

An Innovative Approach to Insider Threat Detection Using Data Mining and Machine Learning

Martyn Brown - 40135758

Submitted in partial fulfilment of
the requirements of Edinburgh Napier University
for the Degree of
BEng (Hons) Computer Security and Forensics

In collaboration with ZoneFox

School of Computing

April 2017

Authorship Declaration

I, Martyn Brown, confirm that this dissertation and the work presented in it are my own achievement.

Where I have consulted the published work of others this is always clearly attributed;

Where I have quoted from the work of others the source is always given. With the exception of such quotations this dissertation is entirely my own work;

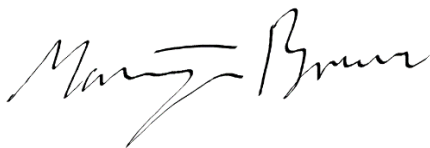
I have acknowledged all main sources of help;

If my research follows on from previous work or is part of a larger collaborative research project, I have made clear exactly what was done by others and what I have contributed myself;

I have read and understand the penalties associated with Academic Misconduct.

I also confirm that I have obtained **informed consent** from all people I have involved in the work in this dissertation following the School's ethical guidelines

Signed:

A handwritten signature in black ink, appearing to read 'Martyn Brown', with a stylized, cursive script.

Date:

Matriculation no: 40135758

Data Protection Declaration

Under the 1998 Data Protection Act, The University cannot disclose your grade to an unauthorised person. However, other students benefit from studying dissertations that have their grades attached.

Please sign your name below *one* of the options below to state your preference.

The University may make this dissertation, with indicative grade, available to others.

The University may make this dissertation available to others, but the grade may not be disclosed.

A handwritten signature in black ink, reading 'Martyn Brown'. The signature is written in a cursive style with a large, stylized 'M' and 'B'.

The University may not make this dissertation available to others.

Abstract

Insider threats accounted for a fifth of all data breaches in 2015, (Verizon, 2016).

With the introduction of stricter data controls such as General Data Protection Regulation, (<http://www.eugdpr.org>, 2016) and many high-profile companies falling victim to data breaches. This project set out to produce an innovative approach to insider threat detection through the use of data mining and machine learning tools.

The project achieved its aim by using a number of literature material to review and develop a novel approach to insider threat detection using machine learning and data mining.

A semi-supervised model is presented consisting of an unsupervised hybrid approach using a selection of anomaly detection algorithms within Rapidminer and a supervised approach using pattern recognition with MATLAB. ZoneFox (ZoneFox, n.d.), a market leader in user behaviour analytic and insider threat protection provided data collected from their system that imitated live user activity. This allowed for the testing of six realistic scenarios including: data theft, accessing sensitive files, security software tampering, unauthorised software installation, user behaviour and use of external contractors.

Evaluation of the presented model shows that all known threats were detectable from only 20% of the processed data after the entire dataset was analysed. A hybrid model that is explored produced highly accurate results when unrestricted and shows that using several algorithms is more accurate than a single algorithm in all test result. The supervised approach performed poorly in the project and suggestions are made for potential future work to improve and build upon the findings of this project. Due to limitations in the size of dataset available for testing, it is suggested that further testing is carried out to ensure accuracy is maintained. The implications of this approach could have a massive positive impact in industry when protecting against insider threats both pro-actively and re-actively.

Table of Contents

1	INTRODUCTION	1
1.1	Context.....	1
1.2	Background	1
1.3	Aims and Objectives	2
1.4	Dissertation Organization	3
2	TERMINOLOGY	5
2.1	Introduction	5
2.2	Data Mining Vs Machine Learning.....	5
2.3	Processing.....	6
2.3.1	STREAM Clustering.....	6
2.3.2	Normalisation.....	6
2.4	Algorithms.....	7
2.4.1	Supervised.....	7
2.4.2	Unsupervised.....	10
2.4.3	Semi Supervised.....	13
2.5	Tools.....	13
2.5.1	Data Mining	14
2.5.2	Machine Learning	14
2.6	Conclusions	15
3	LITERATURE REVIEW.....	17
3.1	Real World	17
3.2	Current Works.....	18
3.2.1	Insider Threat	18
3.2.2	Algorithms and Tools.....	20
3.3	Conclusion	26
4	SCENARIOS & METHODOLOGY	27
4.1	Introduction	27
4.2	Original data	27
4.3	Scenarios.....	29
4.3.1	Data Theft	29
4.3.2	Accessing Sensitive Files.....	29
4.3.3	Security Software	30

4.3.4	Unauthorised Software Installation	30
4.3.5	Contractors	30
4.3.6	User Behaviour	31
4.4	Proposed Methodology	31
4.4.1	Unsupervised Hybrid Model	32
4.4.2	Supervised Model	32
4.4.3	Semi-Supervised Hybrid System	33
5	DATA PRE-PROCESSING	34
5.1	Feature Selection.....	34
5.2	Outlier Identification.....	35
5.3	Data Conversion	36
5.3.1	Date and Time Conversion.....	36
5.3.2	Unique Event Identification and Key Selection	37
5.3.3	Event Conversion	38
5.4	Normalisation.....	38
6	SYSTEM IMPLEMENTATION.....	40
6.1	Rapidminer	40
6.1.1	Tool Implementation.....	40
6.1.2	Algorithm Selection and Implementation	41
6.1.3	Dataset Implementation	45
6.1.4	Results Collection	46
6.2	MATLAB.....	48
6.2.1	Tool Implementation.....	48
6.2.2	Algorithm Selection and Implementation	48
6.2.3	Data Implementation.....	49
6.2.4	Result Collection.....	50
7	EVALUATION	52
7.1	Unsupervised Results.....	52
7.2	Supervised Results.....	54
7.3	Comparison	55
8	CONCLUSIONS AND FUTURE WORK	57
8.1	Conclusions	57
8.2	Future Work	58
8.3	Personal Evaluation.....	58
	REFERENCES	59
	APPENDICES	63

List of Tables

Table 1 Original Data Scenarios	27
Table 2 Modified Fields.....	28
Table 3 Data Theft Scenario	29
Table 4 Accessing Sensitive Files Scenario	29
Table 5 Security Software Scenario.....	30
Table 6 Unauthorised Software Installation Scenario	30
Table 7 Contractors Scenario	31
Table 8 User Behaviour Scenario	31
Table 9 Replacement CSV Files.....	35
Table 10 Date and Time Conversion	37
Table 11 Event UID Ranges	37
Table 12 Assigned Event UID.....	37
Table 13 Data Conversion Process	38
Table 14 Data Normalisation Process	39
Table 15 Unsupervised Algorithms	41
Table 16 Performance Times	54

List of Figures

Figure 1 Insider Threat Figures	1
Figure 2 Insider Type and Motivations.....	2
Figure 3 Standard VS Normalised Data.....	6
Figure 4 KNN example	8
Figure 5 Linear Regression Example.....	9
Figure 6 Decision Tree Example	9
Figure 7 Autoencoder Example	11
Figure 8 K-means, 3 centroids Example (Pandre, n.d.)	12
Figure 9 DBSCAN VS K-means Clustering Example, (naftaliharris, 2015).....	13
Figure 10 AI2 Semi-Supervised Example	25
Figure 11 Proposed Unsupervised Approach	32
Figure 12 Proposed Supervised Approach.....	32
Figure 13 Proposed Semi-Supervised System	33
Figure 14 Operating System.....	40
Figure 15 Detect (LOF) Algorithm Configuration	42
Figure 16 Local (LOF) Algorithm Configurations	42
Figure 17 aLOCI Algorithm Configuration.....	43
Figure 18 Clustering Setup Configurations	43
Figure 19 CBLOF Algorithm Configuration	44
Figure 20 Unsupervised DM Process	44
Figure 21 Import Data 1 of 2.....	45
Figure 22 Import Data 2 of 2.....	46
Figure 23 Unsupervised Results.....	47
Figure 24 Outlier Decoding Process	47
Figure 25 Network Architecture	48
Figure 26 Network Training	49
Figure 27 Data Selection MATLAB.....	50
Figure 28 Data Validation and Testing Configuration	50
Figure 29 Supervised Confusion Matrix.....	51
Figure 30 Test Logs Example	52
Figure 31 Overall Test Results	52
Figure 32 Average Users Vs Full Data Set.....	53
Figure 33 Compiled Dataset Vs Individual User Datasets	53
Figure 34 Supervised Results.....	54
Figure 35 Single Algorithm Vs Hybrid Approach.....	55
Figure 36 Hybrid Comparison.....	55

Acknowledgements

First and foremost, I would like to thank my supervisor, Dr Naghmeh Moradpoor for all the valuable advice and support she provided throughout this project. The feedback provided was instrumental in the production of this work. I would like to also thank Dr Gordon Russell whose feedback at the midpoint and poster presentation was invaluable.

Next I would like to thank Kate Robson and Matt Little from ZoneFox without their support and time. Without which I would have been unable to deliver as what I have. Access to realistic demo data ensured I could deliver all my objectives.

Finally, I would like to thank all my friends and family who have provided steadfast support over the last four years. Especially Lynsey who has tolerated with me during this time for which I am eternally grateful.

1 Introduction

1.1 Context

The threat landscape has evolved from primarily external threats, to more than half of all attacks in 2014 being carried by accidental or malicious insider incidents (IBM Security, 2015, p. 6). Historically inside threat prevention has been conducted by implementing several rules or restrictions to either the system or the end-user including “Enforce separation of duties or least privilege” (Cert, 2009). These approaches require all possible vulnerabilities to be known. For example, if the risk is a member of staff stealing data then all methods of data theft need to be known, such as email and external network access restricted or USB drives disabled and a rule to be created to protect against such a vulnerability. The issue with this approach is that it is not possible to predict all possible attack vectors to ensure a rule is created. With more companies encouraging employees use ‘Bring Your Own Device’ (BYOD), the risk is being able to set rules to fully protect a system while not restricting the device to the point it is unusable. By 2017, it was suggested that half of employers would want their employees to provide their own devices (Gartner, 2013). A survey carried out also found that almost 70% of companies were concerned about compromising personal identifiable information (Cole, 2015). To address these challenges a more dynamic solution is required with the recent advancements in hardware the use of machine learning and data mining tools could hold the answer to this problem allowing better insider threat detection by using event logs to predict abnormal user behaviour.

1.2 Background

Over the space of a year fifty-four companies experienced a combined total of 874 insider incidents, with the financial sector accounting for 17% of these incidents. The same report showed that negligence was the main cause of incidents, as illustrated in Figure 1. The findings also concluded that an average of \$4.3 million being spent to mitigate or correct insider threats each year (Ponemon, 2016). A report published in 2015 found that insider attacks took the longest to

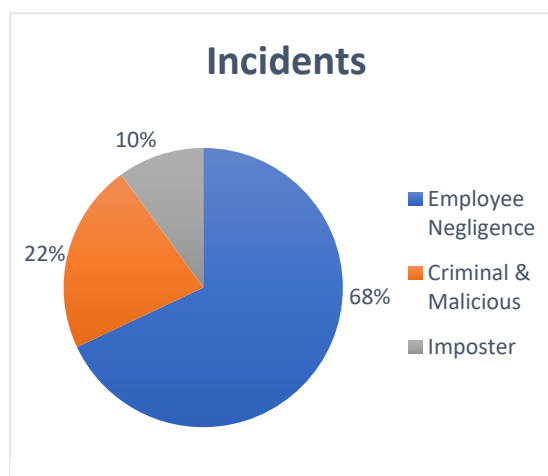


Figure 1 Insider Threat Figures

resolve at fifty-four days and the cost of most averaging around \$144,000 per incident (Draper, 2015).

A follow up report found that the trend of known insider threats continues to sit around 20% of all reported breaches (Verizon, 2016). The report shows that the most common motivation for internal threats is financial with espionage a distant second, these findings are illustrated in Figure 2. Using information from these findings it is possible to improve screening for potential threats and vulnerabilities within a company. Research carried out by Imperva found that undiscovered insider threat events were detected in all of the companies that were investigated (Lonergan, 2016), these findings show that many insider incidents routinely go undetected or reported.

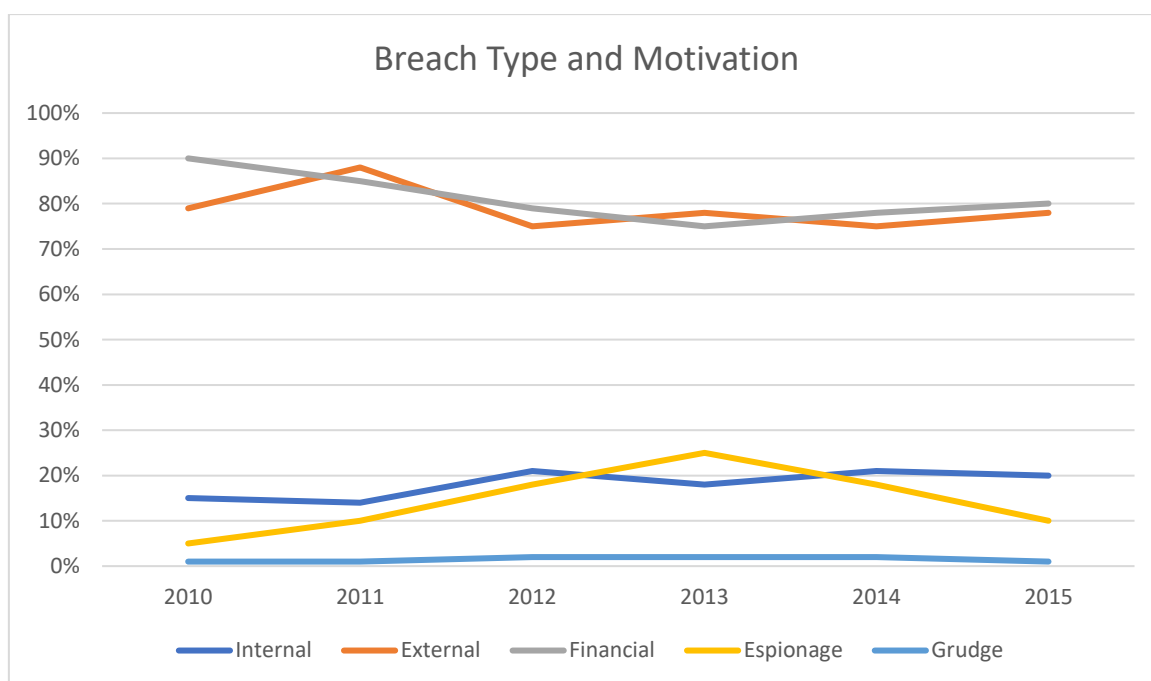


Figure 2 Insider Type and Motivations

1.3 Aims and Objectives

The key aim of this project is to design, implement and evaluate a more dynamic method of detecting abnormal user behaviour. This will be achieved by conducting a detailed literature review of the current methods and tools used to detect and protect against the growing trend of insider threats. This method should address the challenges currently faced in the detection of insider threats found in the literature and technical review carried out.

For this to be achieved the following objectives must be met:

- Review existing methods of insider threat detection and the most common threat vectors. Explore data mining and machine learning approaches to insider threat detection.
- Design a suitable prototype method based on the findings of the literature review. Develop several real-world scenarios to test the proposed method against for evaluation.
- Implement the proposed approach and conduct several experiments using developed scenarios.
- Evaluate results from implantation of the proposed method and experiments, analyse findings and compare the results. Conclusions will be made on the success of the project based on the results produced from the experiments.

1.4 Dissertation Organization

The rest of this dissertation is organized into seven chapters, as follows:

Chapter 2 investigates several alternative tools, terminologies and algorithms to provide the background knowledge that is used to ensure clear understanding of the core concepts used in this paper. Also presents a conclusion on potential use within the practical stage of the project.

Chapter 3 provides a detailed literature review of the key aspects involved within this project to ensure a good level of understanding. A number of conversations with experts are carried out and compared against other sources of information. The results of this chapter are used to provide the foundation for the proposed system.

Chapter 4 identifies the methodology that is use for this project. Data provided from ZoneFox is reviewed and defines each scenario used. The proposed system structure is also presented in this chapter with detailed diagrams and explanations of each stage of the system as well as some potential challenges.

Chapter 5 explores the pre-processing steps that were carried out on the data provided by ZoneFox. All modifications to the data are presented and reasoning behind the rationale are given. The conversion and normalisation stages are also clearly explained and the final data is shown.

Chapter 6 defines how supervised and unsupervised approaches were implemented, the installation of all resources and the configuring of the systems. The chapter concludes by displaying how the results will be collected and a review of challenges that were discovered during the practical work.

Chapter 7 critically evaluates the results of all testing carried out before comparing finding and presenting a conclusion on the performance of the supervised and unsupervised approaches.

Chapter 8 contain a detailed conclusion of the dissertation providing overall findings from the project as a whole along with checking that the aim of the project was achieved. A list of potential future work is provided to also for the project to be taken forward. The chapter concludes by providing a personal reflection of the project including successes, challenges and learnings.

2 Terminology

2.1 Introduction

In this chapter several key terminologies, tools and methods are discussed to help provide a baseline understanding for use within the project. The chapter is broken down into several sections starting with a comparison between Data Mining (DM) and Machine Learning (ML), followed with the processing options available for dataset analysis. The chapter finally explores several different DM and ML algorithms and concludes by looking at a selection of DM and ML tools.

2.2 Data Mining Vs Machine Learning

A detailed description of the difference between DM and ML can be found in (Guyen, 2016).

“ML algorithms need a goal from the domain (e.g., dependent variable to predict). DM focuses on the discovery of previously unknown properties in the data. It does not need a specific goal from the domain, but instead focuses on finding new and interesting knowledge.”

In other words, ML focuses on data that has a predetermined search parameter, for example finding all network attacks matching a DOS signature. While DM is used to find hidden and unknown patterns in data, such as anomaly detection that can be used to identify Zero-day exploits and insider threats. Arthur Samuels produced one of the first examples of ML as early as 1955 by writing a program that understood the game of Checkers and was able to compete against human players, improving its ability with each game. The program went on to beat a former checkers champion (Copeland, 2016).

DM is the newer of the two methods and has been around since the 1980s, it involves the mining of data from large datasets to carry out either supervised mining in which a specific goal is achieved. An unsupervised approach that can be used to detect hidden patterns or provide a better idea of what the data contains. DM techniques can include cluster analyses, classification, regression trees and neural networks. These techniques can use statistics and pattern recognition to group or extract behaviours (Sumeet Dua, 2016). DM is currently the more common approach used, however ML is being identified as a possible solution to the ever-growing amount of big data. With some techniques having already successfully been applied to pattern recognition, finance and entertainment (El Naqa, 2015). Normally ML would be used for a prediction based projects that contains labelled data. With DM being used to find hidden patterns when using an unsupervised algorithm. Selecting a method is important and depending on the selection several additional processes may be required including data normalisation.

2.3 Processing

In this section, a detailed look at STREAM Clustering and Normalisation will be carried out. It is possible that both of these processes might be used within the project and a good level of understanding behind each is required.

2.3.1 STREAM Clustering

One of the challenges raised when dealing with large datasets is the computational power required and the time taken to complete tasks are high, this is especially true when using clustering algorithms. Even small datasets containing less than a thousand events can take hours to run and can use up much, if not all the available RAM on a system. To mitigate this problem the process of stream clustering was proposed, the process involves an ordered sequence of points $x_1 \dots x_n$ being read or scanned in order. The process was developed to deal directly with the challenge of big data (Sudipto Guha, 2003). The basic idea is that when data is being read in it is assigned a cluster identifier, this allows the events to be clustered in batches or runs and then over laid at the end of the process. The paper concluded that there was a trade-off between accuracy and run time. Stream clustering is likely to be out with the scope of this project. However, with clustering data streams being a difficult task some suggest that newer methods of clustering are needed to keep up with evolving needs and data types (Mohamed Elasmr, 2007).

2.3.2 Normalisation

One of the pre-processing stage of DM and ML includes normalisation of the data, the process brings all the input data to within the range of [0,1], this is done to ensure a consistent range of values for each input across the model. A lot of DM tools use a distance metric such as Euclidean distance, if the data features are not normalised they can create biases that would result in skewed results by implying one feature is more important than another based on the numeric value it is assigned. Normalisation is used to mitigate this effect by giving all features the same scale this provides consistency if there are different ranges of data over multiple

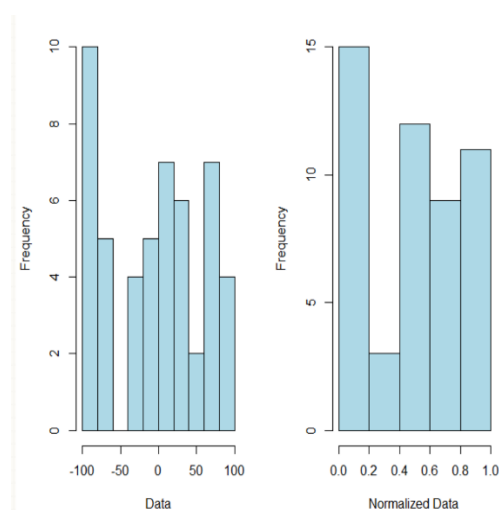


Figure 3 Standard VS Normalised Data

runs. A new possible alternative technique for normalisation of data is explored, however not suitable for the scope of this project (S.Gopal Krishna Patro, 2015). A common formula that is used to normalise data is ' $X = (X - \text{MIN}(X - X_n)) / (\text{MAX}(X - X_n) - \text{MIN}(X - X_n))$ ' (Saitta, 2007). Figure 3 provides a visual comparison between standard data on the left showing a large data range

with widely ranging frequency and normalised data on the right being more refined and showing a reduction in variances. This process is important when using distance based algorithms to prevent biased results.

2.4 Algorithms

This section reviews the main categories of DM and ML that are commonly used including: supervised, unsupervised and semi-supervised. It also reviews and discusses the pros and cons of a selection of algorithms to provide a better understanding of which might be suitable for use within this project. *k*-Nearest Neighbour, *k*-means and decision trees will all be discussed.

2.4.1 Supervised

Supervised learning algorithms rely on a large amount of reliable labelled training data; this means that the system knows what it is looking to find and can be trained to a high level of accuracy. The results are then verified using known unlabelled test data. Supervised learning works by knowing the input data (*X*) and the output result (*Y*) within the training data, $Y = f(X)$. The aim is then to approximate the mapping function so when new input data (*X*) is entered the model will predict the output (*Y*) (Brownlee, 2016). Supervised learning algorithms are mostly used for classification or prediction problems and can be extremely slow with large datasets. Examples of supervised learning include Regression, Decision Tree, Random Forest and *k*-Nearest Neighbour (Ray, 2015).

