Detection of insider threats using data science strategies on Carnegie Mellon University’s Insider Threat Test Data Set for a medium-sized business using extracted features

Tobias Hendricks

University of Wisconsin - Oshkosh

DS 785: Capstone

Professor Alex Korogodsky

April 30, 2023

Abstract

Insider threats, which represent a significant and costly risk to organizations, are challenging to detect, and data science can be successfully used to find and diagnose these issues. Previous research has focused on detection effectiveness using various algorithms with little focus on the features and their impact on the detection. Knowing the feature importance using ensemble decision tree algorithms and an extensible forensic methodology, which compiles and evaluates these features, provides business value by mitigating risk and formulating response strategies. Using labeled data from Carnegie Mellon University Insider Threat Test Dataset, the XGBoost algorithm is able to accomplish this complex detection goal while Isolation Forest struggled to produce meaningful results without producing large numbers of false detections. Based on these results, organizations collecting logs and using insider threat knowledge sets can generate and customize features for detections and risk mitigations.

# Contents

[Abstract 2](#_Toc133698538)

[Contents 3](#_Toc133698539)

[List of Tables 5](#_Toc133698540)

[List of Figures 6](#_Toc133698541)

[Chapter 1: Introduction 7](#_Toc133698542)

[Background 7](#_Toc133698543)

[Problem Statement 9](#_Toc133698544)

[Research Motivation 9](#_Toc133698545)

[Purpose of the Study 10](#_Toc133698546)

[Research Questions 11](#_Toc133698547)

[Definition of Special Terms 12](#_Toc133698548)

[Limitations of the Study 12](#_Toc133698549)

[Significance 14](#_Toc133698550)

[Organization of the paper 15](#_Toc133698551)

[Chapter 2: Literature Review 16](#_Toc133698552)

[Introduction 16](#_Toc133698553)

[Insider threat definition 17](#_Toc133698554)

[Detection of an insider threat 18](#_Toc133698555)

[Real-life scenarios 21](#_Toc133698556)

[Conclusion 22](#_Toc133698557)

[Chapter 3: Methodology and Analysis 24](#_Toc133698558)

[Introduction 24](#_Toc133698559)

[Research design 27](#_Toc133698560)

[Population and Sample 35](#_Toc133698561)

[Bias Handling 36](#_Toc133698562)

[Data Collection 37](#_Toc133698563)

[Data Analysis 39](#_Toc133698564)

[Tool Descriptions 42](#_Toc133698565)

[Conclusion 45](#_Toc133698566)

[Chapter 4: Results 47](#_Toc133698567)

[Introduction 47](#_Toc133698568)

[Summary of Results to Research Questions 47](#_Toc133698569)

[Summary of Results for Algorithm Comparison 49](#_Toc133698570)

[Supervised Learning 51](#_Toc133698571)

[Unsupervised Learning 57](#_Toc133698572)

[Chapter 5: Conclusions 63](#_Toc133698573)

[Summary 63](#_Toc133698574)

[Conclusions 64](#_Toc133698575)

[Recommendations and Future Direction 65](#_Toc133698576)

[References 67](#_Toc133698577)

[Appendix 71](#_Toc133698578)

[Code repository 71](#_Toc133698579)

# List of Tables

[Table 1: File sizes of release 3.1 data set 13](#_Toc133698580)

[Table 2: Core row properties in features dataframe 40](#_Toc133698581)

[Table 3: Quantities of classes after splitting data into training and testing sets. 49](#_Toc133698582)

[Table 4: Summary of evaluated algorithms using test split of data set. 50](#_Toc133698583)

[Table 5: List of optimized hyperparameters tested for XGBoost. 51](#_Toc133698584)

[Table 6: Table describing falsely labeled insider threats. 55](#_Toc133698585)

[Table 7: Table describing undetected insider threat userDates. 56](#_Toc133698586)

[Table 8: List of optimized hyperparameters for Isolation Forest 57](#_Toc133698587)

[Table 9: Prediction and score for insiders detected from Isolation Forest. 59](#_Toc133698588)

# List of Figures

[Figure 1: Changes in roles at DTAA 9](#_Toc133698589)

[Figure 2: Process of acquiring a feature from data set 27](#_Toc133698590)

[Figure 3: Distribution of activity from Accountant role in logon data file 30](#_Toc133698591)

[Figure 4: Process for algorithm evaluation 42](#_Toc133698592)

[Figure 5: Example Decision Tree from XGBoost model for Insider Threat Detection 52](#_Toc133698593)

[Figure 6: Confusion matrix from best model development using XGBoost 53](#_Toc133698594)

[Figure 7: Feature importance from XGBoost best model 54](#_Toc133698595)

[Figure 8: Timeline of insider threat detections 56](#_Toc133698596)

[Figure 9: Distribution comparison between normal (1) and anomalous (-1) userDates by Isolation Forest score. 59](#_Toc133698597)

[Figure 10: Contamination hyperparameter as compared to accuracy of insider detections. 60](#_Toc133698598)

[Figure 11: Waterfall chart depicting feature importance of insider (-1) anomalous detection from Isolation Forest. 61](#_Toc133698599)

[Figure 12: Waterfall chart of feature importance for normal userDate from Isolation Forest - for contrast of anomaly 62](#_Toc133698600)

# Chapter 1: Introduction

## Background

Security researchers have prominently featured insider threats as a significant threat to businesses for many years. An adapted definition provided by the Software Engineering Institute for Insider Threat is “the realized potential for a provisioned individual to negatively affect an organization maliciously or unintentionally through their access” (2023). These threats have large impacts on businesses as well. Insider threats are presented to businesses in numerous ways (Cybersecurity and Infrastructure Security Agency, n.d.):

* + External organizations and malicious actors frequently solicit insiders to perform nefarious acts.
  + Insiders can willfully perform acts of sabotage, espionage, theft, or other criminal acts.
  + Insiders also present accidental and negligent threats when performing normal job activities.

Prevention and detection of these risks, along with avoiding the considerable monetary impact, produce business value.

Insider threats are challenging to discover due to a variety of factors (Clark, 2018). There is typically no single event that can detect an insider threat. Detections are accomplished by identifying the combined anomalous behaviors. Further, malicious insiders typically have increased knowledge of the environment and often have no desire to be caught. Fortunately, numerous compliance controls now exist requiring that businesses collect computer activity logs of users in the environment (Center for Internet Security, 2021). These audit logs aid in discovering malicious insiders but present complexity due to differences in log sources and the volume in which they are generated. Developing detection use cases for insider threats is complicated as many logs establishing this behavior are dispersed throughout numerous applications and log stores and have no common format.

Due to the sensitive nature of this data, businesses are not inclined to share this data due to the risk of breach and privacy concerns. Carnegie Mellon University (CMU) Software Engineering Institute (SEI) has developed a data set that can be used to test detection mechanisms (Lindauer, 2020) and mimics the logs that a medium-sized business collects. This data set presents some of the same challenges faced by medium-sized businesses. The data set is composed of records from multiple applications generated from 1000 employees over a 500-day period. Only a couple (2) of these employees present insider threat behaviors.

The data set generated by the fictitious medium-sized business (dtaa.com) is composed of a single business unit with 22 departments, 37 teams, and 41 roles. Some of these roles generate proprietary business information. While the business starts with 1000 employees, the number of employees remaining with the company at the end of the dataset’s time is reduced by 103. The company has 953 computer assets, with many shared. Nine (9) IT administrators manage the company’s technical assets and provisions access to these users. During the time-period examined by the dataset, there were no role or supervisor changes. The business generates logs for web, email, device, file, and logon activity. The business also has psychometric profiles for each of its employees.

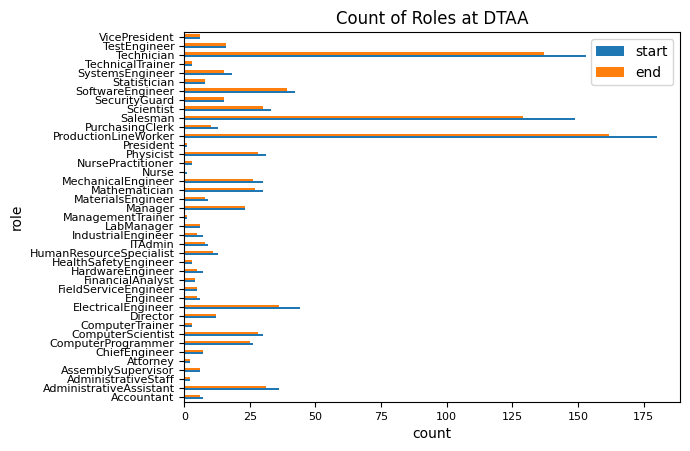


Figure 1: Changes in roles at DTAA

## Problem Statement

The threat of insiders poses a significant risk to businesses, as demonstrated by numerous examples in recent history. Insiders are particularly challenging to detect using traditional cybersecurity methods due to their authorized access to systems and data. While data science techniques have been explored for insider threat detection, other papers have not detailed the features used nor the importance the features have imparted on detections. This knowledge gap can leave businesses unsure of how to act on detection results. Therefore, this study provides example features used in data science-based insider threat detection with feature selection and demonstrates their effectiveness in detecting insider threats; empowering businesses to quickly respond to potential threats with confidence in their actions.

## Research Motivation

My interest in insider threats stems from my experiences as a security analyst for 20+ years. During this time, the perimeter defenses of most organizations have been eroded by cloud-based services used by organizations. One of the results of this change in security philosophy is the more substantial risk posed by an insider. As this threat increased and commodity technical controls started to fail, my interest in detecting insider threats and the utility of data science in the cybersecurity domain grew.

This project aims to assist businesses in detecting insider threats using what has been taught in the University of Wisconsin’s Data Science program. This project has combined my substantive cybersecurity expertise with communication, business value, and data science technical skills. This combination is frequently illustrated as a Venn diagram and has been mentioned through the program’s coursework (Conway, 2010). In providing better insights into the behavior of an organization’s individuals, security analysts will be able to quickly determine the risk an insider might represent. In understanding this risk, analysts can allocate analytical resources and prioritize their workload better; increasing the overall security posture of the organization and reducing the risk present. Medium-sized businesses will be able to automate the function of detecting insider threats performed by security analysts and minimize their impact on a workforce already encountering massive labor shortages ((ISC)², 2022).

Early detection of insiders may also lessen the business' costs associated with loss and remediation. These types of analyses will help to form an insider threat program for cybersecurity departments. Detection of threats, such as those presented by insiders, forms an integral part of multiple compliance standards.

## Purpose of the Study

This project looked specifically at the insider threat scenarios published as part of the Insider Threat Dataset and is specifically tailored to meet the course objectives. The project translated the course objectives into several targeted objectives:

* Engineered, examined, and cleaned data to be utilized by algorithms predicting the probability of an individual being an insider threat during a certain time period. Generated and evaluated features from the CMU data sets that mimic the security personnel’s analysis when diagnosing insider situations. Calculate these features in the context of a medium-sized business wishing to avoid proprietary data loss and brand value harm.
* Using the skills gained through the data science program, programmatically determined insider threat detections based on the behaviors logged from activity at a medium-sized business. Generate value in interpreting these detections to minimize diagnostics, forensics, and impact.
* Critically compared models and determined the merits and accuracy of each model and how businesses might utilize them. Optimized these models toward insider threat detection by selecting and developing the most influential features.
* Communicate the business value and risks of this task to decision-makers and analysts to mitigate the risk and act to address future situations. This project has been submitted and accepted as a conference presentation to the Health Sector – Information Sharing and Analysis Center’s (HS-ISAC) Spring Summit. Alternate audiences may also be included.

## Research Questions

This paper will evaluate and answer multiple questions:

* + How effective are data science strategies at detecting insider threats in the context of logs generated from a medium-sized business?
  + Are ensemble decision trees algorithms, like XGBoost and Isolation Forest, able to detect insider threats?
  + Which data set features are the best indication of the user being an insider threat?
  + After the analysis is performed, how well might a medium-sized business be able to interpret the results of the detection and analysis?
  + Can a forensic methodology be used in detecting insider threats?

## Definition of Special Terms

* + isInsider – The dependent variable associated to all row items in the data set. The value of this variable is different based on the algorithm used to detect insider threats. It takes the value of one (1) for insider threats in XGBoost and minus one (-1) in Isolation Forest.
  + stringDate – The date in a string format matching yyyymmdd. The stringDate is used to merge various data sources and compile the features together.
  + userDate – The unique combination of each user employed at the beginning of the examined period (1000) with the number of dates in the examined period (516) - generating 516000 rows, this string is used as the means of segregating features and labeling isInsider.

