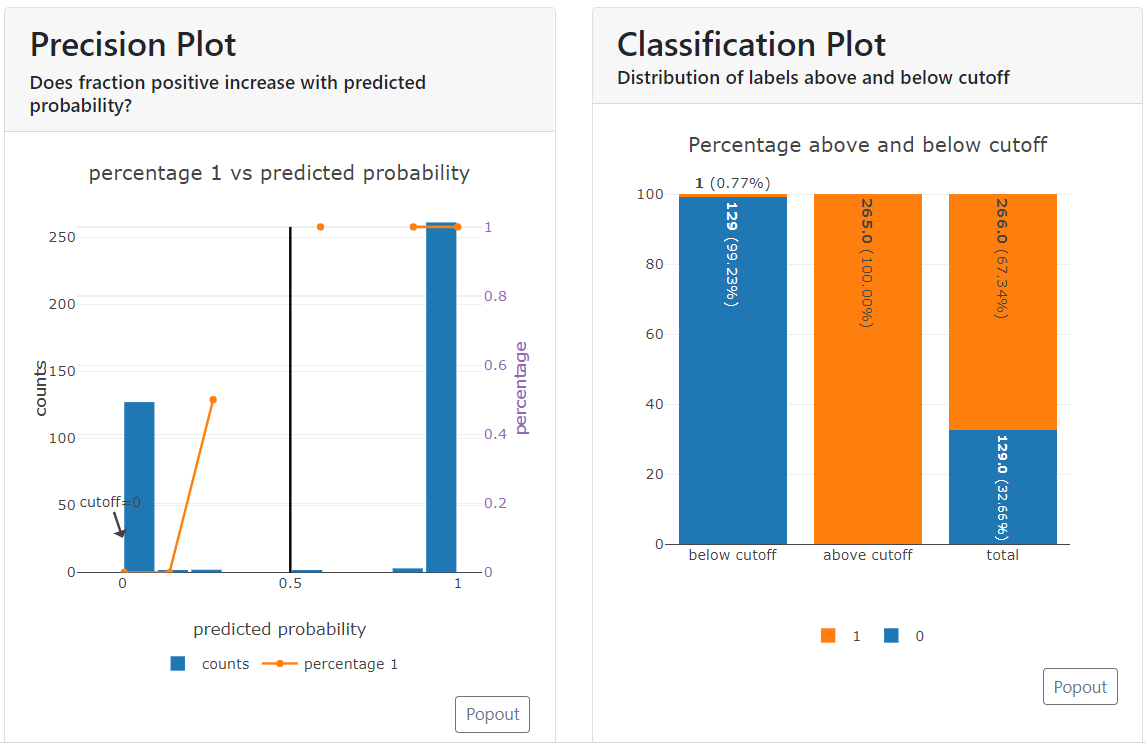
## Visualisation

Visualisation of the analysis results is performed using the Explainerdashboard library. The Python package allows us to deploy an interactive dashboard for each machine-learning model. The dashboard can be broken down into multiple components. The first component of the dashboard displays the model’s performance metrics and the confusion matrix as depicted in Figure 3 below.



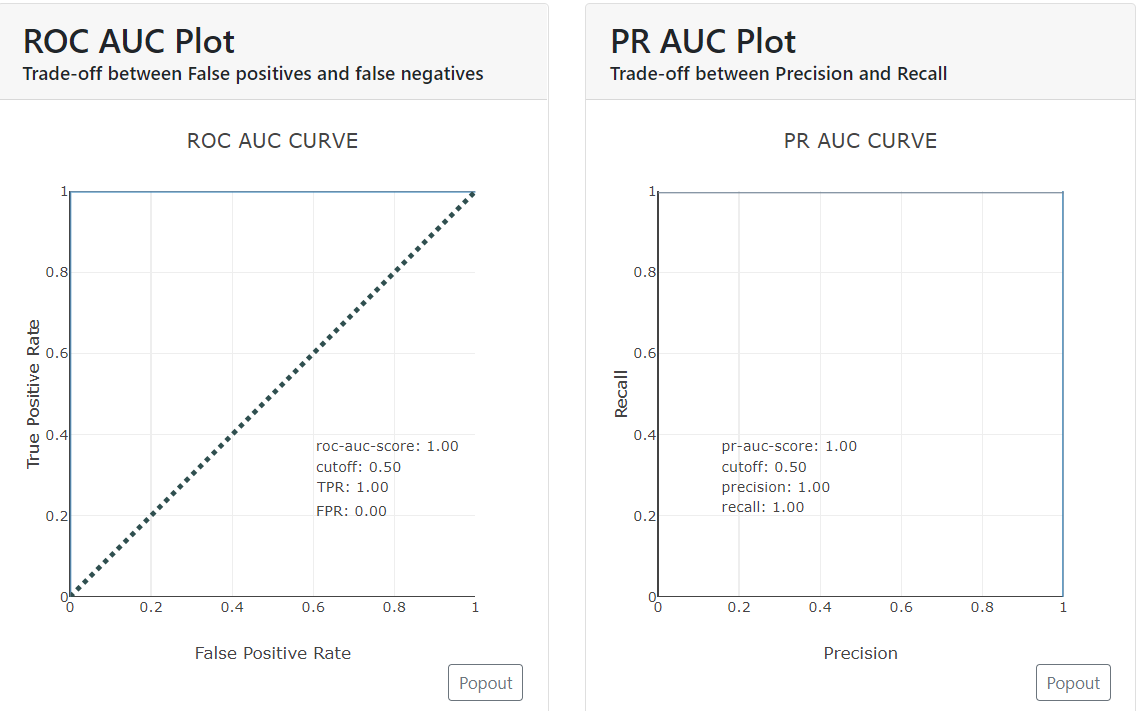
1. Performance Metrics and Confusion Matrix

