DSA4263 AY23/24

Sense-Making in Business & Commerce Fraud Analytics

Exploring Insider Threat Detection

Chan Yi Ru Micole A0221918E
Jazlynn Tang See Lin A0220263W
Melody Tan A0220708M
Nga Pin En Hannah A0221764H
Srivathsan Amruth A0220753M
Wong Shi An A0223539A

April 15, 2024

Contents

1 Abstract

This study focuses on detecting intellectual property theft fraud using a synthetic dataset for insider threat detection. Malicious actors steal sensitive company data before departing the organisation. An ensemble model comprising One-Class Support Vector Machine, Neural Network, and Decision Tree models with confidence scores indicating the likelihood of fraud is developed. The ensemble method exhibits superior F1 and precision scores compared to individual models. Recommendations include employing the model to detect data theft and abuse of privileges, providing organisations with enhanced capabilities to mitigate insider threat risks and protect valuable assets.

2 Introduction

2.1 Background

Insider threat detection involves identifying and mitigating risks posed by individuals within an organisation with access to sensitive data or systems and may misuse that access. These individuals could be former or current employees, contractors or partners with legitimate access to an organisation's resources. The motivations behind insider threats vary widely and can include financial gain, revenge, or simply carelessness. Reports in 2022 have shown that the number of insider threats incidents increased by 44% within 2 years, with costs per incident increasing by 33% [Ins22].

Insider threat detection typically relies on monitoring and analysing various data sources, including network traffic, system logs, user behaviour analytics, and other metadata. By correlating information from these sources, anomalies and suspicious activities can be identified. Machine learning algorithms are often employed to analyse large datasets and identify anomalous behaviour. For our project, we have chosen a synthetic dataset containing labelled insider threat activities and system logs involving employees in a simulated organisation. Insider threats impact a wide range of industries, as they are not specific to any particular sector but rather a pervasive concern across various organisations. Some examples of impacted industries and how insider threat detection solutions can help them are as follows:

- Finance and Banking: Insider threats can lead to significant financial losses, regulatory violations and reputational damage. Detecting and mitigating insider threats can help financial institutions protect sensitive customer data, prevent fraud, and ensure compliance with regulatory requirements.
- Technology and IT Services: Insider threats pose significant risks including intellectual property theft, sabotage and unauthorised access to proprietary systems. By deploying advanced insider threat detection tools and techniques, organisations in the technology sector can safeguard their innovations, trade secrets and business-critical information.

2.2 Aims and Objectives

Our study aims to achieve the following objectives:

1. Minimizing False Positives (FP): Reduce false malicious user alerts and disruptions for normal staff. Our aim is to optimize the trade-off between True Positives (TP), False Negatives (FN) and FP, thus improving the overall efficiency of the detection system.
2. Identifying Malicious Users in the following scenario: User begins surfing job websites and soliciting employment from a competitor. Before leaving the company, they use a thumb drive (at markedly higher rates than their previous activity) to steal data. (Further elaborated in Section 3)
3. Enhancing Explainability: We elucidate our detection system via post model evaluation and incorporating metrics like confidence scores. This will enable stakeholders to understand the model's decision-making process, particularly crucial for sensitive tasks like insider threat detection.

3 Data

There is a lack of real data publicly available due to the confidential nature of employee information. We have thus chosen a synthetic insider threat dataset[Lin20] consisting of system logs and labeled insider threat activities. It was generated by the CERT division of Software Engineering Institute at Carnegie Mellon University based on scenarios containing traitor instances and masquerade activities. The dataset was developed after consulting counter-intelligence experts, using several interdependent systems to create log behaviours of a simulated organisation. It consists of 6 files as described in Figure
Overall, it is important to note that since this dataset is artificial, it may be fully representative of an actual insider threat problem since it contains only a few malicious scenarios. Furthermore, the authors highlighted that the dataset is still cleaner than real-world equivalents, despite efforts to introduce inconsistencies [BJLM21]. Regardless, this dataset is widely used in research.

4 Methodology

In this section, we provide a detailed overview of the methodology employed in our study. We outline the steps taken for data cleaning, pre-processing, resampling methods, exploratory analysis insights, test/train split, model selection, feature engineering, ensemble techniques, feature selection, and parameter tuning.

4.1 Results and Findings

We constructed a novel iterative ensemble method that leverages the use of confidence scoring metrics to generate final predictions compared to most industry solutions, which only flag and provide a binary outcome regarding malicious behaviour. This yields a new paradigm regarding such problems as we can dynamically readjust FP, FN and TP via tuning a ensemble and threshold parameter.

4.2 Data Cleaning and Pre-processing

In data cleaning, our goal was to optimize the dataset for analysis, removing superfluous features, filtering records, and standardizing formats for analysis and feature engineering. Given our dataset's extensive size, over 11GB, it was crucial to reduce and refine the data extensively to ensure loadability and manageability for processing.

4.2.1 Files Dataset

Focusing on high-risk activities within the FILES dataset pertinent to Scenario 2 and data exfiltration risks. This refinement process involved removing columns that were not directly relevant, such as **from_removable_media**, **pc**, and **content**. We then precisely filtered the records to only include those where the activity labeled was ”File Write” and the file was transferred to removable media—two critical indicators of potential data leakage.

4.2.2 HTTP Dataset

Our preprocessing aimed to isolate web traffic that could indicate suspicious activities. We removed non-essential columns like **pc** and **content**, and categorized URLs into job search websites, cloud storage services, and other suspicious domains. This classification helps in later stages to identify users possibly preparing to leave the company or engaging in unauthorized data sharing.

4.2.3 Device, LDAP, Logon Datasets

The DEVICE, LDAP and LOGON datasets were not utilized as they did not provide additional insights beyond what was already available from the other datasets.

4.3 Feature Engineering

We engineered a comprehensive set of features aimed at capturing the genearlised nuances of user behavior and enhancing the predictive power of our models. These features help to identify actions that deviate from normal work patterns, quantify interactions with potentially risky websites, and integrate personality traits that may correlate with risky behaviors.

Usage of Dask for Data Handling
To efficiently handle large datasets, Dask[R+15] was employed as it provides capabilities to work with big data using parallel computing:

4.3.1 Behavioral Features

From HTML Dataset:
First, we extensively quantified user interactions with specific types of web domains, which are critical indicators of user intent and risk profile. For instance:

- **Job Domain Website Activity (JD)**: This metric counts HTTP requests to job-related sites like LinkedIn, Indeed, and Monster. It serves as an indicator of users potentially preparing to leave the organization, correlating with an increased risk of data exfiltration.
- **Cloud Storage Domain Website Activity (CD)**: Accesses to cloud storage domains, such as Dropbox and Google Drive, were tallied to flag potential unauthorized data sharing or storage of sensitive information outside corporate environments.
- **Suspicious Domain Website Activity (SD)**: Interactions with domains that may be related to hacking or surveillance tools—indicated by keywords such as 'spy', 'leak', or 'hack'—were counted to identify possible malicious activities.

From FILE Dataset:
We then quantified file operation activity:

- **File Copy to Removable Media (FC)**: The number of times sensitive files were copied to removable media was recorded, serving as a strong indicator of potential data leakage, particularly when conducted without clear business justification.

4.3.2 Temporal Features

Building on the behavioral indicators, we further refined our feature set by incorporating temporal information to capture context and intent more accurately:

- **(Outside) Work Hours Activity (*(o)wh*)**: A binary feature identifying whether activities, such as file copies (*FC*) to removable media or accessing potentially risky HTTP domains, occurred outside of standard **work hours** (Weekdays 8:00 AM to 5:00 PM). This feature helps flag actions taken during times when there is less oversight and potential for covert operations.
- **Weekend Activity (*wke*)**: Similar to the above, this feature denotes whether such activities took place over the weekend, further highlighting unusual work patterns that could be indicative of malicious intent.

From these temporal and behavioral analyses, we derived a set of features—FCwke, FCowh, FCwh, SDwke, SDowh, SDwh, CDwke, CDowh, CDwh, JDwke, JDowh, JDwh—that provide a comprehensive profile of user activity, painting a clearer picture for anomaly detection and risk assessment.

4.3.3 Psychometric Features

Features derived from psychometric scores include:

- **(O)**, **(C)**, **(E)**, **(A)**, **(N)**: Numerical features representing the scores for each of the OCEAN (Openness, Conscientiousness, Extraversion, Agreeableness, Neuroticism) personality traits from 0-100.

4.3.4 Final Feature-Engineered Dataset

All datasets were then merged into a master frame, carefully managing null values to maintain data integrity. This comprehensive dataset was then cast into optimized data types before being compiled into a final Parquet file for modeling. The boolean, integer, and string data types were selectively applied to each feature based on the maximum value analysis, reducing storage space and enhancing processing efficiency.

The end product of this meticulous process is a feature-rich dataset, streamlined for building robust predictive models that can identify potential insider threats by analyzing a wide array of behavioral signals and psychometric parameters.