## Limitations of the Study

The data set used in this project (Lindauer, 2020) was released in 2020 but was constructed with the methodology published in 2013 (Glasser & Lindauer). This data seeks to represent a realistic 1000-employee company (dtaa.com) for 516 days (about 1 and a half years). The authors of this data acknowledge it was difficult to compare this data with data produced from actual companies. However, based on my experience as a cybersecurity professional, this data follows some of the same collections that real-life companies might retain.

The data set also models a limited number of insider threat scenarios. The scenarios portrayed in the dataset were collected and ratified with the assistance of counter-intelligence experts. However, modeling all potential insider threats would be complex since analysts may model threats differently for different organizations. Additionally, the authors of the data set recognize that it is difficult to draw comparisons between the real-life scenarios modeled in the data set and how the data set illustrates this scenario.

While the data is time-oriented, the focus of this project will be analyzing the content and taking a forensic aspect to detect insider threats. Another opportunity might exist in playing the content back to find insiders. This replay methodology would be more beneficial to the business as the detection would occur in real-time. From this forensic viewpoint, the project will benefit from understanding the entirety of user behavior during their employment.

This dataset is arranged around releases and data sets. Each release is composed of compressed comma separated values (CSV) files from multiple data sources. Uncompressing this release file requires considerable storage space. Processing each of the CSV files also requires substantial computing power. For these reasons, the project will concentrate on analyzing the files of one release – release 3.1. Release 3.1 is a compressed archive 3.95 GB in size composed of seven (7) data sources with the largest data source uncompressing to 12 GB.

Table 1: File sizes of release 3.1 data set

|  |  |  |  |
| --- | --- | --- | --- |
| Data source | # of Files | Columns in data | Size |
| device | 1 | Id – unique identifier for device item  Date – sting of date and time  User – username performing action  Pc – computer in which device has action  Activity – connect or disconnect | 29MB |
| email | 1 | Id – unique identifier for email  Date – string of date and time  To – address(es) email is sent  From – address email is originated  Size – size of email in bytes  Attachments - number of file attachments  Content – word summary of email content | 1.05 GB |
| file | 1 | Id – unique identifier for file  Date – date of file creation  User – username creating file  Pc – computer where file was created  Filename – name of file  Content – work summary of file | 224 MB |
| http | 1 | Id – unique identifer for http session  Date – date of http transaction  User – username of generating http traffic  Pc – computer where http traffic originates  Url – web resource accessed during http activity  Content – work summary of the url web site | 12 GB |
| ldap | 22 | Employee\_name – written name of the employee  User\_id – username of the employee  Email – email address of the employee  Role – role fulfilled by employee  Business\_unit – the number ‘1’  Functional\_unit – number indexed unit of business. 6 unique  Department – number indexed department of user. 22 unique  Team – number indexed team of user. 37 unique  Supervisor – name of supervisor | 2.44 MB |
| logon | 1 | Id – unique identifier for logon/logout  Date – string date and time of activity  User – username of activity  Pc – computer location of activity  Activity – Logon or Logout | 56 MB |
| psychometric | 1 | Employee\_name – written name of employee  User\_id – username of employee  O - Openness  C - Conscientiousness  E - Extraversion  A - Agreeableness  N - Neuroticism | .5 MB |

## Significance

The employment climate after the COVID-19 pandemic has led to a wide-scale change in the relationship between employers and employees (Fischbein, 2022). This change, coupled with increasing job dissatisfaction, has contributed to a rise in an organization’s insider threat incidents – each with significant costs (Ponemon Institute, 2022). The Ponemon Institute found the average cost to resolve an insider threat incident to be $645,997 for the recent fiscal year (2022). This monetary impact on a medium-sized business would have operational consequences. This study has identified behavioral and characteristic features and algorithms to be used in a forensic methodology to identify insider threats from a synthetic data set. This methodology, the algorithms considered, and the adapted features, combined with other security controls, can be used by other businesses to identify and reduce the risk present from insider threats.

## Organization of the paper

The paper is organized into five chapters that align with the course competencies:

* Chapter 1 - Introduction: This chapter aims to provide a working knowledge of insider threats and the project's objectives.
* Chapter 2 – Literature Review: Conduct and communicate previous research performed on insider threat detection. Perform a review of papers surrounding anomaly discovery in a cybersecurity context.
* Chapter 3 – Methodology and Analysis: This chapter describes how the data was gathered, explored, and formatted for detection of insider threats. Descriptions of how algorithms and tools were used to perform detection are also included.
* Chapter 4 – Results: Using appropriate visualizations, results from the project are displayed for communications to the business. Results are associated with the research questions with remarks on that analysis and expected outcome.
* Chapter 5 – Conclusions: Final project summary and what future steps might be taken.

# Chapter 2: Literature Review

**Introduction**

This literature review aims to understand the types of research presently being performed around detecting insider threats and how this research might impact the research questions. In understanding the research trends around this topic, this paper has expanded on this knowledge and validated the existing research against the CMU dataset and the present state of cybersecurity.

The volume of insider threat research since 2010 has produced a significant amount of information towards using data science to detect insiders by various means. Many of these papers align to one of the research questions posed by this paper. A portion of this research is also similar to the topics of this paper. Due to the amount and depth of these resources, only the most relevant articles have been reviewed for this literature review.

This significant increase in interest and knowledge is due to several factors. One of these factors is the rise in popularity of sites like WikiLeaks. WikiLeaks and websites like it allow insiders to reach a larger audience more easily without significantly increasing the risk of revealing their identity. As leaked posts from insiders increased, funding to help prevent and detect these threats also increased. This research became more prominent due to several high-profile whistleblower cases, including Edward Snowden and Chelsea Manning.

Another factor driving increased research into insider threats is the prominence of remote working after the 2020 pandemic. Organizations no longer have the same amount of defense that might be exercised with onsite controls (Samy et al., 2021). Malicious external actors leverage this lack of technical controls to create Unintentional Insider Threat (UIT) situations, where an outside agent uses an insider to create negative circumstances for the organization. Remote work also hampers the ability of an organization to monitor their employees’ activities, contributing to more frequent accounts of insider threats.

Businesses are also impacted by post-epidemic issues where insider threats are likely to occur. Many organizations are performing and facing layoffs. These layoffs exacerbate insider threat issues and make it more likely that a threatening situation from an insider is realized (Jennex et al., 2022). Employees and contractors feel entitled to the work they accomplished with the company and will seek a means to exfiltrate these resources. Compounding this issue in the aftermath of the pandemic, the “great resignation” has increased the number of employees leaving their present working situation. Both circumstances increase the risk to businesses and their current means of detecting insider threats.

Since funding for research on insider threats has increased, researchers produced data sets such as the one used for this paper. Authors can then use these datasets and conduct further research into insider threats (Yuan & Wu, 2021). This research fuels increased knowledge and more information organizations use for defense. Several well-known cybersecurity organizations also now have insider threat research programs and continue to educate and provide solutions.

## Insider threat definition

One item discovered was that various published security documents from well-known security institutions define insider threats in several ways. MITRE (MITRE, 2022a), CMU (Carnegie Mellon University, 2023), and CISA (Cybersecurity and Infrastructure Security Agency, n.d.) all have subtly different means of defining insider threats and how these threats materialize. These slight differences lead to differences in the motives and actions performed by insider threats and give rise to differences in the means of detection as well as the research conducted. Thankfully, many of the papers written establish the definition they accept early in the paper and the thought processes surrounding it. Several ontologies, such as CMU (Costa et al., 2016) and another extension (Greitzer et al., 2016), have been authored to help share data and form a consistent language relating to insider threats.

One example of the issues these definitions create can be drawn back to the data set used for this paper. While some definitions sub-classify insider threats to include compromised insiders and UIT (Carnegie Mellon University, 2023), others will more loosely associate this definition with non-malicious insider threats (MITRE, 2022a). Further, these unintentional sub-classifications are not portrayed or simulated in many documented scenarios and datasets.

This difference in definitions can have impacts on the parameters that would be extracted to detect insider threats. In turn, the strategies used to determine insider threats might also be impacted. For example, a labeled data set may have differences based on how an organization defines inside threats. This definition needs to be considered by businesses as they define their strategy to detect insiders. For the purposes of this paper, the definition by CMU will be used so as to remain consistent with the data set they have authored.

## Detection of an insider threat

The literature reviewed found that insider threat detections deviated from classic malware detections to a substantial degree. Several papers drew comparisons between insider threat detection and the popular MITRE ATT&CK framework (MITRE, 2022b) used for classifying the behaviors of threat actors. The ATT&CK matrix focuses more on atomic detections, while most insider threat risks stem from establishing behaviors and detecting abnormalities. However, there is undoubtedly a cross-over between these two taxonomies, specifically when UIT comes into play. By specific definitions, UIT examines when an insider makes some mistake about a threat that an external threat actor might present. This situation might be present in several of ATT&CK’s techniques (Carnegie Mellon University, 2013). For example, ATT&CK represents phishing as a means performed by an external actor to facilitate unauthorized access. The target of this attack would be construed as a UIT, who has now facilitated access unintentionally to this threat. These types of scenarios have not been demonstrated in this data set but could be used as part of labeling since the frequency of successful phishing attempts are much higher (Greitzer et al., 2014).

Other papers' reviews departed from this classification and examined unchanging characteristics of insider threats (Sanzgiri, 2016). This insider threat-centric approach is more concerned with the insider's personality and attributes rather than the insider's resulting behavioral logs. An organization utilizing this strategy could avoid associating itself with these threats and completely avoid any resulting consequences that might occur from a successful insider threat issue. While some of these psychological detection techniques are far outside this paper's data science scope, some of the products of these characteristics are found in the test data set and used as features. For example, determining the psychological factors that might help determine if an insider threat is probable is not easily performed by data science elements. However, the OCEAN or Big Five personality trait metrics included in the CMU test set can be incorporated into quantitative data science detections.

A more comprehensive review of the literature related to data science detections of insider threats has been performed by Al-Mhiqani et al. (2020). Their review was concentrated on the significant amount of research performed on cyber activities. Many of the sources they reviewed point to retrieving parameters from users' logged activities and then use these features to detect the insiders through various means. While this research is specific to the research questions involving features, each paper they reviewed concentrated on extracting parameters from a single data source, such as network logs or file access. This paper does not focus on any single such source and instead retrieved features from all the data sources provided by the test set. Zou et al. (2020) performed similar research and looked through various data sources while listing extracted features. Unlike their research, the number of parameters evaluated for this paper's analysis will be more significant, allowing more scenarios to be detected or seen as abnormal. Additionally, little context is provided for their selection of parameters – something that expertise in the subject should provide. Another paper also reviewed the idea of pulling parameters from the log sources but took this analysis to another level with correlation of insider threat events (Ambre & Shekokar, 2015). By correlating events together and calculating the probability that events might occur within log sources, researchers could produce fewer false-positive events.

Few of the papers reviewed gave much insight into how the parameters for the algorithms were chosen. Their analysis also falls short of understanding each parameter's magnitude in detecting insider threats. This importance information gives better context to the analysts that may need to respond to these insider threats. In understanding how these features impact the model, an analyst is provided the details on where first to focus the analysis and how the combination of factors indicated the detection was an insider threat.

Many papers on insider threat detection were focused on algorithms. These researchers concentrated their papers on deep learning and how it might be applied to insider threat detection. A broad range of solutions has been developed in this area. Including topics on autoencoders (Liu, 2018) and comparisons of various neural network methodologies, most prominently looking at recurrent neural networks (RNN) (Kim et al., 2018). This same paper by Kim et al. (2018) also discusses attempts made with support vector machines (SVM) and decision trees. Supervised machine learning strategies were primarily accompanied by discussions on the high amount of imbalance found within data sets and real-life scenarios. While deep learning strategies were effective in detecting insider threats, these algorithms are less interpretable than the ensemble machine learning strategies this paper studies.

Each of these papers on algorithms sought to measure their performance using a variety of metrics. Generally, the papers found that grading their algorithms was best achieved using the precision, recall, and receiver operating characteristic (ROC) curve (Liu et al., 2018a) (Zou et al., 2020). Alternately, Jason Brownlee (2021) provided a flowchart in his book illustrating the best metric for evaluating machine learning algorithms in the context of imbalanced data (p.308). Based on this analysis and diagram, an F1 score would be a better metric for measuring performance.

## Real-life scenarios

The original creators of the test data set (Glasser & Lindauer, 2013) acknowledge in their future work that the scenarios displaying insider threat should be modeled to better emulate real-world scenarios. This is made further obvious in CERT’s Common Sense Guide to Mitigating Insider Threats (2022) in stating that the approaches used in generating insider threat data are constrained by sensitivity and validity of the data. This constraint can limit the effectiveness of models constructed from such scenarios as they may not be tuned to detect all potential insider threats or scenarios.