2.4.1.1 *k*NN

k-Nearest Neighbour (*k*NN) is a type of clustering known as instance based learning and is primarily a supervised learning algorithm used when a little knowledge about the dataset is known. Findings also show that using similar clustering methods in an anomaly detection scenario can produce very high accuracy of around 98% (Guyen, 2016). This relies on several factors including feature selection, method and algorithm selection as well as test dataset type. This method could be adapted to work within the scope of this project. *k*NN has been used recently to detect insider threats as a possible replacement for community-based anomaly detection system (CADS). The research also found that using a slightly modified *k*NN algorithm detected abnormal behaviour events. These events are defined as outliers as they do not conform to the normal pattern of behaviour. The findings concluded that using a modified *k*NN algorithm produced better results as it could be implemented without any additional user data (Aruna Singh, 2014). Using the *k*NN clustering algorithm can be very computationally expensive which is exacerbated with large datasets. Due the algorithm using distance based functions such as Euclidean distance, all variables are required to be normalised to ensure no feature bias is created that adversely affects the results. An example of *k*NN can be found in

Figure 4, the first stage seen on the top left depicts the raw unclassified data that has been plotted. The second stage of the process seen on the top right shows k being set, this is also known as the centroid. This is done randomly with the user only specifying how many clusters are needed and the distance between events. The third stage involves the data points being assigned to their nearest cluster, this is shown on the bottom left. Any data points that do not fit in to the original clusters are defined as outliers. The final stage of the process is to assign all outliers to their nearest cluster to allow for convergence.

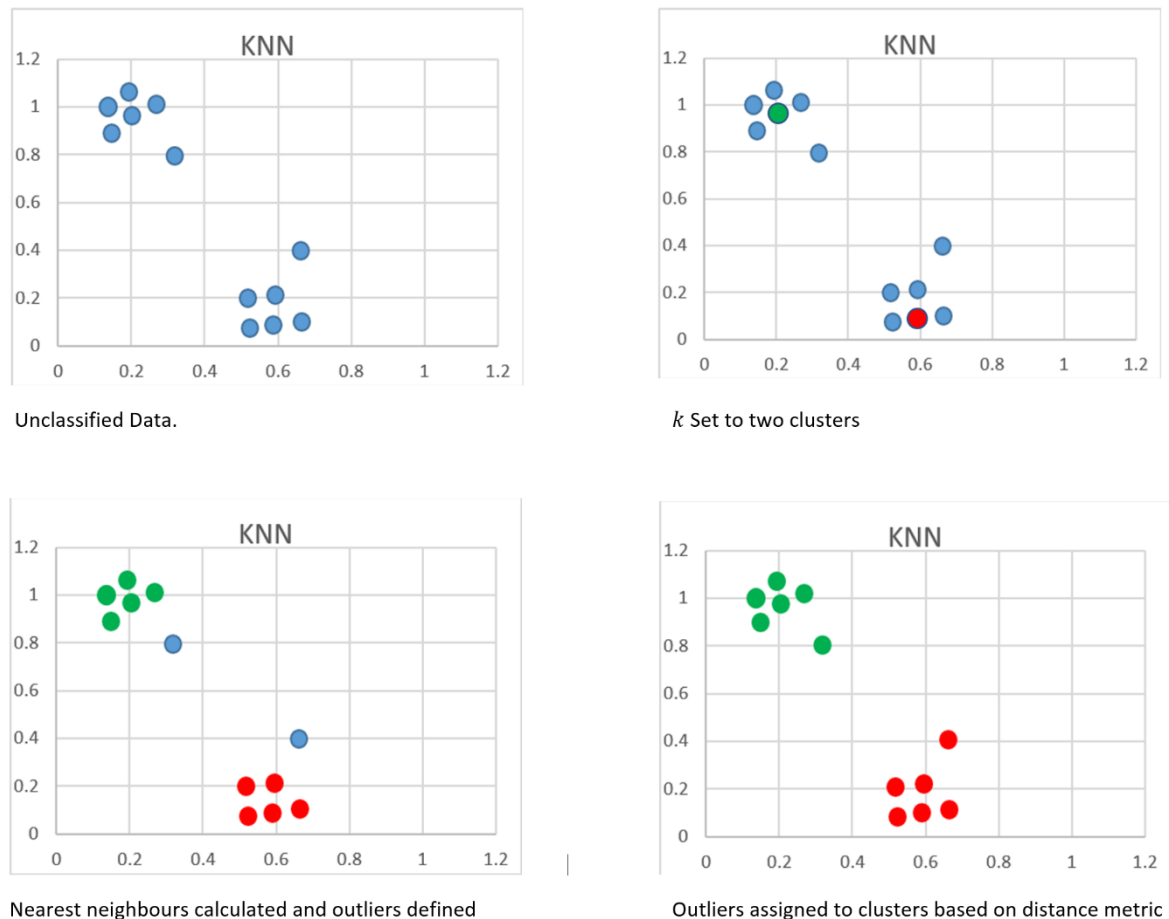


Figure 4 KNN example

2.4.1.2 Regression

There are a couple of different algorithms that can be used for regression problems including Linear regression and Random forest. Regression algorithms are used when the output variable is a real value such as currency or temperature. This research focuses on linear regression and its possible use for insider threat detection. Linear regression requires that you need both input and the output data or labelled datasets (Teoulas, 2016).

A common example of linear regression is arranging people in order of weight without knowing the actual weight of the individuals, the people would be visually assessed and arranged based on build (Y) and height (X). In this example, the variable Y is known as the dependant variable and the independent variable is X, if they both increase, such as the build increases with height then the regression would be positive. However, if the Y variable increased while the X variable decreased then it would be a negative regression (Statisticsfun, 2012). Once each person is identified a best fit line or regression line is added using the equation $Y = a + bX$. An example of positive regression can be seen in Figure 5.

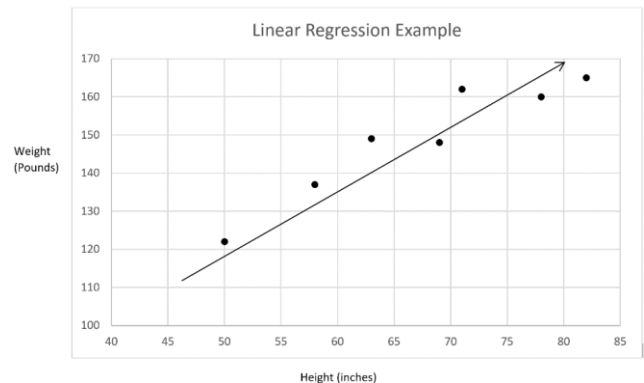


Figure 5 Linear Regression Example

2.4.1.3 Decision Tree

A Decision Tree (DT) algorithm follows a tree-like structure where a decision point defines the outcome. This allows for highly accurate classification of data, one of the most common and accurate algorithms used for DTs is the C4.5 algorithm (Guyen, 2016). However one of the key draw backs to DTs shows that feature selection can become biased as the number of levels within the tree grows.

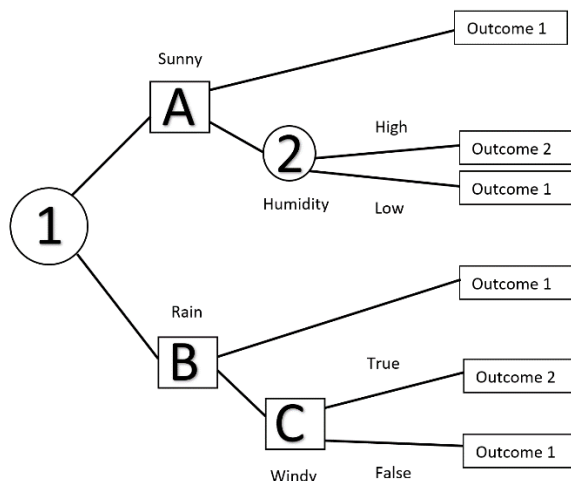


Figure 6 Decision Tree Example

then it is likely that outcome 2 will apply, however if it's raining and not windy outcome 1 would mostly likely apply, as each decision node is reached a decision is made based on the data available and the process moves to the next level, this process is shown in Figure 6.

Decision trees work well when output value is a Boolean (e.g. yes or no), the method also works well when some of the data is missing within the dataset or the dataset contains errors (Mitchell, 1997). Published findings found that when comparing DT and Support vector machines (SVM) for intrusion detection, DT outperformed SVM consistently and showed a similar or higher accuracy in all the tests. With accuracy, as high as 99.86% (Sandhya Peddabachigari, 2004). A common example of DT is predicting if a sports game will be played based on data about the weather. For example, if it is raining and windy

2.4.2 Unsupervised

With supervised learning algorithms relying on labelled data then, unsupervised learning algorithms can best be described as the opposite. Unsupervised learning is used when only the input data (X) is available. Unsupervised algorithms are used to detect patterns in the unlabelled data, this is helpful when looking for anomaly detection and finding complex patterns that might otherwise be unknown. Unsupervised learning tends to focus on clustering of data. One of the challenges with this method when using large amounts of data is understanding what the algorithm is finding. It is important to understand that no prediction is being made, it merely groups patterns together for classification. As unsupervised algorithms focus on detecting patterns and anomalies, it is important to take in to account that if malicious behaviour has been happening regularly then it is highly likely that it will continue to go undetected and any similar attacks using the same approach will also be undetected. To that end, it is important an unsupervised model is used in conjunction with additional security measures. Examples of unsupervised learning includes Autoencoders, k -means clustering and DBScan (Donalek, 2011).

2.4.2.1 Autoencoders

An autoencoder neural network is an unsupervised anomaly detection algorithm that uses backpropagation, matching the target values to the input values ($y(i) = x(i)$). Autoencoders can be made up of many hidden layers yet consist of at least three layers, Input, Encoded and Decoded (Standford, n.d.). A detailed look at several different autoencoder models and good explanation of the underlying mathematics involved are available (Baldi, 2012). However, autoencoders are likely out with the scope and expertise for this project. The process that the autoencoder follows can be simplified to a few steps:

Step 1: Input unlabelled data and encode, reducing the data to smaller key point similar to compression.

Step 2: Decode or recreates the data from the pattern in the encoded data based on weights and biases of the connections and nodes.

Step 3: Uses optimiser such as gradient descent to adjust the weights and biases to reduce the loss between the recreated data and the input data.

A visual understanding of this method can be found in Figure 7. The outlier is generated by the autoencoder calculating the loss, this is data that is not part of the recreated pattern. With the recreated data being based on the patterns of the encoded data, the recreated will fit a normal pattern. Thus, meaning that any outliers will have a large loss value and likely be abnormal events (Standford, n.d.).

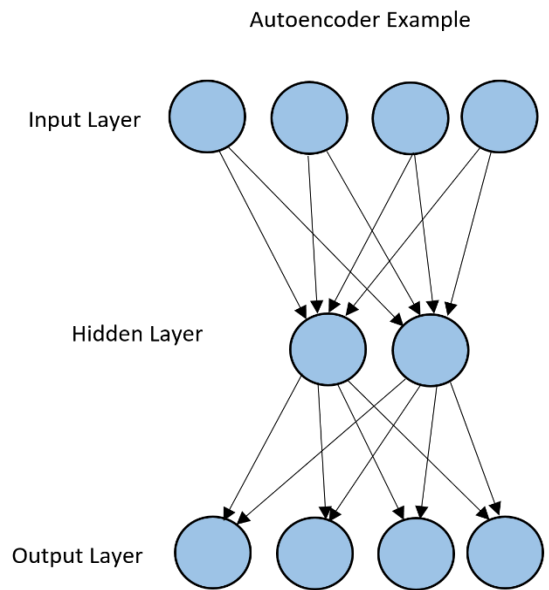


Figure 7 Autoencoder Example

2.4.2.2 *k*-Means

An unsupervised algorithm that uses clustering is *k*-means, this algorithm aims to create *k* clusters with a minimal within-cluster sum of squares. In other words, the algorithm partitions the dataset into a set number of clusters based on what has been specified. The algorithm then randomly assigns the specified number of *k* points, these are defined as centroids. Convergence is reached by the iteration of two key steps (Ray, 2015):

- I. Each data point forms a cluster with the closest centroids.
- II. Each centroid is reallocated to the centre (mean) of all data points assigned to it.

Outliers are detected based on the data points furthest from the centroids after convergence has been reached, this can be done using a distance metric such as Euclidean distance. Depending on dataset type can lead to optimisation and distance metric selection. The accuracy of *k*-means can be negatively affected by the presence of outlier as the “mean” will not be optimised. This can increase the time it takes to reach a convergence and affect the results (Steinberg, 2007). Finally, *k*-means like *k*NN is computationally expensive and with large dataset it might require optimisation such as stream clustering to improve performance.

An example of *k*-means can be seen in Figure 8. Iteration one shows that all data points have been plotted and the three centroids randomly placed, the centroid location is then recalculated using the mean of the data points assigned to it’s cluster. The process is repeated until convergence is achieved or until a defined number of iterations is carried out (Iteration 2 -5). The sixth iteration shows that each of the three centroids has been assigned to three clearly defined clusters.

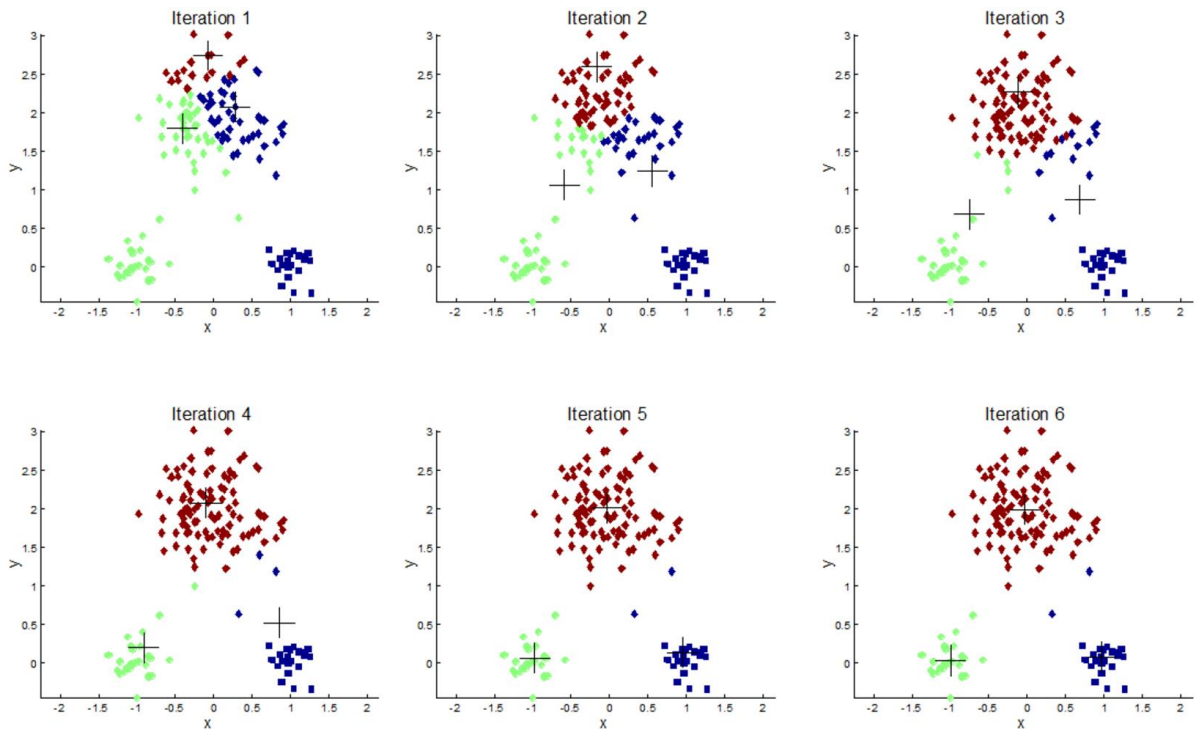


Figure 8 K-means, 3 centroids Example (Pandre, n.d.)

2.4.2.3 DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) was put forward as an alternative to current clustering models such as *knn* or *k*-means. Results published in (Martin Ester, 1996) found that DBSCAN was significantly more effective at discovering patterns in arbitrary shapes. DBSCAN works by estimating the density around each data point based on user assigning point limits or the minimum number of data points in a cluster (minPts), this allows for clustering of data points that are more likely to be part of a pattern. The algorithm only needs to be told the minimum distance from a cluster point (eps-neighbourhood) and minPts (Michael Hahsler, 2016). As DBSCAN uses a distance metric similar to other clustering algorithms, such as Euclidean distance the accuracy can be adversely affected. For high-dimensional data can be affected by an effect known as “curse of dimensionality” that creates problems in finding a value for *E*. Naftali Harris provides a fantastic visual example of DBSCAN and a comparison to *k*-means, the example provided in Figure 9, shows a smiley face with DBSCAN identifying five clusters and outliers, were as the *k*-means example has been set to four centroids and provides a completely different result (naftaliharris, 2015).

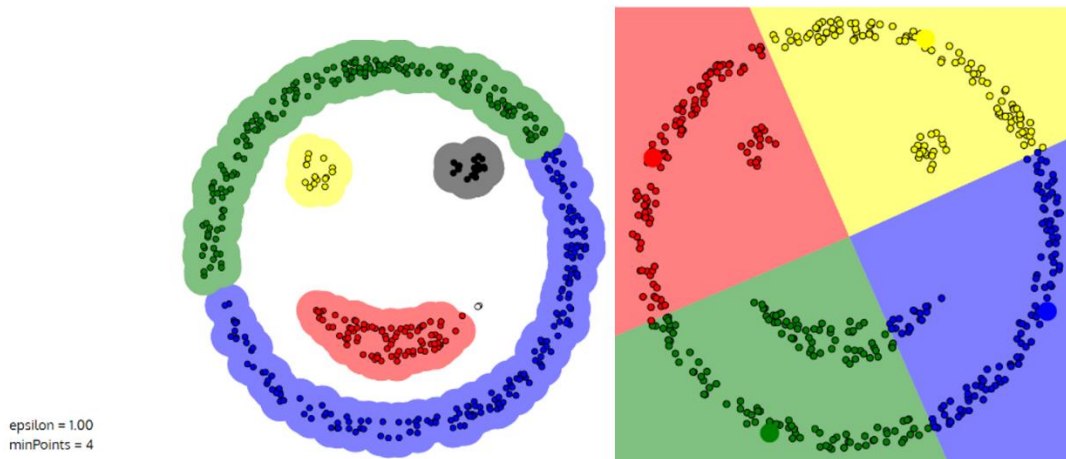


Figure 9 DBSCAN VS K-means Clustering Example, (naftaliharris, 2015)

2.4.3 Semi Supervised

Supervised and unsupervised algorithms both have their own strengths and weaknesses; supervised algorithms are able to produce highly accurate results, however need large amounts of reliably labelled data which can be difficult to produce. Unsupervised algorithms identify unknown patterns and find outliers the user might not expect, they are however unclassified and require additional action to make the data useable. Semi-supervised/reinforced learning is an attempt to mitigate these weaknesses while taking advantage of their strengths. A possible approach is using unsupervised anomaly detecting and clustering algorithms to detect patterns and highlight outliers. Once detected these outliers are then classified and built in to a supervised algorithm. MIT and PatternEx discuss the shortcomings of supervised and unsupervised algorithms and present an alternative approach (Kalyan Veeramachaneni, 2016). To achieve the best results a number of open source DM and ML tools will be reviewed for use within this project.

2.5 Tools

Brownlee discusses the benefits of using tools when carrying out any task to improve the results (Brownlee, 2016). This is true in all aspects from IDS systems to none computer related tasks. For this project, the use of open source tools will allow for faster and easier use of the data as many platforms provide a graphical user interface, with an underlying command line interface for more direct interaction with the algorithms. Open source tools are important as they allow any findings to be easily duplicated in the future without proprietary software. Within this section three open source tools are discussed and compared.

2.5.1 Data Mining

A review of three available open source DM tools including Rapidminer, Weka and R-programming. These are described as the top three open source DM tools (Goopta, 2014) even though this article is a few years old the platforms discussed are still widely used and available today as seen in (Immanuel, 2016) and (Zhao, 2016).

2.5.1.1 Rapidminer

Rapidminer (Rapidminer, 2017), provides a template based framework that allows for advanced analytics using Java (Goopta, 2014). There is no need for the user to have experience in coding to accurately use this tool. The platform uses a four-stage process that allows user to prep data, select model, validate findings and operationalize. This platform is used by many large companies including BMW, Cisco and Amnesty international (Rapidminer, 2016).

2.5.1.2 Weka

Weka (Waikato, 2016), is a ML tool which focuses mainly on DM, there are several detailed and informative tutorials available online (Witten, 2016) that start at the entry level and work up to advance DM. Weka contains tools for pre-processing, classification clustering and more. One of Weka's drawbacks is that it formats datasets in a unique way and this could possibly create issues depending on the format of the extracted system logs (Waikato, 2016).

2.5.1.3 R Programming

R (The R foundation, 2016), is an integrated platform that allows for data manipulation, calculations and more, R is written in C and Fortran focusing on statistical computing and graphical modelling. There is detailed documentation available and the ease of use has contributed to the popularity of R (R Development Team, n.d.). R also allows for classification and clustering of data, this might be useful when searching for abnormal behaviour that is hidden.

2.5.2 Machine Learning

There are many ML platforms available on the market currently, some paid for, however a large number are open source. "[Artificial Intelligence] can help us in everything from accomplishing our daily tasks and travels to eventually tackling even bigger challenges like climate change and cancer diagnosis." (Dean, 2016) this seem to be the opinion shared by several large companies including Yahoo, Google and Microsoft who have all made at least some of their ML code open source. Desale 2016, provides a list of their top 15 frameworks

for ML, although this is aimed towards experts. It does allow for an insight in to what platforms are currently available (Desale, 2016).

2.5.2.1 Tensorflow

Tensorflow is one of the most well-known ML tools and was developed by Google brain team (Wikipedia, 2016) and released late 2015 under Apache 2.0 open source. There are a wide range of detailed tutorials and documentation available on the Tensorflow website, this allows a user with no experience to jump straight in and get an idea of what the platform can do. Tensorflow is used both python and C/C++ API's. Tensorflow also provides TensorBoard a visualisation tools to demonstrate computational graphs and outputs (Google, 2016).