4.4 Exploratory Analysis Insights on Post-FE Dataset

This section provides insights into the dataset after feature engineering, highlighting key findings and their implications for fraud detection.

4.4.1 Class Imbalance

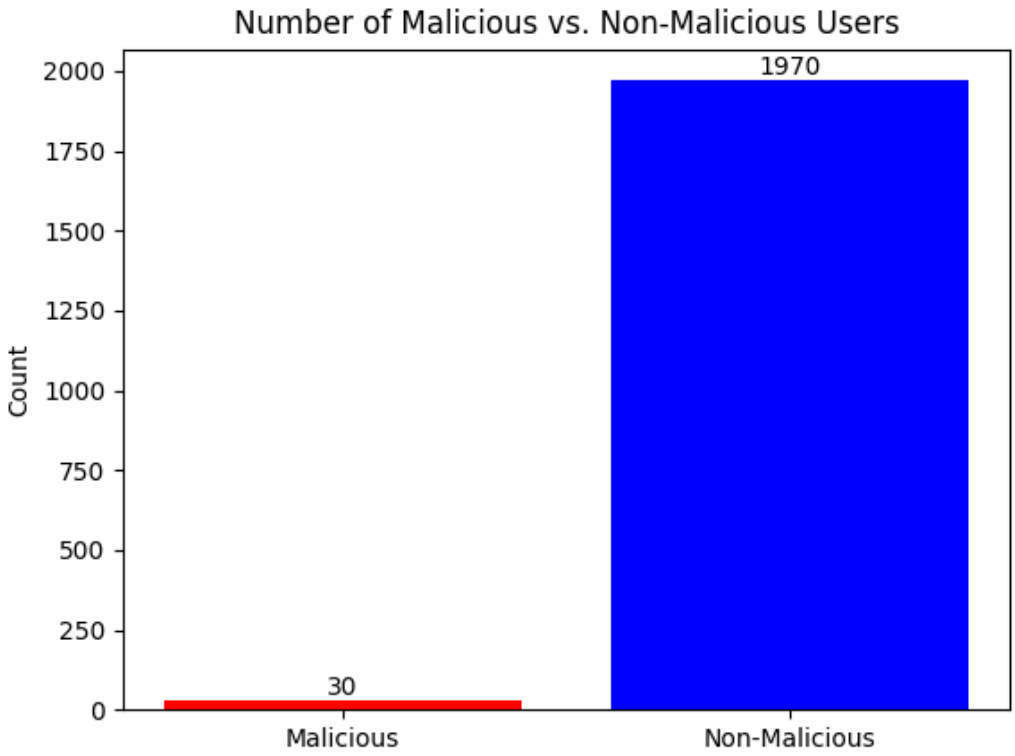


Figure 1: Class imbalance: Malicious vs Non-Malicious Count

A stark class imbalance is evident[1], with 1970 non-malicious users compared to only 30 malicious users. Given this disparity, implementing oversampling techniques like SMOTE is crucial to balance the dataset and improve the performance of the fraud detection model.

4.4.2 Complex Nature of Human Behavior

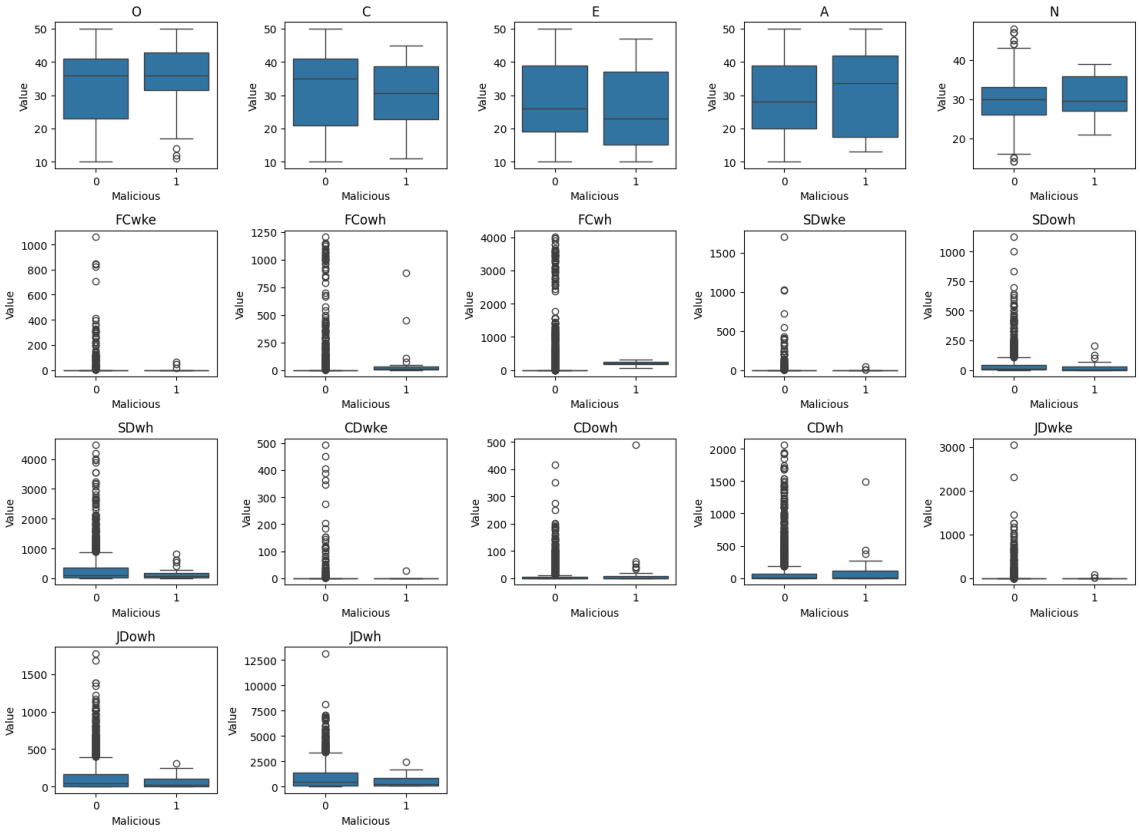


Figure 2: Comparison of Features: Malicious vs. Non-Malicious Users

Findings:
Comparatively[2]:

- **File Copy Activity:** Malicious users copy files less on weekends (Mal: -3.17 vs. Non-Mal: 6.25).
- **Website Activity:** Malicious users visit less in suspicious domain (Mal: 25.53 vs. Non-Mal: 40.15). and job domain (Mal: 4.13 vs. Non-Mal: 23.09) websites on weekends.

Insights:
These findings significantly challenge conventional assumptions about the behavior of malicious users. The observed unexpected patterns in file copy, suspicious domain, cloud

domain, and job domain activities reveal that user behavior is more complex and nuanced than traditionally anticipated.

4.4.3 Deep Dive into Personality Traits

This section explores personality traits to identify patterns of fraudulent behavior using a Personality Similarity Graph Network. The analysis involves several key steps:

- **Graph Construction:**
  - Nodes representing users are created in a graph with malicious users having larger nodes.
  - Edges are formed between users based on their similarity in personality traits, calculated using the Manhattan distance metric.
- **Community Detection and Analysis:**
  - The graph utilizes the greedy modularity communities method to detect communities, which are then analyzed and visualized.
  - This visualization highlights interconnected clusters, potentially indicating groups of users with similar behavioral traits.
- **Visualization:**
  - A network graph displays the nodes with color coding by community and size differentiation based on the maliciousness of the user behaviors.

Insights:
The analysis of the Personality Similarity Graph Network[3] reveals distinct personality trait differences between Community 0 and Community 1, with contrasting levels of OCEAN. Despite these differences, both communities exhibit a high proportion of malicious users. Basic features like personality traits or file activity alone may not suffice, emphasizing the importance of advanced analytics for enhancing fraud detection accuracy.

4.5 Hypotheses Formulation

This leads us to the two hypotheses formulated regarding the nature of fraud in our dataset.