The mechanics of supervised learning require models to be constructed from data sets specific to the scenarios learned within those data sets. As such, finding a model that can determine all potential insider threat scenarios seems unlikely. Moreover, specific scenarios may not pose a threat to some organizations. Slight variations in identified scenarios can also cause problems for detection methods. While detecting an insider threat is necessary to prevent further damage, investing in prevention is a better approach to mitigate the impact of insider threats (Musthaler, 2008). Unfortunately, preventative controls are not typically modeled in test sets.

To address this real-life issue, it is crucial to develop models that can detect insider threats across various scenarios, rather than relying on models trained on a limited set of scenarios. This approach would require the generation of more diverse data sets that incorporate a range of insider threat scenarios. Additionally, incorporating preventative controls in such data sets would enable the development of models better suited for mitigating insider threats. Ultimately, developing such models would contribute to a more comprehensive and practical approach to managing insider threats, which is critical for protecting organizations from the potentially devastating impact of insider attacks.

## Conclusion

This literature review has shown a significant increase in research on detecting insider threats, particularly given the increased popularity of remote work and prominent high-profile whistleblower cases. Although there are numerous published studies and papers on this subject, this literature review has been concentrated to the research questions of this paper and specifically illustrates where studies have deviated from each other or more specifically follows the research of this paper. These differences include:

* + The definition of an insider threat and how that relates to the means used to discover these issues.
  + Insider threats are seen differently than malware infections or other cybersecurity detections.
  + While there are many papers that speak about features, few detail the means in which features are extracted and enumerate through the features retrieved.

The cybersecurity field is rapidly evolving, and more research must be done to detect insider threats more effectively. With the continued advancement of cybersecurity and the increasing use of remote work, staying current with emerging insider threat detection trends is essential for businesses. Overall, there is a need for future research to delve deeper into understanding and detecting insider threat behavior and the use of multiple data sources in detecting these threats. One avenue of this research is data science and parameter extraction from data log sources. This future research, including this paper, will help organizations better protect their data and assets from insider threats in the rapidly changing world of cybersecurity.

# Chapter 3: Methodology and Analysis

## Introduction

Per the literature review, data science can detect insider threats through various means. However, little published research describes extracting features, selecting the most relevant features, and calculating the importance determined by the algorithms. Organizations can use these features to validate, diagnose, and triage insider threats based on the importance that the algorithms have placed on them. The research conducted with this paper has provided some insight into the mechanics of generating these features. A forensic methodology successfully accomplished this paper’s research and helped answer the research questions in Chapter 1. The details of this methodology are presented in this chapter.

This forensic viewpoint directly relates to one of the research questions. A forensic viewpoint looks through the entirety of the logs rather than replaying the user’s activities, simulating the activities. This analysis gains the perspective of all the times and can compare a particular user, role, or slice of time to an earlier portion. This method allows analysts and algorithms to compare what is expected and how it might abstract itself from being normal.

A forensic examination is something that a medium-sized business might initiate if there is a suspicion that insider threat activity has taken place in the organization. Alternatively, the company may periodically review its logs to determine whether insider threats exist. This investigation requires a window of logs to examine from which features can be extracted. In the case of this test set, this window of time was 15 months. Depending on the organization's ability to collect and store these logs, this may not be feasible. However, in this business's case, it can keep and process the information accordingly.

Forensically studying this data set differs from real-time detection capabilities. This research has summarized a user’s activity, behavior, and characteristics on a calendar day basis – named the userDate. Analyzing this data more atomically per session, as might be presented between login and logout, is certainly feasible. However, analysts might lose the ability to compare one session with another. For example, a session lasting 14 hours for an employee is not easily compared to a session lasting 4 hours for another user. Some features can compensate by regarding the metric gathered by the period it occurred. Others would face difficulty or lose some meaning because of this conversion.

To compensate for this complexity, a session in the context of this report will be taken as one calendar day with the assumption that all data source logs are held within the same time zone. This assumption can be easily made as most organizations facilitate a logging infrastructure and would make the necessary corrections to compensate for devices logging from various physical locations and time synchronization. Additionally, as features are compiled, logical checks will accompany them.

Using a daily session window for all users, or userDate, allows analysts to directly compare users since they all have the same time frame. Unfortunately, it also means that a detection would be delayed by at least the length of a full calendar day. Since the Ponemon Institute indicates that the mean time to contain an insider threat is 85 days (2022), this is an acceptable time to wait to respond to an insider situation. Additionally, as seen in the scenarios included in the data set, an insider threat might not happen in a single calendar day. Indications of an insider becoming a threat occur over a more extended period than just a single day. As a result, some features are aggregated over longer periods of time. When this occurs, a value will remain static for the period of aggregation across multiple userDates. As designed with this analysis, some features take on monthly values while others are done by as short as one hour – as would be assigned to a userDate for the first login for a user during a particular userDate. In the case of psychometric values, this feature remains constant for each user through all userDates.

This forensic design with static day-long sessions helps to iterate over other research questions, such as identifying the most appropriate parameters to detect insider threats and which algorithms can be used to detect insider threats. By keeping this time frame static, this paper concentrates more on the features that have the most utility to analysts. This methodology can also be used to develop new features to account for other scenarios rapidly. This capability is instrumental in this study as parameters can be quickly developed and included in the analysis.

The XGBoost and Isolation Forest algorithms have been selected to evaluate the features extracted from the data set. These decision tree algorithms can choose the relevant features in the data set by generating many decision trees. These algorithms are also able to deal with the imbalanced classes in the data set through adjustment in their hyperparameters. Using these two algorithms will help to answer three of the research questions. Namely, the effectiveness of data science strategies in detecting insider threats, which user features and the best indication of an insider threat, and how decision tree algorithms can detect insider threats.

After completing the algorithmic analysis of the features extracted, the paper will make conclusions about how this work was performed and if the forensic methodology was successful in detecting threats. These conclusions have objectively and quantitatively summarized the effectiveness of data science strategies in detecting insider threats. By examining the output of these efforts, future directions and recommendations have been formed.

## Research design

The project requires features to be acquired from these data sets to be used by the tools and algorithms. As described in the data collection section, the project environment has been designed to develop and store these features into a features dataframe rapidly. Features constructed are then stored in a singular dataframe that has been saved to the file system [Figure 2]. This feature dataframe is the target of the algorithms addressed in this paper.

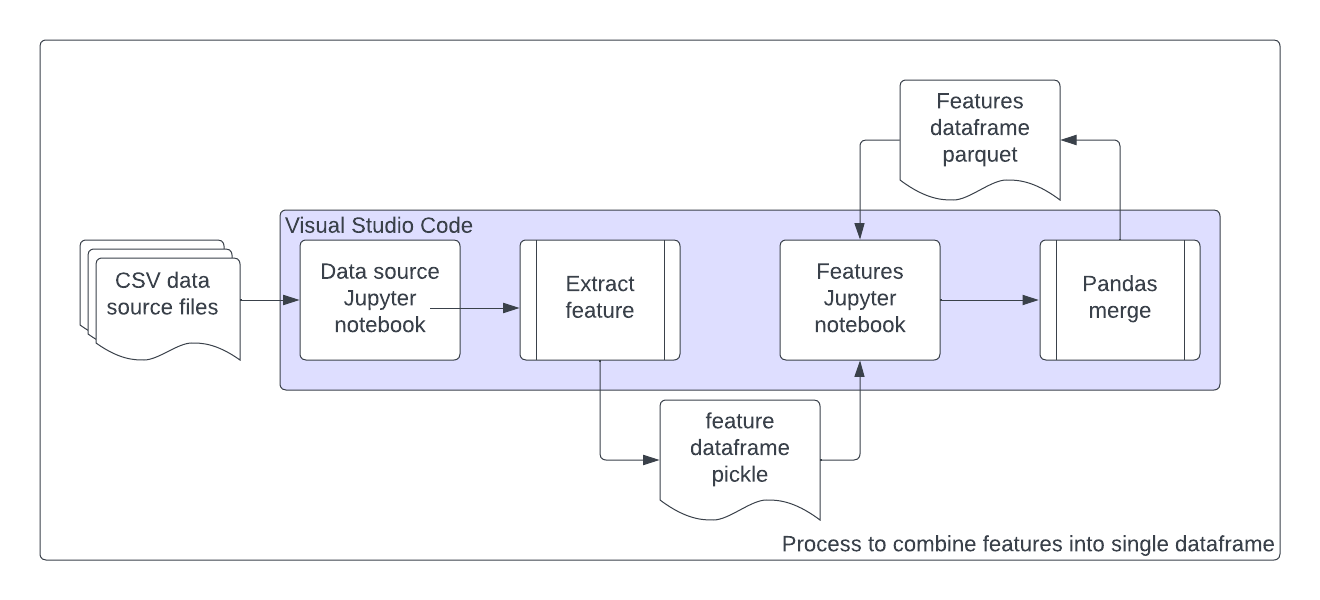


Figure 2: Process of acquiring a feature from data set

The features defined below are tuned to the scenarios described by the data set but also have an emphasis on known threats to businesses from insider threats:

**isEmployed**: Developed from the LDAP data source, isEmployed signifies if a user is present in the LDAP dump for that month. The LDAP data source files for each month were combined into a single dataframe using pandas and python. Since the LDAP data file is only found on a monthly frequency, it may only be determined if the user was employed on the first of that month. However, other features should indicate if the user was logged on. A function was developed in the feature notebook that uses a groupby dataframe to find the maximum value string for a user. Subtracting one from this value provides for the last month that the user was seen in the LDAP source files. A function is defined to compare this value with the date field in the features dataframe and find if the end date is greater than the value already present in the dataframe. The function's output is Boolean, and a 1 or 0 is provided in the new isEmployed column for each userData row. As visually seen in Figure 1, DTAA started with 1000 employees and lost employees over time. Stranger still, no employee was found to have started with the company based on this data set in the time window examined. This employee decrease may indicate that the company is downsizing, impacting employee morale.

**hasLogin**: The hasLogin feature is calculated from the logon.csv data source file after importing this file into a pandas dataframe. This feature shows the number of times a user logged on to the company's resources for that specific day. The date column of the logon is converted into a datetime type to facilitate this. This datetime type has the hour, day of the month, and calendar day extracted from it and stored in separate columns. The calendar variable is taken as a date type and later converted into a string named stringDate. As these features will frequently demonstrate, the dataframe object is transformed into a pickle file and stored in the logon directory. This pickle file is then restored into an object as part of the features notebook. Since the userDate field is the same for both dataframes, the hasLogin feature can be merged in such a way that the features dataframe preserves values that do not match the hasLogin dataframe. Due to the position of the features dataframe in the merge command, this takes the shape of a left merge. Columns are renamed to indicate how it is associated with hasLogin. CMU has constructed the data set so that when isEmployed is zero, hasLogin will always be zero. It would be highly unusual for a logon to occur for a user who is no longer an employee.

**firstActivity**: firstActivity is acquired from the device data source. This data source collects and examines when external devices are connected to computing devices. The connections might occur due to a USB device being plugged into the resource. Both connections and disconnects are gathered in this data source. As the name implies, firstActivity gathers the userDate that a Connect event first took place. A dataframe is collected and converted to a pickle, where it is then read into the primary features dataframe. This information is then merged into the dataframe. firstActivity is an important feature that examines the potential of a user to exfiltrate a business' information when no previous activity had been witnessed.

**hasConnect**: Similar to firstActivity, hasConnect is pulled from the device data source. Unlike firstActivity, this feature counts the number of device Connect events by userDate. In the same manner, a dataframe is created and then transported as well as merged into the dataframe with the other features. hasConnect illustrates a time period in which the user connected a device multiple times in a tracked period. This behavior may represent a user attempting to gather information from various devices. The behavior could also show a user propagating a malicious piece of code, proliferation of misinformation, or a means of disabling a business' resources.

**probMinLogonRole**: Incorporating the user's role in the features, probMinLogonRole collects the probability that a user’s first logon occurs during the hour seen during the examined userDate session. A change in logon behavior may indicate user nefariousness while attempting to hide their activities from other co-workers during who might arrive at the same time. For example, a user who historically logged in at 8 AM every day and suddenly started logging in at 5 AM may be trying to hide threatening activities. At the same time, a user who also logged in at 8 AM and suddenly started logging in at 10 AM might indicate an issue in employee morale. This metric was calculated using the kernel density estimator (KDE) from scikit-learn's sklearn.neighbors (Pedregosa et al., 2011). Using each distinct role and for each hour, a dataframe was generated to show the probability that a particular role might logon during a specific hour. This dataframe was stored in a pickle and transported to the feature notebook page. After merging the role and dateHour to the feature dataframe, the probability was looked up for each userDate and stored in the probMinLogonRole column. Checks were run on each role to validate that the sum of all the hour's probabilities was equal to one.