The second component of the dashboard displays the precision and classification plots. The precision plot allows us to see the relation between the predicted probability that an index belongs to the positive class, and the percentage of the observed index in the positive class. The classification plot shows the percentage and number of each class above and below the cut-off point.



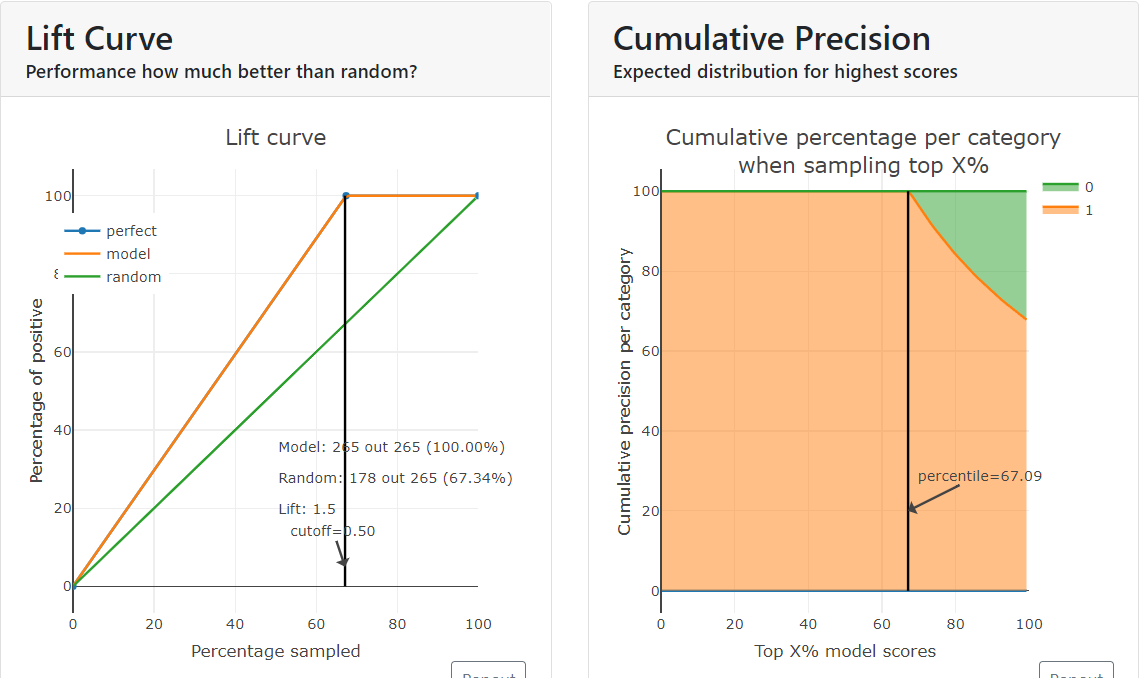
1. Precision and Classification Plots

The third component of the dashboard displays the ROC AUC and PR AUC Plots. The ROC AUC plot visualizes the trade-off between the true positive rate (TPR) and the false positive rate (FPR), whereas the ROC AUC plot is a curve that combines precision (PPV) and Recall (TPR) in a single visualization.



1. ROC AUC and PR AUC Plots

The final component of the dashboard displays the lift curve and cumulative precision plots. The lift curve visualises the relation between the number of instances which were predicted positive and those that were indeed positive. The cumulative precision plot shows the percentage of each label that one can expect when only sampling the top x% highest scores.



1. Lift Curve and Cumulative Precision Plots

*Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated*

*Text

Description automatically generated*

1. Programming Results

# Dataset Limitations

Although all 4 models achieved a high accuracy score, there are a few things to take note of regarding the dataset used.

## Size

The total number of rows in the 10 groups’ dataset only amounts to 9001, with most of them not being data exfiltration logs. While this might seem like a lot, real-world applications of machine learning involve training the model with millions of logs. Thus, such high accuracy produced should be taken with a grain of salt.