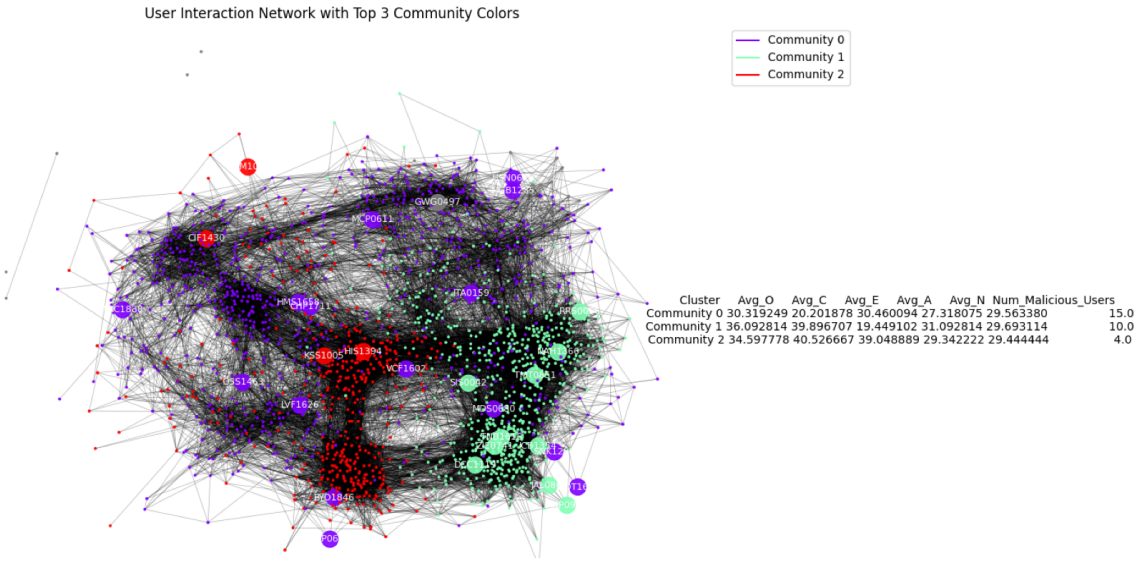


Figure 3: Personality Trait Similarity Network

4.5.1 Hypothesis 1

Since the dataset is heavily imbalanced, adopting oversampling methods for the minority class of the dataset should improve model performance significantly.

4.5.2 Hypothesis 2

A more complex machine learning algorithm could unveil more insights into the subtle yet complex patterns of human behavior involved in fraud.

4.6 Test/Train Split and Validation Approach

The dataset was randomly split into 80% training and 20% testing. The 20% testing data serves as the holdout set to provide an unbiased estimate of the model's performance on unseen data. During hyperparameter tuning via Grid Search, 5-fold cross validation was employed to select optimal parameters.

4.7 Resampling Methods

With a significantly imbalanced train dataset, Synthetic Minority Oversampling Technique (SMOTE) was adopted to generate synthetic samples for the minority class. This reduces bias towards the majority class and improves model performance across all 3 adapted models. Oversampling increased the minority class to 40% of the majority without introducing excessive synthetic data and potential noise as shown in Figure 4.

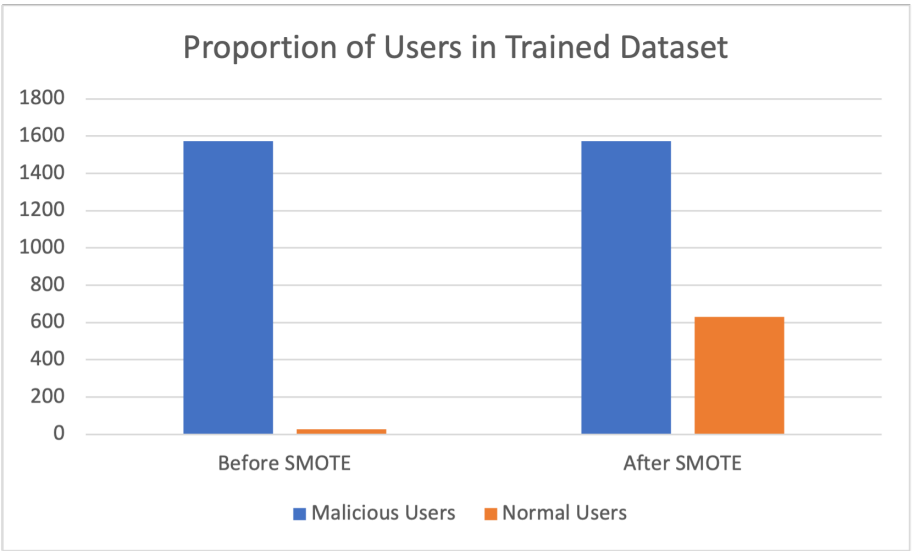


Figure 4: Chart showing Proportion of Users in the Trained Dataset

4.8 Baseline Model: Decision Tree

Decision Tree (DT) was chosen as the baseline model because of its simplicity and interpretability.

4.8.1 Reason for Model Selection

First, DT is well-suited for fraud detection tasks because of their ability to capture nonlinear relationships and complex decision boundaries. Hence, the model can identify subtle patterns indicative of malicious behaviour. Second, the straightforward nature of the model provides a baseline performance benchmark against more complex[4.5.2] challenger models. Third, the model's interpretability is particularly useful for an entirely numerical dataset.

As each decision node represents a specific numerical threshold for a feature, it is easy to understand how the model makes predictions. This enhances model explainability, which is crucial to identify the reasons behind FP and refine the model to reduce such cases. By visualising and tracing decision paths, one can gain insights into the characteristics of malicious users.

4.8.2 Hyperparameter Tuning

The hyperparameters are chosen to handle imbalanced dataset (*class weight*) or control the DT complexity via stopping criteria (*max depth, min samples split, min samples leaf, max features*) and pruning (*ccp alpha*).

To search for the best combination of hyperparameters, Grid Search was performed with 5-fold cross validation. By optimising F1[5.2] score, the performance of minority cases improve via increasing TP as well as decreasing FN and FP.

Through Grid Search, the maximum depth of tree was limited to 10 and the minimum number of samples required at a leaf node increased to 3. Introducing such stopping criteria reduces model complexity and promotes more robust generalization to unseen data. As a result, the model detected one more malicious user correctly and the F1 score improved from 15.38% to 26.67%.

4.8.3 Visualisation

The DT visualisation[5] offers insightful observations as follows.

1. The balanced distribution of colours within the decision nodes suggests that SMOTE improved the model's ability to make balanced predictions for normal (orange colour) and malicious (blue colour) users.
2. The top features (i.e. Fcwh and FCowh) of the DT visualisation underscore their significant influence on the model's predictions.
3. The node statistics such as number of samples, class distribution and Gini scores, provide insights into the model's prediction confidence.

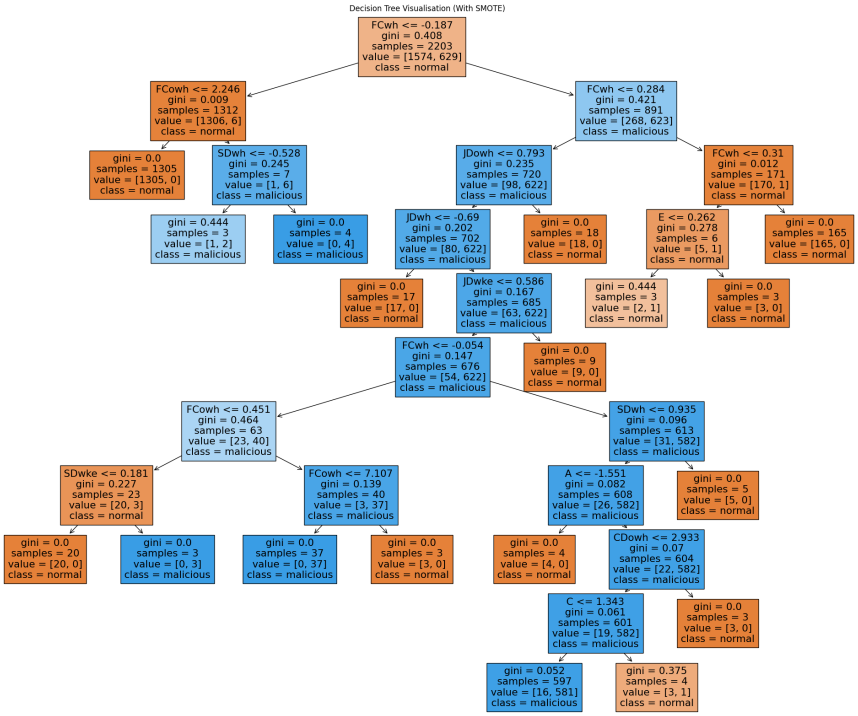


Figure 5: Decision Tree Visualisation

4.8.4 Feature Importance

Feature importance provides model explainability through insights into key behaviours associated with insider threat. It is computed based on how much each feature contributes to reducing Gini impurity at the decision nodes. From the chart[6], the top 3 most important features which contribute to 90% of DT prediction are **FCwh**, **FCowh** and **JDowh**.