2.5.2.2 MATLAB

MATLAB is widely used in both academia and the research community the platform provides a lot of user flexibility when writing their own models. MATLAB is used for ML, signal processing as well as data classification (MATLAB, 2017). With MATLAB, having been around for the best part of 20 years, the reliability and user interface is of a very high standard. A potential advantage to using MATLAB with this project would be the scalability, with large datasets the ability to upscale quickly is key and having a tool that is capable of such functionality would present a key advantage.

2.5.2.3 Caffe

Caffe is widely used for research and has been built with Spark in mind, thus allowing for large scalability. It includes C++, Python, and MATLAB interfaces, however is mainly focused toward image recognition and provides both GPU and CPU support (Yangqing Jia, 2016), with images having higher dimensions of features compared to system logs.

2.6 Conclusions

The research highlighted the fundamental differences between ML and DM. Surprisingly showing that DM is newer than ML, even though ML is starting to return to the foreground with the development of advanced processors and the ability to use complex algorithms effectively. The importance of normalisation was explored and the impact of not normalising data when using a distance based function was made clear. Several supervised and unsupervised algorithms were explored with their advantages and disadvantages being discussed.

In conclusion findings from this chapter identified that Rapidminer will be used as the DM tool as it presented an easy to use GUI and provided plenty of documentation via the developer's website. The ML tool that will be utilised for this project is Tensorflow, the advantages include the high number of online resources and support. Both tools allow for Comma Separated Values (CSV) files to be read directly in to the tool without any addition modification.

Initial selection of algorithms for this project will consist of k -NN, Regression, DT and Clustering, these algorithms appear to be well suited for anomaly detection and classification and present the best potential to ensure high accuracy is achieved. However further changes may be required depending on initial results and further review. The next chapter contains a literature review of published findings that contain examples of insider threats, DM and ML. These are reviewed to provide a good understanding of previous work and highlight any challenges that may arise.

3 Literature Review

This chapter provides a detailed literature review on some of the key aspects involved within this project. The first section contains real-world scenarios within the scope of this project, allowing for a clear understanding of real life challenges faced in insider threat detection. To ensure that previous experiences are learnt from and a new approach is can be suggested to try and improve upon current methods, the second section contains a review of current models and works along with limitations of these methods. The chapter ends with a conclusion on the literature found and the usefulness within the scope of the project.

3.1 Real World

Several conversations with Kate Robson and Matt Little from ZoneFox (See Appendix 4 & 5) provided some real-life examples of the type of insider threats that have been seen already. Some of the findings from this exchange highlighted that insider threats are not limited to a single industry or staff type. Although individuals with a gripe are known to be a threat, the majority of incidents are financially motivated.

Kate was happy to share her personal thoughts on a number of questions that were put to her, the scale of insider threats is underestimated by many companies with a common response being “oh our employees wouldn’t do that”, Kate also raised the fact that with harsher fines and penalties coming in to force with the new GDPR companies are suddenly becoming more interested in where their data is and what is happening to it. Finally, Kate raised several key challenges in detecting insider threats including the amount of data and time restrictions on when the information is usable to the fact that rule based systems are not practically in anomaly detection. The full interview can be found in Appendix 5.

An example that the team at ZoneFox has worked on previously was a large engineering company who worked on developing innovative technology. One of the high-level members of staff were in the process of changing employers and from the surface the work they were conducting was completely normal. However, ZoneFox could prove that the member of staff was creating backups using unauthorised software and stealing proprietary data. This example illustrates the ease that individuals in positions of trust can deceive others and carry out malicious activities. It also indicates that no matter how secure a system is perceived to be, someone can still breach it when they are on the inside and have access.

There are several steps that can be taken to help protect against insider threats. These steps are the most basic level of insider security and should be used as a building block for more advanced and secure policies and systems. The six suggestions come from (ObserveIT, 2016), however they are generally shared by most IT security companies and specialists.

Step 1: Ensure all intellectual property is properly secured and access is restricted. Provide detailed training on policies to all staff.

Step 2: When staff leave the company, change any common passwords, and ensure all departments are aware of the termination of employment.

Step 3: Check departing staff to ensure no company data is in their personal possession.

Step 4: Ensure that only staff that need access to certain data have the access and when their need is removed, change their access privileges.

Step 5: Have staff use secure passwords and provide correct training on how to use secure login details that do not need written down. Having a password on a posted note is common.

Step 6: Use monitoring software and ensure staff are aware of its use, people are less likely to do anything malicious if they know they are already being watched.

3.2 Current Works

This section reviews a selection of papers and research material that identifies work carried out regarding insider threat detection, looking at the definition and some background related to the threat. The section continues by comparing publications that have used DM and ML to detect insider threats.

3.2.1 Insider Threat

Defining the Insider Threat, Bishop 2008 argues that even though many organisations have carried out research on insider threats, there has not been a concise definition of what an insider threat is and the different types of insider threats present (Matt Bishop, 2008). This has led to the inability to tailor detection techniques to detect certain types of insider threats. The paper states that an 'insider' can be defined by two key actions:

“violation of a security policy using legitimate access.”

“violation of an access control policy by obtaining unauthorized access.”

This means that an insider threat does not need to be a member of staff but could also be someone who has internal access to the system or network resources. Bishop 2008 provides

an opinion that varying levels of access control and conflicting concerns can constitute different levels of “insiderness”. If the fear is someone taking hardware or hard copies of data out the building, then anyone with physical access is an insider threat from CEO to cleaner. Alternatively, if the fear is someone might remove security processes then only individuals with access to do so would be classified as a higher insider threat. The approach that is taken allows for a lattice approach to be implemented, this requires a risk assessment to be carried out on components of the organization that need protected the most. This determines who has access to these components and assigns them as a higher risk factor. This can include not only direct employees but also allows for the identification of contractors.

This approach is interesting as it brings to light the fact that an insider threat might not be a direct employee but can be anyone who has access to critical resources and infrastructure. Bishop 2008 identifies challenges in defining an insider threat showing clearly that many reports differ in their definition from a trusted person with access and knowledge to just someone with access, having no reference to the trust or knowledge aspect. The use of this approach would allow for more precise threat detection by reducing the amount of data that needs to be processed in real time allowing for highlighted groups of insiders to be prioritised over others. It is however important to emphasise that groups with lower rankings still need to be monitored to ensure detection of all possible breaches. Although this definition is not referred to by everyone working in the field it does provide a unique view.

Insider threat is further discussed in Miltiadis 2010 when researchers from Athens University explored the ability to predict an insider threat by identifying behaviours and patterns in user activity. Including both technical action and psychological patterns along with user taxonomy, the system works by using both a rule based and ML approach. This allows the system to measure user activity against known security policies and to pattern mine user search habits to help predict malicious intent. The framework is broken into three factors, Motive, Opportunity and Capability with users being assigned a risk ranking based on their score for each factor. A user with high levels of stress might for example be more motivated due to personal challenges (Miltiadis Kandias, 2010).

Several limitations were identified in this research such as all data sent across the network needed to be sent in the clear to allow for data processing. This showed an inherent weakness to insider exploitation from users with the capability. Miltiadis 2010 also finds that the amount of data needed grows at a higher rate the longer the system is active and additional over heads would also be required. The approach of this method is interesting as it allows for both a psychological evaluation of staff along with the technical monitoring of users. That however is

also the main weakness of this approach to insider threat detection. The costs and legal requirement to perform psychological evaluations on all staff and constant monitoring would restrict this system to organisations with restricted material or restricted locations such as high-level security and military organisations. On the other hand, if a system could be used to collect the user data and automatically produce a psychological profile to better predict insider threats this might be worth exploring. This however presents its own challenges in being able to use large datasets.

DM with big data has become something of a necessity as the size of datasets have exploded in the past several years. In 2013 Xindong and team submitted that 90 of the data generated had been done so in the last couple of years (Xindong Wu, 2014). The paper looks at the challenges faced with big data and proposes a model for processing data to using in a DM perspective. The challenges are later compared to a group of blind people sizing up an elephant and the limited view of each leading to a biased conclusion. Xindong proposes using the “heterogeneous, autonomous sources with distributed and decentralized control, and seeks to explore complex and evolving relationships among data.” (HACE) Theorem approach.

Xindong 2013 looks at several areas referring to DM including available tools and methods currently used, the challenge of sharing data across organisations while ensuring data protection is also looked at with the option of anonymising sensitive fields that would allow for data to be shared. The paper concludes by acknowledging the challenges and costs both in monetary and computational cost of DM with big data, while understating the need to continue to evolve and keep up with ever changing data types. Critically the paper presents many of the same challenges and methods that have been reviewed previously with several tools and algorithms that work well with DM listed including, k-Means, logistic regression and naive Bayes.

3.2.2 Algorithms and Tools

Guyen 2016 presents a detailed review of several different ML and DM methods used in security applications. The paper aimed to provide as much advice and references to allow a user to select an algorithm based on the results of previous works done. A discussion about cyber security datasets is also carried out and compared, this includes network data, public datasets and packet level data. The survey takes several ML and DM methods and discusses them in detail along with references to works that have used these approaches in a cyber-security aspect. The methods explored include artificial neural networks, clustering, bayesian network, decision trees, naïve bayes (Guyen, 2016).

The paper concludes that due to the large number of critical criteria, such as accuracy, complexity and speed of a system it is not possible to establish any recommendation that shows which method is best suited for a cyber-security scenario. The survey also suggests that ensuring reliable labelled training and test datasets is important as it would allow for improved performance of a system. This paper provided a wide look at different methods that could possibly be implemented in this project. The additional information provided examples of their use in a cyber-security role previously along with a detailed review of metrics including speed, capacity, and reliability. The additional research referenced in this work such as the anomaly detection and hybrid detection is promising as this will allow for further investigation. Looking at several methods along with examples of tools used such as Weka (Guvén, 2016), highlighting any pitfalls that have been discovered previously. Unfortunately, a lot of the datasets that have been used are almost over 15 years old this along with the data types containing mainly networked based data potentially limits the usefulness of the methods used for insider threat detection. However, the appearance of clustering and semi-supervised learning presents a possible avenue for further research.

An example of DM and using k –means clustering is presented in work by Gerhard Munz, Sa Li and Georg Carle (Gerhard M"unz, 2007). They look at using an unsupervised algorithm combined with Network flow records to detect patterns in the data and identify any abnormal activity including intrusion or network attacks. The approach used was focused around three steps:

1. Datasets containing the net flow records is normalised or converted to feature datasets.
2. The dataset is divided in to clusters of normal and abnormal data based on the use of k – means algorithm.
3. The resulting cluster centroids are used to run new data against to allow for faster anomaly detection.

The distance metric used in the clustering was weighted Euclidean distance this allowed for the detection of normal, abnormal and outlier events. The evaluation of the system showed that when the researchers used simulated network traffic there was expected results, however when using life data, there was a large anomaly when the UDP traffic was clustered. The anomalous cluster being eight times larger than the normal date, this was linked to an online game that created more outgoing one-way traffic as it was constantly requesting a status update on available servers and resources. Although not used for insider threat detection the work carried out in this paper demonstrated that clustering is a promising approach to DM of datasets as long as the data is pre-processed correctly. Compared to several other papers this one is better presented to allow for clear understanding of all aspect of the framework while not saturating the paper with complex equations. The last thing that was made clear in this

work is the importance of knowing that there might not be any malicious activity in the dataset especially if using a small dataset or the malicious behaviour has been going on that long that it is now classified as a pattern.

An alternative clustering algorithm that has been suggested is k Nearest Neighbours, in 2014 Singh and Patel put forward a paper that suggests a way of using a modified k -nearest neighbour (k NN) clustering algorithm to help and detect against insider threat on a collaborative information system (Aruna Singh, 2014). An explanation of the challenges that are faced when trying to detect insider threats on such a system are given, showing that with so much information available to be accessed at any time it is difficult to audit all users or even restrict users in a way that would not negatively impact performance. The proposed framework uses a modified anomaly detection algorithm together with a framework called Community-based Anomaly Detection System (CADS) “which infers the user communities from the access logs of the users”, suggesting that using just k NN with its distance function may not be enough to detect all anomalous activity and outliers.

Singh 2014 claims that the use of the modified k NN along with the CADs framework produces a noticeable improvement on anomaly detection over the original CADs or non-modified algorithms. This is hard to verify as the results are unlabelled and difficult to read. The paper does provide several suggestions that could be adapted in to this project. Graph based anomaly detection is mentioned along with the idea of using unsupervised algorithms as there is less pre-processing needed and this can improve the speed of a system.

An example of work showing unsupervised learning for insider threat detection is explored in (Pallabi Parveen B. T., 2012). The aim of the work presented was to show that as the same patterns are presented to an unsupervised algorithm the classification accuracy increases. This was achieved by first looking at stream mining and uses an unsupervised solution, finally the work addresses the challenges of unlabelled data. The dataset is then pre-processed to ensure a normative baseline is established to better predict the abnormal activity. This process is referred to as “compression method” and “Quantized dictionary”. Finally, the datasets are tested with the system as abnormal activity was so rare it is important that a baseline is established, to achieve this the team injected 25 anomalies into predetermined sections of the dataset.

The results of the work contained showed that across all seven tests the results for identifying true positives was between 97.5% and 98.3% with false positives being around the 40% mark (Pallabi Parveen B. T., 2012, p. 3). The overall conclusion from the work is that the suggested

approach shows a marked improvement over the current static models used. The work highlights a concern, for example as a user's ability grows with experience so does the risk and ability of an insider threat. Although this work focuses on a different type of dataset from this project it does provide a point of comparison when looking at potential challenges and directions to take the project. Another example of an unsupervised clustering approach is explored in Kumar 2016 the team proposed an improved method of DBSCAN. Within this paper, advantages of DBSCAN are compared against normal clustering algorithms and look at the additional costs in using this method in terms of time and computational power (K. Mahesh Kumar, 2016). The aim of their project was to implement an algorithm group to improve the speed of neighbour search queries. This is achieved by running two distinct processes against the data. The first involves groups to be set, groups are formed by assigning a selection of master nodes these are defined by ensuring their distance is equal to the eps, if the eps are double the distance the point is assigned to another group or a new master node is created. The second stage involves the master groups being merged with their neighbour groups based on the density of the groups. As the groups encompassed their neighbour the system reaches convergence. In short, the approach is similar to standard DBSCAN however instead of assigning a single centroid to start with and running through the data in a linear motion the system sets multiple centroids across the dataset and processes the data in small groups allowing for faster convergence.

The results show that after each algorithm is run against the benchmark datasets used the proposed method is faster than DBSCAN by a factor of 1.5 to 2.2. The results show a clear improvement in speed from the proposed method in all dimensional data from $5n - 65n$ (K. Mahesh Kumar, 2016, p. 8). The running time of the algorithms improved across the board being almost a third faster than current DBSCAN. The paper shows that group methods are also efficient in dealing with noise in the data and produced identical results to normal DBSCAN. Although the use of this method is outside the scope of this project several findings are highlighted including the power and complexity of patterns that can be used with DBSCAN, yet it also presents the relatively long processing time that this method is subject too, this is a common issue with trade-offs between speed and accuracy being common. A way of mitigating this effect is to look at a combination of supervised and unsupervised algorithms working in conjunction.

The advantages of using a semi-supervised approach is touched on by researchers from the University of Texas when looking at detecting insider threats by applying unsupervised stream mining to detect hidden abnormal behaviour. With less than 0.03% (Pallabi Parveen J. E., 2011, p. 1) of events showing as insider related, for the best results an organisation should

retain as much user event logs as possible to accurately train the system. Although the system presented did show an improvement over time in the number of false positive alerts the bulk of these fell within the first two iterations over three weeks dropping from 920 to 188. Finally, after nine weeks the final number was 150 (Pallabi Parveen J. E., 2011, p. 6). The final findings of the paper concluded that the ensemble-based approach did detect all abnormal events with fewer false positives than a single stage model. The research carried out looks at an innovative way of confronting insider threats, the use of graph based detection is however out of the scope of this project. The literature does provide an interesting look at unsupervised works completed previously and reinforces the growing trend that using a single algorithm or method of detection can return findings, yet for higher levels of accuracy a combination of methods should be looked at.

Veeramachaneni 2016 present an “analyst-in-the-loop system”. The aim of the system was to improve the detection of abnormal events and reduce the number of false positives in current intrusion detection systems while reducing time taken to act on the data (Kalyan Veeramachaneni, 2016). This was achieved by designing a framework that took advantage of both supervised and unsupervised algorithms. The model that they produced contained a data processing system that allows for feature selection based on input data, this is then put through three different unsupervised algorithms: density, matrix decomposition and replicator neural networks. The outliers are then ranked per the likelihood that they are indeed abnormal behaviour and the top n amount is selected and presented to a human analyst for classification. Once this is completed the newly classified data is then written into a supervised algorithm and built into the platform for future data to be run against. This process is repeated each time new data is added and the accuracy and speed of the system improved each time. The basic approach suggested involves any data being fed through both a supervised and unsupervised system with the results from the unsupervised system being sent to a human analyst to be classified and built in to the supervised system. This process is repeated each time the system is run and helps to improve performance and accuracy over time. This is illustrated in Figure 10.

The results that Kalyan found shows that using this approach allowed for a detection rate of 86.8% even with a low number of 200 daily events, this was a tenfold improvement over just using unsupervised outlier detection. The system was also able to limit the number of false positives to only 4.4%. Although this model focuses on deep ML and firewall logs data, the general approach to having a human analyst involved in the reviewing of outlier identification and providing feedback speaks for itself. As it shows the higher levels of accuracy that it provides along with the improvement in reducing false alarms over using a completely

automated system. An additional advantage to this approach is that zero-day attacks are more likely to be detected and identified by a human that will allow for faster resolution along with improved speed of the system as time goes on the less man hours that are needed as the system filters out data with the supervised feedback. This approach to semi-supervised learning may be adapted for this project depending on the tool selected for the project and the algorithms available to work with both the tool and dataset.

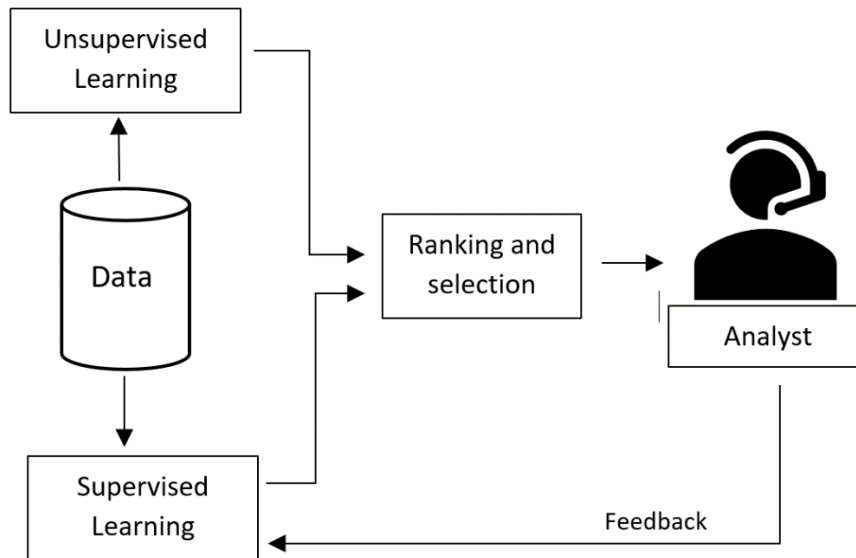


Figure 10 AI2 Semi-Supervised Example

A team of researchers from Wroclaw University in 2009 presented comparison of KEEL (An open source DM tool similar to Rapidminer and Weka), Rapidminer and Weka (Magdalena Graczyk, 2009). Where the researchers present the results of several different ML algorithms in conjunction with DM tools to predict the value of property. The team compare the differences in results when the dataset is run again, the available methods within the tools used included neural networks, decision trees and linear regression. They achieved this by selecting the best parameters using a trial and error method. Once this was achieved normalisation of the data was carried out as well as 10-fold cross validation.

The results from the experiments show that although the tools are similar in function the results differ greatly with prediction results falling in to the 16.2% to 19.3% range, with Rapidminer doing better with results as high as 25.3%. This is more impressive given the incomplete data that was used in the datasets. The paper concluded by stating that with the same algorithm being used the result should have been similar however this was only seen for KEEL and Weka when used with a linear regression method. However, for each algorithm there was a significant difference between KEEL and Rapidminer. Critically this work showed that although many tools are believed to achieve the same results when presented with the same data and algorithm, this is not always the case and small variances in the process can produce large

differences in the results. The approach used is a common one where incomplete data is used to predict the value of a house, however this work provides a good understanding of the different tools and the capabilities of these tools.

3.3 Conclusion

From this review of literature, a clearer understanding of insider threats and methods that can be used to detect or mitigate the impact of such threats can be drawn. A clear definition of what constitutes an insider threat is also explored and compared. Many of the challenges highlighted in the reviewed literature show that the pre-processing of data is important as well as effective algorithm selection. Using the wrong algorithm with the wrong data type or for the wrong purpose can produce very inaccurate results. Real world scenarios have also been provided through detailed interviews with members of the team from ZoneFox helping to provide an alternative perspective. A review of available open source tools was conducted allowing for Rapidminer and Tensorflow to be identified as the most likely tools to be used for this project.

In conclusion based on the overall findings from both chapter two and three several unsupervised algorithms including, decision trees, KNN and DBSCAN will be used in any unsupervised model proposed within this project to help predict the abnormal behaviour. A supervised regression algorithm will be implemented for any supervised sections of the project. The project will also use Rapidminer as the DM tool with Tensorflow being adopted for any ML sections of the project. The work carried out by MIT in their AI2 work shows many strengths that will be adopted for this project and a semi-supervised approach will be attempted.

4 Scenarios & Methodology

4.1 Introduction

The evolution of the proposed approach has been influenced by the findings in previous chapters. Due to the data type presented and the aim of detecting abnormal activity with no pre-defined characteristics, the decision has been made to start with an unsupervised approach as this presents the best chance of identifying abnormal behaviour from unlabelled data. Once this is done the results will be classified and an attempt to build them into a supervised algorithm implemented. The use of DM over a ML approach has been decided on as one of the more popular mediums at this stage (El Naqa, 2015). With available DM tools presenting a wider range of unsupervised algorithms to pick from the advantages of using DM over ML for the first stage of the project is clear. The first section of this chapter will look at the dataset used in detail, concluding with a detection model which will be proposed and described with the use of diagrams to provide clear understanding of the process.

