|  |
| --- |
| ASP.NET Vulnerability Assessment  Part 2: Machine Learning  Mikolaj M. Mroz  2003114  BSc (Hons) Ethical Hacking, 2023  CMP417: Engineering Resilient Systems 1 |

|  |
| --- |
| School of Design and Informatics  Abertay University |

Contents

[Context 2](#_Toc133346461)

[Critical Comparison 2](#_Toc133346462)

[Support Vector Machines 2](#_Toc133346463)

[Decision Trees 3](#_Toc133346464)

[Implementation 5](#_Toc133346465)

[Data Collection 5](#_Toc133346466)

[Pre-processing 6](#_Toc133346467)

[Removal of redundant data features 6](#_Toc133346468)

[Nominal conversion 6](#_Toc133346469)

[Synthetic dataset cleaning 6](#_Toc133346470)

[Normalisation 6](#_Toc133346471)

[Modelling 6](#_Toc133346472)

[Results Analysis and Evaluation 7](#_Toc133346473)

[Accuracy - Confusion Matrix and Heatmap 7](#_Toc133346474)

[Precision and Recall – Macro-averaged F1 8](#_Toc133346475)

[Degradation - Data Drift 9](#_Toc133346476)

[Results Communication 9](#_Toc133346477)

[Simple 9](#_Toc133346478)

[Complex 10](#_Toc133346479)

[Conclusion 10](#_Toc133346480)

[Bibliography 10](#_Toc133346481)

[Appendices 12](#_Toc133346482)

[Appendix 1 – Tables 12](#_Toc133346483)

[Appendix 1A – A table of redundant dataset features present 12](#_Toc133346484)

[Appendix 1B – A visualisation of a packet classifier confusion matrix 12](#_Toc133346485)

[Appendix 1C– A visualisation of a packet classifier heatmap 13](#_Toc133346486)

[Appendix 2 - Equations 13](#_Toc133346487)

[Appendix 2A – Precision equation example. 13](#_Toc133346488)

[Appendix 2B – Recall equation example 13](#_Toc133346489)

# Context

In this part of the investigation, recommended Machine Learning (ML) implementations are discussed in relation to network packet analysis and more specifically, the design of a machine learning based classifier for ScottishGlen. The aim of the classifier is to categorise the various possible attacks that may be detected on the company network based on the network packet capture provided as a sample. Two appropriate algorithms are to be chosen and compared to be best fit for the job. The specific pipeline development stages are thoroughly discussed, including a detailed description of the evaluation metrics along with reasoning.

## Critical Comparison

Based on the results gathered from these investigations, two algorithms can be compared to tackle the task of ML-based network packet analysis: Decision Trees (DT) and Support Vector Machines (SVM). On a fundamental level, DT and SVM use very different processes to achieve their results.

### Support Vector Machines

SVMs make use of non-parametric linear supervised learning methods meaning they learn wholly from the data they are given. SVMs make unique use of a ‘kernel trick’, allowing non-linearly separable results to be clearly divided into anomalous and non-anomalous by projecting them onto a higher n-dimensional plane, where an appropriate margin can be set as shown in Figure 1.

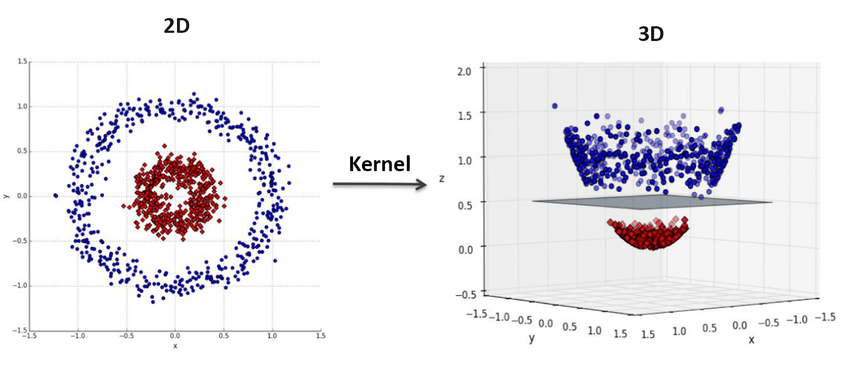


Figure – A visualisation of indivisible results being divided in a higher dimensional plane using SVM (Hachimi, et al., 2020)

The strengths of SVM in network packet classification lie in its ability to effectively handle continuous data streams, separate linear and non-linear data by making use of the kernel trick and make very efficient use of memory. SVM is specialised for use in situations where a clear margin of separation needs to be set, such as the classification of network packets, and can do so to a great degree of accuracy as outliers are less likely to skew results than in comparable models. The benefits of SVM enable it to be a favoured model for implementations of intrusion detection systems (IDS) (Mohammed & Sulaiman, 2012 & Jha & Ragha, 2013).