4.8.5 Confidence Scores

The confidence score indicates how certain a model is of it's prediction. One idea of a confidence score stems from Gini impurity at a decision node, where higher Gini indicates

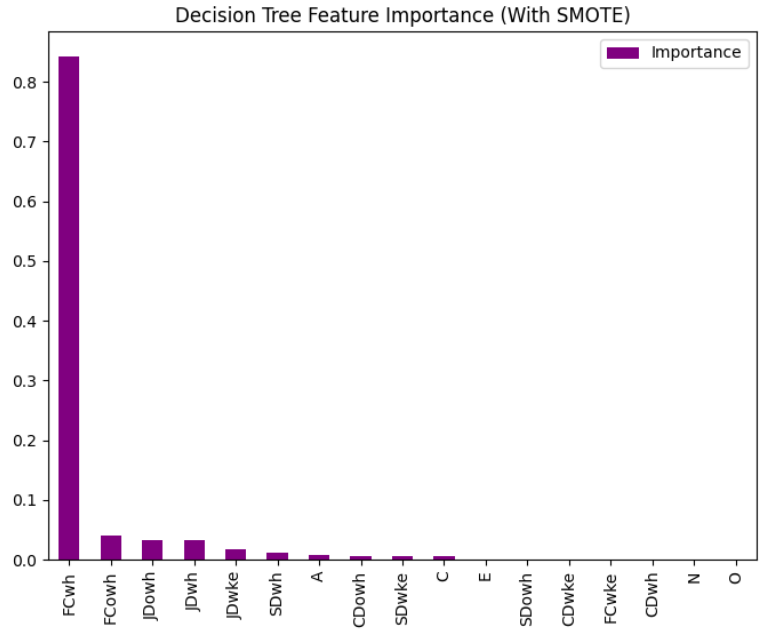


Figure 6: Chart showing Feature Importance for Decision Tree

higher risk in the classification decision made by that node, thereby lowering the confidence score. Hence, we compute confidence score as follows.

1. Weigh the Gini score by the number of samples at a decision node
2. Sum the weighted Gini scores across all nodes in a decision path
3. Min-max scale the result from step 2 from 0 to 1
4. Inverse the result from step 3 since higher Gini score means lower confidence score

We then experimented with other models for this fraud detection task.

4.9 Challenger Model 1: Support Vector Machine

4.9.1 Reason for Model Selection

For our first challenger model, we employed a One-Class Support Vector Machine (OCSVM). OCSVM is an unsupervised algorithm used for anomaly detection. Unlike traditional SVMs, which are typically used for binary classification tasks, OCSVM constructs a decision boundary that encapsulates the normal data points in the feature space, aiming to maximize the margin between this boundary and the nearest normal instances. This boundary effectively separates normal instances from potential anomalies. OCSVM's capability to learn solely from normal data makes it especially suitable for our imbalanced dataset, and provides resilience to high-dimensional datasets and intricate non-linear patterns, thereby aligning well with the characteristics of our dataset. Moreover, we chose this simplistic model to explore our second hypothesis[4.5.2].

4.9.2 Hyperparameter Tuning

We used sci-kit learn's GridSearchCV to perform hyperparameter tuning on our OneClassSVM model with F1 score as the scoring metric. The hyperparameters considered for tuning include the regularization parameter (ν), the kernel type, and the kernel coefficient (γ). F1 score was used as the scoring metric as it considers TP, FP and FN, which we aim to optimize. The model was fitted on normal users only, in order for it to learn the boundary of the normal class. The best performing parameters are {'γ': 1.0, 'kernel': 'sigmoid', 'ν': 0.01} on both OCSVM models with and without SMOTE.

4.9.3 Confidence Scores

The distance of data points from the learned decision boundary serve as a basis for our confidence scoring system. The scores are normalized across all data points. By scaling the scores relative to the minimum and maximum distances observed within the dataset, we establish a standardized range from 0 to 1. where a higher value reflects a greater confidence of prediction.

4.10 Challenger Model 2: Neural Network

4.10.1 Reason for Model Selection

The second challenger model used is a regular Neural Network (NN). The NN is highly effective in detecting anomaly behaviour due to their ability to learn complex patterns and relationships in the data. For the insider threat dataset, malicious users often have behaviour that blends in well with non-malicious users' activities, allowing the NN to learn and reconstruct normal patterns accurately while highlighting anomalies that deviate significantly from the learned representations. This capability is crucial for detecting insider threats, as malicious users often exhibit behavior that is nuanced and context-dependent, reflective of the high-dimensional dataset, requiring a deep understanding of normal user behavior to identify suspicious activities effectively. Additionally, the adaptability of NN allows them to continuously learn and update their understanding of what constitutes anomalous behavior, making them robust against evolving insider threat tactics.

4.10.2 Hyperparameter Tuning

Hyperparameter tuning was done by testing out different combinations of the dimensions for the first and second layers of the NN, iterating the different dimensions over different learning rates to find the best parameters for training. For threshold value tuning, we chose to create a function that directly maximises TP while minimising FP and FN. Different weights are attached to each TP, FP and FN in order to penalise FP and FN by a greater proportion, maximising recall and precision values of the model. For this project, we focus on penalising FP greatly to reduce the number of wrongly classified non-malicious users, while noting the importance of minimising FN in the real world as well.

4.10.3 Confidence Scores

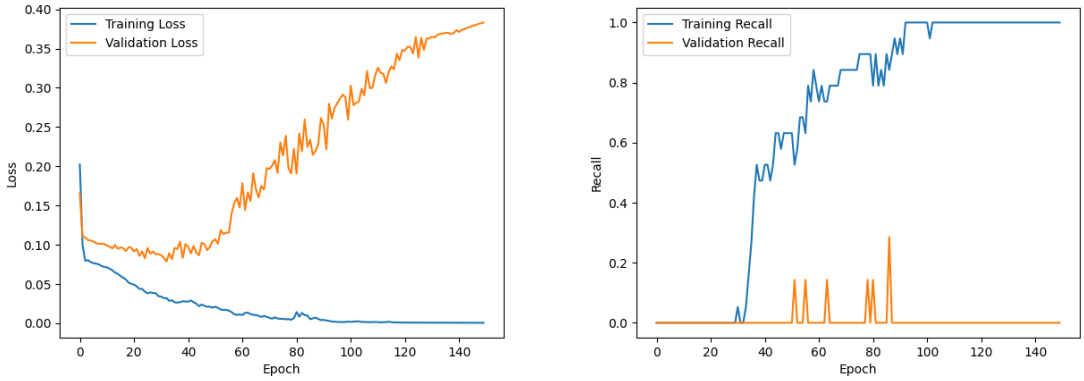
The confidence scores calculated from this model's predictions and a specified threshold represents the probability of successful prediction of each user's real label - malicious or not. The absolute values of each the difference between the threshold and predicted probabilities were taken, followed by scaling to the range of 0 to 1 inclusive to normalised the confidence scores. This scoring algorithm is applied to both the train data and test data with a predetermined threshold that proved to give the best outcome. Similar to the OCSVM model, higher confidence scores indicates greater confidence in the classification outcome for each individual user, and a lower score would indicate otherwise.

4.10.4 Model Performance

Contrary to the previous models, the neural network model was evaluated for its performance based on is loss and recall values.

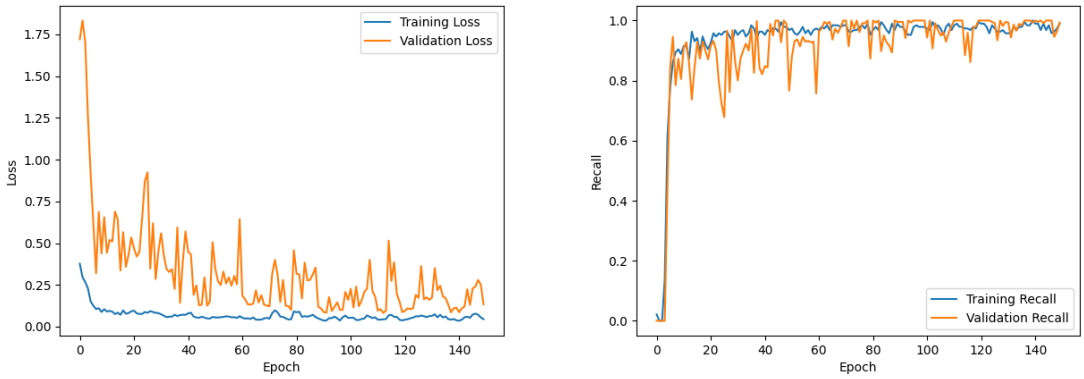
The model was initially trained without SMOTE, according to Figure 7, the model performs well on the training data (decreasing train loss) but poorly on the validation data (increasing validation loss), indicative of the model overfitting on the data. The corresponding recall plot in Figure 7b shows the model's poor performance in successfully identifying TPs consistently for train and validation data, also indicative of an overfitted model.

Integrating SMOTE into the model training proved to give better results, indicative in Figure 8, converging loss and recall plots shows a model that generalised the data well to the unseen data. Together with proper hyperparameter tuning, the model is learning from the training data effectively and better at capturing the nuances of malicious users. The erratic lines for validation loss and recall could mean that the model has high variance, a tendency to overfit the validation data. But concern for this erratic behaviour is expected for a moderately complex neural network that has multiple dense layers.



(a) Loss without SMOTE (b) Recall without SMOTE

Figure 7: Charts showing Neural Network performance without SMOTE.



(a) Loss with SMOTE (b) Recall with SMOTE

Figure 8: Charts showing Neural Network performance with SMOTE.