4.2 Original data

One of the key challenges to this project is to ensure reliable results, this will be accomplished by working with data that was as realistic as possible. ZoneFox was able to provide invaluable support in the form of several datasets that contained an array of different scenarios all relating to insider threat. This data was captured on site by ZoneFox software over the course of three days and was presented in a raw csv format, an example of this can be found in *Appendix 6*. The original dataset consisted of three demo scenario csv files, a demo scenario description file and a data format text file, as seen in *Appendix 6*. The first task was to analyse each file to ensure clear understanding of what each files contents and functions were. A description of each file can be found in Table 1.

Table 1 Original Data Scenarios

<i>File Name</i>	<i>File Type</i>	<i>Description</i>
<i>Dataformat</i>	Text Document	This file contained the column headers of the csv files, this allowed for each column to be identified and the importance of the contents to be assessed.
<i>DemoScenarios</i>	Excel Worksheet	This file contained the different insider demo scenarios that ZoneFox had simulated and defined the abnormal behaviour.
<i>Demoscenario1</i>	CSV File	This file was made up of a selection of User and system events that had been captured over a certain time.
<i>2new</i>		
<i>Demoscenario3</i>	CSV File	This file was made up of a selection of User and system events that had been captured over a certain time.
<i>4</i>		
<i>Demoscenario5</i>	CSV File	This file was made up of a selection of User and system events that had been captured over a certain time.

The data format file allowed for each of the columns in the csv files to be identified and defined. This allows for better refinement of the dataset and helps to ensure a consistent result when testing. It was clear that a few of columns contained useless data and might negatively affect the results, to mitigate this possibility both “IndexName” and “id” columns were removed from the dataset. A complete breakdown of changes to the dataset can be found in Table 2.

Table 2 Modified Fields

<i>Header</i>	<i>Description</i>	<i>Data Type</i>	<i>Removed, Added or Kept.</i>
<i>IndexName</i>	ZoneFox identifier	Text	Removed
<i>Unique Id</i>	Unique identifier to enable outliers that were detected with original data.	Numeric	Added
<i>Doc datetime</i>	Data and time format.	Date and time (contain text)	Kept
<i>Id</i>	ZoneFox identifier.	Text	Removed
<i>Machine</i>	Machine identifier.	Text	Kept
<i>User</i>	User Name.	Text	Kept
<i>Application</i>	Application that was accessed by the user.	Text	Kept
<i>action</i>	Process that the user or system carried out.	Text	Kept
<i>Resource</i>	Resource that was accessed, file system, removable device or networked drive.	Changed from Text to Boolean (Binary)	Changed to Boolean (Binary)
<i>Outlier</i>	Identified abnormal activity based on demo scenario file.	Boolean (Binary)	Added

After this the individual demo scenario csv files were looked at and it was observed that the size of the files and the number of events in each file differed greatly, this large difference in available data had the potential to adversely affect the result as the smaller datasets may not contain enough data to allow for any meaningful patterns to be detected. To overcome this challenge, the decision was taken to combine all csv files in to one file. This had the benefit of allowing a single large csv to be used for testing, it is important to note that although this csv was deemed large when compared to the smaller csv file it was in fact small when compared to the large amounts of data that companies deal with on a daily basis. The newly combined csv was then organised by date and time to ensure that user event happened in the correct order. The next stage was to review the demo scenarios defined by ZoneFox within the demo scenario file and ensure familiarisation of the insider threats that had been simulated. The original file was a little jumbled and included scenario examples for users that were not present in the data. Due to this a new file was created to provide a consistent content, DemoScenarios_refined.csv, each scenario is defined below.

4.3 Scenarios

This section identifies the scenarios that will be used as a testing bed for the proposed approach of this project. The scenarios consist of three main groups of staff: permanent (e.g. Laura), temporary (e.g. Timmy) and third party (e.g. Colin). These users are then simulated carrying out an internal threat: data theft, accessing sensitive files or folders, accessing security software and third party software installation.

4.3.1 Data Theft

Scenario one includes the threat of data theft from employees this threat was also identified previously in the research section of this project. The scenario uses a proposed insider threat kill chain by ZoneFox, (ZoneFox, 2016). This kill chain follows a number of predefined stages that are likely to detect or highlight a data theft. Looking at staff who have handed in their notice but who begin to look around the data servers creating backups of files or attempting to transfer data to removable devices. Within this scenario, Charlotte one of the engineer staff is backing up files to a removable drive. The key data for this scenario can be found in Table 3.

Table 3 Data Theft Scenario

<i>Scenario</i>	<i>Person</i>	<i>Application</i>	<i>Resource</i>	<i>Alert Fired</i>
Scenario 1 – Data Theft, follows insider threat kill chain 1) Leaving. 2) Looking around Data server. 3) Downloading backup software. 4) Copy zip file to RM.	Charlotte - Engineer 2	bbackup.exe	rm:\\e:\\myusb\\backup.zip	File Written to RM.

4.3.2 Accessing Sensitive Files

Scenario 2 comprises the threat involved in companies who take on temporary staff over busy periods and the access these staff would need to confidential data. This scenario looks at Timmy accessing a file and folder that he does not need access to. The key data for this scenario can be found in Table 4.

Table 4 Accessing Sensitive Files Scenario

<i>Scenario</i>	<i>Person</i>	<i>Application</i>	<i>Resource</i>	<i>Alert Fired</i>
Scenario 2 – Privileged User Data Breach.	Timmy – Temp 1		nfs:\\.....\\fileshare2\\boardminutes\\minutes - february.txt	Protect Sensitive Folders – Boardminutes.

4.3.3 Security Software

Scenario 3 explores when an inside threat attempts to deactivate certain applications that are put in place to either detect, stop or protect against malicious activity. This scenario identifies Laura, a member of the sales team, deactivating the anti-virus software. The key data for this scenario can be found in Table 5.

Table 5 Security Software Scenario

<i>Scenario</i>	<i>Person</i>	<i>Application</i>	<i>Resource</i>	<i>Alert Fired</i>
Scenario 3 – Endpoint Security Processing.	Laura – Sales 1	Savservice.exe (av software)	Laura's stopping of AV software looks suspicious in conjunction with her other activities	Symantec AV Disabled.

4.3.4 Unauthorised Software Installation

Scenario 4 encompasses the threat posed to a company when staff install unauthorised or undetected software known as Shadow IT. This can prevent the company from implementing the correct safe guards. In this scenario, Rebecca, a member of the engineering team installs and uses third party software such as Dropbox. The key data for this scenario can be found in Table 6.

Table 6 Unauthorised Software Installation Scenario

<i>Scenario</i>	<i>Person</i>	<i>Application</i>	<i>Resource</i>	<i>Alert Fired</i>
Scenario 4 – Shadow IT Risk.	Rebecca – Engineer 3	dropbox.exe	c:\users\engineer3\dropbox\plan2.doc	Files backed up to cloud. Skype File Upload.
	Rebecca – Engineer 3	Skype.exe	nfs:\...\fileshare1\engineeringplans\plan1.doc nfs:\...\fileshare1\engineeringplans\plan2.doc	

4.3.5 Contractors

Scenario 5 identifies the threat companies may come across when third party contractors are employed to work directly on their network. In this scenario, Colin who works as a contractor, accesses files he is not supposed to be accessing and he also writes data to a removable drive. The key data for this scenario can be found in Table 7.

Table 7 Contractors Scenario

<i>Scenario</i>	<i>Person</i>	<i>Application</i>	<i>Resource</i>	<i>Alert Fired</i>
Scenario 5 – Contractor Data Security.	Colin – Contractor 1		nfs:\.....\fileshare1\engineeringplans\plan1.doc	Contractor accessing file they're not supposed to – Limited Contractors Access.
	Colin – Contractor 1		Rm:\f:\copyto\remdrive\ipdata.txt	File Written to RM.

4.3.6 User Behaviour

The final scenario also looks at Colin who has accessed sensitive information that he should have no need to access. This scenario shows Colin accessing a patient medical file on the network. The key data for this scenario can be found in Table 8.

Table 8 User Behaviour Scenario

<i>Scenario</i>	<i>Person</i>	<i>Application</i>	<i>Resource</i>	<i>Alert Fired</i>
Scenario 6 – Protect Sensitive Folders – Patient Data.	Colin – Contractor 1		nfs:\.....\fileshare1\PatientData\Alice BrownMedicalRecord.txt	Contractor accessing highly sensitive information relating to Professor Brown.

4.4 Proposed Methodology

As discussed in the literature review a number of possible approaches were found to help with insider threat detection, including the use of unsupervised algorithms for anomaly detection and pattern recognition, supervised ML approach and a semi-supervised approach. Based on the findings from research carried out in this project the method proposed is a semi-supervised approach, this approach will take advantage of the unsupervised DM ability to use powerful anomaly detection algorithms to highlight abnormal behaviour. This in turn can be given to a human analyst who will then be able to build this labelled data in to a supervised algorithm where it can be used to help improve the overall accuracy of the final system. The approach used by MIT, (Kalyan Veeramachaneni, 2016) provided a fundamental starting point for this project to be built on. The proposed system is constructed of two modules with each module filling a key function in the process. Due to constraints on knowledge, skills and available resources not all modules will be implemented fully within this project and a proposal submitted

for future work and possible implementation provided. Each design stage as well as the final proposed approach is clearly presented with the use of diagrams in Figure 11 - Figure 13.

4.4.1 Unsupervised Hybrid Model

The first stage of the project is to use a selection of unsupervised algorithms for pattern recognition and anomaly detection, the first module proposed is constructed of four unsupervised algorithms available within the Rapidminer DM tool. This approach aims to identify the most likely outliers and ensure a higher level of accuracy over any single algorithm. All datasets would be put through the unsupervised module and the results recorded and compared. The setup of this module is illustrated in Figure 11.

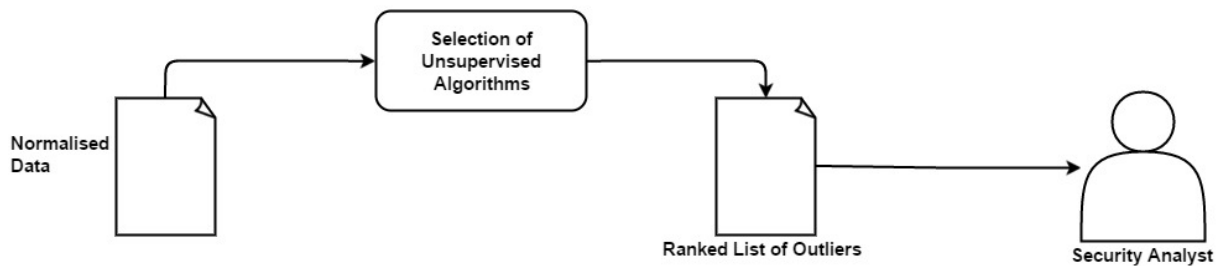


Figure 11 Proposed Unsupervised Approach

4.4.2 Supervised Model

Once the data has been processed by the unsupervised algorithm and a list of outliers has been produced. The outliers will be classified by comparing the outliers from the unsupervised results with the original data to define normal and abnormal behaviour. This stage is very important as not all outliers will be threats and an unsupervised system would not be able to define this. The now labelled data will be built in to a supervised model and the network trained with the results being recorded for later analysis. This module is illustrated in Figure 12.

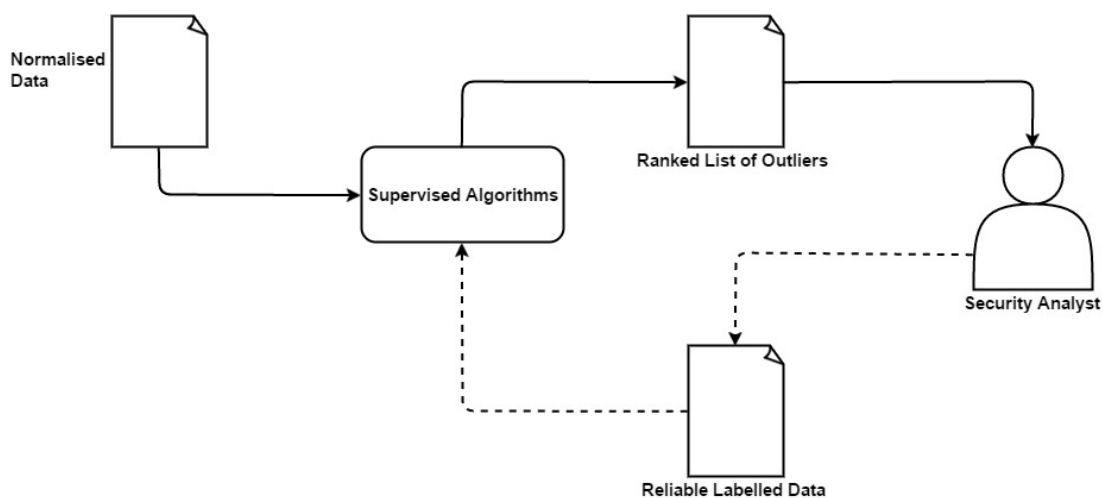


Figure 12 Proposed Supervised Approach

4.4.3 Semi-Supervised Hybrid System

The final stage is to combine the unsupervised and supervised systems to produce a semi-supervised approach. This approach will use the classified data from the unsupervised approach to produce an accurate supervised system and help to reduce the number of future outliers while improving performance and accuracy. The complete approach is illustrated in Figure 13.

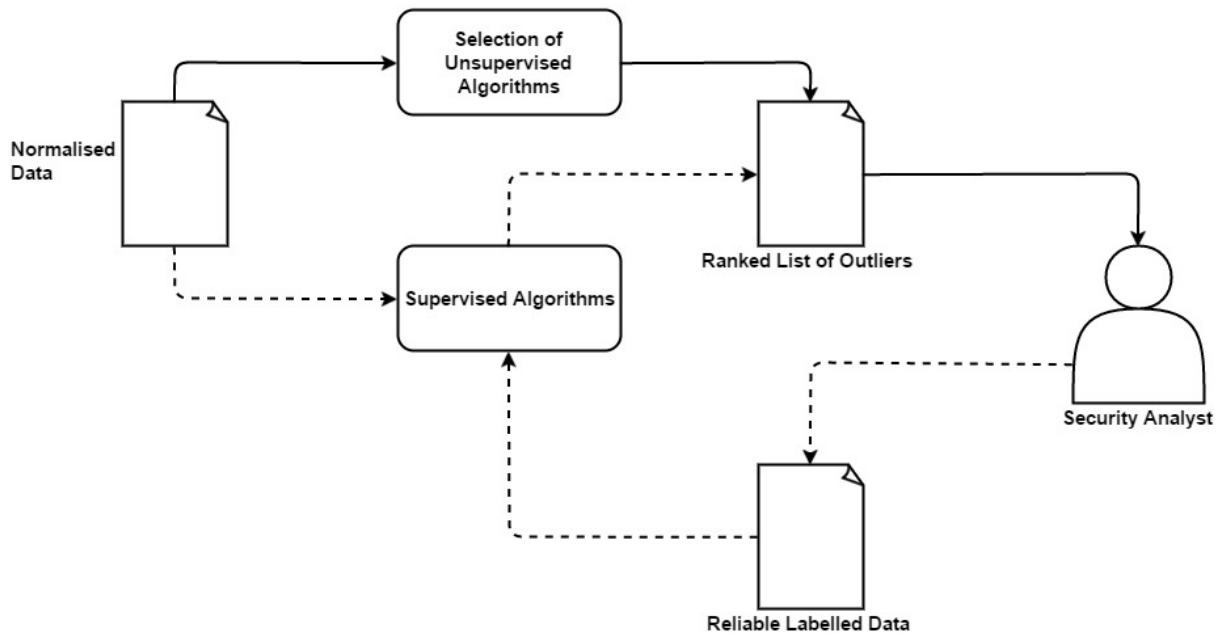


Figure 13 Proposed Semi-Supervised System

5 Data Pre-Processing

This section of the dissertation identifies a selection of pre-processing stages that will be carried out on the data provided from ZoneFox. There are several pre-processing stages that can be used to prepare data including: data cleaning, parsing, standardising, normalising and data staging. Some of these stages will be carried out before the selected dataset can be implemented into the proposed system. The first section of this chapter explores what features were selected from the original data. Next outlier identification is carried out to enable the use of a supervised approach by labelling the original data. The process includes cleaning the data to remove any unwanted features or corrupt events. The process is completed by carrying out several conversions and normalisation processes that are discussed in chapters two and three.

5.1 Feature Selection

One of the most important stages identified in chapter 3 for preparing data for use with any DL or ML process is feature selection. Ensuring the correct features are used improves the chances of achieving high accuracy when identifying abnormal behaviour. During this stage, a number of features were removed from the original data, this included ZoneFox unique identification codes and a column containing no unique event. The features that were selected are as follows:

- **Date and Time** - Would highlight if a user's work pattern had changed, a possible data theft might take place out with normal working hours.
- **Machine Id** – This enables any user that changes machine to carry out malicious activity to be detected. The user might move to another machine in hopes of hiding their identity.
- **User ID** – Allows each user to be identified.
- **Application** – Allows the algorithms to produce a pattern of usage and identify any anomalies.
- **Action** – Allows the algorithms to produce a pattern of usage and identify any anomalies.
- **Resource** - Allows the algorithms to produce a pattern of usage and identify any anomalies. This field was converted from text to a Boolean value either [1:0] as the entire path name was too long to convert into a useable numeric value given the number of possible path lengths. The value 1 is used to represent that a resource is used and 0 is used to denote that no resource is used.

5.2 Outlier Identification

One of the main challenges when trying to detect insider threat is the lack of reliable labelled data to allow for any supervised system to be trained accurately. Using the demo dataset provided by ZoneFox it is possible to create labelled data based on the scenarios discussed above and to accurately identify these events within the full csv file. This approach was looked at for a few reasons firstly running the data against both supervised and unsupervised models would allow for a comparison in results to be viewed. Secondly, as looked at in the literature review, the option of a semi-supervised approach was suggested and the possible implementation of a supervised method in conjunction with the unsupervised methods has the potential to improve detection accuracy.

To allow this option to be explored, the original full csv file was investigated and the abnormal events based on content of demo scenarios csv were highlighted, once highlighted each event unique identifier was recorded. This is to allow for the unsupervised results to be verified and a detection result to be defined. The final stage involved adding a column to identify the abnormal user events. This was achieved by assigning a Boolean value were 0: Benign and 1: Insider threat, this ensures that when used with a supervised method the system can be trained correctly. During the investigation of the dataset six users were identified including an Administrator. One of the concerns raised from this was that several of the user events were generated by the Administrator. However, the abnormal activity was not conducted by the Administrator therefore it was unclear how/if this would affect results. As a result, the decision was taken to create five additional CSV files, one for each of the following users. Contractor_1, Engineer_2, Engineer_3, Sales_1 and Temp_1. These CSV files would contain only events by that user and no other. This provided six final CSV files, a breakdown of each file is listed in Table 9. When looking at the Original_data_Sales_1.csv file user Laura is seen deactivating the security software and this action is identified as the outlier. This outlier is highlighted using the Boolean method discussed above.

Table 9 Replacement CSV Files

<i>File</i>	<i>No. Events</i>	<i>No. Outliers</i>	<i>Description</i>
<i>Original_data_Full.csv</i>	2643	33	Contains all user events captured by ZoneFox in order of date and time. Including abnormal behaviour highlighted.
<i>Original_data_Engineer_2.csv</i>	146	6	Contains only events by user Engineer_2, arranged in order of date and time. Including abnormal behaviour highlighted.
<i>Original_data_Engineer_3.csv</i>	75	5	Contains only events by user Engineer_3, arranged in order of date and time. Including abnormal behaviour highlighted.
<i>Original_data_Contractor_1.csv</i>	57	11	Contains only events by user Contractor_1, arranged in order of date and time. Including abnormal behaviour highlighted.

<i>Original_data_Sales_1.csv</i>	135	1	Contains only events by user Sales_1, arranged in order of date and time. Including abnormal behaviour highlighted.
<i>Original_data_Temp_1.csv</i>	28	10	Contains only events by user Temp_1, arranged in order of date and time. Including abnormal behaviour highlighted.

5.3 Data Conversion

One of the challenges faced in the pre-processing stage of this project was determining how the DM tool would interpret the data within the date set. The format of the data did not conform to a single type and due to this it was unlikely that the DM or ML tools would be able to process findings and produce accurate results. To overcome this a simple solution was employed, this involved converting all the data into a simple data type as the algorithms suggested for this project rely on a distance metric it was clear that only numeric values would work. The conversion stage involved a few steps as follows.

5.3.1 Date and Time Conversion

The data that ZoneFox provided contained standard date and time format that contain both numeric and text characters. The first thing that was carried out on this data was to separate the date and time. This needed to be converted into only numeric so the T and Z characters in the data needed to be removed. The T value was used to separate the date and time section of the string and the Z value denoting the zero-hour offset or Zulu time. Once this was achieved the process of converting both the date and time in to a single numeric value could begin. The decision was made that a Unix time format would be used as this allowed for quick conversion and reversal and contained only numeric values that would be easily processed in the system.

The conversion was carried out in Excel, due to the small scale of the data that was being converted it was quicker to complete the work this way than to look at developing the skills and understanding to build scripts to convert the data. In excel the data was formatted to ensure the date and time contents were correct. The next stage involved the date being converted in to Unix format this was achieved using the following formula “=(A1-DATE(1970,1,1))*86400”, (Extended Office, n.d.). A similar formula was then used to convert the time, “C1*86400”. Once both date and time were converted the two values were combined and placed in to a Unix. The date and time were converted to ensure the value represented the data correctly, if the Unix conversion was not correct then it could adversely affect the results of the experiment. A visual representation of this process can be found in Table 10.

Table 10 Date and Time Conversion

<i>Original</i>	<i>2016-02-23T16:26:33Z</i>	
<i>Step 1 - Split Data</i>	2016-02-23T	16:26:33Z
<i>Step 2 - Removal of T and Z</i>	2016-02-23	16:26:33
<i>Step 3 - Format Date and Time</i>	23/02/2016	16:26:33
<i>Step 4 - Convert to Unix Data and Time</i>	1456185600	59193
<i>Step 5 - Combine to form Final Unix Data and Time</i>	1456244793	

5.3.2 Unique Event Identification and Key Selection

One of the biggest challenges in processing the data was deciding on a method of conversion in order to best mitigate any adverse effect on the testing. To ensure this was achieved the approach was taken to identify each unique event in the data and assign it a numeric value. For each column or header in the data a range of numeric values was assigned, this would allow a consistent approach based on the type of event. For example, all users in the data would be assigned a value between 15000 and 19000. The complete range of assigned values can be found in Table 11.