Chart, line chart, histogram

Description automatically generated

Figure 3: Distribution of activity from Accountant role in logon data file

**probMaxLogoutRole**: Similarly to probMinLogonRole, the maximum logout time was found for each userData using the last logout event. This event was used to lookup against a transported dataframe containing the roles and dateHour. This particular calculation gathers the likelihood that a user might stay late rather than logon early. It is also calculated by function with role and hour inputs since certain roles are more likely to remain in the office or work different hours than others.

**probLogonUser**: Also similar to probMinLogonRole, probLogonUser calculates the probability that a user, rather than a role, will likely logon during a particular hour. Unlike probMinLogonRole, the distribution used by KDE was taken from the distribution of the user's logon times – not the entirety of the user's role. This transported dataframe was then used to lookup the probability based on the minimum logon time for the user. It is important to note that the difference in these values is comparing a user's logon activity to their role's behavior versus their own. Since a user's behavior might be unique for their role, this column prevents a user from being penalized for performing their role but accomplishing it during a different time.

**probLogoutUser**: In the same vein as probLogonUser and probMaxLogoutRole, probLogoutUser determines the probability for a user's last logout for a userDate based on the entirety of the user's previous Logouts over the year and a half period. By a similar nature, probLogoutUser finds when a user's logout for the day is outside of a typical time for this user and the dateHour in question.

**changeNumRoles**: The changeNumRoles finds the difference in the number of employees within a role for a given userDate. This feature was calculated using the LDAP data source and is sensitive to a month – meaning that a calendar month of dates will all have the same value for this feature. As noted prior, DTAA seems to be retracting the number of employees. If the same amount of work were being done with fewer people in that role, this could impact morale and the company's general function. This exodus of talent and workforce could affect the remaining employees and make them prone to become a threat. Additionally, the employees leaving may feel entitled to the work they performed with the company and attempt to copy this company information to be used by their next employer, who might also be a competitor.

**numMonthsEmployed**: Calculated based on each employee ID rather than by userDate, this value counts the number of times the employee ID is found in the combined LDAP data source. A user found throughout the entire period will have a value of 18. The minimum value observed was 2. This feature is the result of viewing the entirety of the data set and inline with a forensic methodology.

**webCount**: The feature webCount was calculated from the http data source. This massive file requires the Modin library to allow the http data source to be ingested as a dataframe. The Modin library allows pandas-like functionality with multi-threading and memory swapping. Once converted to a dataframe, the userDate column is introduced by tabulating the originating user and converting strings to datetimes. Row items in the data frame are grouped by the userDate and then counted. Since userDate also exists in the feature dataframe, this newly calculated dataframe is easily ported to a pickle file and then reconstituted in the feature dataframe's kernel, where it is merged. Potential insiders may have a more significant degree of web usage on company resources. This activity distracts from their primary responsibilities to the company and could be used as a potential conduit to exfiltrate company data. Additionally, the web might be used by malicious actors soliciting company employees to perform actions not in the best interest of their employers.

**webQuestionableUsage**: Like webCount, webQuestionableUsage counts the number of rows for a userDate column. Unlike webCount, webQuestionableUsage counts the number of rows in the http data source where the site visited may be used by an insider in a suspicious capacity. The category of the website is determined by parsing the domain from the website and then referencing the domain's information from the Cisco Umbrella (Cisco, 2023) domain categorization database via API call. This category is gathered for each unique domain within the http data source. The complete inventory of domain categories was then reviewed, and any classification that an insider might use in an un-business-like practice was marked. After these domains with suspicious categories were determined, the dataframe was grouped by the userDate and the number of domain accesses summed. With this task completed, the dataframe was exported to a pickle file and transported to the kernel, compiling all the features of the userDate. A large count of questionable web use for a particular userDate might show that a user is consciously attempting web-related activities not in the business's best interest.

**countPC**: The countPC feature records the distinct number of devices used during a particular userDate. This information is extracted from the logon data source. This feature could detect a user with increased access rights working to propagate a malicious piece of code to infect many devices. Many logins to different devices might occur when an insider has been leveraged to impact the business with ransomware. Like other features, the metric was calculated from its data source notebook and transported into the features notebook. Checks were also performed to validate that the countPC value was never greater than zero when the hasLogin was also zero.

**numExtEmails**: The number of external emails is found by combining several data sources. Only the LDAP data source associates the user id with the email address, so this data source is needed. After merging this information, the userDate can be combined for each row item in the email data source. Once the userDate is combined with the email address, any row item that has the string 'dtaa.com' in the 'to' field is eliminated from consideration since this would mean the email was being sent to internal recipients. Finally, the content of the filtered dataframe was grouped, and the number of row items was counted. Stored as a pickled dataframe with other related email measures, this feature is transported and merged into the main feature dataframe. Insiders may use company resources such as email to communicate with outside entities. Tracking the number of external emails sent can help detect if this should happen.

**sumExtEmailAttachments**: When the numExtEmails was calculated (above), an ‘agg’ function was used to record other parameters from the LDAP/email combined data source. One of these features was sumExtEmailAttachments. The email data source includes an 'attachments' column with the count of the number of attachments. Summing this column for all attachments grouped by the userDate illustrates the number of attachments sent to all external email addresses. Insider threats may attempt to exfiltrate company information using email to outside email addresses. Analysts may assess the likelihood that this information is sensitive based on the number of attachments.

**sumExtEmailSize**: The feature sumExtEmailSize was calculated in the same manner as sumExtEmailAttachments except using a different column of the email data source. Using the 'size' column of the filtered email data source, the numerical value was summed for each userDate. Users attempting to send large magnitudes of company information may try to do so through email attachments. A large-sized file may indicate a compressed archive of email information.

**df\_psychometic**: The psychometric data source contains measures describing an employee's personality traits of openness (psyc\_O), conscientiousness (psyc\_C), extraversion (psyc\_E), agreeableness (psyc\_A), and neuroticism (psyc\_N). These traits are tabulated for each user\_id and held constant throughout the examined period. The metrics are communicated through the first letter of each of these traits forming the acronym 'OCEAN’ in the dataset but take the psyc prefix as labeled in the parenthesis above for the features dataframe. Each value of these attributed are in a range between 10 and 50. Each of these traits is joined to the features dataframe for the user\_id associated with each userDate value. Certain personality combinations are predisposed to becoming insider threats (Padayachee, 2022).

## Population and Sample

This data set was synthetically created to test insider threat detection capabilities, as discussed. For this project, the data set is used in its entirety and will not have a sample drawn from it. However, viewed more holistically, a business will likely have more logs than what has been described in this data set. From this perspective, the data set could be seen as a sample or summary of a greater whole.

The scenarios embedded in the test data set are realistic and have merit in being detected. In the data set used for this project, two users performed malicious activities to the company's business. While the scenarios were mapped directly to userDates, other events also occur in this data set that are unusual and should be investigated with the processes that DTAA uses to detect insider threats. These two events do not represent all events that might occur from insider threats, but facets of their behavior are representative of other forms of insider threat.

Based on the descriptions from CMU, the following are the scenarios [data sources in brackets]:

* A user starts to exhibit strange behaviors that they have not performed before. Namely, the user begins logging in after work hours [login] and using removable drives [device]. After completing these activities, the user is found to upload data to wikileaks.org [http]. The user leaves the business after this upload [ldap] – potentially undetected and exfiltrating business data. Depending on the line of business that DTAA might be in, this might have devastating consequences. This user was interested in willfully causing harm to the company.
* Another user visits numerous job websites [http]. This user also sends emails soliciting employment from competing businesses [email]. Before leaving the company [ldap], a removable drive [devices] is used to steal data and information from the business. This situation can be more associated with industrial espionage and may be in breach of the user's employment contract.

## Bias Handling

One of the most significant problems with this project was dealing with the severe imbalance between regular users and insider threats. There are 50 userDates categorized as isInsider while there are 515950 rows not categorized as dealing with an insider. This issue creates a severe imbalance that algorithms have difficulty exploring. Algorithms attempting to find the minority class in this situation will find difficulty in doing so, as there is an assumption that the classes are equally distributed.

Using the XGBoost algorithm, there is a hyperparameter to adjust that impacts the weights of the losses differently for each class. This is accomplished by finding the frequency for each class in the data set and then setting the scale\_pos\_weight parameter in XGBoost to account for this imbalance. Other methodologies, such as undersampling and oversampling, can also address this situation but may lead to different biases.

Another bias addressed in this research was dealing with the range of values that might be present in one certain role but not in another. One example is the number of machines a system administrator needs to log into versus research scientists. As part of their role, a system administrator is expected to log into numerous devices throughout their userDate, creating a large number found to be peculiar to an unsupervised machine learning algorithm due to the limited number of system administrators. To address this issue, the features of the unsupervised algorithm are scaled so that a system administrator’s number of machines is compared with other system administrators, and research scientists are compared to others as well. Both role and individual features do this scaling to ensure that personal choices are considered.

## Data Collection

The data used to detect insider threats is some of the most sensitive logs gathered by a company. Insider threats also require a tactful response and controlled release of information. Businesses typically are careful to share this information and may never do so due to the legal and business repercussions. CMU has synthetically created this data set to mimic the logging that a company would collect. This data set includes several realistic and probable insider threat scenarios that might be investigated or detected through various methods. Like most businesses, the length and retention of the CMU log files is satisfactory for most compliance standards and allows the ability to establish regular activity in the company.

The compressed archives containing the Insider Threat Test database were retrieved from CMU (Lindauer, 2020) via a purpose-built repository. Files from the website are named to represent their release and data set number. For example, the file r3.1.tar.bz is a BZipped (bz) archive (tar) from the third revision (r3) and is the first data set (.1). This is the file from which initial development of the detection methodology was performed. The third revision first data set was picked due to the size of the download and the constraints on the platform at that point. CMU maintains six revisions.

After the file was downloaded and the integrity of the file was checked against the sha256 hashes posted to the website, the file was uncompressed and unarchived. However, each source data file was extracted to an independent directory based on the data source's name. In some of these data sources, multiple files compose the collection – see Table 1. The parent directory of these data source files was named to match the revision and version of the data set - opening the opportunity to simultaneously host other revisions in the future. This directory configuration differs from how the archive was compressed and shared from the website.

The project files specific to the data source are stored in these directories. These files included python notebook files (\*.ipynb), pickle files (\*.pkl), newly created data files (\*.csv), and parquet files (\*.parquet). By storing the files close to the data sets they originated, it was immediately apparent what generated the file and how it might be used in the future.

As features were taken from the data set, various tests were performed to validate the consistency of the data set. The complexity and artificial nature of this data set created opportunities where unlikely and impossible situations might occur. Some examples of these situations might occur when a user who might have a login is not found in the LDAP data source or a user who connects a device and has not logged in to a computer. In other cases, these consistencies were outside the typical capacities of computer systems. For instance, logout events are challenging to capture on computer systems. A missing logout might occur on a laptop where a user hibernates the system or disconnects from the network. In these situations, an abstraction of events can be viewed as the real-life equivalent but contextually logged differently in the data set. As these checks were performed, no inconsistencies were found. The data set seemed whole and self-validating for the parameters collected and the purpose of this paper.

## Data Analysis

In addition to data source directories, two other types of directories were created. One housed the primary or master files for all the features calculated or obtained from the data source files. Other directories were also established for the models and techniques to investigate the dataframe containing the features and predictors. As needed, the notebooks homed in these directories also made changes to the features dataframe to make it compatible with the explored algorithm but never saved beyond the model directory. For example, when using unsupervised algorithms, the notebook would drop the predictor column and define dummy variables for other non-numerical features but never save files beyond the directory where this notebook was housed.

Each row of the features dataframe required the user\_id, userDate, dateString, role, and employee\_name columns. These columns allowed for joining other features from other data sets. Another value, isInsider, was used as the predictor of the data set. This predictor variable was coded by hand to the userDate as interpreted from the separately maintained answers data set. This answers data set described the scenarios for each revision and the data set created from CMU.

This features dataframe was created with a key for each user during each day. This column, userDate, in the dataframe took the shape <user>\_<yyyymmdd> and is a unique value for each row. Many of the features engineered are aggregated daily, as described in the chapter three introduction, with all log sources having times associated with the days and fields.

Table 2: Core row properties in features dataframe

|  |  |
| --- | --- |
| Command: | |
| df\_master[['user\_id', 'dateString', 'userDate','role','isInsider','employee\_name']].iloc[0] | |
| user\_id | MCN0973 |
| dateString | 20100101 |
| userDate | MCN0973\_20100101 |
| role | ElectricalEngineer |
| isInsider | 0 |
| employee\_name | Macey Colleen Nash |

Ultimately, the problem of finding insider threats is a classification problem. Unfortunately, the ratio of insider threats to normal users is seriously imbalanced, and we can make no assumptions about the distribution of the data. Given this situation, a non-parametric algorithm is a better solution than others to minimize the number of transformations to the data required if possible. The data set the project targets are also large compared to many of the data sets used for class. Additionally, the paper's strategy has generated features and provided structure to the data sets by combining the various data file's information. Feature selection is a task that needs to be performed. Finally, if a business finds an insider threat, there is a need to understand why the algorithm made this conclusion. Analysts working on insider threat detection will need to understand how to follow up and confirm this detection while having the ability to communicate this information to non-technical groups like legal and human resources.