4.11 Final Model Ensemble

4.11.1 Reason for Model Selection

We chose to implement an Ensemble Model (EM) to incorporate the complementary strengths of our individual models to capitalize on their respective advantages while mitigating their weaknesses. This strategy mitigates the risk of overfitting, ensuring that the model generalizes well to unseen data and maintains high performance.

4.11.2 Hyperparameter Tuning

In our EM, we selected the optimal combination of models by exhaustively evaluating all possible permutations among the 6 distinct models employed previously. Initially, the ensemble was trained to optimize probabilities, prioritizing the maximization of TPs. Subsequently, during training, a threshold fine-tuning process was employed to further enhance the TP count while simultaneously mitigating FPs and FNs, albeit expense of sacrificing some TPs. The classification threshold, t_score, was based on a scoring function:

$$\text{t\_score} = \text{TP} \times \text{tp\_weight} - \text{FP} \times \text{fp\_weight} - \text{FN} \times \text{fn\_weight}$$

The hyperparameters tp_weight, fp_weight, and fn_weight refer to the importance or penalty for each classification error type. Final predictions result from a weighted sum of the selected model's confidence scores.

4.11.3 Confidence Score

Subsequently, confidence scores are calculated using the formula below.

$$C = \sum_i \left(\frac{C_{M_i}^2}{\sum_i C_{M_i}} \right)$$

where each $c \in C$ refers to the confidence score for a single prediction, and $c_{Mi} \in C_{M_i}$ refers to the confidence score given by the i-th selected model in the ensemble for that prediction.

5 Results

5.1 Performance of All Models

The charts[9][10] show the confusion matrix for all models. EM performs the best out of the 7 models, with the highest number of TP as well as lowest FP and FN.

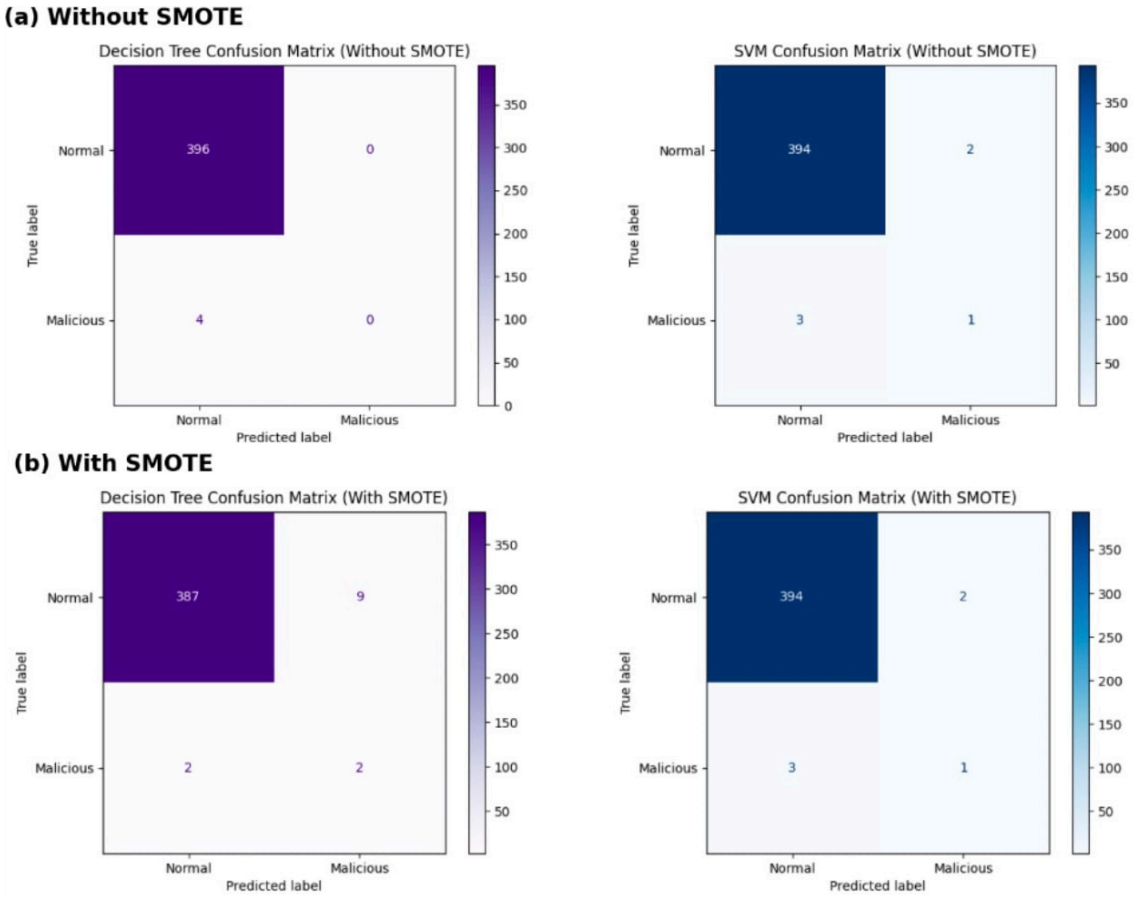


Figure 9: Confusion Matrix for Decision Tree and Support Vector Machine

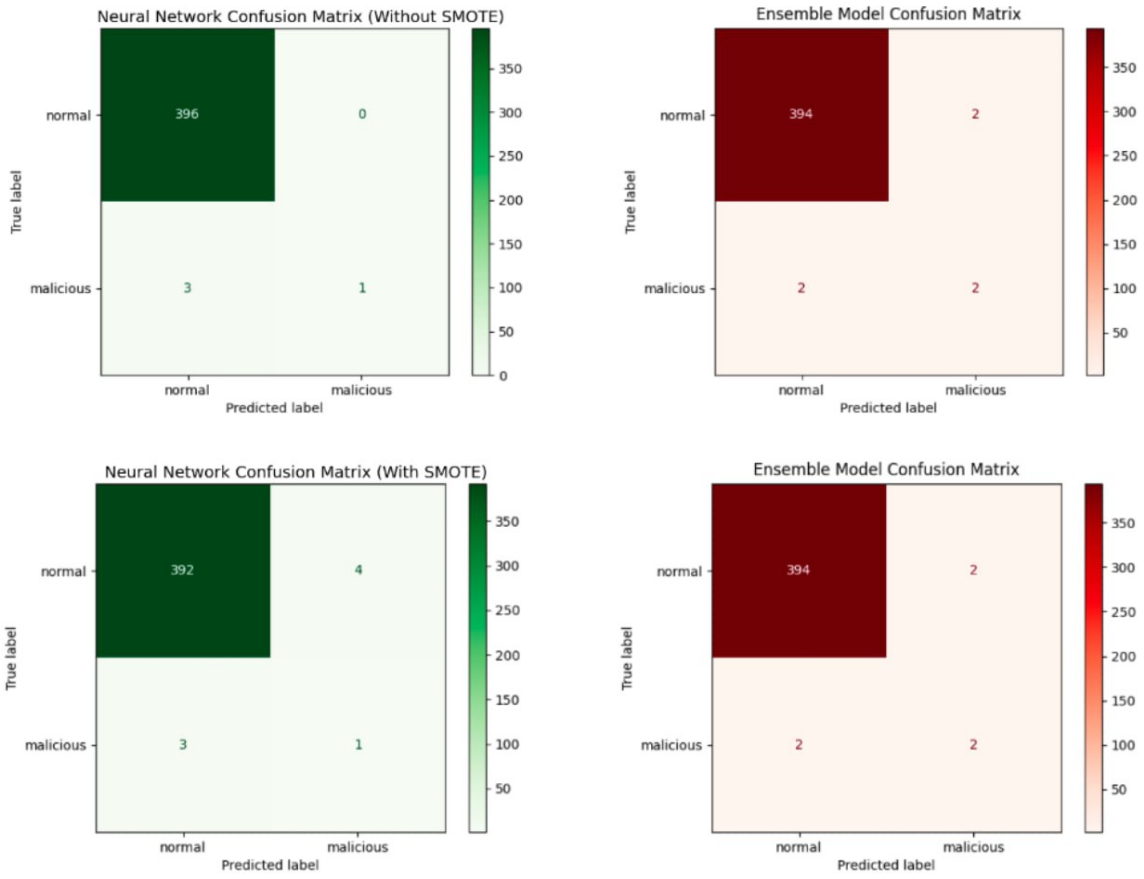


Figure 10: Confusion Matrix for Neural Network and Ensemble Model

5.2 Best Performing Model

Following the model results, we generated Figure 5.4.1 depicting the metrics (Accuracy, Precision, Recall and F1-Score) of each model, values highlighted in bold refer to the highest values for each metric. The neural network model performed best in terms of accuracy and precision, while the decision tree and ensemble model performed best for recall, and ensemble model the best for F1-score. More importance was placed on recall and F1-scores for deciding which model performed best, doing so allows us to identify which model has a greater ability to correctly identify TP without missing out on many of them (higher recall value), and see which models gives the best balance between precision and recall (higher F1-score) , especially with the high imbalance between classes in the dataset.