Table 11 Event UID Ranges

Column contents	Range
<i>Machine Identifier</i>	15000 – 19000
<i>User</i>	1000 – 1500
<i>File Actions</i>	400 – 499
<i>System Processes</i>	200 – 399
<i>User Applications</i>	50-99
<i>System Applications</i>	20-49

Secondly, each unique event was assigned a numeric value based on the ranges above, this key can be found in Table 12.

Table 12 Assigned Event UID

Original	Replaced	Original	Replaced
2hCsuqP	16001	explorer.exe	24
4RcZBZz	16002	lsass.exe	25
6CY6z99	16003	mmc.exe	26
AV8WF5u	16004	netsh.exe	27
		rdpclip.exe	28
acmeltd__administrator	1000	savservice.exe	29
acmeltd__engineer2	1202	svchost.exe	30
acmeltd__engineer3	1203	taskeng.exe	31
acmeltd__sales1	1250	tstheme.exe	33
acmeltd__exec1	1299	process	34
acmeltd__temp1	1301	servermanager.exe	40

acmeltd__contractor1	1351	taskhost.exe	41
		taskhostex.exe	42
user logged on	201	taskmgr.exe	43
user logged off	202	werfault.exe	44
new process created	301	wermgr.exe	45
process stopped	302	wsmprovhost.exe	46
file created	401	dropbox.exe	61
file deleted	402	dropboxclient_3.14.7.exe	62
file read	403	dropboxinstaller.exe	63
file written	404	dropboxupdate.exe	64
file renamed	405	dropboxupdateondemand.exe	65
file moved	406	skype.exe	66
		notepad.exe	67
cmd.exe	20	notepad++.exe	68
configure-smremoting.exe	21	openwith.exe	69
conhost.exe	22	wordpad.exe	70
dwm.exe	23	calc.exe	71
		bbackup.exe	72
		backup4all.exe	73
		btray.exe	74

5.3.3 Event Conversion

The final stage involved carefully converting the original data with the newly assigned numeric values, this was carried out in excel using the find and replace function and allowed for all data to be found easily and also allowed for any data that was missed to be highlighted and corrected. The conversion process is shown in Table 13, where the original and converted data is compared.

Table 13 Data Conversion Process

	Data & Time	Machine ID	User ID	Application	Action	Resource
Original Data	2016-02-23T16:26:33Z	4RcZBZz	acmeltd__administrator	configure-smremoting.exe	process stopped	N/A
Converted Data	1456244793	16002	1000	21	302	0

5.4 Normalisation

As already discussed the algorithms most used for anomaly detection relies on using some form of distance metric, for this project that is likely to be some variation of Euclidean distance. As during the conversion process each event was assigned a numeric value ranging from 0 up to 19000 it is possible for the DM tool to create a bias in the results where it assigns an event as more likely to be an outlier due to its higher numeric value than an actual outlier. To overcome this problem a method of normalisation is used, this process was explored in earlier chapters and is used to ensure a consistent range of values are used for each input when being put in to the model, this is achieved by ensuring all data is ranged between 0 and 1.

This process was also completed in excel and carried out by a common formula that is used to normalise data is ' $X = (X - \text{MIN}(X - X_n)) / (\text{MAX}(X - X_n) - \text{MIN}(X - X_n))$ ' (Saitta, 2007). Using this formula allowed the large range of varying numeric values to be restricted to between [0:1]. This process was completed for each event in the data file and this was the final phase of the data processing stage, a complete example of the data processing cycle can be found in Table 14 to better illustrate the process. All of these stages were completed on the six CSV files to ensure that the final normalised data files are able to be used on the DM and ML tools to produce the most accurate results.

Table 14 Data Normalisation Process

	Data & Time	Machine	User	Application	Action	Resource
Original Data	2016-02-23T16:26:33Z	4RcZBZz	acmeltd_administrator	configure-smremoting.exe	process stopped	N/A
converted Data	1456244793	16002	1000	21	302	0
Normalised Date	0	0.333333	0	0.018519	0.492683	0

6 System Implementation

This chapter covers the implementation of the proposed model from chapter five and it will explore the tools that have been used in the testing of the proposed approach along with the algorithms that were selected and the reason behind those decisions. The first section of this chapter looks at the implementation and execution of the unsupervised DM aspect of the experiment using Rapidminer. The next section reviews the implementation and running of the supervised approach using MATLAB, the final section of this chapter looks at the challenges that were discovered and how these were overcome.

6.1 Rapidminer

Rapidminer was chosen for the unsupervised section of the project based on the findings from earlier in the report where it was highlighted that Rapidminer was one of the most popular DM tools (Immanuel, 2016). This was reinforced with the fact that the application is compatible and easily installed on the system used to carry out all the experimentation, the system details can be found in Appendix 7. Finally, the application provided a simple GUI interface and sported a large user base that had produced a large amount of detailed documentation.

6.1.1 Tool Implementation

Installation of the Rapidminer was simple and straight forward, it is possible to install the application on a selection of operating systems including Windows, Ubuntu and Apple. Initially the plan was to install a virtual system and run both ML and DM tools on it. Due to a number of challenges and a change in ML tool, the decision was taken to install directly on to Windows. The operating system details can be seen in Figure 14 this allows for a performance based line of the experiments to be established and compared when run using more powerful processors to see any improvement of performance time.

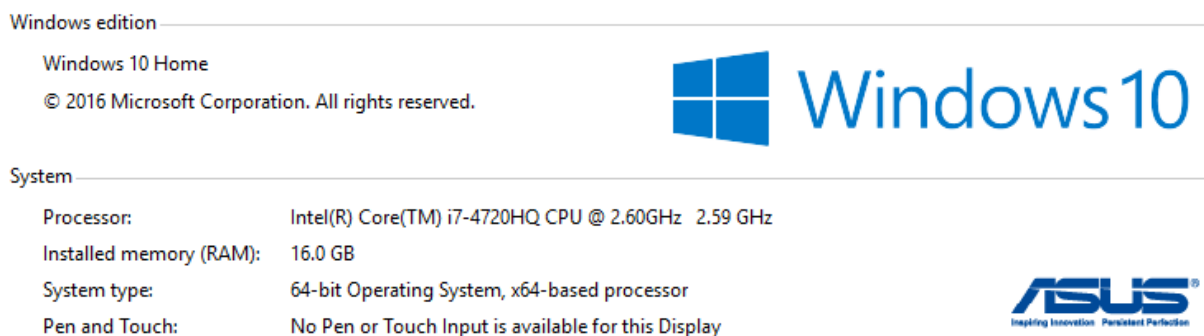


Figure 14 Operating System

The number of preinstalled algorithms for outlier detection was limited to only four, after using Rapidminer support, it was discovered that a number of additional algorithms could be added with the installation of an extension. This process was very simple and easy to manage resulting in an additional fifteen anomaly detection algorithms to select from.

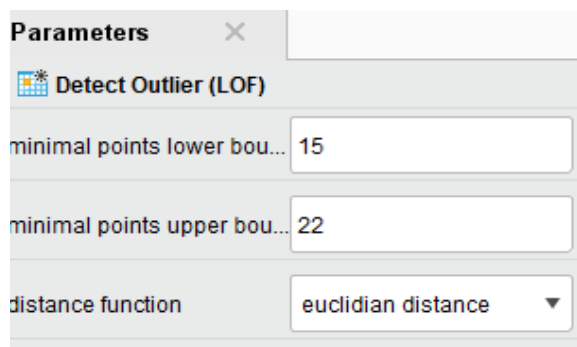
6.1.2 Algorithm Selection and Implementation

As discussed in chapter three a number of potential algorithms were selected for the unsupervised DM part of the proposed model, after the additional anomaly detection extension was installed and nine algorithms were selected and implemented. One of the algorithms that was used was Cluster-Based Local Outlier Factor (CBLOF) this algorithm works by taking the clustered data points and categorising the cluster into two classification alpha and beta based on the configurations from the user. The outlier score can then be calculated as the distance of a data point to the cluster centre. The additional algorithms can be found in Table 15 with data from (Rapidminer, 2017).

Table 15 Unsupervised Algorithms

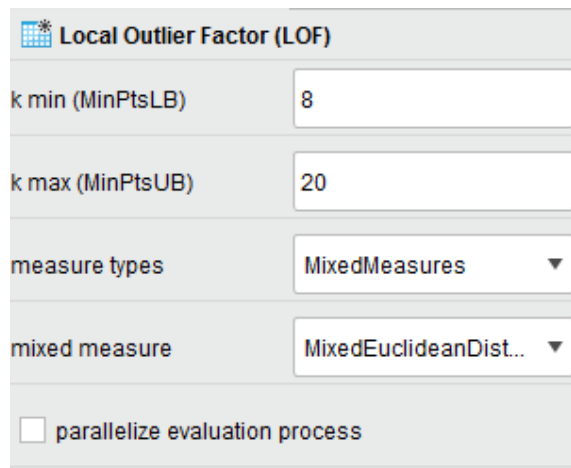
<i>Algorithm</i>	<i>Description</i>
Local Outlier Factor	Calculates the outlier score based on Local Outlier Factor implementation.
<i>Detect Outlier (Local Outlier Factor)</i>	This operator identifies outliers in the given ExampleSet based on local outlier factors (LOF). The LOF is based on a concept of a local density, where locality is given by the k nearest neighbours, whose distance is used to estimate the density. By comparing the local density of an object to the local densities of its neighbours, one can identify regions of similar density, and points that have a substantially lower density than their neighbours. These are considered to be outliers.
<i>approximate Local Correlation Integral (aLOCI)</i>	Calculates the max outlier score based on the approximate Local Correlation Integral (aLOCI).
<i>Cluster-Based Local Outlier Factor (CBLOF)</i>	Calculates the outlier score based on cluster-based local outlier factor.
<i>Detect Outlier (Distance)</i>	This operator identifies n outliers in the given ExampleSet based on the distance to their k nearest neighbours. The variables n and k can be specified through parameters.
k-NN Global Anomaly Score	Calculates the outlier score based on k-nearest-neighbours implementation.
<i>Connectivity-Based Outlier Factor (COF)</i>	Calculates the outlier score based on Connectivity Based Outlier Factor.
<i>Robust Principal Component Analysis Anomaly Score (rPCA)</i>	Computes an anomaly score based on a robust PCA estimation.
<i>Histogram-based Outlier Score (HBOS)</i>	Calculates an outlier score by creating a histogram with a fixed or a dynamic binwidth.

After some initial testing these were reduced down to the four most accurate algorithms and further refined to allow for adjustments in their configurations so as to better improve the accuracy of the results, the four selected algorithms and the final configurations can be seen in Figure 15 - Figure 19. These configuration settings can be manually changed to define how the user would like each algorithm to measure and calculate certain metrics. Such as defining the upper and lower distance limits that each data point can be from another before being defined as in the same cluster or as an outlier. Looking at Figure 18 shows that the distance limit is set to 1.3 epsilon thus defining the neighbourhood size. The min points were set to 6 denotes that a minimum number of 6 data points are needed to create a cluster.




Parameters	
Detect Outlier (LOF)	
minimal points lower bou...	15
minimal points upper bou...	22
distance function	euclidian distance ▼

Figure 15 Detect (LOF) Algorithm Configuration




Local Outlier Factor (LOF)	
k min (MinPtsLB)	8
k max (MinPtsUB)	20
measure types	MixedMeasures ▼
mixed measure	MixedEuclideanDist... ▼
<input type="checkbox"/> parallelize evaluation process	

Figure 16 Local (LOF) Algorithm Configurations

 **approximate Local Correlation Integral (aLOCI)**

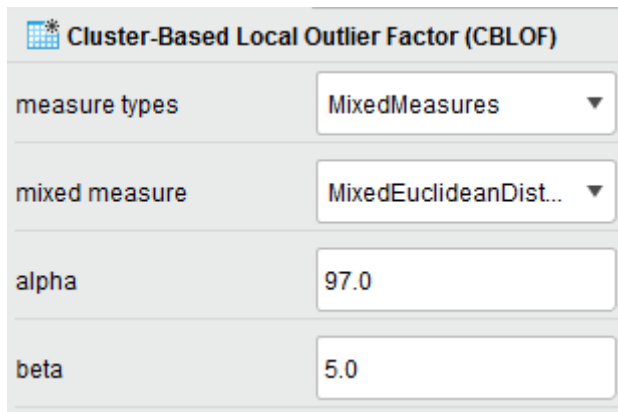
difference of levels L	<input type="text" value="8"/>
tree depth (levels)	<input type="text" value="20"/>
number of grids	<input type="text" value="20"/>
n min	<input type="text" value="20"/>
<input type="checkbox"/> parallelize evaluation process	
measure types	<input type="text" value="MixedMeasures"/>
mixed measure	<input type="text" value="MixedEuclideanDist..."/>

Figure 17 aLOCI Algorithm Configuration

 **Clustering (2) (DBSCAN)**

epsilon	<input type="text" value="1.3"/>
min points	<input type="text" value="6"/>
<input checked="" type="checkbox"/> add cluster attribute	
<input type="checkbox"/> add as label	
<input type="checkbox"/> remove unlabeled	
measure types	<input type="text" value="MixedMeasures"/>
mixed measure	<input type="text" value="MixedEuclideanDist..."/>

Figure 18 Clustering Setup Configurations



Cluster-Based Local Outlier Factor (CBLOF)	
measure types	MixedMeasures
mixed measure	MixedEuclideanDist...
alpha	97.0
beta	5.0

Figure 19 CBLOF Algorithm Configuration

The final model structure can be seen in Figure 20. This diagram shows the process direction of data through the system. Each CSV file is located on the left side of the diagram, to allow all the algorithms to run at the same time a multiplier is added. This can be seen in orange and works by simply duplicating the normalised data to be run through each algorithm in one go. This sends the data to each of the four selected algorithms identified in purple and the results are then generated for collection and interrogation. The clustering algorithm has an additional stage that is highlighted in green. This stage of the algorithm plots the data points and passes this data to the outlier detection part of the algorithm to calculate the outlier score.

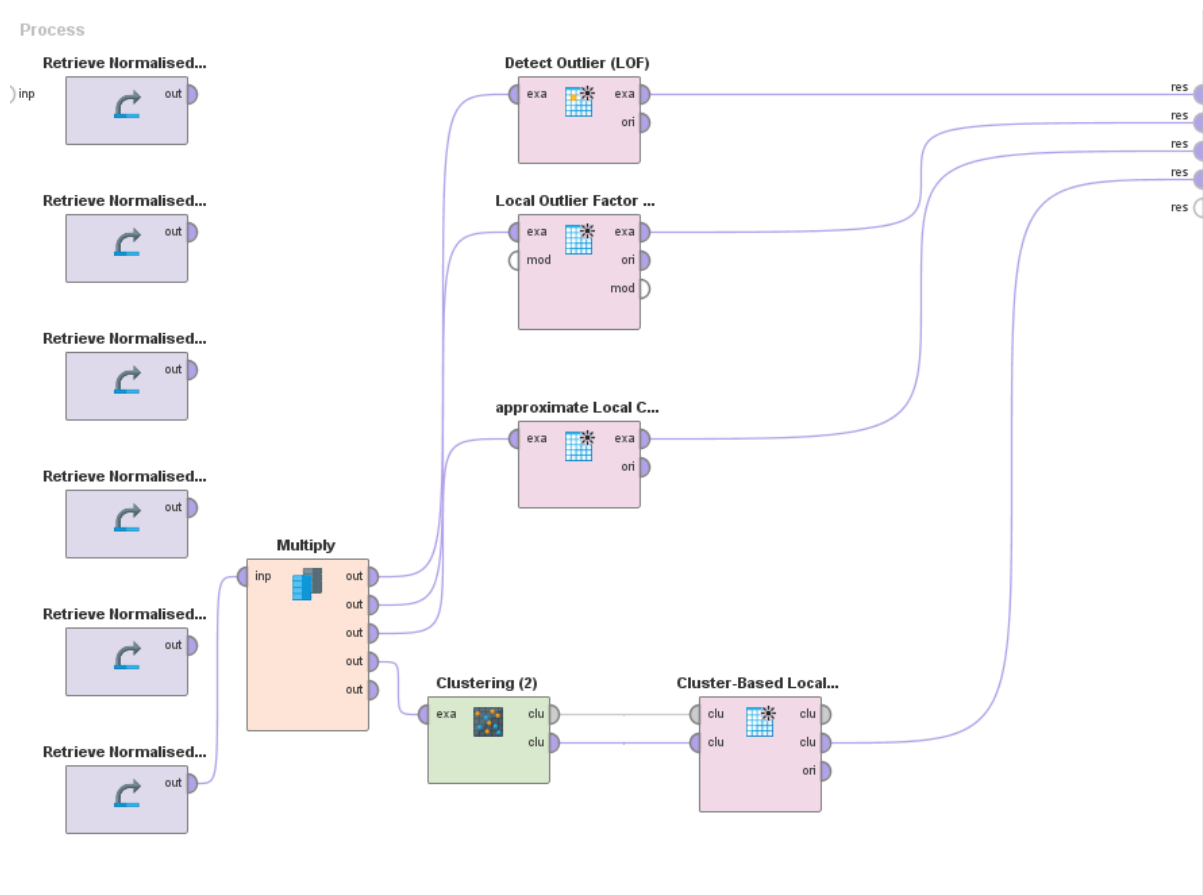


Figure 20 Unsupervised DM Process

6.1.3 Dataset Implementation

Rapidminer allows the use of standard .csv files to be imported directly and during the import process a number of modifications can be made to the data. This is a massive advantage over other DM tools such as WEKA which uses an alternative data formats such as .arff (Weka, 2008). The first section allows the removal of any header rows along with the ability to define how the columns are separated (Figure 21). The following stage allows for the formatting of columns and the removal of features that are not important for this stage of the process this is useful if the dataset contains unique identifiers or anomaly identification key, (Figure 22). Finally, the dataset is available to be used in the system and only requires being dragged and dropped in to the process area.

Import Data - Specify your data format

Specify your data format

☒ Header Row: 1

Start Row: 1

Column Separator: Comma ,

File Encoding: windows-1252

Escape Character: \

Decimal Character: .

☒ Use Quotes: "

☐ Trim Lines

☒ Skip Comments: #

1	1	0	0.333333333	0	0.018518519	0.492682927	0	0
2	2	0	0.333333333	0	0.018518519	0.487804878	0	0
3	3	0	0.333333333	0	0.037037037	0.492682927	0	0
4	4	0	0.333333333	0	0.037037037	0.487804878	0	0
5	5	7.81E-05	0	0.575498575	0.055555556	0.487804878	0	0
6	6	7.81E-05	0	0.575498575	0.259259259	0	1	0
7	7	7.81E-05	0	0.575498575	0.166666667	0.487804878	0	0
8	8	7.81E-05	0	0.575498575	0.148148148	0.492682927	0	0
9	9	7.81E-05	0	0.575498575	0.240740741	0.487804878	0	0
10	10	7.81E-05	0	0.575498575	0.148148148	0.487804878	0	0
11	11	8.20E-05	0	0.575498575	0.259259259	0.004878049	0	0
12	12	8.20E-05	0	0.575498575	0.055555556	0.492682927	0	0
13	13	8.98E-05	0	0.575498575	0.981481481	0.487804878	0	0
14	14	0.000101508	0	0.575498575	0.981481481	0.985365854	1	0
15	15	0.000101508	0	0.575498575	0.240740741	0.492682927	0	0
16	16	0.000105412	0	0.575498575	0.981481481	0.985365854	1	0
17	17	0.000105412	0	0.575498575	0.981481481	0.985365854	1	0
18	18	0.000105412	0	0.575498575	0.981481481	0.985365854	1	0

no problems.

Previous Next Cancel

Figure 21 Import Data 1 of 2

Import Data - Format your columns.

Format your columns.

Date format: MMM d, yyyy h:mm:ss a z ☐ Replace errors with missing values ⓘ

	att1 integer	att2 real	att3 real	att4 real	att5 real	att6 real	att7 integer	att8 integer
1	1	0.000	0.333	0.000	0.019	0.493	0	0
2	2	0.000	0.333	0.000	0.019	0.488	0	0
3	3	0.000	0.333	0.000	0.037	0.493	0	0
4	4	0.000	0.333	0.000	0.037	0.488	0	0
5	5	0.000	0.000	0.575	0.056	0.488	0	0
6	6	0.000	0.000	0.575	0.259	0.000	1	0
7	7	0.000	0.000	0.575	0.167	0.488	0	0
8	8	0.000	0.000	0.575	0.148	0.493	0	0
9	9	0.000	0.000	0.575	0.241	0.488	0	0
10	10	0.000	0.000	0.575	0.148	0.488	0	0
11	11	0.000	0.000	0.575	0.259	0.005	0	0
12	12	0.000	0.000	0.575	0.056	0.493	0	0
13	13	0.000	0.000	0.575	0.981	0.488	0	0
14	14	0.000	0.000	0.575	0.981	0.985	1	0
15	15	0.000	0.000	0.575	0.241	0.493	0	0
16	16	0.000	0.000	0.575	0.981	0.985	1	0
17	17	0.000	0.000	0.575	0.981	0.985	1	0
18	18	0.000	0.000	0.575	0.981	0.985	1	0

no problems.

Previous Next Cancel

Figure 22 Import Data 2 of 2

6.1.4 Results Collection

The final stage of the unsupervised model process was the result collection and interpretation. Rapidminer easily identifies the outliers in the dataset by assigning an outlier score to the record as can be seen highlighted by red in Figure 23. The higher the outlier score the more likely it is that the identified event is an abnormal behaviour.

The row number is used to identify the row from the original data that was read into the DM tool this can help the analyst to identify the action in the clear data. Rapidminer also provides the features that were used in the calculations with att2 referring to column two within the original data. This feature was not important for this project as using the outlier score and the row number was enough to compare the normalised results against the original clean data.