Decision trees satisfy these requirements and are easy to interpret based on these needs. One ensemble learning implementation of decision trees is the XGBoost algorithm (Chen & Guestrin, 2016). When using supervised learning for insider threat detection, using XGBoost to create and determine a model makes sense since it can handle large amounts of data while remaining performant. This algorithm is also more easily implemented with structured data, such as what has been created within our features dataframe. XGBoost also has multiple parameters to deal with imbalanced data sets and overfitting (Tsukerman, 2019). For these reasons, XGBoost will be explored for detecting insider threats in a supervised learning capacity.

While XGBoost can be easily used to perform this form of detection, the algorithm is limited because it will only be trained with the scenarios found in the data set. While there is utility in performing this, a business would also be interested in detecting insider threats that do not conform to the scenarios outlined in the data set. To contrast XGBoost’s supervised learning, unsupervised learning will also be performed and compared with this supervised learning. While XGBoost can be used for unsupervised learning in specific capacities, this paper will explore using the decision tree algorithm for detecting anomalies – Isolation Forest.

An Isolation Forest is also an ensemble machine learning algorithm based on decision trees (Liu et.al., 2008). Since it is designed to find outliers, the imbalanced data set is ideal for analysis with this algorithm. An Isolation Forest also has various hyperparameters to be tuned and requires some idea about the number of anomalies present in the data set. These hyperparameters can be adjusted and calculated with the knowledge of the existing scenarios. By describing this contamination value, a business accepts that it will be researching a certain number of threats by analyzing a time period. This increased effort might be ideal for a business since the algorithm will find unusual issues even if not defined as insider threats.

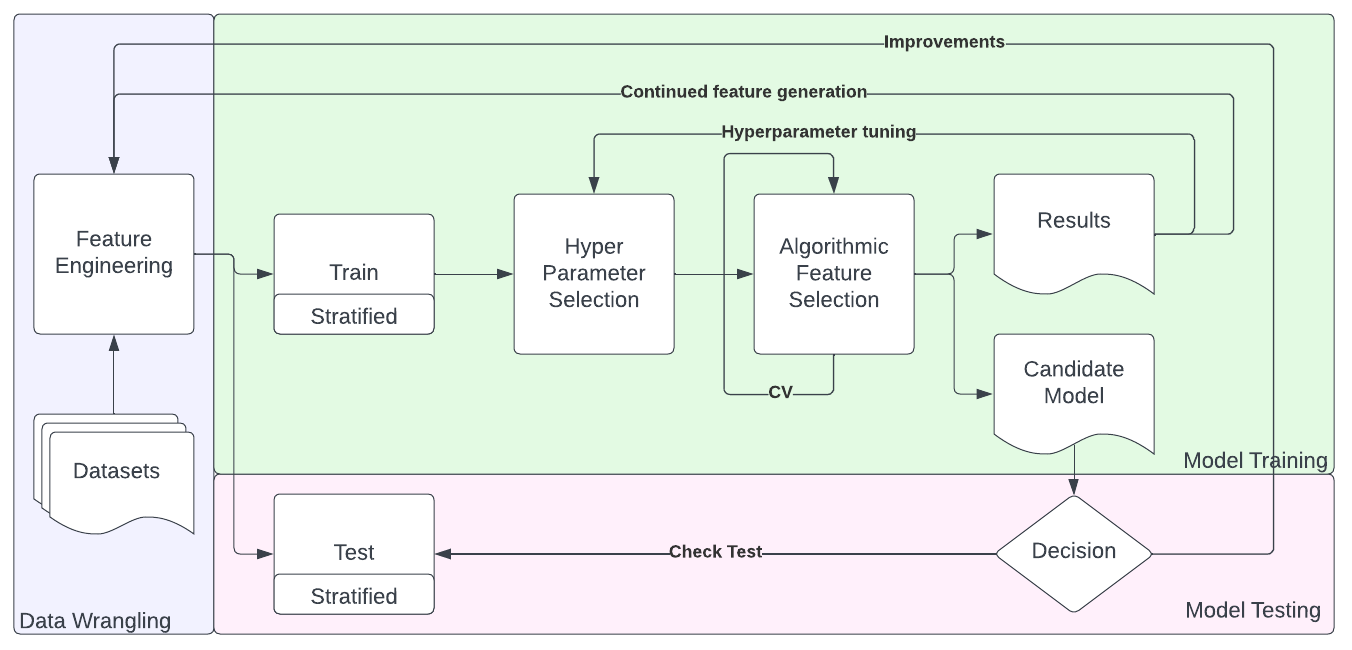


Figure 4: Process for algorithm evaluation

These two algorithms are also helpful in that they can determine the most appropriate features to be used to find insider threats. This capability allows the research to be more concerned with creating the most interesting features using the institutional knowledge from cybersecurity experience in combination with the description of the scenarios that CMU proposed. The algorithms will also provide clear, intuitive explanations of how the userDate was determined to be an insider threat based on the importance of the features. By using these algorithms, the business will incur less risk and analysts will be more informed as to the means in which an insider threat was detected. This permits the business’ response to be aligned with the situation, potentially preventing future insider threat situations. Using these two algorithms, the paper will address the research questions presented with this paper.

## Tool Descriptions

During the project's beginning phases, attempts were made to conduct the analysis using cloud-based resources. While it would seem ideal to leverage these pre-constructed and targeted resources, the costs incurred from using cloud resources in the early weeks of the project indicated that their utility would not meet the personal budget constraints placed on the project. Exploratory analysis was expensive when the charges were summed for compute, storage, and application per time usage. Extrapolating this cost throughout the semester demonstrated that purchasing a project-specific hardware platform was more cost-effective.

In the project's context, this also makes sense. A medium-sized business, that seemingly has little placement in cloud resources based on the lack of cloud logs contained in the data sets, would also leverage on-premises resources or purchase a resource to perform the project's activities since the additive charges of this analysis would be expensive and, in the end if using cloud infrastructure, they would have no tangible resources beyond the results from the analysis. If a project-specific device were purchased, the business could perform similar data exploration activities using the data already hosted on its site. Exploratory analysis of user logs would not be the project to justify an enterprise commitment to cloud initiatives. This conclusion may change should the analysis be ongoing, continuous, production, or if the logs were collected and gathered in cloud storage devices.

One of the reasons cloud resources were so expensive was the memory requirements of processing large CSV files and storing these in a manner that was accessible to the Python resources. Pandas (NumFOCUS, 2023) and Modin (Modin Developers, 2022) libraries both make extensive use of physical memory. In testing, machines with 16 GB of memory found difficulty in both retrieving and processing the larger CSV files of the data set. Larger cloud platforms with more memory were more expensive to provision and use than smaller devices. The acquired hardware-specific platform equipped 64 GB of memory and had no trouble processing these files.

Python, with the associated libraries for Modin, Pandas, and scikit-learn, were used for this project. Based on familiarity with Pandas, feature extraction is performed readily using this library and, with the ability to also integrate with algorithms, data analysis can also take place. The R programming language was also considered for use in this project but, relating to the interpretability of the research outcomes and extensibility for other businesses, cybersecurity analysts are more likely to know Python and their related libraries.

Python also has a large support community that has readily authored libraries to perform a myriad of tasks. For example, using requests and other native libraries, Python was able to enrich the data of the http calls in the CMU data set with categories which were then used to abstract additional features about web conduct of users. This interoperability between data science libraries, data management functions, and visualization capabilities makes Python an ideal choice for this project.

Visual Studio Code using Python notebooks was used as the coding interface for this project. Coding cells allow for experimentation within the notebooks and kept a history of the how attempts were coded. Direct integration for git was also provided. Alternatively, the project could have been coded as a flat file in Python. Unfortunately, this would lead to situations which are less exploratory. Should the model generation be made production and automated, Python scripts could be used but error-checking and alerting would need to be incorporated.

The entire project hierarchy was backed up to a public git repository at <https://github.com/hendt5997/capstone>. As changes were made to the projects' various files, they were committed and pushed to the repository. Using Git created a backup of all the work performed on the project and provided a transport methodology that kept multiple machines working on the project coordinated with the duplicate files. This strategy is similar to what might be employed by a medium-sized business for code development. However, due to the sensitivity of this issue and data surrounding insider threat detection, this information would be made a private Git repository inaccessible to the public.

Git does have certain limitations that were encountered with the project. Git only allows files with a maximum size of 150 MB by default. For this reason, the data set files were not stored with notebooks and other computer data files. In cases where files exceeded this size but were computed as part of the project, the dataframe or object was compressed or converted to the parquet file format.

## Conclusion

While other methods to determine insider threats were also considered, the means of performing this analysis using these methods were likely to fail to satisfy the research questions or timeline of the project. Using the methodology detailed above shows how features are extracted from the data set and describes these features and how they may indicate an insider threat. As highlighted in the literature review, most researchers did not concentrate on the features extracted for their analysis, nor did they explain the impact these features might have on the detection itself.

This methodology is also extensible and can include other data sets and features. For example, today’s businesses will have various forms of cloud infrastructure. By ingesting these logs and generating features from these logs, a business will be able to incorporate these logs into the features dataframe described and use the same algorithms and processes to find insiders and abnormalities. The tools selected for this project lend themselves to this extensibility and work toward the knowledge set of the target audience of cybersecurity management and analysts.

The algorithms used by this research are also interpretable by the business and analysts. While ensemble methods add complexity to these interpretations, there are resources to extract the most important features used for detections. These visualizations help to answer what features are best suited for detecting insider threats. By the very nature of the algorithm, feature selection is performed and evaluated – providing richer insights than might a single decision tree that may not correctly weigh the outcomes for the imbalanced data set.

# Chapter 4: Results

## Introduction

The results and value gathered from the data set generated will be presented in this chapter. The audience of this chapter’s illustrations and results will be DTAA executives and more technical security analysts who are in the throes of performing the post-mortem analysis of insider threats and seek to develop a means of detecting insider threats using data science strategies. In this manner, the analysis results are specifically written to illustrate business value and be delivered concisely with enough technical content to be evaluated by the analysts.

## Summary of Results to Research Questions

This section presents a focus of the results directed towards the research questions as might be provided as an executive brief. This concise summary is directly associated to the research questions. It describes the type of analysis and the result, if this result matched the hypothesis, and how the outcome related to the research question:

* + ***How effective are data science strategies at detecting insider threats in the context of logs generated from a medium-sized business?*** Data science strategies in detecting insider threats depend on the number and type of features provided to the algorithm. In the context of supervised learning, detection is dependent on the labeled data set and scenarios illustrated. Unsupervised learning will find anomalies in the data set, but these anomalies may not be related to insider threats. This outcome matched the expectation of this finding.
  + ***Are decision tree algorithms, like XGBoost and Isolation Forest, able to detect insider threats?*** Yes, these algorithms can detect insider threats based on the feature set extracted in this research. Supervised learning, using XGBoost, is more effective at performing this task and produces fewer investigable events than unsupervised learning as performed using Isolation Forest. The magnitude of false positives with Isolation Forest was a surprising outcome of the analysis conducted – regardless of the features provided to the algorithm.
  + ***Which data set features are the best indication of the user being an insider threat?*** Both XGBoost and Isolation Forest were able to determine the features that held the most importance. The features important to one algorithm did not fully coincide with those important to the other. XGBoost indicated that the features that were the most predictive were those that were based on behaviors of the userDate rather than characteristics of the user. From an analyst's perspective, this indicates that equal attention should be placed on employees of merit and those needing improvement. This did not intuitively match the predicted result, but it may be that additional features based on performance reviews and human resource feedback may change this result.
  + ***After the analysis is performed, how well might a medium-sized business be able to interpret the results of the detection and analysis?*** The detailed explanation of algorithms enables businesses to quickly triage the detection process by comparing it to non-detections. These importance tables reduce the risk of insider threats and limit the potential for false accusations of wrongdoing against employees. This assessment capability aligns with previous research and meets expectations.
  + ***Can a forensic methodology be used in detecting insider threats?*** Yes, a forensic perspective can be used for detecting insider threats. This methodology gains insights from the more extensive data set and can provide features that might not otherwise be made. For example, the number of months a present employee has worked would be calculated from his start time to their termination. If the analysis were conducted in real-time, this would simply reflect their tenure at the company. If the analysis were completed after they had ceased working, it would represent the length of their employment. Forensic analysis will increase the risk as the insider threat may have departed or continue conducting malicious actions. A model produced from a forensic methodology might be used in real-time detections.