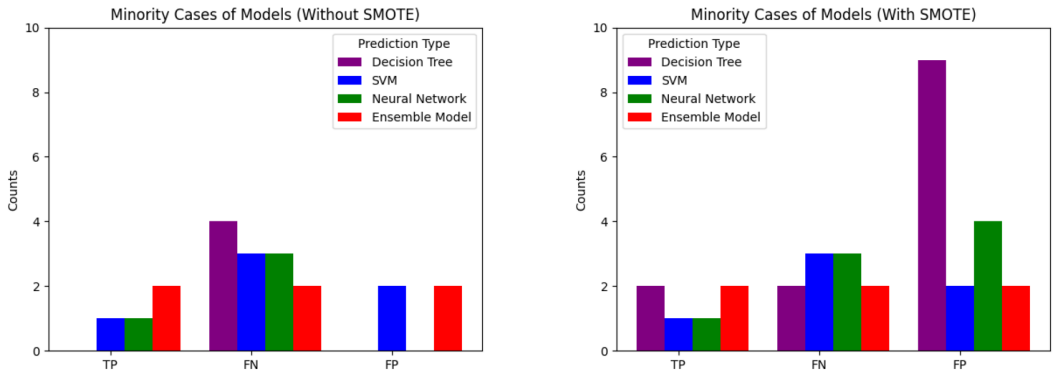


Figure 11: Chart showing Minority Cases of All Models

$$\text{F1-Score} = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \text{Precision} = \frac{TP}{TP + FP}$$

With our EM performing best in terms of recall and F1-scores, we can conclude that it gives a more desirable result in insider threat scenarios. High recall reduces the the more costly error, Type II, from being made, as capturing a lower number of FN can mitigate the severe impacts actual malicious users can bring after being incorrectly detected as non-malicious. EM's higher F1-score proves it has a better balance between being able to correctly identify positive instances among all predicted positives (precision) and to capture a higher proportion of positive instances (recall), concluding itself as the best performing models out of all models.

Amongst all models in Figure 13, the EM ended up giving the highest Area Under Curve (AUC) value that is closes to 1, indicating that the EM can differentiate between the malicious and non-malicious users with a higher degree of accuracy, allow it to be better at assigning **higher probabilities to the true cases compared to the false cases**[2].

	Accuracy %	Precision %	Recall %	F1 Score %
Decision Tree (No SMOTE)	99.00	0.00	0.00	0.00
SVM (No SMOTE)	98.75	33.33	25.00	28.57
Neural Network (No SMOTE)	**99.25**	**100.00**	25.00	40.00
Decision Tree (SMOTE)	97.25	18.18	**50.00**	26.67
SVM (SMOTE)	98.75	33.33	25.00	28.57
Neural Network (SMOTE)	98.25	20.00	25.00	22.22
Ensemble Model	99.00	50.00	**50.00**	**50.00**

Figure 12: Table showing Model Performance

5.3 Model Explainability

5.3.1 Confidence Scores

Comparing the distributions of confidence scores across all 7 models as seen in Figures 14 and 15, DT with SMOTE and NN models had the highest frequency of high confidence predictions. Despite the high confidence scores, this only indicates that these models are highly confident ”guessers”, and has no implication on the correctness of their predictions.

5.3.2 Decision Path Visualisation

Visualising decision paths constructed by the DT provides model explainability by illustrating the features and splitting criteria that led to correct or incorrect predictions. This transparency helps identify areas where the model may be making incorrect assumptions or where certain features are not adequately capturing underlying patterns in the data. Hence, such insights contribute to model refinement for improved performance and trust in predictions.

However, there still exist cases where the decision path visualisation[16] are identical for TP and FP. This implies the existence of certain decision criteria that are not effective in distinguishing the two.

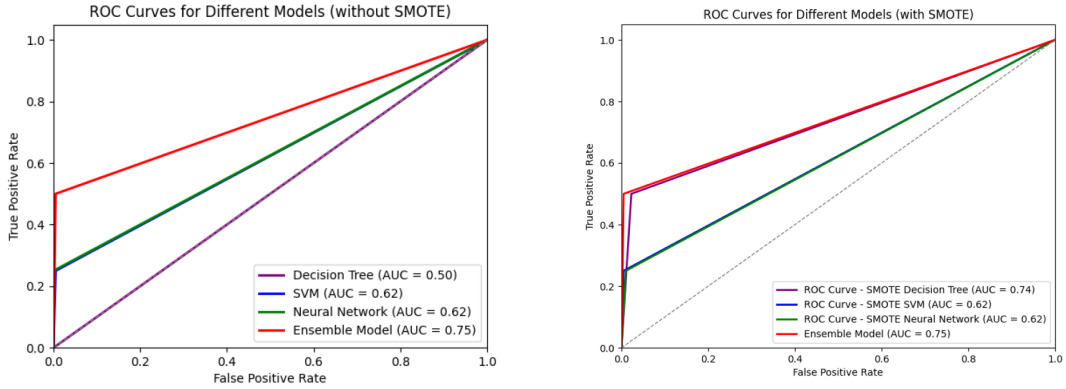


Figure 13: Chart showing ROC Curves for All Models

5.4 Addressing the Hypotheses

5.4.1 Hypothesis 1 4.5.1

Comparing the model performance from Figure 5.4.1, SMOTE improves model performance for DT, worsens performance for NN, and had no effect on OCSVM.

For DT, improved model performance upon resampling was due to increased representation of the underrepresented class during training, enabling the model to learn a more balanced decision boundary, reducing bias towards the majority class. Surprisingly, we observed a decline in performance for the neural network (NN) when using SMOTE compared to its performance without SMOTE, as depicted in Figure . This finding is particularly noteworthy considering that the NN model exhibited signs of overfitting on validation data when SMOTE was not applied.

As OCSVM is designed to detect outliers, it was expected that SMOTE had no impact on model performance due to the resampled points maintaining the same density as before oversampling, and thus constructing the same decision boundary.
Overall, it is proven that our hypothesis of oversampling methods being necessary for balancing the minority does not hold true.

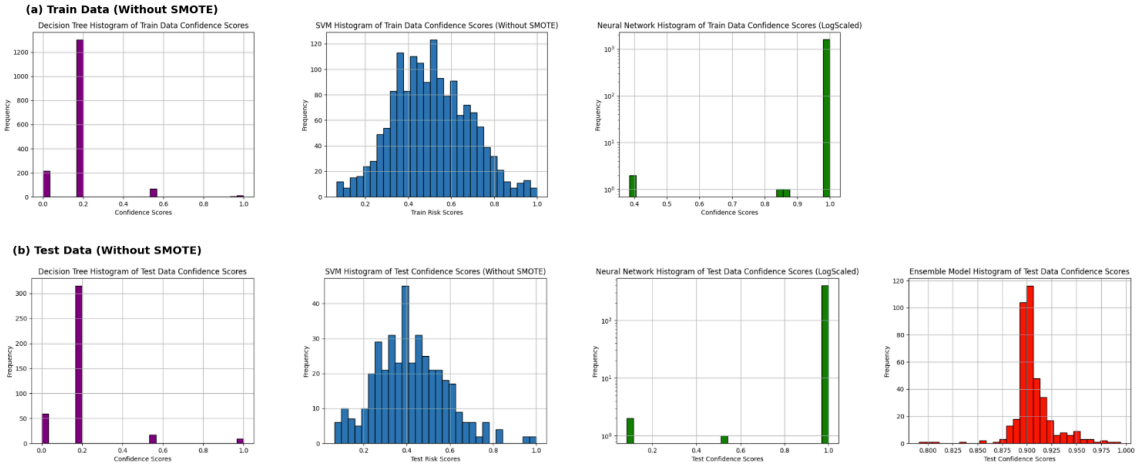


Figure 14: Chart showing Confidence Scores for All Models **(Without SMOTE)**

5.4.2 Hypothesis 2 (4.5.2)

Based on the findings presented in Figure 13, it is evident that the OCSVM model performed poorly with an AUC score of 0.3775. This outcome indicates that our second hypothesis which states human behaviour is too complex to be modelled with simple models, holds true. This reaffirms our results that the NN model outperformed the OCSVM and DT models, as seen from Figure 5.4.1. NN models have complex architectures with multiple layers, each one containing numerous neurons. In contrast, OCSVM typically involves finding a hyperplane that separates data from the origin in the feature space, which is conceptually simpler.

In brief, complex machine learning algorithms are able to unveil the subtle and complex nature of human behavior involved in fraud.