Row No.	outlier ↓	att2	att3	att4	att5	att6	att7
6	7376.667	0.000	0	0.575	0.259	0	1
2459	6507.919	0.352	0	0.712	0.259	0.985	1
2517	1459.002	0.643	0	0.578	0.259	0.976	1
2518	1457.907	0.643	0	0.578	0.259	0.980	1
2495	1271.543	0.643	0	0.578	0.167	0.488	1

Figure 23 Unsupervised Results

The interface then acts similar to excel and allows the user to arrange the data by any of the features in ascending or descending order. For the purposes of this project each algorithm results were sorted by outlier score from highest to lowest. To provide the most accurate results all of the .csv files were used, this involved the entire dataset being used and then the individual user datasets being processed and the results collected for comparison. Two test logs were taken with the top 20% and 10% of outliers being identified and compared from the entire dataset, all test logs can be found in Appendix 8. The row number highlighted by blue (Figure 23) shows the row that the outlier identifies and corresponds with the original data, this allows the security analyst to clearly identify abnormal activity. This can be seen in Figure 24 with the second two steps not including the outlier score as the data is traced back to the original data.

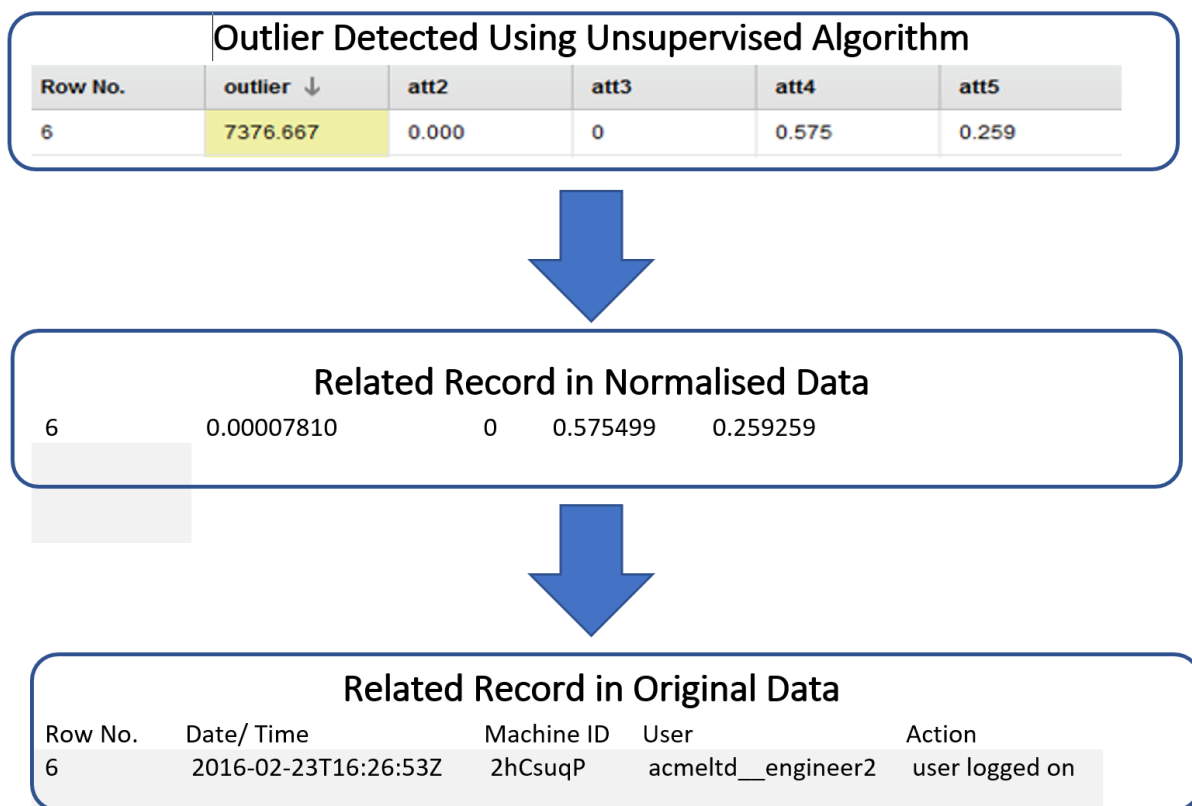


Figure 24 Outlier Decoding Process

6.2 MATLAB

MATLAB was used to carry out the supervised section of the research as the application was widely used in both academia and research fields. The addition of a simple GUI interface, online tutorials and university based expertise added to the ease of use. The advantages of using MATLAB included the fact that ML coding expertise was not needed as the tool provided a couple of preinstalled pattern recognition algorithms that the user only needs to activate. This was completely different from Tensorflow that was initially selected for this stage of the project that relied on a deep level of coding knowledge. The application allows the uses of a ML approach to be used when presented with reliable labelled data.

6.2.1 Tool Implementation

The original plan was to use Tensorflow for the ML section of the project, however due to a number of technical challenges and the need to build complex supervised ML algorithms a decision was taken to change to MATLAB. The installation process was simple and quick being installed on the same system as Rapidminer as seen in Figure 14.

6.2.2 Algorithm Selection and Implementation

MATLAB allows the use of an artificial neural network to train and test the ML algorithm against the selected dataset. For this section of the project the pattern recognition algorithm was selected as this was most likely to give accurate results. The architecture of the network can be seen in Figure 25, along with training results in Figure 26.

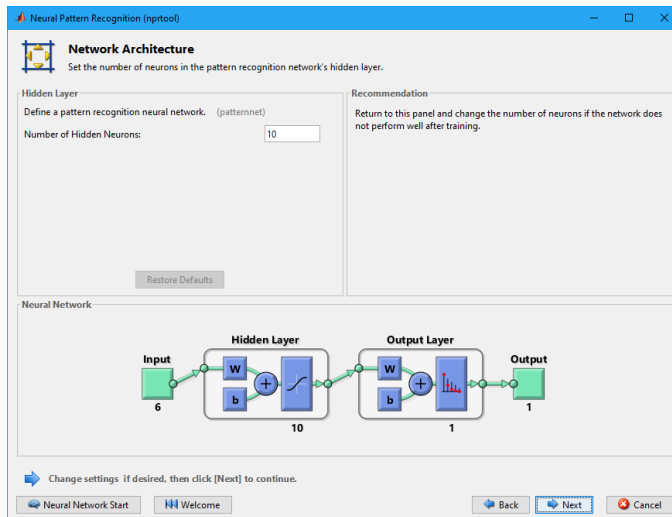


Figure 25 Network Architecture

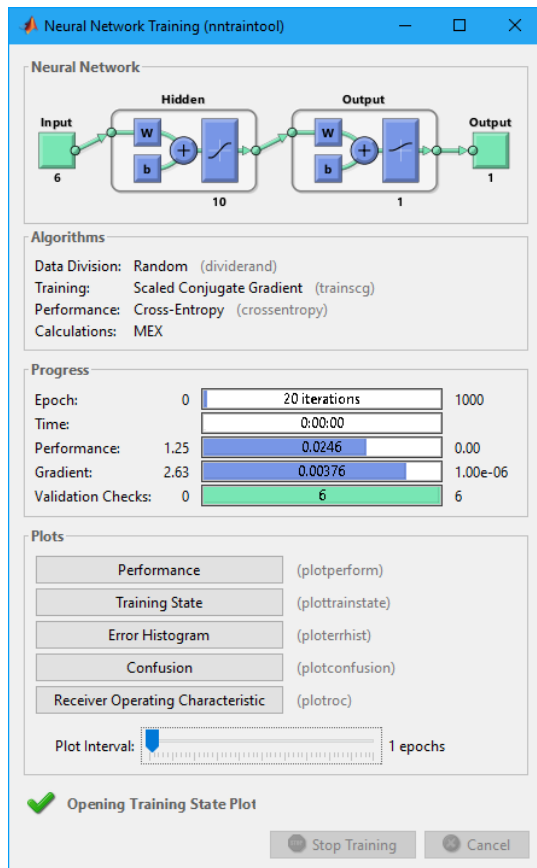


Figure 26 Network Training

6.2.3 Data Implementation

Importing data in to a MATLAB is more complex than Rapidminer as the data needs to be labelled to allow for the supervised model to classify the data and carry out predictions. This involved separating the outlier label from the main test data and reading the files in separately. This included assigning the outlier labels to Boolean from integers, this ensured the results were as accurate as possible. Data selection can be seen in Figure 27, this data was then used for the validation and testing of the network (Figure 28).

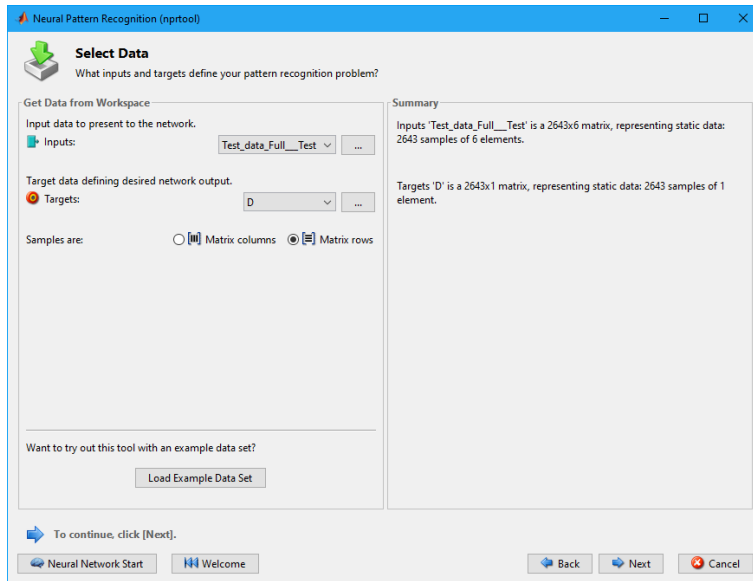


Figure 27 Data Selection MATLAB

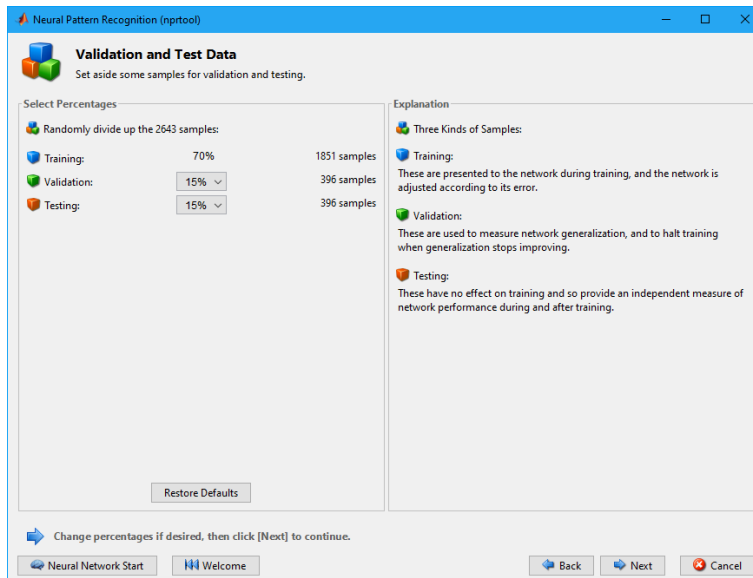


Figure 28 Data Validation and Testing Configuration

6.2.4 Result Collection

Results produced from the supervised algorithm using MATLAB consisted of several results that identify findings and network performance for all of the outputs. Including Training performance, Error Histogram and ROC, these can be found in Appendix 9. The prediction results for the network can be found in Figure 30 and display the training, validation, test and full confusion matrix.



Figure 29 Supervised Confusion Matrix

7 Evaluation

7.1 Unsupervised Results

Several testing stages were carried out during this project, this was completed for a couple of reasons. Firstly, it allowed for the algorithms to be refined and configured to produce the most accurate results. This was achieved by selecting the nine highlighted unsupervised algorithms from chapter six and four variations of the algorithms configurations being compared, an example of this test log is displayed in Figure 30. For each .csv file that was used in the testing the top 10% of the outliers were selected from the ranked list of the entire dataset. This number was chosen as it allowed a minimum of 264 outliers to be reviewed when testing the fully compiled dataset, 10% is also a reasonable number of outliers for a security individual or security team to review before adding the classified results to the supervised model.

Detect_Outlier_Distance Exec 1				k-NN_Global_Anomaly_Score Exec 1				Connectivity-Based_Outlier_Fact Exec 1				Detect_Outlier_(LOF) Exec 1				
K=2 K=5 K=10				K=2 K=5 K=10				K=2 K=5 K=10				L2, U10 L8, U10 L5, U20 L15, U20				
Outliers				Outliers				Outliers				Outliers				
5	2	7	7	5	5	5	5	5	2	2	5	5	2	N/A		
	10	11	10		7	7	7		10	10	7		7	5	N/A	
Temp 1				Temp 1				Temp 1				Temp 1				
K=2 K=5 K=10				K=2 K=5 K=10				K=2 K=5 K=10				L2, U10 L8, U10 L5, U20 L15, U20				
Outliers				Outliers				Outliers				Outliers				
3	1	17	17	3	19	19	19	3	19	19	19	3	19	19	19	19
	12	18	18		17	17	17		13	13	21		13	13	17	17
4	18	19	19	4	23	23	23	4	26	26	17	4	26	26	23	23
6	19	21	21	6	21	21	21	6	27	27	23	6	27	27	21	21
7	20	23	23	7	24	24	24	7	11	18	18	7	23	17	13	24
	24	24	24		18	18	18		2	24	24		21	23	26	18

Figure 30 Test Logs Example

The second stage of testing reduced the number of algorithms down to four, with the final stage refining the settings of those algorithms further still. This stage was carried out twice to ensure the results were accurate and consistent. A breakdown of each stage of testing results can be seen in Figure 31, along with the accuracy results. The accuracy score was generated by comparing the outliers identified by the system with the labelled data that was produced based on the demo scenarios in chapter four.

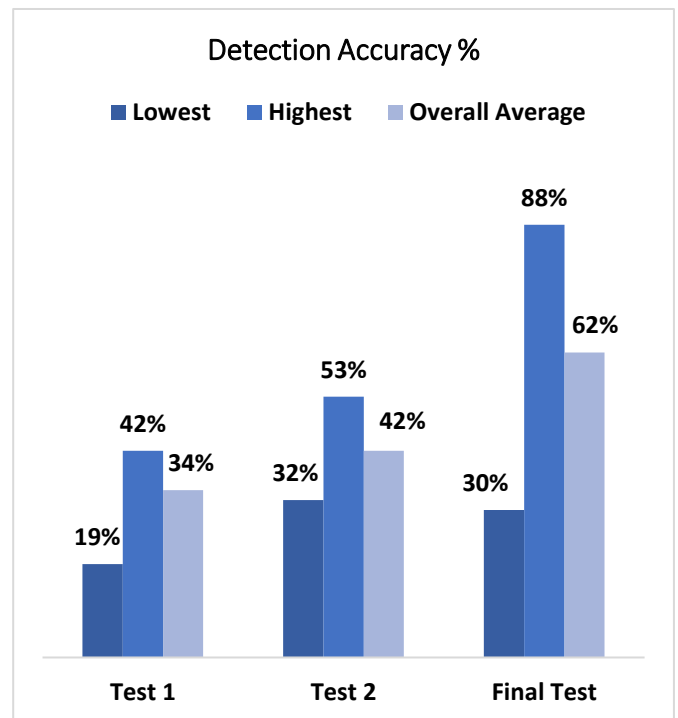


Figure 31 Overall Test Results

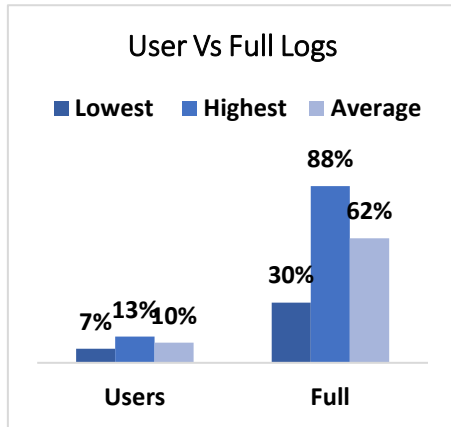


Figure 32 Average Users Vs Full Data Set

The final results from each individual user .csv file and the compiled .csv file are displayed in Figure 33, it is clear to see that the unsupervised approach produced higher levels of accuracy when run against the compiled file versus the individual user files, there are several possible reasons for this. The individual user .csv files are in comparison very small such as 28 events, it can be difficult for the algorithms to detect the pattern in such a small dataset and still limit the selection of outliers to the 10% that has been suggested for collection. This is further highlighted when the lowest, highest, and average accuracy are compared (Figure 32).

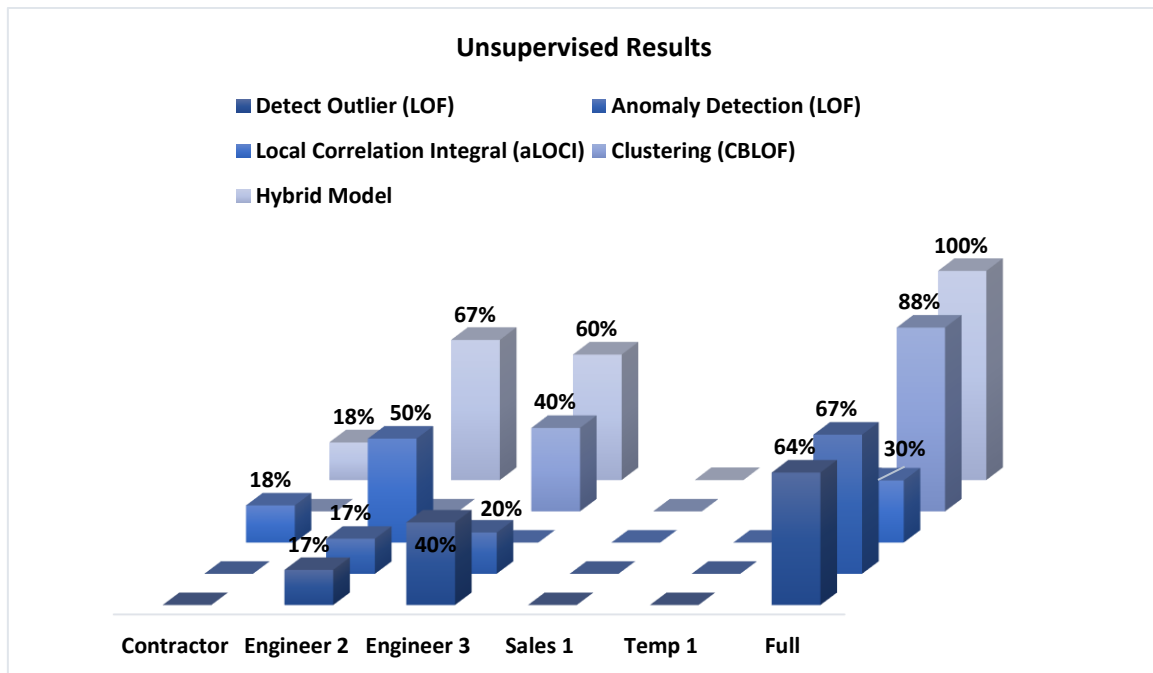


Figure 33 Compiled Dataset Vs Individual User Datasets

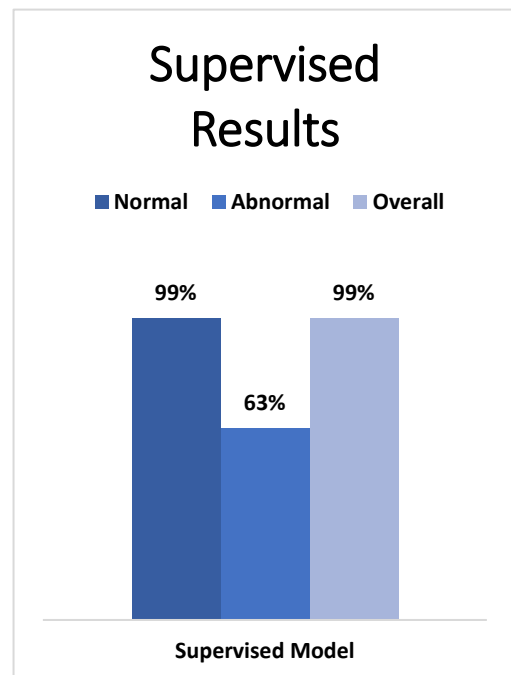
The performance of the unsupervised algorithms used is limited to a relatively small data set however this can be used in conjunction with the system information presented previously to show how changes to the hardware might impact performance. The time taken for each algorithm to be run against the entire dataset and the individual users can be seen in Table 16.

Table 16 Performance Times

<i>Algorithm</i>	<i>Entire Dataset</i>	<i>Contractor</i>	<i>Engineer 2</i>	<i>Engineer 3</i>	<i>Sales 1</i>	<i>Temp</i>
Local Outlier Factor	<1 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec
<i>Detect Outlier (Local Outlier Factor)</i>	<28 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec
<i>approximate Local Correlation Integral (aLOCI)</i>	<7 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec
<i>Cluster-Based Local Outlier Factor (CBLOF)</i>	<2 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec
<i>Complete combined system</i>	<38 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec	<1 Sec

7.2 Supervised Results

Results from the supervised pattern recognition algorithm used in MATLAB, show that the supervised section of the model was very accurate in being able to predict the normal behaviour at around 99% (Figure 34). The model however showed very poor results when predicting the abnormal outliers. The accuracy in this section peaks at around 62%, yet is as low as 30%. These results are illustrated in Figure 29. From these findings, it is possible to conclude that the current supervised model is not suitable for implementation in to the system and an alternative would need to be used. This is discussed more in the Future Work section. The inclusion of the supervised part of the model is important as the classified results produced from the security analysts and based on

**Figure 34 Supervised Results**

the outlier list generated by the unsupervised process

will ensure that duplicate outliers are removed in future, saving man hours and improving detection rates. This cannot be achieved or trusted as the supervised accuracy remains at current low levels.

7.3 Comparison

As was seen in Figure 31 the unsupervised model improved through each stage of testing. The benefits of using a hybrid approach to unsupervised insider threat detection is clear, the highest detection rate of a single algorithm was 87.9% with the hybrid approach detecting all 100% of the known threats, (Figure 35). In 2012 work published on insider threat detection using unsupervised approach resulted in a detection rate between 97.5% and 98.3%, (Pallabi Parveen B. T., 2012). Even though further testing is required it is clear to see the potential improvements on insider threat detection using the hybrid model suggested in this project.