## Summary of Results for Algorithm Comparison

As has been mentioned, DTAA is a medium-sized company that has been retracting its workforce from 1000 individuals. This company has multiple departments and employee roles. DTAA has learned that two (2) insiders have acted as threats in the past 15 months. This development has led to a forensic examination of their logs and data sets. These logs are stored as CSV files specific to the activity or function they log. The company seeks a means to detect insider threats using data science capabilities. Using the knowledge of the scenarios, 50 individual dates have been tied to the threatening activities of these two (2) employees.

Table 3: Quantities of classes after splitting data into training and testing sets.

|  |  |  |  |
| --- | --- | --- | --- |
| User type | # of userDate in training set | # of userDate in test set | dates |
| CSF0929 - insider | 5 | 2 | 20100701-0703  20100708-0709  20100714  20100716 |
| CCH0959 - insider | 35 | 8 | 20100802-0806  20100809-0813  20100816-0820  20100823-0903  20100907-0910  2010091300917  20100920-0924  20100927-0930 |
| Non-isInsider | 412760 | 103190 |  |

Table 3 illustrates the distribution of the imbalanced classes between the test and training sets. This table also details which dates have been labeled as insider threats for each of the two users found to be threats in this data set. Observationally, the dates when the users posed a threat are tightly grouped for each user over the 15 months.

This audience's goals align with the research questions addressed in this paper, specifically aimed at the effectiveness of the algorithms and analysis to detect insider threats and how analysts might interpret them. This interpretation is equally important as there is legal risk present should an employee be accused of being an insider threat when no issue is present (Berdal, 2018). If some harmful action is taken against the employee and the impacted employee takes legal action, that lack of interpretation would leave the company without a detailed explanation of why analysts acted on what the algorithm determined.

Table 4: Summary of evaluated algorithms using test split of data set.

|  |  |  |
| --- | --- | --- |
|  | XGBoost [supervised ML] | Isolation Forest [unsupervised ML] |
| Requires labeled training | Yes | No |
| Number of insider users | 2 | 2 |
| Detected insider users | 2 | 1 |
| Labeled insider events | 10 | 10 |
| Detected insider events | 8 | 2 |
| Incorrect insider detection events | 6 | 579 |
| Missed insider detection events | 2 | 8 |

Table 4 summarizes the effectiveness of the two algorithms applied to the test data set after training on the training data set. This table is targeted at the company's executives, who might use this information to decide if the data science approach to detection requires more investigation and can be used by the company to reduce risk in the future from insider threats.

## Supervised Learning

Supervised learning was conducted using the XGBoost algorithm (Chen & Guestrin, 2016). The model generated from this algorithm was trained using a cross-validated set of selected hyperparameters; testing five stratified folds of 81 candidate parameters for 405 models. The algorithm performs feature selection as part of its design. The train and test split were performed with scikit-learn using 80% train and 20% test. As a result of this split, the test had ten userDates of isInsider, and the algorithm training was configured with 40 indications of insider threats. Scoring for the training was performed with AUC (Area under the Curve) due to the several items: class imbalance, lack of availability of scoring F1 in XGBoost, and the relative importance of false positives and false negatives. The cross-validated folds were stratified to ensure that samples from both classes were present during the training.

Table 5: List of optimized hyperparameters tested for XGBoost.

|  |  |
| --- | --- |
| Hyperparameter tune | Tuning value |
| max\_depth | 10 |
| n\_estimators | 500 |
| learning\_rate | 0.3 |
| subsample | 0.8 |

From the hyperparameters selected, the best model had 500 estimators used. Since it is unlikely that analysts would look through 500 different decision trees to understand the model, the interpretability of the model is limited to the F score calculation of the features – see Figure 7. Sacrificing some interpretability is to be expected with the XGBoost algorithm and the trade-off for accuracy is warranted in this case. This optimization also found that a limited number of features would be used in each of these 500 trees. In terms of the business, maintaining this sense of interpretation is permissible so long as the number of detections is limited and would not overwhelm cybersecurity analysts performing triage on these detections so that time could be invested to diagnose the detection and determine the most questionable behaviors taking place during the userDate.

Based on the max\_depth and subsample hyperparameters, XGBoost calculates all the decision trees based on the observations in the training set. Predictions are made for each row item by the algorithm using and weighting these trees.

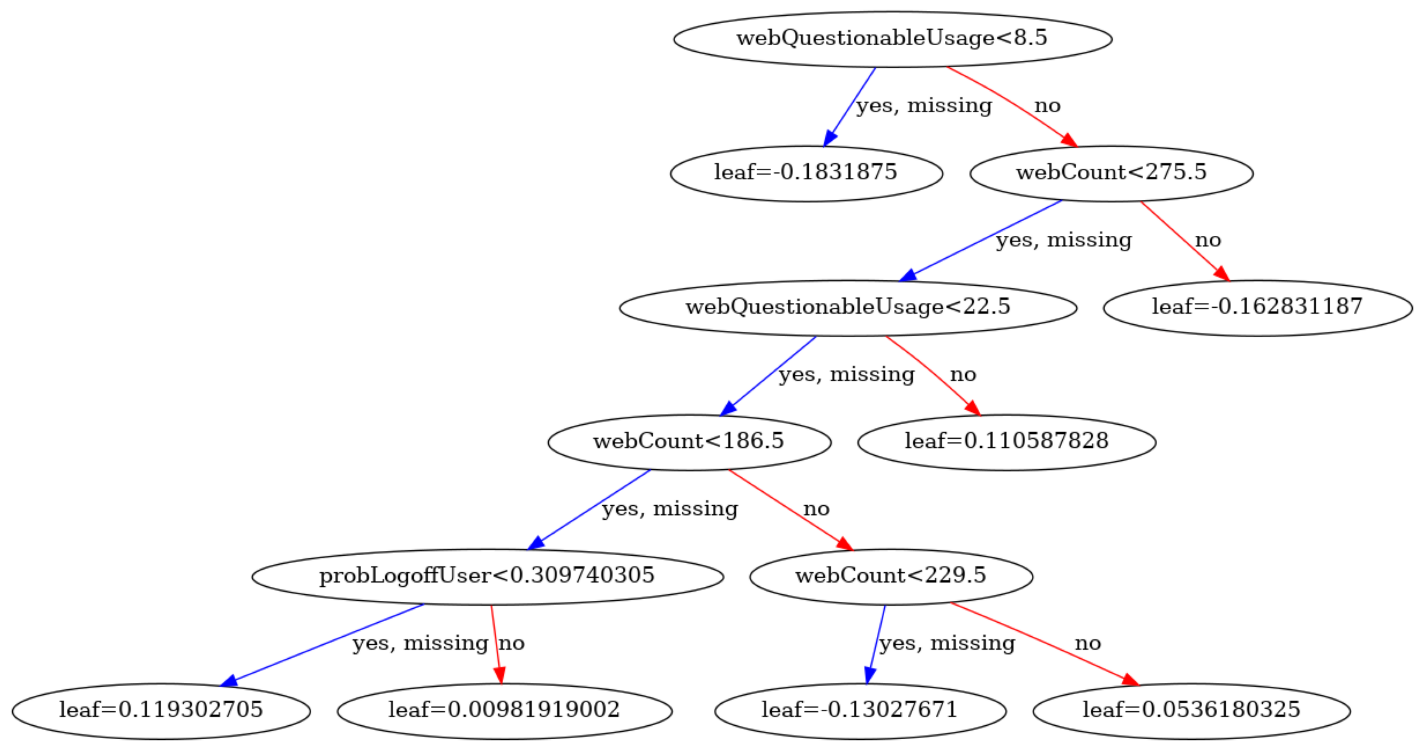


Figure 5: Example Decision Tree from XGBoost model for Insider Threat Detection

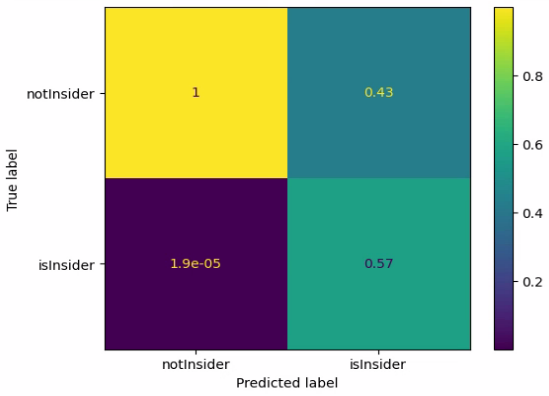
One of the 500 models used for making predictions about an insider is shown in Figure 4. This particular decision tree illustrates three different features in a 5-layer depth with webCount and webQuestionableUsage iterated in several layers; adding to their F score and importance in the ensemble model – seen in Figure 7. 

Figure 6: Confusion matrix from best model development using XGBoost

In the confusion matrix [Figure 6], the best model made eight (8) correct predictions of the ten (10) insider userDates in the test set. This best model also predicted six (6) userDates to be an insider threat when the isInsider field was not set [false positive]; yielding a precision score of 0.57 and a recall of 0.8.

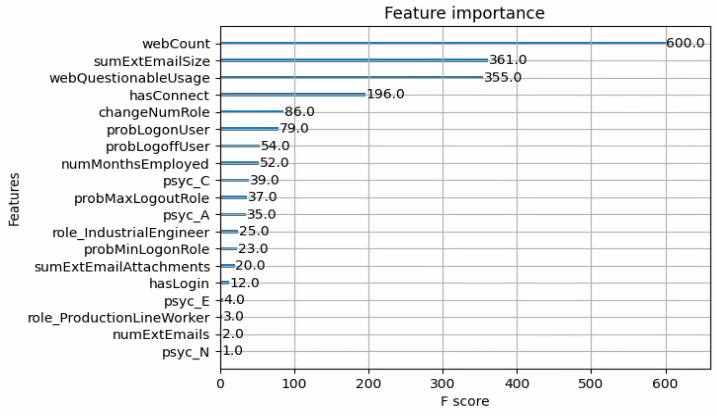
While this might not seem entirely promising and was not the expected value for the algorithm, the model can correctly detect at least one userDate from each of the two users involved as insider threats. From a business perspective, this is ideal in that no situation from any one user would be entirely lost. Both insider threat users were detected and can be analyzed to determine the threat they present. Unfortunately, these detections cost six (6) userDates that were not isInsider from two (2) users. Still, as the analysis later shows, these falsely detected userDates are from the same users as those detected as insider threats - only for different non-threat labeled dates.

Figure 7: Feature importance from XGBoost best model

XGBoost also facilitates a method to determine the feature importance within the model. From this Figure 7, we can see four (4) noteworthy features at the top of the table. These four (4) features, as determined by the algorithm, compose the bulk of how the algorithm will make decisions about the observations. Also notice that these features are wholly dependent on the actions of the user and not a characteristic of the user as might be defined as a role or psychometric. Based on the results of this feature importance graph, an analyst will be able to diagnose the detection and provide intuition on the discovery more easily. For example, in the event of a detection, the analyst can primarily look at these features to determine how the individual userDate might compare with other userDates.

The model incorrectly predicted six (6) insider threats positives (6) [false positives]. The features of these errors were examined to see how the model might be improved. The false positive userDates determined from this data set were found to belong to the same user\_id as the labeled insider threats but from dates not associated with malicious behavior, as seen in Table 6.

In most of these false positives, the detection was made prior to the actual user’s labeled event. These detections are a consequence of the forensic analysis performed on this data set. Had the detections occurred in real-time, the business may use these unrealized pseudo-detections to help prevent insider threats from happening in the future. Still, their utility brings some ethical dilemmas in what response might be required from an event that has not happened, or if the investigation of this type of false positive might result in the event never happening. However, the false positives illustrate that acting on detections, that might later be determined to be benign, could have benefit to the business.

Table 6: Table describing falsely labeled insider threats.

|  |  |  |
| --- | --- | --- |
| User\_id | User insider labeled? | Date of false positive |
| CSF0929 | labeled user for other dates | 20100723 – 7 days after first detection |
| CCH0959 | labeled user for other dates | 20100201 – 182 before first detection |
| CCH0959 | labeled user for other dates | 20100222 – 161 before first detection |
| CCH0959 | labeled user for other dates | 20100329 – 126 days before first detection |
| CCH0959 | labeled user for other dates | 20100505 – 89 days before first detection |
| CCH0959 | labeled user for other dates | 20100714 – 19 days before first detection |

The XGBoost model tuned with optimized hyperparameters did not detect two (2) of the ten (10) labeled insider threats (Table 7). When looking at the entirety of these userDate’s features, no obvious irregularity was detected, so it is uncertain that a cybersecurity analyst might make the same conclusion. This missed detection may indicate a missing feature unique to these two particular userDates. Unfortunately, one of these events was the first date of one of the user’s insider threat userDates. In a situation where this model might be made to perform near real-time analysis, the business has missed an opportunity to respond to the insider threat and deter further losses quickly. However, the timeline of that user’s [CSF0929] events is compact and limited (2), and that insider was found to commit a string of malicious actions.