However, it is let down by its struggles with noisy datasets, which are a common characteristic of network data streams that only get worse with network complexity and time. The training time for the model is also greatly dependant on the volume of data within a dataset (Kawulok & Nalepa, 2014), which furthers the issue of increased complexity and preparation time for use within bigger and more intricate networks. SVMs are a model ultimately best suited for smaller sets of data.

### Decision Trees

DTs, like SVMs, are also classed as a non-parametric supervised learning models but take a different approach during processing. DTs are a prediction model that use given target data features to produce ‘branches’ of the actions that can be taken to reach an appropriate result (or ‘leaf’) [Figure 2] Its efficiency improves with larger datasets as more results allow the model to determine anomalous results more accurately from non-anomalous.

Diagram

Description automatically generated

Figure - An example of a decision tree being used to predict a player's golf score based on environmental conditions (Master's in Data Science, 2023).

The most noteworthy benefits of a DT model include its ease of visualisation, which can be utilised to make use of DT’s general flexibility. DT algorithms are also much easier to set up than SVM requiring less preparation and pre-processing. Its flexibility extends to its ability to be incorporated into other decision-making techniques and can be expanded into a Random Forest (RF), or group of trees, to improve accuracy when determining if a network packet is deemed malicious or not using a majority vote system. This allows several trees to classify a given packet according to their own unique rules, after which the most popular classification among those trees (anomalous or non-anomalous) can be determined.

However, research suggests the DT model often performs worse in terms of accuracy compared to other options including SVM and is more costly in terms of performance and memory usage. This particularly applies to more complex trees with many chance variables as the paths of the trees grow, and as the model grows, it is prone to interpreting unnecessary data leading to the issue of overfitting. Trees may require maintenance in the form of pruning, optimizing the number of possible branches to prevent overfitting. They may also be included as part of an RF as an alternate solution but are complicated to set up and require much more computational power and memory to function effectively.

# Implementation

Based on the critical comparison of the two potential algorithms, DT and SMV, research suggests that a DT RF algorithm is best suited for this classification scenario – RFs are typically more accurate than a standalone DT model due to its majority vote process which helps determine the best course of action for a given network packet. This ultimately reduces the potential for undesirable false positive and false negative results and alleviates the risk of overfitting the model.

Diagram

Description automatically generated

Figure - A diagram of an RF model. (DotNetTutorials, 2023)

## Data Collection

Before any training can begin, the relevant network packet data must first be collected. The full UNSW-NB15 IDS dataset is recommended here as it was developed as a solution to the lack of thorough network traffic datasets relevant to modern network structures. Some of the most prominent sets such as the KDD98 quickly grew outdated in recent years, failing to represent modern attack vectors and in-depth network packet information gathering. The UNSW-NB15 dataset accounts for these issues, providing a “hybrid of the real modern normal and the contemporary synthesized attack activities of the network traffic” to train the modern network packet classifier in an effective manner. It also has many more packets than the given training dataset allowing for more precise learning, though must be considered carefully due to the risk of synthetic datasets overfitting the model.

## Pre-processing

Pre-processing is an act of optimizing the dataset or otherwise transforming it to better fit the developed model. Based on research by Zoghi & Serpen (2021), several improvements can be recognised to improve the potential of the dataset.

### Removal of redundant data features

Several features within the dataset are irrelevant to attack detection and should be removed. By removing irrelevant data, the time spend training the model can be decreased. See Appendix 1A for a list of identified redundant features.

### Nominal conversion

To improve the compatibility of the dataset, the remaining features should be converted to numerical from their original nominal (or descriptive) values using nominal-to-numerical label encoding. This does not impact the model’s accuracy but allows for easier restructuring if another algorithm is chosen in the future.