Individual Vs Hybrid

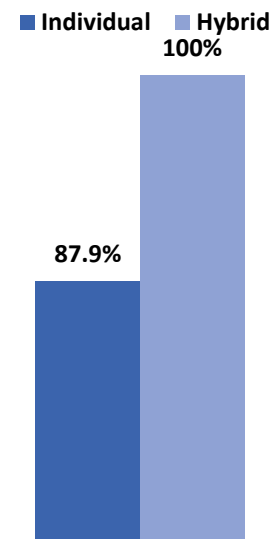


Figure 35 Single Algorithm Vs Hybrid Approach

Hybrid Results

■ Average (20%) ■ Limited (15%)
■ Limited (10%)

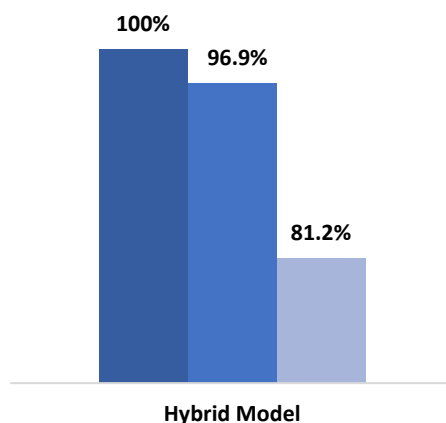


Figure 36 Hybrid Comparison

Work carried out by MIT using a semi-supervised approach similar to the one suggested in this project, showed that even by limiting the number of outliers collected to as little as 200, the accuracy can still be relatively high, with the MIT model having a 86.8% detection rate (Kalyan Veeramachaneni, 2016). Within this project, steps were taken to compare these findings as the available test dataset was only 2600 events in size. A full comparison is not possible and suggestion to improve this are presented in the Future Work section. The hybrid results were calculated by taking all the outliers from the four used unsupervised algorithms and removing any duplications. This lead to an unlimited model producing around 480 outliers, this was around 20% of the original dataset. With a human analyst

classifying and feeding this back in to a supervised model, these levels of outliers would be expected to drop with each consecutive time the system is run, this is something suggested within the Future Work section. However even with the limited dataset a comparison was

carried out to see how limiting the number of outliers collected by the hybrid model affected the accuracy. This was achieved by collecting the outliers along with their outlier score, the outlier score was then normalised using the same approach as was used in section five, the outliers from all four algorithms were then ranked based on their outlier score and duplications were removed. Finally, the top 15% and 10% of outliers were taken and compared, these findings are presented in Figure 36. It is clear to see that based on the dataset used in this project an unrestricted model would produce the highest accuracy, and the more restrictive the model is made the lower the accuracy.

8 Conclusions and Future Work

8.1 Conclusions

A detailed review for research and work in the field of insider threat detection was conducted and used to provide a starting point for the project. The works carried out by (Kalyan Veeramachaneni, 2016), (Miltiadis Kandias, 2010) and (Matt Bishop, 2008) provided the catalyst for the proposed approach that was taken within this project, the use of a semi-supervised approach in conjunction with user activity logs supplied by ZoneFox were used to produce highly accurate means of detecting protentional insider threats.

In conclusion, the main aim and objectives this project set out to achieve have been achieved.

A detailed review of the insider threat and methods currently being used to confront this threat were carried out. This showed that although many companies were starting to understand the importance of insider threat protection many still did not have any systems or policies in place to fully protect against them. It was also highlighted that using DM and ML to analyse big data is becoming more common and positively seen as a solution to how this can be done quickly.

A novel approach to insider threat detection was achieved based on findings, the research carried out from published literature helped to provide the framework for the approach presented. Several real-world examples were used as a test bed with data provided from ZoneFox. These scenarios included: data theft, installation of unauthorised software and accessing data that is out-with the user's job role. Test results showed promise, once several modifications had been carried out on the configurations for each algorithm, high levels of accuracy were achieved in the final test results. Based on the findings from this work it is clear that the hybrid approach proposed has the ability to detect more potential abnormal activity than a single algorithm. However, it is important to note that the unsupervised system can only identify potential threats and it is up to the security analyst to classify these outliers and use them to produce a supervised algorithm. Preliminary results from the supervised model were not as successful as the unsupervised results, this area is something that would potentially benefit from future work.

8.2 Future Work

The supervised approach used in the project did not perform as well as was hoped, the algorithms ability to detect abnormal activity was very low as was discussed in chapter 7. It is suggested that future work would enable possible improvements by building a supervised algorithm that would be tailored exactly for the circumstance it is needed instead of using an open source generic algorithm as available in MATLAB. With the positive results shown in the unsupervised section of the model, it is proposed that testing the system against larger datasets and archived data logs from a live company would allow for initial results to be verified in the real world. If a high level of accuracy is maintained, then a full-scale system could be built and implemented to fully complete the proposed model. Publication of this project is something that could be explored as part of the future work.

8.3 Personal Evaluation

This whole experience contained several learning experiences and I feel that I have come away with a lot of knowledge that I would otherwise not have gained. While implementing the approach that I wanted I had to overcome many challenges including hardware and software configuration issues. Being able to trouble shoot these will help if the same problems arise in future work. One of the biggest challenges involved the written sections of the work including the dissertation, poster and other hand in sections. This is due to the fact I find it difficult to adequately convey what I am trying to say in writing.

My successes include having learnt how to use a number of additional software tools including MATLAB and Rapidminer that I would otherwise never have needed to use. This will be helpful when moving in to industry as it increases my knowledge base. I was very happy with the results found from my project as it identified a novel way of using ZoneFox data to help detect insider threats. Working with ZoneFox also allowed me to work on my communication skills when dealing with a real-world client.

References

- Aruna Singh, S. S. (2014). Applying Modified K-Nearest Neighbor to Detect Insider Threat in Collaborative Information Systems. *International Journal of Innovative Research in Science, Engineering and Technology*, 14146 - 14151.
- Baldi, P. (2012). Autoencoders, Unsupervised Learning, and Deep Architectures. *JMLR: Workshop and Conference Proceedings*, 37-50.
- Brownlee, J. (2016, 12 28). *Machine Learning Tools*. Retrieved from machinelearningmastery.com: <http://machinelearningmastery.com/machine-learning-tools/>
- Brownlee, J. (2016, 03 16). *Supervised and Unsupervised Machine Learning Algorithms*. Retrieved from machinelearningmastery: <http://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/>
- Cert. (2009). *Common Sense Guide to Prevention and Detection of Insider Threats 3rd Edition*. Unknown: Carnegie Mellon.
- Cole, D. E. (2015). *Insider Threats and the Need for Fast and Direct Response*. SANS Survey.
- Copeland, B. (2016, 03 15). *Artificial intelligence (AI)*. Retrieved from Britannica: <https://www.britannica.com/technology/artificial-intelligence#ref739470>
- Dean, S. (2016, 05 05). *Open Source Projects Are Transforming Machine Learning and AI*. Retrieved from <https://www.linux.com>: <https://www.linux.com/news/open-source-projects-are-transforming-machine-learning-and-ai>
- Desale, D. (2016, 04). *Top 15 Frameworks for Machine Learning Experts*. Retrieved from kdnuggets.com: <http://www.kdnuggets.com/2016/04/top-15-frameworks-machine-learning-experts.html>
- Donalek, C. (2011). *Supervised*. California: Caltech.
- Draper, S. (2015, 10 07). *Insider Attacks Were the Most Costly Breaches of 2015*. Retrieved from Securonix: <http://www.securonix.com/insider-attacks-were-the-most-costly-breaches-of-2015/>
- El Naqa, I. L. (2015). *Machine Learning in Radiation Oncology*. Springer.
- Extended Office. (n.d.). *-excel-timestamp-to-date.html*. Retrieved 01 11, 2017, from <https://www.extendoffice.com>: <https://www.extendoffice.com/documents/excel/2473-excel-timestamp-to-date.html>
- Gartner. (2013, 05 01). *Gartner Predicts by 2017, Half of Employers will Require Employees to Supply Their Own Device for Work Purposes*. Retrieved from www.gartner.com: <http://www.gartner.com/newsroom/id/2466615>
- Gerhard Münz, S. L. (2007). *Traffic Anomaly Detection Using K-Means Clustering*. Tuebingen: University of Tuebingen.
- Google. (2016, 10 29). *tensorflow*. Retrieved from Tensorflow: <https://www.tensorflow.org/>
- Gopta, C. (2014, 10 7). *Six of the Best Open Source Data Mining Tools*. Retrieved from thenewstack.io: <http://thenewstack.io/six-of-the-best-open-source-data-mining-tools/>

- Guven, A. L. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, VOL. 18, NO. 2, 1154.
- <http://www.eugdpr.org>. (2016, 09 11). *GDPR Key Changes*. Retrieved from <http://www.eugdpr.org>: <http://www.eugdpr.org/the-regulation.html>
- IBM Security. (2015). *IBM 2015 Cyber Security Intelligence Index*. United States: IBM Corporation.
- Immanuel. (2016, 10 28). *top-free-data-mining-software*. Retrieved from [predictiveanalyticstoday.com](http://www.predictiveanalyticstoday.com): <http://www.predictiveanalyticstoday.com/top-free-data-mining-software/#comments>
- K. Mahesh Kumar, A. R. (2016). A fast DBSCAN clustering algorithm by accelerating neighbour searching using groups method. *Pattern RECOGNITION* 58, 39-48.
- Kalyan Veeramachaneni, A. C.-I. (2016). *AI2 : Training a big data machine to defend*. San Jose: MIT.
- Lonergan, K. (2016, 03 31). *Researchers find undetected insider threats in 100% of companies*. Retrieved from [information-age](http://www.information-age.com): <http://www.information-age.com/researchers-find-undetected-insider-threats-100-companies-123461176/>
- Magdalena Graczyk, T. L. (2009). Comparative Analysis of Premises Valuation Model Using KEEL, Rapid Miner, and Weka. *ICCCI* , 800-812.
- Martin Ester, H.-P. K. (1996). *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*. Munich: aaai.org.
- MATLAB. (2017, 03 20). *matlab.html*. Retrieved from <https://uk.mathworks.com>: <https://uk.mathworks.com/products/matlab.html>
- Matt Bishop, C. G. (2008). *Defining Insider Threat*. Tennessee: CSIRW.
- Michael Hahsler, M. P. (2016). *Density Based Clustering of Applications with Noise (DBSCAN) and Related Algorithms*. Maryland: CRAN.
- Miltiadis Kandias, A. M. (2010). *An Insider Threat Prediction Model*. Athens: Athens University.
- Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Science.
- Mohamed Elasmr, P. T. (2007). *Clustering Data Streams* .
- naftaliharris. (2015, 01 24). *Visualising*. Retrieved from [naftaliharris](http://naftaliharris.com): <https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/>
- ObserveIT. (2016, 09 21). *6 Steps to Protect Against Insider Threat from High-Risk Employees*. Retrieved from [ObserveIT](http://www.observeit.com): <http://www.observeit.com/blog/6-steps-protect-against-insider-threat-high-risk-employees>
- Pallabi Parveen, B. T. (2012). Unsupervised Incremental sequence Learning for Insider Threat Detection. *IEEE*, 141-144.
- Pallabi Parveen, J. E. (2011). *Insider threat detection using stream mining and graph mining*. Dallas: University of Texas.
- Pandre, A. (n.d.). *cluster analysis*. Retrieved 01 08, 2017, from [apandre](http://apandre.wordpress.com): <https://apandre.wordpress.com/visible-data/cluster-analysis/>
- Ponemon. (2016). *2016 Cost of Insider Threats*. Ponemon Institute LLC.
- R Development Team. (n.d.). *The R Manuals*. Retrieved 11 15, 2016, from <https://cran.r-project.org/>: <https://cran.r-project.org/manuals.html>
- Rapidminer. (2016, 10 29). *Data Science Behind Every Decision*. Retrieved from rapidminer.com: <https://rapidminer.com/>
- Rapidminer. (2016, 11 30). *Rapidminer Documentation*. Retrieved from [Rapidminer](http://docs.rapidminer.com): http://docs.rapidminer.com/studio/operators/modeling/predictive/trees/decision_tree_multiway.html
- Rapidminer. (2017, 03 01). *Rapidminer Tool*. Boston, USA.

- Ray, S. (2015, 10 10). *Essentials of Machine Learning Algorithms (with Python and R Codes)*. Retrieved from analyticsvidhya:
<https://www.analyticsvidhya.com/blog/2015/08/common-machine-learning-algorithms/>
- S.Gopal Krishna Patro, K. K. (2015). *Normalization: A Preprocessing Stage*. Odisha,: Department of CSE & IT.
- Saitta, S. (2007, 07 10). *Standardization vs. normalization*. Retrieved from dataminingblog: <http://www.dataminingblog.com/standardization-vs-normalization/>
- Sandhya Peddabachigari, A. A. (2004). *Intrusion Detection Systems Using Decision Trees and Support Vector Machines*. Oklahoma: Oklahoma State University.
- Standford. (n.d.). *Autoencoders*. Retrieved 01 08, 2017, from [standford.edu: http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/](http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/)
- Standford. (n.d.). *Autoencoders*. Retrieved 10 13, 2016, from <http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/>
- Statisticsfun (Director). (2012). *An Introduction to Linear Regression Analysis* [Motion Picture].
- Steinberg, X. W.-H. (2007). *Top 10 algorithms in data mining*. London: Springer.
- Sudipto Guha, A. M. (2003). *Clustering Data Streams: Theory and Practice*. Palo Alto: Stanford University.
- Sumeet Dua, X. D. (2016). *Data Mining and Machine Learning in Cybersecurity*. CRC Press.
- Teoulas. (2016, 09 16). *supervised learning,unsupervised learning ,regression*. Retrieved from Stackoverflow:
<http://stackoverflow.com/questions/22419136/supervised-learning-unsupervised-learning-regression>
- The R foundation. (2016, 10 29). *What is R?* Retrieved from www.r-project.org:
<https://www.r-project.org/about.html>
- Theano. (2016, 10 21). *theano*. Retrieved from [deeplearning.net: http://www.deeplearning.net/software/theano/](http://www.deeplearning.net/software/theano/)
- user25658. (2013, 09 23). *How to noralize data to 0-1 range*. Retrieved from stackexchange: <http://stats.stackexchange.com/questions/70801/how-to-normalize-data-to-0-1-range>
- user3415874. (2014, 04 19). *how-to-perform-0-to-1-normalization-in-excel*. Retrieved from <http://stackoverflow.com>:
<http://stackoverflow.com/questions/22952516/how-to-perform-0-to-1-normalization-in-excel>
- Verizon. (2016). *2016 Data Breach Investigations Report*. USA: Verizon.
- Waikato. (2016, 10 29). *Weka 3: Data Mining Software in Java*. Retrieved from [cs.waikato.ac.nz: http://www.cs.waikato.ac.nz/ml/weka/](http://www.cs.waikato.ac.nz/ml/weka/)
- Weka. (2008, 11 01). *ml/weka/arff.html*. Retrieved from <http://www.cs.waikato.ac.nz/ml/weka/arff.html>
- Wikipedia. (2016, 10 29). *TensorFlow*. Retrieved from wikipedia.org:
<https://en.wikipedia.org/wiki/TensorFlow>
- Witten, I. (2016, 10 27). *WekaMOOC*. Retrieved from youtube.com:
<https://www.youtube.com/user/WekaMOOC>
- Xindong Wu, X. Z.-Q. (2014). Data Mining with Big Data. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 97-108.
- Yangqing Jia, E. S. (2016, 10 29). *Caffe*. Retrieved from <http://caffe.berkeleyvision.org/>: <http://caffe.berkeleyvision.org/>

Zhao, Y. (2016, 10 28). *tools*. Retrieved from rdatamining.com:

<http://www.rdatamining.com/resources/tools>

ZoneFox. (2016). *Introducing-the-insider-threat-kill-chain/*. Edinburgh: ZoneFox.

ZoneFox. (n.d.). *platform-features/*. Retrieved 03 16, 2017, from <https://zonefox.com>:

<https://zonefox.com/platform-features/>

Appendices

Appendix 1

Initial Project Overview

[Martyn Brown]

[40135758]

Initial Project Overview

SOC10101 Honours Project (40 Credits)

- ▲ **Title of Project: Assessing the viability of using machine learning in conjunction with windows system logs to detect anomalous user behaviour.**

Overview of Project Content and Milestones

This project is to assess the viability of taking Windows system log files that will be produced in a virtual environment and put these through some form of machine learning or data mining tool so as to detect patterns in user behaviour and see if that can be used to detect abnormal activity. This will be done in a number of sections. First is to produce a detailed timeline including a Gantt chart that will have all dates and milestones. My initial time line is below, please note that a number of the time to completions will run at the same time and will be clearly defined in the Gantt chart: -

- Detailed plan, 1 week
- Research stage will be ongoing however initially will take 3 weeks
- Decision on user profiles for the simulated environments, 1 week
- Production on simulated environments and user activity, 4 – 8 weeks to collect enough usable data.
- Research on available tools and implantation of possible tools for development. 4-8 weeks
- Extraction of the training data, 1 week
- Testing stage of using the data collected and the prototype system, 2 weeks
- Analysis of finding, 1 week
- Production of final report with all findings and documentation, 2 weeks

The Main Deliverable(s):

The project has a number of deliverables which will build in to my main deliverable of producing a report on the viability of using machine learning to detect abnormal user behaviour, my hope is that as each stage is completed and the findings match my initial hypothesis I will be able to use these findings to continue and produce my final results.

My Main deliverable: -

- Final report on the Viability of the findings.

The primary deliverables include

- Labelled training data set.
- Report on findings of research already completed in the area.
- A prototype system to take the training data set and give a result.
- An analysis on the results vs the labelled training data.

•

The Target Audience for the Deliverable(s):

The results from this project can be used to help in future research or development at not only an academic level but also an industry level. Although I can't predict the complete final target audience, I have included my main target audiences that I had in mind when I proposed the project. My hope is that at the end of my research I will have a prototype system that can be implemented easily to a client system or edited easily to accept different feature selection.

1. Future academic research / Supervisor projects.
2. Organisations / Security services / Cyber Security Teams.
3. Open source research.

The Work to be Undertaken:

I initially will be carrying out detailed research and investigation in to what academic and industry papers are available already in the subject area. This will help to identify any dead-end research and give me a starting point to potentially compare results. Next I will produce a detailed plan of what I am going to need to do and specifications on how this will be achieved along with a timeline.

My proposed approach at this stage (which may need to be amended ~~at a later date~~) is to create a number of virtual machines and collect a couple of weeks' worth of system log data. This will serve as the training / test data. While this is happening, I will research tools to help me process and analysis my data.

Next the data will be processed and put through one of the tools that I have found to hopefully detect the known abnormal behaviour. This step will be repeated to ensure consistent results.

Finally, I will evaluate my finding and produce a report along with documentation on all my findings and any prototype that I manage to build, also my proposal for future work.

Additional Information / Knowledge Required:

As machine learning is not a subject I have ever covered in my studies I believe there will be ~~a number of~~ new knowledge, skills and tools I will need to gain so as to be able to produce a 1st grade project. These will include gaining an intermediate understanding of machine learning. Learning how to capture and use the Windows system logs, building on my current understanding of virtual systems, different OS and the installation and running of new software. I will also need an understanding of current research and findings in this field.

Information Sources that Provide a Context for the Project:

<https://zonefox.com/>

<http://www.btplc.com/>

[Martyn Brown]

[40135758]

The Importance of the Project:

Although machine learning and AI technology has been around for a number of years, it has taken a more centre stage place lately as advanced persistent threats (APT) and sophisticated attacks become more popular. Along with the ever-growing number of connected devices make it impossible for a human analyst to detect, find and stop attacks, leaks or threats in a real-time environment. I believe that an autonomous system that can sit in the back ground and detect hidden patterns that find abnormal behaviour of users is very important. This field is still in its infancy and is not common place to see or hear of these systems in general use.

The Key Challenge(s) to be Overcome:

Given my experience in this field I have a couple of expected challenges and plans to overcome these or to minimise the effect they have on the final results. The production of a bespoke prototype to process the data is my main concern and my hope is to be able to learning enough code to write or to use current open source code to achieve this. If this becomes too difficult or time restrictive, I will look at current systems that can process the data for me and compare the results against expected findings.

Appendix 2

Week Nine Formal Review Report

SOC10101 Honours Project (40 Credits)

~~**SOC10102 Honours Project (60 Credits)**~~ (please delete one)

Week 9 Report

Student Name: Martyn Brown

Matriculation Number: 40135758

Programme (and any specialisation):

Supervisor: Naghmeh Moradpoor

Second Marker: Gordon Russell

Date of Meeting: 16.11.2016

Can the student provide evidence of attending supervision meetings by means of project diary sheets or other equivalent mechanism? yes no*

If not, please comment on any reasons presented

- to change 2 week meetings to weekly meeting

Please comment on the progress made so far

- literature review should be improved
- the narrative is a little weak
- Introduction: should be improved

Is the progress satisfactory? yes no*

Can the student articulate their aims and objectives? yes no*

If yes then please comment on them, otherwise write down your suggestions.

- Objectives need to be pinned out
- Evaluate
- objectives are limited (need more broad)

* Please circle one answer; if no is circled then this must be amplified in the space provided

Does the student have a plan of work? yes no*

If yes then please comment on that plan otherwise write down your suggestions.

- There are a lot of long bars with no caption

~~copy~~

Does the student know how they are going to evaluate their work? yes no*

If yes then please comment otherwise write down your suggestions.

- A little bit narrow at this stage
- general: priority

Any other recommendations as to the future direction of the project

- number & quality references should be improved
-

Signatures: Supervisor *Naghib* Second Marker *Lythwell*
Student *MT*

Please give the student a photocopy of this form immediately after the review meeting; the original should be lodged in the School Office with Leanne Clyde

* Please circle one answer; if no is circled then this must be amplified in the space provided

Appendix 3

Diary Sheets & Project Management

EDINBURGH NAPIER UNIVERSITY

SCHOOL OF COMPUTING

PROJECT DIARY

Student: Martyn Brown

Supervisor: Naghmeh Moradpoor

Date: Thursday 6th October

Last diary date: N/A

Objectives:

- Start literature review and bring in for feedback at next meeting.
- Speak to Bobby Souter about creating a virtual environment on the cloud to capture data sets.
- Create a detailed action plan along with Gantt chart.
- Seek feedback from Russell on IPO.