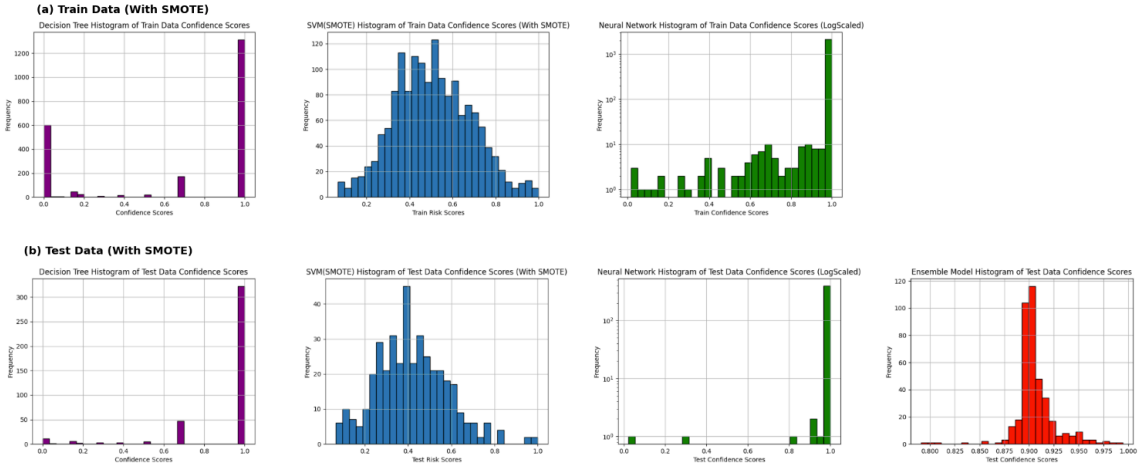


Figure 15: Chart showing Confidence Scores for All Models **(With SMOTE)**

6 Error Diagnosis

False positives have numerous negative consequences such as operational disruption and wasted resources, hence we decided to perform error diagnosis on our models to understand why such errors were made. First, users were separated into True Negatives (TNs), FPs, and TPs, then perform exploratory data analysis to examine which features contribute most to FP misclassifications.

As shown in Figure 17, the graph on the left illustrates the difference in average feature values between FP and TP instances. It can be seen that features with the smallest differences in values between false positive and true positive instances include the number of times cloud storage websites were accessed outside work hours (CDowh) and the number of file copies outside work hours (FCowh). The graph on the right illustrates the difference in average feature values between FP and TN instances. Features showing largest differences in values between FP and TN instances also include CDowh and FCowh. These two graphs indicate that the features which contribute most to false positive predictions include CDowh and FCowh.

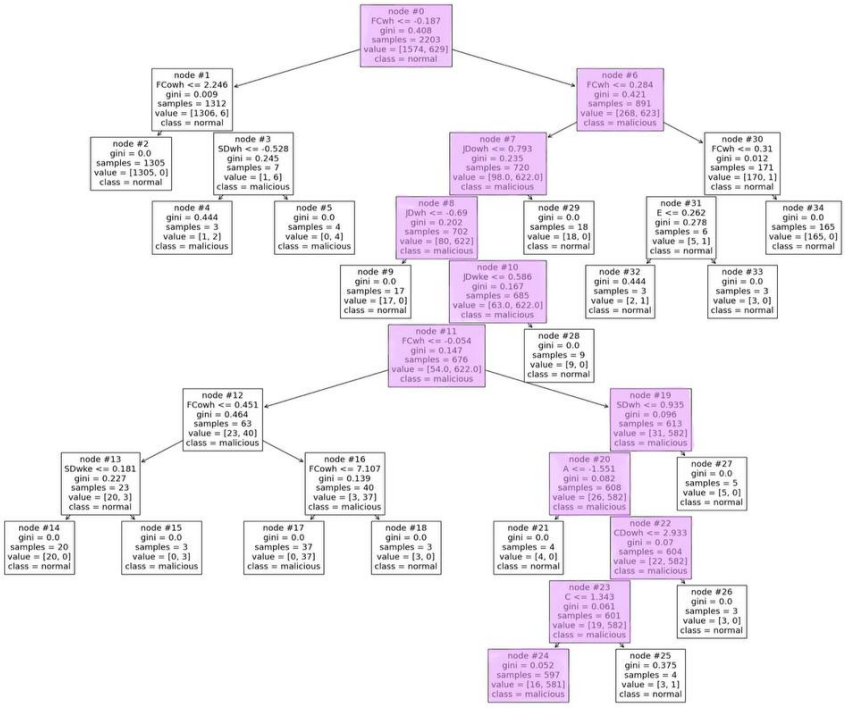


Figure 16: Decision Path Visualisation for TP and FP

Figure 18 shows boxplots of feature values for FP, TN and TP instances. For features such as CDowh, FP instances have a more similar value distribution to TP instances than TN instances, suggesting why the models predicted them as positive (malicious). This suggests that FP instances do not exhibit the same patterns as TN instances with respect to these features, indicating they are ineffective features which do not necessarily suggest malicious intent.

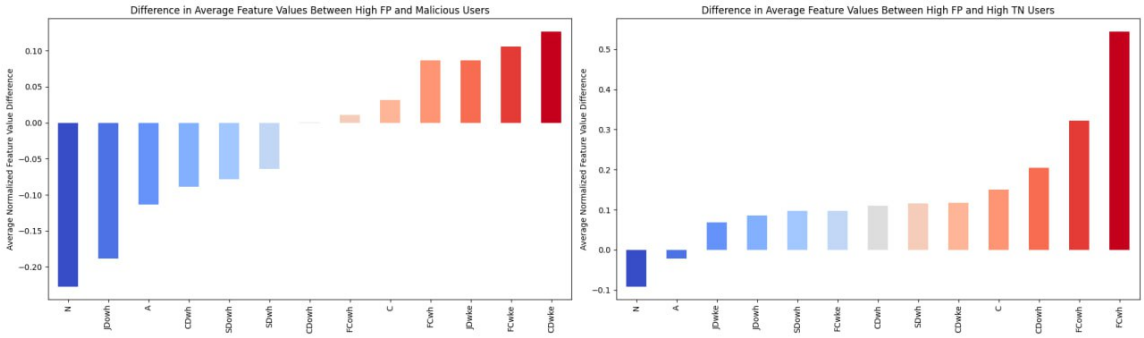


Figure 17: Difference in Average Feature Values between FP and TN/TP

Hence, it is clear that anomalous file activity or website browsing behaviour don't necessarily indicate insider threat. Some employees need to access job or cloud domain websites more often than others on average, due to the nature of their role. For example, recruiters may need to visit job websites more often. Secondly, some employees may click cloud storage websites on accident or out of curiosity. Hence, URL clicks are not informative of a user's intent when isolated from contextual factors. The following improvements can be considered.

- Group more behaviours instead of only focusing on URL clicks. For example, only count URL clicks for cloud storage websites if they are accompanied by file uploads of company data.
- Examine deviations from normal patterns of file copies, job website visits, etc. instead of absolute counts per employee. This is because each employee could have a different baseline behaviour, taking into account contextual factors such as job responsibilities.

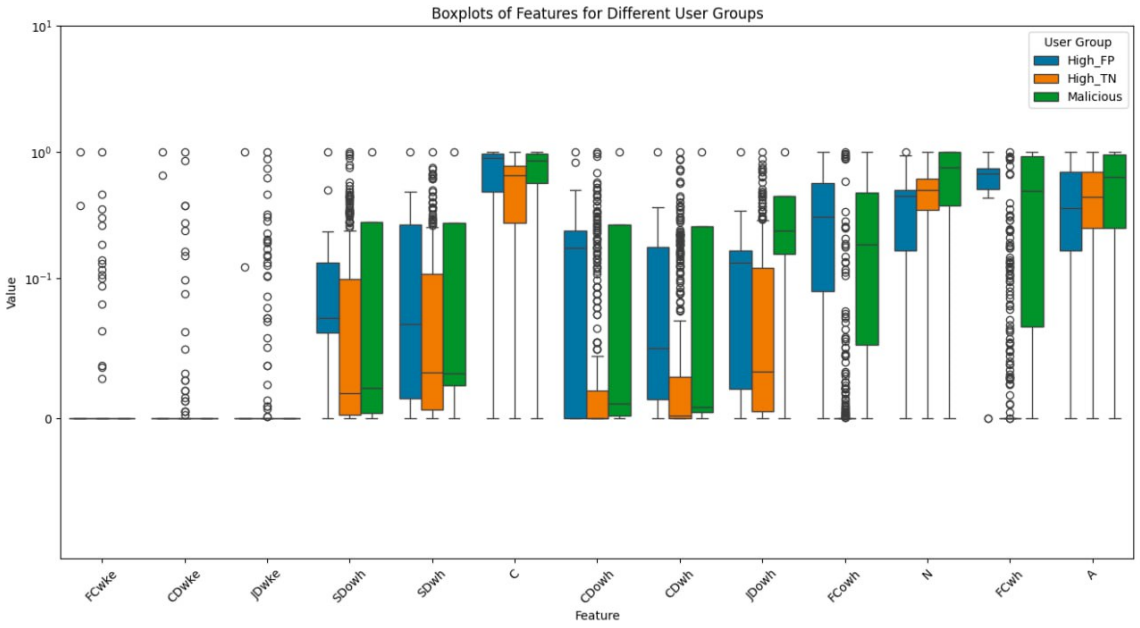


Figure 18: Box Plot of Different Features

7 Conclusion

7.1 Summary of Results

Our ensemble method with confidence scores for insider threat detection can be highly significant to various industries, especially those dealing with sensitive information, financial transactions or critical infrastructure. Confidence scores generated by ensemble methods can help prioritise alerts or incidents based on their likelihood and potential impact. This allows security teams to focus their attention on the most significant threats, improving response efficiency, reducing FP and optimising resource allocation. Secondly, ensemble methods help by combining multiple models to produce more accurate predictions than any single model. Lastly, by tuning model classification thresholds to penalise FP more than FN, the detection system is more aligned with asymmetrical, real-life misclassification costs since FP errors can cause false alarms and unnecessary costs involved in investigating threats.