### Synthetic dataset cleaning

The UNSW-NB15 dataset is synthetic, therefore requires minimal pre-processing; It has already been altered by its authors to best fit a network classifier. Should the client wish to use a real dataset, such as a true sample of network traffic, observations such as outliers, duplicates, structural errors, and missing data will need to be removed.

### Normalisation

As the RF model is tree-based, aspects of the data such as feature scale do not need to be considered as they have virtually no effect on the decision-making process, and therefore data does not require normalisation.

## Modelling

Visualising (modelling) an RF is relatively straightforward. Because an RF can be simplified as democratic vote between multiple trees, a decision plot can be constructed for each individual tree and culminated into a forest thereafter [Figure 3]. However, these can grow quite complex and therefore computationally expensive, in which case a Gini index can be used to simply display the information gain of each feature. Alternatively, a feature importance graph can be calculated to give an idea of which features have the most impact on results.

## Results Analysis and Evaluation

Analysis of the model’s results should take the following metrics into account: Accuracy, Precision, Recall, Feature Importance and Data Drift.

### Accuracy - Confusion Matrix and Heatmap

The confusion matrix is named so as it allows the model’s incorrect guesses to be visualised. It is best used for binary multi-class models such as the RF network packet classifier and is particularly useful for tweaking it during the analysis phase of the pipeline especially. When paired with a normalized heatmap, it can display the ratios of each incorrect category prediction.

Graphical user interface

Description automatically generated

Figure - An RF Confusion Matrix heatmap using different tree species as an example. Results have been normalized for easier visualisation. (Kreiger, 2020)

In the case of this specific classifier model, the ‘true label’ is the true attack category (‘attack\_cat’ in the testing and training datasets), and the predicted label is the potential predicted attack category. See Appendix 1B for an example confusion matrix and Appendix 1C for its heatmap.

This allows us to accurately measure classifier accuracy using the following equation.

A is accuracy, TP and TN are the true positive and true negative predictions and FP and FN are false positives and false negatives. However, accuracy on its own is not sufficient to gauge the performance of a classifier.

### Precision and Recall – Macro-averaged F1

Precision can be described as the of correct positive identifications. As an example, this may be simplified as a measure of how many times the model correctly classed a packet as DoS against all the packets that actually were DoS within a given dataset. See the equation below.

Recall is another useful metric and can be described as a measure of positives identified correctly.

Ideally this rate should be as close to 1 as possible to show absolute precision but does not give much informational value unless compared to another.

The F1 score combines the benefits of calculating precision and recall into a harmonic mean equation, meaning it more harshly punishes results where precision and recall deviate from each other. Its macro-averaged format is only useful for datasets with the same number of datapoints within each possible class, as it calculates the average of class specific F1 scores. This applies to the given dataset and is therefore the most relevant format of the F1 score for network packet analysis.

### Degradation - Data Drift

Degradation, known within ML as Data Drift, is a metric of how outdated a model becomes with time. This is particularly important in the field of network packet analysis, as new attack vectors, techniques, and classifications are discovered regularly leading to models and datasets become outdated.

Data drift calculations can be implemented into the ML model lifecycle as Data Drift Monitoring and can be calculated using the Kolmogorov-Smirnov (K-S) test. K-S testing is recommended as its requirements are already included within a typical ML model lifecycle: the datasets of the training data, which was already obtained, and post-training data.

The K-S test compares the data distribution between these two sets of data, with the ideal outcome being that the distribution within the datasets is the same. If this is not the case, then a drift is present within the data, and the dataset will need to be recalibrated or recaptured under new, up-to-date conditions.

## Results Communication

The level at which results should be communicated depends entirely on the individual or group on the receiving end of the analysis. Therefore, it is necessary to evaluate the audience and their potential level of knowledge before piecing together visualisations of results.

### Simple

For stakeholders and managers, it is generally best practice to keep results and language used as simple as possible. The technical knowledge can vary between individuals though they will most likely be looking for similar results from the presentations: business impact, a list of important features, an initial estimate of accuracy or precision, visualisations of the output, and future work. In this case, it is recommended to use simple visualisations such as feature importance graphs and confusion matrices. In the case of the confusion matrix, a simple positive against negative results visualisation can effectively get the point across without naming each possible classifier outcome.

### Complex

Coders, data engineers, and forensic investigators require details on the inner workings of the model, and the visualisations displayed should reflect this. Continuing to use the confusion matrix as an example, a large multi-class matrix should be utilized to give the most detailed picture of accuracy. A macro-averaged F1 metric can be incorporated into the confusion matrix itself to display the chance of each attack category being guessed correctly.