## Validity

Each group’s dataset is cleaned and categorised based on the MITRE ATT&CK Technique. However, accuracy of the models is difficult to evaluate for this project as it has not been deployed for real world applications.

## Sanity

While processing the dataset for the models, there were many instances of missing column headers or information in some logs. This was due to the presence of many different variables caused by the various groups’ logging methods. Columns had to be created with null values to concatenate all the datasets together such that the team can consistently train the models. Hence, this heavily affects the models’ training as there were not much reliable data for the model to use and differentiate from non-data exfiltration logs.

# Conclusion

The team have presented an analysis of the four classification algorithms for identifying data exfiltration patterns in cyber security logs. Based on the metric scores for each algorithm (refer to *Table IV*), the team have concluded that the Random Forest is the better algorithm for the dataset that the team was using. Hence, the Random Forest machine learning classification algorithm is the most befitting algorithm in detecting data exfiltration.

## Future Works

For future works, the team would attempt to gather more comprehensive exfiltration data from online reputable sources to enrich the currently implemented machine learning algorithms in the project. With a valid and informative dataset, deep learning techniques can be performed to achieve improved accuracy and prediction. Lastly, additional features such as capabilities in detecting real-time anomalies/attack detection can also be implemented.

## Task Allocation & Contributions

The table below shows the task allocation and contributions from each team member for the project.

1. Member’s Contribution

| **Team Member** | **Contributions** |
| --- | --- |
| Ang Wei Herng | Data Pre-processing, Random Forest Model |
| Chong Wei Bing Nicholas | Pre-processing Flow & Encoding |
| Chow Wen Jun | Data Pre-processing, K-Nearest Neighbors Model |
| Lim Yang Jun | Data Pre-processing, Multinomial Naïve Bayes Model |
| Tan Zhi Yu | Data Pre-processing, Random Forest Model |
| Xavier Lim Gui Ming | Data Pre-processing, Logistic Regression Model |

# Acknowledgment

The team would like to give special thanks to Associate Professor Dr Guo Huaqun for her guidance which helped in shaping our research and the outcome of this paper.

##### References

1. L. Enhao, “Logistic Regression Model for Predicting Warning “Incident” Rates and Implications for the Common Vulnerability Scoring System,” *2017. Available at*  https://etd.ohiolink.edu/apexprod/rws\_etd/send\_file/send?accession=osu1494008131762435&disposition=inline.
2. M.C.Andrew, “Graphical Models Lecture 2: Bayesian Network Representation,” 2011, Available: https://people.cs.umass.edu/~mccallum/courses/gm2011/02-bn-rep.pdf [Accessed Nov. 15, 2022]
3. S.Madeh Piryonesi, Tamer E. El-Diraby, “Role of Data Analytics in Infrastructure Asset Management: Overcoming Data Size and Quality Problems,” 2020, Available: https://ascelibrary.org/doi/10.1061/JPEODX.0000175 [Accessed Nov. 15, 2022]
4. H.Trevor, T.Robert, J.H.Friedman, “The elements of statistical learning: data mining, inference, and prediction,” 2001, Available: https://www.worldcat.org/title/46809224 [Accessed Nov. 13, 2022]
5. G. Ruiling, D. Jiawen, C. Xiang and S. Shouyou, "A DNS-based Data Exfiltration Traffic Detection Method for Unknown Samples," *2022 7th IEEE International Conference on Data Science in Cyberspace (DSC)*, 2022, pp. 191-198, doi: 10.1109/DSC55868.2022.00032.
6. K.Trusova, “Comparison of Two Classification Machine Learning Models of Avionics Systems for Health Analysis” 2020, Available: https://ieeexplore.ieee.org.singaporetech.remotexs.co/stamp/stamp.jsp?tp=&arnumber=9243971 [Accessed Nov.15, 2022]
7. Shao, Enchun (2019): Encoding IP Address as a Feature for Network Intrusion Detection. *Purdue University Graduate School. Thesis. https://doi.org/10.25394/PGS.11307287.v1*
8. G. Chhabra, V. Vashisht and J. Ranjan, "A Classifier Ensemble Machine Learning Approach to Improve Efficiency for Missing Value Imputation," *2018 International Conference on Computing, Power and Communication Technologies (GUCON), 2018, pp. 23-27, doi: 10.1109/GUCON.2018.8674904.*
9. J. Mishra, S. K. Sahay, H. Rathore and L. Kumar, "Duplicates in the Drebin Dataset and Reduction in the Accuracy of the Malware Detection Models," *2021 26th IEEE Asia-Pacific Conference on Communications (APCC), 2021, pp. 161-165, doi: 10.1109/APCC49754.2021.9609892.*
10. R. Ramachandran, S. Neelakantan and A. S. Bidyarthy, "Behavior model for detecting data exfiltration in network environment," *2011 IEEE 5th International Conference on Internet Multimedia Systems Architecture and Application*, 2011, pp. 1-5, doi: 10.1109/IMSAA.2011.6156340.