Progress:

- Literature review has been started however not as far along as I would have liked this is due to misunderstanding the structure needed.
- Meet with Bobby and now have an active environment on the cloud.
- Action plan complete.
- Emailed Russell about feedback still waiting on response will follow up if not heard in a reasonable time.

Supervisor's Comments:

We agree to:

- ~~to~~ clarify literature review
- to start writing up the literature review
- the papers are relevant to the project topic

→ It seems that the student is on the right path.

NM

EDINBURGH NAPIER UNIVERSITY

SCHOOL OF COMPUTING

PROJECT DIARY

Student: Martyn Brown

Supervisor: Naghmeh Moradpoor

Date: Wednesday 23rd November

Last diary date: Wednesday 16th November

Objectives:

- Take feedback from midterm meeting and update work accordingly.
- Format in the correct way to meet report standard
- Change writing style to 3rd person
- Redo introduction and structure properly along with expanding the information
- Set more precise timeline in the project Gantt chart
- Move meeting to weekly
- Set date for lit review to be completed
- Make aims and objective more measurable

Progress:

- Report has been formatted the correct way and broken in to chapters to be compiled as completed.
- Now written in 3rd person
- Meetings have been moved to weekly every Wednesday at 1pm
- Propose date of the 7th of December for the complete lit review
- Introduction redone and almost complete background needs to be finished
- Aims and Objectives complete, giving more clarity
- Project Gantt chart New timeline not completed yet and will be added to next weeks Objectives

Supervisor's Comments:

EDINBURGH NAPIER UNIVERSITY

SCHOOL OF COMPUTING

PROJECT DIARY

Student: Martyn Brown

Supervisor: Naghmeh Moradpoor

Date: Thursday 12th January

Last diary date: Wednesday 14th December

Objectives:

Review current work and receive feedback.
Discuss data set from ZoneFox
Tech Review to be completed for net meeting with final draft of Lit review.
A possible start on Chapter 4

use feedback to make changes
to lit review and improve.

Progress:

Work on Lit review 90% finished just waiting on response from ZoneFox and changes from feedback.
Demo Data Set received from ZoneFox

Supervisor's Comments:

- Excellent progress compared with the previous report.
- Introduce & lit Review chapters to receive from Martyn

EDINBURGH NAPIER UNIVERSITY

SCHOOL OF COMPUTING

PROJECT DIARY

Student: Martyn Brown

Supervisor: Naghmeh Moradpoor

Date: 6th April 2017

Last diary date: 30th March 2017

Objectives:

- Submit draft of dissertation for feedback

Progress:



- Dissertation at a stage that is suitable for feedback.

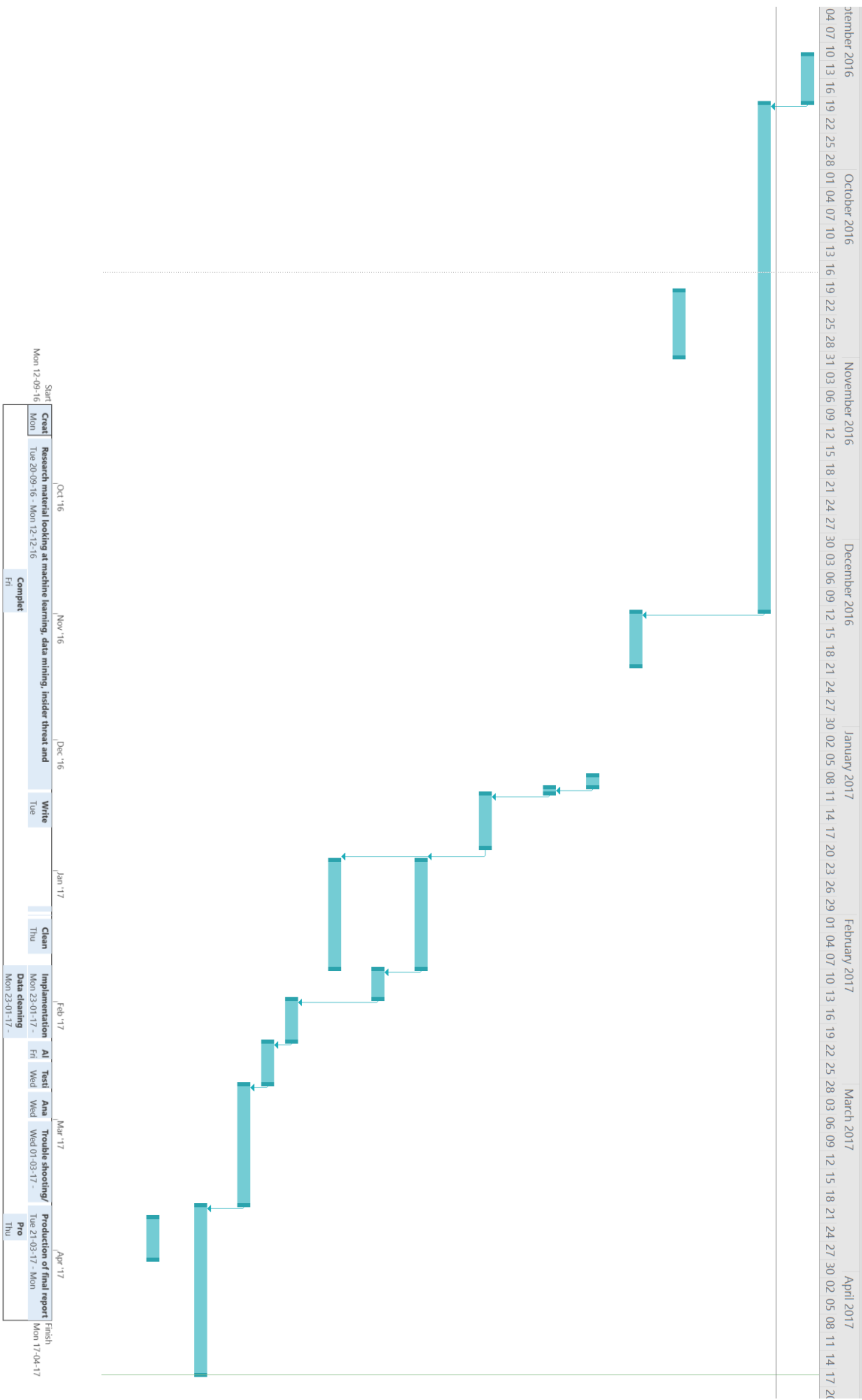


Supervisor's Comments:

Feedback provided.

Martyn Brown – BEng (Hons) Computer Security and Forensics

Task Mode	Task Name	Duration	Start	Finish	Predecessors
✦	Create detailed action plan	6 days	Mon 12-09-16	Mon 19-09-16	
✦	Research material looking at machine learning, data mining, insider threat and methods of detection	60 days	Tue 20-09-16	Mon 12-12-16	1
✦	Complete week 9 report for second marker	7 days	Fri 21-10-16	Mon 31-10-16	
✦	Write up a final literature review	7 days	Tue 13-12-16	Wed 21-12-16	2
✦	Talk to Zonefox about using dataset	2 days	Mon 09-01-17	Tue 10-01-17	
✦	Meet with ZoneFox and discuss the dataset provided along with analyse of scenarios	1 day	Wed 11-01-17	Wed 11-01-17	5
✦	Clean and organise the ZoneFox data to ensure its use within the testing stage	7 days	Thu 12-01-17	Fri 20-01-17	6
✦	Implamentation Data mining and machine learning tools.	14 days	Mon 23-01-17	Thu 09-02-17	7
✦	Algorithm implementation	3 days	Fri 10-02-17	Tue 14-02-17	8
✦	Data cleaning and Normalisation	14 days	Mon 23-01-17	Thu 09-02-17	7
✦	Testing and result extraction	5 days	Wed 15-02-17	Tue 21-02-17	9
✦	Analysis of results	5 days	Wed 22-02-17	Tue 28-02-17	11
✦	Trouble shooting/ Retesting	14 days	Wed 01-03-17	Mon 20-03-17	12
✦	Production of final report	20 days	Tue 21-03-17	Mon 17-04-17	13
✦	Produce poster presentation	5 days	Thu 23-03-17	Wed 29-03-17	



Appendix 4

Matt ZoneFox Conversations

Hi Martyn,

Sorry for the delay in getting back to you, I have just returned from a couple of day's holiday.

Are you attending the event at Napier on Monday run by Bill? I believe that there is a workshop showing people how to get up and running with ZoneFox. I would suggest that you attend that and then we can talk about what other support we can offer you going forward.

Is that ok?

Cheers

Matt

Matt Little, CTO

ZoneFox
www.zonefox.com

tel: 0131 618 2925

twitter: @zonefox

On 10 November 2016 at 15:36, Brown, Martyn <40135758@live.napier.ac.uk> wrote:

Afternoon Matt,

I am not sure if you will remember me, my name is Martyn Brown. I worked on the visualisation project that Napier done for ZoneFox last year.

I am writing to you today as I have been talking with Bill Buchanan about the possibility of using the ZoneFox software that Napier currently has on their cloud, to help with my honors project. I am looking at using machine learning in conjunction with windows system logs to predict abnormal user behaviours.

I remember that the unlabelled data that you gave us during the visualisation project was perfect for this and is what gave me the idea of using your software. I am planning on creating a virtual system with active user accounts that I will use to imitate activity on the system and I am hoping that I would be able to use your software to extract these user events in a CSV format that I can then feed in to the prototype system that I complete and compare the results with the labelled data.

I was talking to Bobby about using the software and he informed me that a member of your team was out to install the software on the cloud, I was wondering if it would be possible if they could come back out and provide some training on how to use the software correctly ?

I would be happy to share any of my findings at the end of the project.

Many thanks

Martyn

Martyn Brown – BEng (Hons) Computer Security and Forensics

Sent Items

Hi Matt,

That is perfect.

Thanks,

Martyn

From: Matt Little <m.little@zonefox.com>

Sent: 10 January 2017 07:57:02

To: Brown, Martyn

Subject: Re: Honours Project

Hi Martyn

I'm fine with you using that quote.

Best of luck with your honours!

Regards

Matt

On 9 January 2017 14:35:46 "Brown, Martyn" <40135758@live.napier.ac.uk> wrote:

Hi Matt,

I hope you are well? I wanted to thank you for putting me in touch with Kate as she has been a great help with my project. I remember last year when we met at your office to discuss the work ZoneFox had done and I wanted to reference some of that conversation in my honours work, if that is OK. I have included a copy of the content for your approval. I have kept all the sensitive details out.

"Several conversations with Kate Robson and Matt Little from ZoneFox, provided some real-life examples of the type of insider threats that have been seen already. Some of the findings from this exchange included, that insider threats are not limited to a single industry or staff type. Although individuals with a gripe are known to be a threat, the majority of incidents are financially motivated. An example that the team at ZoneFox has worked on previously was a large engineering company who worked on innovative technology. One of the high-level members of staff were in the process of changing employer and from the surface the work they were conducting was completely normal. However, ZoneFox were able to prove that the member of staff was creating backups and stealing proprietary data. This example illustrates the easy that individuals in positions of trust can fool others and carry out malicious activities. It also indicates that no matter how secure a system is perceived to be, someone can still breach it when they are on the inside and meant to be on your side."

Many Thanks

Martyn

Appendix 5

Kate ZoneFox Conversations

Re: Catch up



Kate Robson <k.robson@zonefox.com>

Wed 04/01, 10:48

You ↵

↩ Reply |

Hi Martyn,

Happy New Year.

The Napier version currently uses demo data. There isn't a huge amount of user events but it would be a good start and would probably help you test your model as I know what types of scenarios are within the data ie files written to removable etc. I already have these as CSV files as the data is uploaded nightly. Would you like me to zip these up and send them to you?

I'm also in the office all the rest of the week if you want to pop in?

Kate

On 29 December 2016 at 13:03, Martyn Brown <martynbrown1987@hotmail.co.uk> wrote:

Hi Kate,

I hope you had a nice Christmas and if you don't read this till the new year, A Happy New Year!

Apologies for not getting in touch sooner, with exams before we broke up things were a little hectic. I wanted to continue our talk from our meeting at Napier, as I said I am looking to use the ZoneFox software already on the cloud system, for my Honours project. I am wanting to log and extract user events in to CSV format to then put through a data mining/ Machine learning algorithm to detect abnormal behaviour.

Would it be possible to arrange a time for me to come and talk this over and answer any questions you might have?

Martyn Brown – BEng (Hons) Computer Security and Forensics

1. Do you feel that companies understand the scale of insider threats?

No, I don't. They are far more aware of external threats such as ddos attacks etc. Insider "incidents" aren't always maliciously done

but can have an even more devastating effect than external attacks as the data being accessed can be of a highly sensitive nature.

There also seems to be a false sense of security - "oh our employees wouldn't do that" - but often it can be accidental rather than malicious.

I've also heard companies say that they know where every bit of their data is going only to find that there are applications being downloaded and used without their knowledge and are often quite shocked at the data leaving their organisation.

Companies also don't want to be seen as having a problem. There needs to be a culture change in business that shows doing something about your security stance is a positive step, it's not highlighting you have a problem, on the contrary, it is highlighting that you are taking steps to reduce the threats faced by your organisation.

2. Have you seen a change in approaches from industry to deal with insider threats?

In the past year we have had a huge increase from companies interested in knowing where their data is going.

I also think things like GDPR (with hefty fines, and having to declare breaches within 72 hours) for data breaches will have an impact and increase companies interest in being able to track and audit data movement.

3. What is the biggest challenge of detecting an insider threat?

The problem is you don't always know what you are looking for, so you have to look at everything. This amount of data is difficult to assimilate in a timely and coherent manner. This is where UBA comes into play, as rules won't be required, to highlight potential threats.

The problem areas - "the needle in the haystack"

- amount of data generated by an organisation (and it's storage)

- how to query that data effectively and quickly

- ability to reduce false positives (any method has to give you useful information you can act on and relay to others) otherwise admins switch off and ignore the important info.






Please be aware these are my personal thoughts and not Zonefox thoughts.

On 31 January 2017 at 10:03, Martyn Brown <martynbrown1987@hotmail.co.uk> wrote:

Hi Kate,

Appendix 6

ZoneFox Dataset

	dataformat	Text Document	1 KB	No
	demoscenario5	Microsoft Excel Comma S...	2 KB	No
	demoscenario12new	Microsoft Excel Comma S...	14 KB	No
	demoscenario34	Microsoft Excel Comma S...	4 KB	No
	DemoScenarios	Microsoft Excel Worksheet	9 KB	No

A	B	C	D	E	F	G	H
1	events.usr.alias	2016-02-23T16:33:50Z	Vhos9bF9R5M7emSUqBFU2P	2hCsuqP	acmelt_d_engineer2	savservice.exe	process stopped
2	events.usr.alias	2016-02-23T16:30:26Z	2TZdl2V8pthG3x4ea6ELlx	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
3	events.usr.alias	2016-02-23T16:30:26Z	KnzuC1VY5Aumx6uHrr2zUu	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
4	events.usr.alias	2016-02-23T16:30:26Z	NSPhTo5L5PaIXv3UKhodJS	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
5	events.usr.alias	2016-02-23T16:30:26Z	JZ1Gw6YFWNaxPZ88UWPEKA	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
6	events.usr.alias	2016-02-23T16:30:26Z	7reSbGx58ILH2SaPvRTEMN	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
7	events.usr.alias	2016-02-23T16:30:26Z	XE7ZkzA8XhYQwsu1MPvic7	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
8	events.usr.alias	2016-02-23T16:30:26Z	Wfo1UJzwh8VRjdCpRam3N	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
9	events.usr.alias	2016-02-23T16:30:28Z	CPHNARHGT4nuMrWazp17ca	2hCsuqP	acmelt_d_engineer2	backup4all.exe	file written
10	events.usr.alias	2016-02-23T16:30:28Z	9aq8tL1S1qJK15MecCqLoh	2hCsuqP	acmelt_d_engineer2	backup4all.exe	file read
11	events.usr.alias	2016-02-23T16:30:33Z	EAAW12KnXg4EHWWXbbYFFY	4RcZBZz	acmelt_d_administrator	configure-smremoting.exe	process stopped
12	events.usr.alias	2016-02-23T16:30:27Z	DApp28AxZPGvLF1yrWSqznK	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
13	events.usr.alias	2016-02-23T16:30:27Z	XZ2Mc6VdUzUjfumXB7aarv	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file created
14	events.usr.alias	2016-02-23T16:30:27Z	Vr9J7b3KNrtLpiPRHf6XsC	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file read
15	events.usr.alias	2016-02-23T16:30:27Z	BMXdqKeajLMxgz6daTwpqe	2hCsuqP	acmelt_d_engineer2	bbackup.exe	file deleted
16	events.usr.alias	2016-02-23T16:26:33Z	EAAW12KnXg4EHWWXbbYFFY	4RcZBZz	acmelt_d_administrator	configure-smremoting.exe	process stopped

Appendix 7

System Information

Item	Value
OS Name	Microsoft Windows 10 Home
Version	10.0.14393 Build 14393
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	GAMING
System Manufacturer	ASUSTeK COMPUTER INC.
System Model	G751JL
System Type	x64-based PC
System SKU	ASUS-NotebookSKU
Processor	Intel(R) Core(TM) i7-4720HQ CPU @ 2.60GHz, 2594 Mhz, 4 Core(s), 8 Logical ...
BIOS Version/Date	American Megatrends Inc. G751JL.202, 20/01/2015
SMBIOS Version	2.7
Embedded Controller Version	255.255
BIOS Mode	UEFI
BaseBoard Manufacturer	ASUSTeK COMPUTER INC.
BaseBoard Model	Not Available
BaseBoard Name	Base Board
Platform Role	Mobile
Secure Boot State	On
PCR7 Configuration	Binding Not Possible
Windows Directory	C:\Windows
System Directory	C:\Windows\system32
Boot Device	\Device\HarddiskVolume1
Locale	United Kingdom
Hardware Abstraction Layer	Version = "10.0.14393.206"
User Name	Gaming\Gamer
Time Zone	GMT Standard Time
Installed Physical Memory (RAM)	16.0 GB
Total Physical Memory	16.0 GB
Available Physical Memory	12.1 GB
Total Virtual Memory	18.3 GB
Available Virtual Memory	13.8 GB
Page File Space	2.38 GB
Page File	C:\pagefile.sys
Hyper-V - VM Monitor Mode E...	Yes
Hyper-V - Second Level Addres...	Yes
Hyper-V - Virtualization Enable...	Yes
Hyper-V - Data Execution Prote...	Yes

[illegible]

2nd Test Log				
	Detect Outlier (LOF)	Anomaly Detection	Approximate Local Correlation	Clustering
Scenario 1 & 2				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	8	11	6	10
Within 3	13	23	6	23
Set	13	13	13	13
Accuracy	62%	85%	46%	77%
Engineer 2				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	4	0	4	3
Within 3	5	3	1	8
Set	7	7	7	7
Accuracy	57%	0%	57%	43%
Exec 1				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	1	1	0	
Within 3	0	1	0	1
Set	1	1	1	1
Accuracy	100%	100%	0%	0%
Temp 1				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	0	0	0	0
Within 3	0	1	1	1
Set	4	4	4	4
Accuracy	0%	0%	0%	0%
Scenario 3 & 4				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	3	3	2	5
Within 3	7	6	2	4
Set	11	11	11	11
Accuracy	27%	27%	18%	45%
Engineer 3				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	6	6	0	5
Within 3	5	5	6	4
Set	10	10	10	10
Accuracy	60%	60%	0%	50%
Sales 1				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	1	1	1	0
Within 3	3	3	1	1
Set	1	1	1	1
Accuracy	100%	100%	100%	0%
Scenario 5				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	3	3	6	0
Within 3	3	3	3	2
Set	8	8	8	8
Accuracy	38%	38%	75%	0%
Contractor 1				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1.3, 6
TP	3	3	0	6
Within 3	2	2	4	1
Set	8	8	8	8
Accuracy	38%	38%	0%	75%

3rd Test Log				
	Detect Outlier (LOF)	Anomaly Detection (LOF)	Approximate Local Correlation	Clustering
Scenario 1 & 2				
Full	L15, U20	L8, U20	8/20/20/20	CBLOF 1,3,6
TP				
Within 3				
Set	13	13	13	13
Engineer 2	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	7	7	7	7
Exec 1	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	1	1	1	1
Temp 1	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	4	4	4	4
Scenario 3 & 4				
Full	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	26	26	26	26
Engineer 3	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	16	16	16	16
Sales 1	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	28	28	28	28
Scenario 5				
Full	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	14	14	14	14
Contractor 1	L15, U20	L8, U20		CBLOF 1,3,6
TP				
Within 3				
Set	12	12	12	12

Final Test Logs

Test 10%					
Detect Outlier (LOF)	Anomaly Detection (LOF)	Approximate Local Correlation	Clustering	Hybrid	
ZoneFox Data Set					
Contractor 1	L15_U20	L8_U20	8/120/20/20	EBLOF 1.3	Hybrid
TP	0	0	2	0	2
Within 3	2	2	2	0	4
No Outliers	11	11	11	11	11
Accuracy	0%	18%	0%	0%	18%
Engineer 2	L15_U20	L8_U20	8/120/20/20	EBLOF 1.3	Hybrid
TP	1	1	3	0	4
Within 3	2	2	2	0	4
No Outliers	6	6	6	6	6
Accuracy	17%	17%	50%	0%	67%
Engineer 3	L15_U20	L8_U20	8/120/20/20	EBLOF 1.3	Hybrid
TP	2	1	0	2	3
Within 3	3	2	0	2	3
No Outliers	5	5	5	5	5
Accuracy	40%	20%	0%	40%	60%
Sales 1	L15_U20	L8_U20	8/120/20/20	EBLOF 1.3	Hybrid
TP	0	0	0	0	0
Within 3	1	1	0	1	1
No Outliers	1	1	1	1	1
Accuracy	0%	0%	0%	0%	0%
Temp 1	L15_U20	L8_U20	8/120/20/20	EBLOF 1.3	Hybrid
TP	0	0	0	0	0
Within 3	0	0	0	0	0
No Outliers	10	10	10	10	10
Accuracy	0%	0%	0%	0%	0%
Outlier Selection					
CSV	Records	Selected %	Selected	Selected %	Selected
Full	2643	10%	264	10%	264
Contractor 1	57	20%	12	20%	12
Engineer 2	146	20%	30	20%	30
Engineer 3	75	20%	16	20%	16
Sales 1	135	20%	28	20%	28
Temp 1	28	20%	6	20%	6