Both the simple and complex presentations need to discuss data drift; managers, stakeholders, and developers will all need to know if the model requires a recalibration. This is because recalibrating the model requires time and money and may need to be considered or even prioritised in the development workflow.

## Conclusion

In conclusion, an RF algorithm was chosen as the best fit for a network packet classifier due to its popular use in intrusion detection systems and documentation within the field. The implementation pipeline follows a largely conventional structure of pre-processing the dataset, modelling, evaluating results, and communicating them appropriately to an audience. The main evaluation metrics selected were accuracy, precision, recall, and degradation as they are relevant to RF’s confidence in correctly classifying a network packet and its longevity, needing recalibration in the event of new network packet technologies.

# Bibliography

Alqudah, N. & Yaseen, Q., 2020. Machine Learning for Traffic Analysis: A Review. *Procedia Computer Science,* Volume 170, pp. 911-916.

DotNetTutorials, 2023. *Random Forests in Machine Learning.* [Online]   
Available at: https://dotnettutorials.net/lesson/random-forests-inmachine-learning/  
[Accessed 24 April 2023].

Hachimi, M., Kaddoum, G., Gagnon, G. & Illy, P., 2020. *Multi-stage Jamming Attacks Detection using Deep Learning Combined with Kernelized Support Vector Machine in 5G Cloud Radio Access Networks.* Montreal, Canada, IEEE.

Jha, J. & Ragha, L., 2013. Intrusion Detection System using Support Vector Machine. *IJAIS Proceedings on International Conference and workshop on Advanced Computing 2013,* Volume 3, pp. 25-30.

Kawulok, M. & Nalepa, J., 2014. *Dynamically Adaptive Genetic Algorithm to Select Training Data for SVMs.* Santiago de Chile, Springer Cham.

Kreiger, J., 2020. *Evaluating a Random Forest model.* [Online]   
Available at: https://medium.com/analytics-vidhya/evaluating-a-random-forest-model-9d165595ad56  
[Accessed 25 April 2023].

Master's in Data Science, 2023. *Decision Tree.* [Online]   
Available at: https://www.mastersindatascience.org/learning/machine-learning-algorithms/decision-tree/  
[Accessed 22 April 2023].

Mohammed, M. N. & Sulaiman, N., 2012. Intrusion Detection System Based on SVM for WLAN. *Procedia Technology,* 1(1), pp. 313-317.

Zoghi, Z. & Serpen, G., 2021. *UNSW-NB15 Computer Security Dataset: Analysis through Visualisation,* Ithaca: ArXiv.

# Appendices

## Appendix 1 – Tables

### Appendix 1A – A table of redundant dataset features present

|  |  |  |
| --- | --- | --- |
| Feature | Description | Reason |
| record\_start\_time | The time the packets were initially sent. | There already exists a feature (dur) which displays the difference between the start and time. |
| record\_last\_time | The time the packets were received. |
| Source IP address | The IP address the packets originated from | These features are not relevant to attack classification as they vary between individual attackers. |
| Destination IP address | The IP address the packets were sent to |
| Source port number | The port number the packets originated from |
| Destination port number | The port number the packets were sent to |

### Appendix 1B – A visualisation of a packet classifier confusion matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Normal** | xxx | 890 | xxx | xxx |
| **DoS** | xxx | 50000 | xxx | xxx |
| **Recon** | xxx | 31 | xxx | xxx |
| **Backdoor** | xxx | 7412 | xxx | xxx |
|  | **Normal** | **DoS** | **Recon** | **Backdoor** |

### Appendix 1C– A visualisation of a packet classifier heatmap

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Normal** |  | 0.01525 |  |  |
| **DoS** |  | 0.85714 |  |  |
| **Recon** |  | 0.00531 |  |  |
| **Backdoor** |  | 0.12706 |  |  |
|  | **Normal** | **DoS** | **Recon** | **Backdoor** |

## Appendix 2 - Equations

### Appendix 2A – Precision equation example.

If and

Then our precision for the model can be equated to

Showing that the chances of the model correctly identifying a Denial-of-Service packet are 90.2%.

### Appendix 2B – Recall equation example

If and Then the recall score can be identified using

Giving a recall score of 0.862.