Table 7: Table describing undetected insider threat userDates.

|  |  |  |
| --- | --- | --- |
| User\_id | Date of activity | Notes |
| CCH0959 | 20100802 – first date of insider label | Features gathered show no obvious irregularity |
| CCH0959 | 20100810 | Features gathered show no obvious irregularity |

Chart, scatter chart

Description automatically generated

Figure 8: Timeline of insider threat detections

This trained model has worked well for the business and detected both scenarios detailed in the data sets. While, by numbers, the algorithm has made several mistakes, these mistakes were in the same realm as the actual insider threats. Analysts working to determine the root cause of the issue can determine why the algorithm behaved as it did and have easily ruled out the other falsely detected issues, potentially incorporating new features in the model. If this algorithm functioned as the events were produced, aggregated, and run daily, there is a strong chance that the investigation into the false positives would prevent the actual events. The investigation might lead them to believe that they are monitored and therefore deter the activities – which would also be an unrealized win for the business.

## Unsupervised Learning

Unsupervised learning was also conducted using an Isolation Forest algorithm. This model was generated by optimizing the hyperparameters that were scored using f1\_micro since false positives and false negatives are important to the business and the data set is imbalanced. The stratified train-test split was held at the same ratios as the supervised learning (80:20), and the random state of this split keeps the composition the same between the two algorithm tests - see Table 3.

Table 8: List of optimized hyperparameters for Isolation Forest

|  |  |
| --- | --- |
| Hyperparameter tuned | Tuned value |
| bootstrap | True |
| contamination | 0.005354957160342718 |
| max\_features | 24 |
| max\_samples | 0.7 |
| n\_estimators | 100 |

Using gridsearchCV, the stratified folds of the training set were each tested to find the best hyperparameters for the data set using the Isolation Forest algorithm. Notably, this algorithm was trained on f1\_macro since it was available – unlike XGBoost which was trained on AUC. Also, these models exhibit very high accuracy as numerous examples (103192) in the data set were correctly attributed to not being an insider threat userDate; the majority class. Another item that was different in the two algorithms is that the label for isInsider needed to be changed from ‘1’ to ‘-1’ as this is how Isolation Forest records abnormalities.

As with XGBoost, the test data set consisted of ten (10) insider threat userDates from two (2) users. The algorithm predicted two (2) of these userDates to be abnormal. Both predicted userDates were also from the same user, so one insider threat label was not detected at all. This insider threat accuracy was also only achieved by increasing the contamination hyperparameters, which also increased the number of non-insider threats predicted to be anomalous. In detecting these two (2) events and achieving the highest ratio of actual insider threats to predicted insider threats, the algorithms also predicted 579 other anomalies that were not insider threats [false positives].

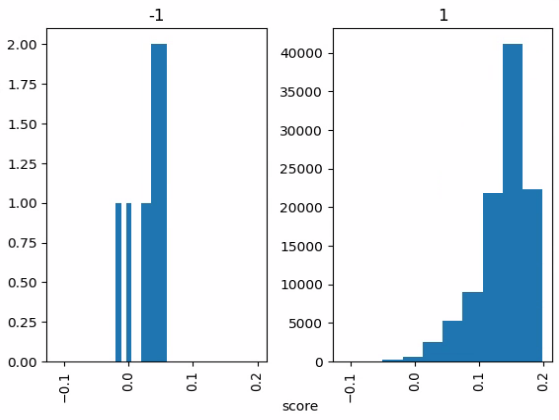


Figure 9: Distribution comparison between normal (1) and anomalous (-1) userDates by Isolation Forest score.

The above histogram shows that most samples score between 0.1 and 0.2 and are labeled as normal. A smaller set of items fall below 0 and are labeled as anomalous. For those labeled as insiders with –1 [title of Figure 9], only two (2) observations fall below 0 [histogram on left]. Numerous non-insider labeled events (579) labeled 1 [histogram on right], also fall below 0.

Table 9: Prediction and score for insiders detected from Isolation Forest.

|  |  |  |
| --- | --- | --- |
| User\_id [index] | Score | Prediction |
| CCH0959 [434186] | -0.003913 | -1 |
| CCH0959 [434213] | -0.019784 | -1 |

The contamination hyperparameter is especially relevant as it provides the expected number of anomalies in the data set (Carletti, 2019). Ideally, this value would be the number of userDates with the isInsider set divided by the total number of userDates in the data set. When this ideal contamination was tested with the test data set, the number of accurate isInsider detections was zero. By increasing the contamination, the zero mark of the histogram is shifted. This shift will envelop more of the distribution of both classes. Better results have been obtained by testing the accuracy of insider threat detections against the contamination hyperparameter.

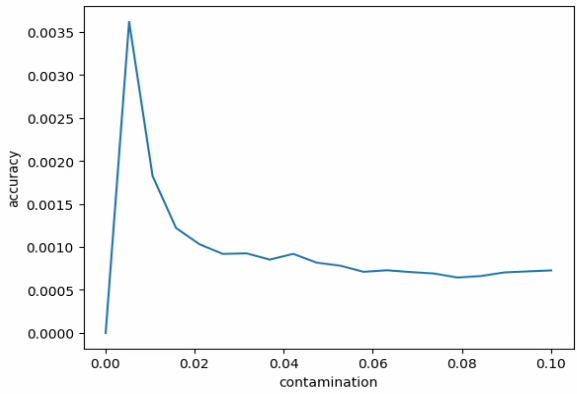


Figure 10: Contamination hyperparameter as compared to accuracy of insider detections.

After the initial hyperparameter search, the contamination factor was evaluated with the other optimized hyperparameters using a finer amount of granularity and based on the accuracy of the detections for the insider class. Based on the results, this optimized contamination factor with the other hyperparameters produced the more accurate insider detections displayed above.

Isolation Forest also suffers from the same interpretability issues as XGBoost. To provide some interpretation of the results of an Isolation Forest, the SHAP (Shapley Additive exPlanations) library was used (Lundberg, 2018). This library assigns credit to the features involved in making the determinations for the Isolation Forest (Kartha et.al., 2021). After calculating the expected values in SHAP, the library can detail each feature's contribution.

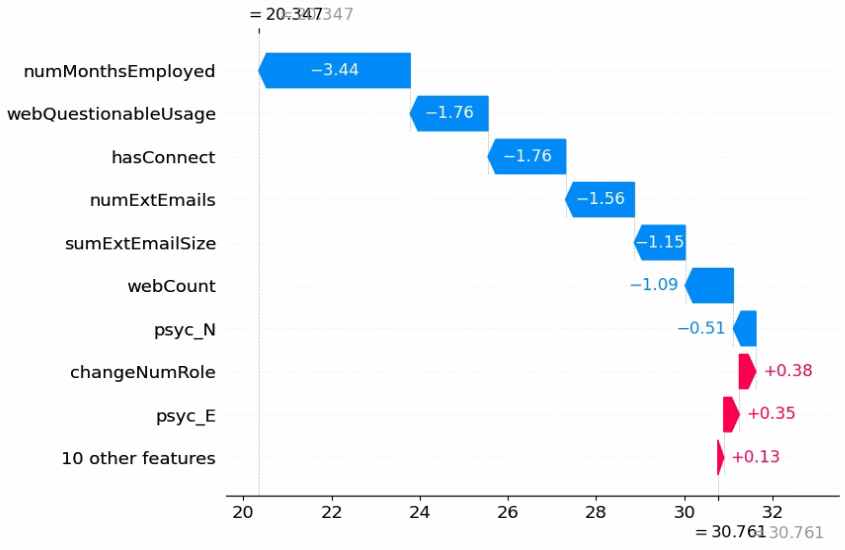


Figure 11: Waterfall chart depicting feature importance of insider (-1) anomalous detection from Isolation Forest.

When a CHAP waterfall plot was constructed from an abnormal result – Figure 11, we can immediately see the features that had the most significant impact on the decision that the observation was abnormal. The image in Figure 11 represents CCH0959\_20100819. This observation was labeled as an insider threat and was correctly predicted by the Isolation Forest. From the Figure, we see the numMonthsEmployed and webQuestionableUsage both have a high degree of influence in this decision. From this row item, numMonthsEmployed is only 9, with a mean of 17.1 and a mode of 18 for the entirety of userData in the data set. WebQuestionableUsage has a mean of 3.6, while this row entry is 26. Finally, hasConnect has a mean of 0.4, while this userDate has a value of 3.0. This combination of features has undoubtedly made the entry abnormal, and it can be visually evaluated that the features impact has pushed it away from the expected value [bottom value of waterfall graph].

For comparison, an example is provided of a waterfall chart of a normal user who was not labeled as an insider threat [Figure 12]. In this example, the features impact on the user's score [top] falls near the expected score [bottom]. It is also seen that the values associated with the features are small in comparison to the waterfall chart of the abnormal userDate [Figure 11].

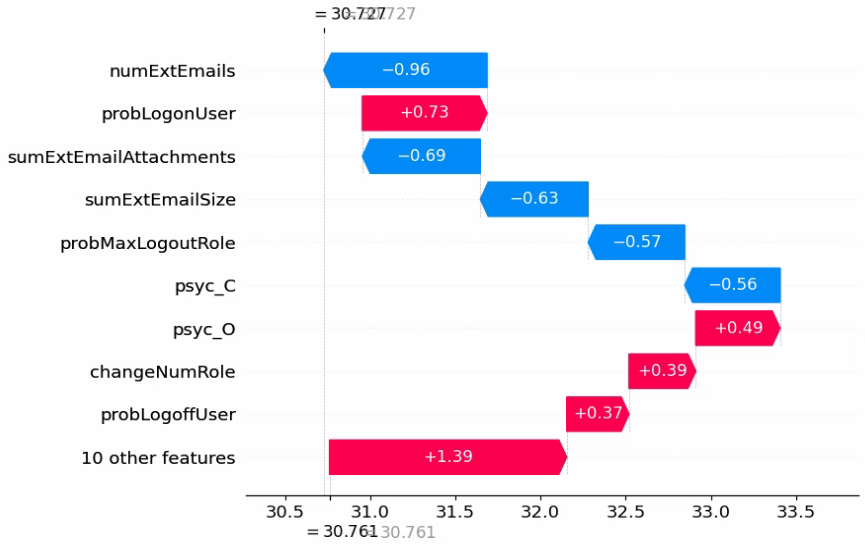


Figure 12: Waterfall chart of feature importance for normal userDate from Isolation Forest - for contrast of anomaly

# Chapter 5: Conclusions

## Summary

The research questions of this paper were answered in the context of this data set as collected from a medium-sized business. As demonstrated, both algorithms were able to detect insider threats with incomplete accuracy. These capacities were limited by how the algorithms functioned.

For XGBoost, working as a supervised machine learning algorithm, this constraint is found in having labeled data that the algorithm can use to find similar instances based on the features of the user sessions. With this information, the algorithm performs adequately and has detected both scenarios in this data set. When the hyperparameters were optimized, XGBoost found most of the user sessions labeled as insider threats in the data set as well. As the algorithm performed these detections, there were few errors, and those errors made were explainable. In using this ensemble decision tree-based algorithm, XGBoost found the most influential features to detect insider threats. Methods in the algorithm reported these features, which can be used by responding analysts. Analysts can use this information to diagnose the detection and react accordingly without exposing the business to accusations of employee discrimination and limiting the impact that the threat might have. While this forensic methodology has looked at historic system logs, the model generated from this examination can also be used on near real-time features collected.

The Isolation Forest algorithm was also tested on the extracted features of the insider threat data set. Isolation Forest is an unsupervised machine learning algorithm that does not require labeled data to understand anomalies. Isolation Forest detected fewer insider threat events and only one of the scenarios found in the test data set. In order to make these detections, many non-insider threat anomalies were also found when passing the same features to this algorithm. Interpreting these detections can be accomplished with little effort, and feature importance can be derived. Understanding these features' impact allows for detailed analysis of the detection, allowing a medium-sized business to interpret and triage it. This forensic examination of the data set and the model generated has detected the presence of insiders. Still, the algorithm has also generated numerous false positives, which a business will also need to research – a process that will take resources and time away from other activities and may not merit much actionable results.

## Conclusions

As a result of this research, the following conclusions can be made in direct association with the research questions of this paper:

* + A medium-sized business that generates audit logs can also extract features from these logs. When paired with algorithms that select important features, insider threat detections can be made. These detections will reduce the risk and impact presented by insider threats.
  + The XGBoost and Isolation Forest algorithms can detect insider threats in a heavily imbalanced data set where one class is a significant minority in the data set. These algorithms determine the most influential features. A business can interpret this feature importance and reduce the risk in acting. Both algorithms have drawbacks associated with their use.
  + Analysts can use the output from feature importance to diagnose and triage insider threats. The features that had the most impact on predicting an insider threat were behavior-based. An organization might use this insight to increase monitoring of employees’ activities on business assets.
  + Since the features were compiled from the data sets and the feature importance was detailed from the algorithm, a medium-sized business should be able to interpret the meaning of the detections. When paired with the detection itself, this understanding produces business value through risk mitigation and insider threat incident response.
  + Throughout this paper, a forensic methodology was developed and used to detect insider threats. This methodology is modular and can be used by businesses to build and include features. These features can be tailored to the organization’s use and threats in the environment. Due to the non-parametric and feature selection capabilities of the ensemble decision tree algorithms reviewed, a business will not require much data science knowledge to use these results.

## Recommendations and Future Direction

While this project has addressed the research questions presented, the project created other questions for potential future papers. Due to the scope and time limitations, these questions could not be addressed as it might delay the delivery of the capstone project. The potential topics identified have been captured here:

* This paper has studied the impact of a selected number of extracted parameters developed from the personal experience of a single cybersecurity analyst after studying the scenarios found in the data set. Incorporating other analyst opinions and collecting additional features may impact the accuracy of these algorithms. Since the algorithms are able to select the most relevant features, increasing the number of features should have little negative influence.
* CMU maintains more than a single data set used for insider threat detection. While this project only focused on one, other projects might train their models on one data set and then test these models on another one. This would allow the model to be evaluated on scenarios completely unseen and unrelated to the scenarios in which the model was trained. Researchers would be able to understand the effectiveness of models where behaviors might not precisely match those in which it has been trained.
* In the course of this project, the Isolation Forest was seen to generate numerous false positives to become partially effective at detecting insider threats. Should a business pay consideration to these anomalies, they would effectively create a labeled and more balanced dataset for detecting insider threats. Research into how Isolation Forest and XGBoost might be used in a cooperative manner could take place from this data set. There would also be increased value to the organization in understanding their anomalies as well as insider threats in the environment.

# References

Al-Mhiqani, M. N., Ahmad, R., Zainal Abidin, Z., Yassin, W., Hassan, A., Abdulkareem, K. H., Ali, N. S., & Yunos, Z. (2020). A Review of Insider Threat Detection: Classification, Machine Learning Techniques, Datasets, Open Challenges, and Recommendations. In Applied Sciences (Vol. 10, Issue 15, p. 5208). MDPI AG. <https://doi.org/10.3390/app10155208>

Ambre, A., & Shekokar, N. (2015). Insider Threat Detection Using Log Analysis and Event Correlation. In Procedia Computer Science (Vol. 45, pp. 436–445). Elsevier BV. <https://doi.org/10.1016/j.procs.2015.03.175>

Berdal, S. J. (2018). A holistic approach to insider threat detection. (Master's thesis). Retrieved from http://urn.nb.no/URN:NBN:no-68010

Brownlee, J. (2021). Imbalanced Classification with Python (1.2 ed.). Independently published.

Carletti, M., Masiero, C., Beghi, A., & Susto, G. A. (2019). Explainable Machine Learning in Industry 4.0: Evaluating Feature Importance in Anomaly Detection to Enable Root Cause Analysis. In 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC). 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC). IEEE. <https://doi.org/10.1109/smc.2019.8913901>

Carnegie Mellon University. (2013). Unintentional Insider Threats: A Foundational Study. Carnegie Mellon University. <https://doi.org/10.1184/R1/6585575.V1>

Carnegie Mellon University. (2023). Software Engineering Institute. Insider Threat. Retrieved March 5, 2023, from <https://www.sei.cmu.edu/our-work/insider-threat/#:~:text=Insider%20Threat%20%5Bis%5D%20the%20potential,Enterprise%20Threat%20and%20Vulnerability%20Management>

Center for Internet Security. (2021, November 29). CIS Critical Security Controls navigator. CIS Critical Security Controls Navigator. Retrieved March 15, 2023, from <https://www.cisecurity.org/controls/cis-controls-navigator/>

CERT National Insider Threat Center. (2022). Common Sense Guide to Mitigating Insider Threats, Seventh Edition. Carnegie Mellon University.

Chen, T., & Guestrin, C. (2016). XGBoost. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD ’16: The 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM. <https://doi.org/10.1145/2939672.2939785>

Cisco. (2023). Cisco Umbrella. Retrieved April 15, 2023, from <https://umbrella.cisco.com/>

Clark, J. (2018, August 29). Challenges Facing Insider Threat Programs and Hub Analysts. Retrieved March 15, 2023, from <http://insights.sei.cmu.edu/blog/challenges-facing-insider-threat-programs-and-hub-analysts-part-1-of-2/>

Costa, D. L., Albrethsen M. J., Collins M. L., Perl S. J., Silowash G. J., & Spooner D. L. (2016). An Insider Threat Indicator Ontology. TECHNICALREPORT CMU/SEI-2016-TR-007. Pittsburgh, PA: SEI

Conway, D. (2010, September 30). The data science Venn diagram. THE DATA SCIENCE VENN DIAGRAM. Retrieved February 20, 2023, from <http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram>

Cybersecurity and Infrastructure Security Agency. (n.d.). *Defining insider threats: CISA*. Defining Insider Threats. Retrieved March 15, 2023, from <https://www.cisa.gov/defining-insider-threats>

Cybersecurity and Infrastructure Security Agency. (n.d.). Insider threat mitigation. Insider Threat Mitigation. Retrieved March 5, 2023, from <https://www.cisa.gov/topics/physical-security/insider-threat-mitigation>

Fischbein, J. (2022, May 10). *Insider threats on the rise due to the great resignation*. World Economic Forum. Retrieved April 17, 2023, from <https://www.weforum.org/agenda/2022/05/insider-threats-how-the-great-resignation-is-impacting-data-security/>

Glasser, J., & Lindauer, B. (2013). Bridging the Gap: A Pragmatic Approach to Generating Insider Threat Data. In 2013 IEEE Security and Privacy Workshops. 2013 IEEE CS Security and Privacy Workshops (SPW2013). IEEE. <https://doi.org/10.1109/spw.2013.37>

Greitzer, F.L., Imran, M., Purl, J., Axelrad, E.T., Leong, Y.M., Becker, D.E., Laskey, K.B., & Sticha, P.J. (2016). Developing an Ontology for Individual and Organizational Sociotechnical Indicators of Insider Threat Risk. Semantic Technologies for Intelligence, Defense, and Security. Available from: <https://www.researchgate.net/publication/310751829_Developing_an_Ontology_for_Individual_and_Organizational_Sociotechnical_Indicators_of_Insider_Threat_Risk> [accessed Mar 01 2023].

Greitzer, F. L., Strozer, J. R., Cohen, S., Moore, A. P., Mundie, D., & Cowley, J. (2014). Analysis of Unintentional Insider Threats Deriving from Social Engineering Exploits. In 2014 IEEE Security and Privacy Workshops. 2014 IEEE Security and Privacy Workshops (SPW). IEEE. <https://doi.org/10.1109/spw.2014.39>

(ISC)². (2022, October 17). (ISC)² 2022 Cybersecurity Workforce Study. Retrieved February 21, 2023, from <https://www.isc2.org//-/media/ISC2/Research/2022-WorkForce-Study/ISC2-Cybersecurity-Workforce-Study.ashx>

Kartha, N. S., Gautrais, C., & Vercruyssen, V. (2021). Why Are You Weird? Infusing Interpretability in Isolation Forest for Anomaly Detection (Version 1). arXiv. <https://doi.org/10.48550/ARXIV.2112.06858>

Kim, M., Kim, K., & Lee, H. (2018). Development trend of insider anomaly detection system. In 2018 20th International Conference on Advanced Communication Technology (ICACT). 2018 20th International Conference on Advanced Communications Technology (ICACT). IEEE. <https://doi.org/10.23919/icact.2018.8323762>

Lin, L., Zhong, S., Jia, C., & Chen, K. (2017). Insider Threat Detection Based on Deep Belief Network Feature Representation. In 2017 International Conference on Green Informatics (ICGI). 2017 International Conference on Green Informatics (ICGI). IEEE. <https://doi.org/10.1109/icgi.2017.37>

Lindauer, B. (2020). Insider Threat Test Dataset [Data set]. Carnegie Mellon University. <https://doi.org/10.1184/R1/12841247.V1>

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation Forest. In 2008 Eighth IEEE International Conference on Data Mining. 2008 Eighth IEEE International Conference on Data Mining (ICDM). IEEE. <https://doi.org/10.1109/icdm.2008.17>

Liu, L., De Vel, O., Chen, C., Zhang, J., & Xiang, Y. (2018). Anomaly-Based Insider Threat Detection Using Deep Autoencoders. In 2018 IEEE International Conference on Data Mining Workshops (ICDMW). 2018 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE. <https://doi.org/10.1109/icdmw.2018.00014>

Liu, L., Chen, C., Zhang, J., De Vel, O., & Xiang, Y. (2019). Insider Threat Identification Using the Simultaneous Neural Learning of Multi-Source Logs. In IEEE Access (Vol. 7, pp. 183162–183176). Institute of Electrical and Electronics Engineers (IEEE). <https://doi.org/10.1109/access.2019.2957055>

Lundberg, Scott (2018). Welcome to the SHAP documentation. Retrieved April 14, 2023, from <https://shap.readthedocs.io/en/latest/index.html>

MITRE. (2022, June 5). Our approach. MITRE Insider Threat Research & Solutions. Retrieved March 5, 2023, from <https://insiderthreat.mitre.org/insider-types/>

MITRE. (2022, October 25). MITRE ATT&CK v12. MITRE ATT&CK. Retrieved March 5, 2023, from <https://attack.mitre.org/>

Modin Developers. (2022). Scale your pandas workflow by changing a single line of code#. Scale your pandas workflow by changing a single line of code - Modin. Retrieved April 15, 2023, from <https://modin.readthedocs.io/en/stable/>

Musthaler, L. (2008, June 2). 13 best practices for preventing and detecting insider threats. Network World. Retrieved April 3, 2023, from <https://www.networkworld.com/article/2280365/13-best-practices-for-preventing-and-detecting-insider-threats.html>

NumFOCUS, Inc. (2023). Pandas. pandas. Retrieved April 15, 2023, from <https://pandas.pydata.org/>

Padayachee, K. (2022). An Exploratory Factor Analysis of Personality Factors: An Insider Threat Perspective. In Human Aspects of Information Security and Assurance (pp. 253–264). Springer International Publishing. <https://doi.org/10.1007/978-3-031-12172-2_20>

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.

Ponemon Institute. (2022, February 7). 2022 Cost of Insider Threats Global Report. Proofpoint. Retrieved January 18, 2023, from <https://www.proofpoint.com/sites/default/files/threat-reports/pfpt-us-tr-the-cost-of-insider-threats-ponemon-report.pdf>

Samy, G. N., Maarop, N., Shanmugam, B., Radhakrishnan, M., Perumal, S., & Rahim, F. A. (2021). Multidimensional Insider Threat Detection Model For Organization. Journal of Theoretical and Applied Information Technology.

Sanzgiri, A., & Dasgupta, D. (2016). Classification of Insider Threat Detection Techniques. In Proceedings of the 11th Annual Cyber and Information Security Research Conference. CISRC ’16: 11th Annual Cyber and Information Security Research. ACM. <https://doi.org/10.1145/2897795.2897799>

Software Engineering Institute. (2023). Software Engineering Institute. Insider Threat. Retrieved February 22, 2023, from <https://www.sei.cmu.edu/our-work/insider-threat/>

scikit-learn. (n.d.). IsolationForest. Retrieved April 13, 2023, from <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html>

Tsukerman, E. (2019). Machine Learning for Cybersecurity Cookbook: Over 80 recipes on how to implement machine learning algorithms for building security systems using Python. Packt Publishing.

Yuan, S., & Wu, X. (2021). Deep learning for insider threat detection: Review, challenges and opportunities. In Computers & Security (Vol. 104, p. 102221). Elsevier BV. <https://doi.org/10.1016/j.cose.2021.102221>

Zou, S., Sun, H., Xu, G., & Quan, R. (2020). Ensemble Strategy for Insider Threat Detection from User Activity Logs. In Computers, Materials &amp; Continua (Vol. 65, Issue 2, pp. 1321–1334). Computers, Materials and Continua (Tech Science Press). <https://doi.org/10.32604/cmc.2020.09649>

# Appendix

## Code repository

Code for the project can be found at <https://github.com/hendt5997/capstone>.