7.2 Hypothetical Data Pipeline

For an organization concerned with insider threat detection, the model could be integrated into the company's data security infrastructure. The system would operate continuously, analyzing user activities in real-time. It would draw from data sources such as system logs, email traffic, network activity, and file access records to detect potential insider threats. The model would be run as part of a weekly security audit cycle, pulling the latest user behavior data, which has been aggregated over the week. Upon detection of a potential threat, the model would output its findings and confidence scores into a security dashboard, which is monitored by the cybersecurity team. This dashboard would allow for immediate review and action. Additionally, detailed reports could be compiled and sent to the relevant department heads and IT security teams to inform them of any necessary follow-up actions. This enables a proactive stance against insider threats with minimal disruption to the workforce and ensures that any potential risks are managed promptly and efficiently.

7.3 Fraud Recommendations

Other fraud patterns that our model can detect include inappropriate use of privileged access, such as employees with privileged access rights abusing their privileges, such as accessing restricted data or systems for personal gain. Next, our model can also detect patterns of network activity that deviate from normal behaviour, such as accessing sensitive systems or downloading large amounts of data during non-standard hours. Lastly, our model can detect attempts to steal sensitive data or access information without authorisation. Products and tools that may enhance fraud detection include User and Entity Behaviour Analysis, which can analyse user behaviour across multiple data sources to detect anomalies and identify potential insider threats. Furthermore, Endpoint Detection and Response solutions can provide visibility into endpoint activity and help identify indicators of compromise associated with insider threats or external attacks targeting sensitive data.

Feature	Description
O	Openness Score
C	Conscientiousness Score
E	Extraversion Score
A	Agreeable- ness Score
N	Neuroticism Score
FCwke	File Copy to Removable Media during Weekend
FCowh	File Copy to Removable Media Outside Work Hours
FCwh	File Copy to Removable Media during Work Hours
SDwke	Suspicious Domain Website Activity during Weekend
SDowh	Suspicious Domain Website Activity Outside Work Hours
SDwh	Suspicious Domain Website Activity during Work Hours
CDwke	Cloud Storage Domain Website Activity
CDowh	Cloud Storage Domain Website Activity
CDwh	Cloud Storage Domain Website Activity
JDwke	Job Domain Website Activity during Weekend
JDowh	Job Domain Website Activity Outside Work Hours
JDwh	Job Domain Website Activity during Work Hours

Table 1: Table of Data Description

8 Appendix

8.1 Data Dictionary

Observed in Table 1

8.2 Citations

References

[BJLM21] Filip Wieslaw Bartoszewski, Mike Just, Michael A. Lones, and Oleksii Mandrychenko. Anomaly Detection for Insider Threats: An Objective Comparison of Machine Learning Models and Ensembles. In Audun Jøsang, Lynn Futcher, and Janne Hagen, editors, *36th IFIP International Conference on ICT Systems Security and Privacy Protection (SEC)*, volume AICT-625 of *ICT Systems Security and Privacy Protection*, pages 367–381, Oslo, Norway, June

2021. Springer International Publishing. Part 7: Machine Learning for Security.

[Ins22] Ponemon Institute. 2022 ponemon institute cost of insider threats: Global report today., 2022.

[Lin20] Brian Lindauer. Insider Threat Test Dataset. 9 2020.

[R+15] Matthew Rocklin et al. Dask: Parallel computation with blocked algorithms and task scheduling. In *SciPy*, pages 126–132, 2015.

8.3 Meeting Minutes and Contributions List

8.3.1 Meeting 1

Date: 15 March 2024, 9pm - 10pm
Present: Melody, Shi An, Amruth, Micole, Jazlynn, Hannah
Agenda: Decide case study, dataset, roles in the group, division of work, setting of timeline across the next 5 weeks
Things-To-Do:

- Clarify with prof to what extent the synthetic dataset can be
- Come up with 2-3 problem statements, consult with prof to see which is more feasible
- Additional questions to discuss with prof

Notes:
Discussed various potential fraud in case studies
Keeping in mind the development of graphs to display results
Research papers to read and refer to: (Amruth, Jaz)

- https://www.ijcai.org/proceedings/2021/0505.pdf
- https://arxiv.org/pdf/2008.08692.pdf

Potential datasets to consider:

- Online dating fraud (Micole)
  - OKCupid profiles: age, status, sexual orientation, etc details of the profiles

  - How to identify fake profiles/ catfishing/ fake users with malicious intentions
  - Find patterns like abnormal messaging rates, use of stock image, repeated images
  - https://harshalvaza.medium.com/online-dating-platform-analysis-okcupid-14d9704b6b02
  - https://www.kaggle.com/datasets/andrewmvd/okcupid-profiles
  - https://www.kaggle.com/datasets/jmmvutu/dating-app-lovoo-user-profiles
- Crowdfunding, kickstarter projects (Micole)
  - Sudden spikes in funds from accounts with little to no backing history, or unusual pattern in their comment activity and withdrawal attempts post-campaign
  - https://dl.acm.org/doi/fullHtml/10.1145/3501247.3531541
  - https://www.kaggle.com/datasets/kemical/kickstarter-projects
- Online restaurant reviews on Yelp (Micole)
  - Clustered posting times, similarity in language or rating patterns, or discrepancies between review content n what the restaurant actually offers
  - https://github.com/asiamina/FakeReviews-RestaurantDataset
  - https://link.springer.com/chapter/10.1007/978-3-031-00129-1_9
  - https://cs229.stanford.edu/proj2019aut/data/assignment_308875_raw/26552575.pdf
- Mobile fitness apps: rewards/ incentive abuse, GPS spoofing, irregularities in speed, altitude changes, super-human patterns (Micole)
- Insider threat (Hannah)
  - threat black box: variational auto-encoder
  - employees in companies who have the potential to leak internal data
  - tracks employee behaviour on company devices
  - when, where of downloads, logins to determine leak threats
  - https://arxiv.org/ftp/arxiv/papers/2109/2109.02568.pd
  - https://kilthub.cmu.edu/articles/dataset/Insider_Threat_Test_Dataset/12841247

  - https://arxiv.org/pdf/2005.12433v1.pdf
  - value add using explainable AI

- Political/ election/ voting fraud (Melody, Shi An)
  - potentially hard to get a proper dataset e.g. 2020 US Elections
  - data such as twitter comments
  - identify who the specific users are, their activity levels, to see if they are active specifically during the election period
  - https://www.kaggle.com/code/arlene025/votefrauddata
  - https://www.kaggle.com/datasets/paultimothymooney/voterfraud2020-election-fraud-claims-on-twitter

Final selected dataset: CMU Insider Threat

8.3.2 Meeting 2

Date: 21 March 2024, 9am - 10am
Present: Melody, Shi An, Amruth, Micole, Jazlynn, Hannah
Agenda: Present findings of EDA: Melody, Micole, Jazlynn, Hannah. Present findings from research paper for solutions to insider threat: Shi An, Amuth
Things-To-Do:

- data storage: local $\rightarrow$ docker afterward
- use dask instead of pandas
- use sstable instead of csv
- feature extraction for 5.2 using the same git code

Notes:
Dataset Initial EDA (Micole, Melody, Hannah, Jazlynn)

- Columns have a lot of potential for network analysis, using the http values
- Difference between versions 4.2 and 5.2: Use dataset 5.2 (encompasses more scenario cases)

- Using other versions of the dataset as testing/ validation: *Check if can combine 5.2 and 6 (in case of data duplication → data leakage)*
- Adding labels to the training dataset: Binary labelling, Need resampling
- Consider: removing user-id so the model does not recognise potential threat through user activity but rather the device activity instead
- Train-test split using the same dataset? Or test using a separate version of the dataset
- Need to check if the dataset versions are overlapping in data
- Final use dataset 5.2, ignore scenario 5
- Feature engineering: Group by user id and time series (time stamp)

Literature review findings: (Amruth, Shi An)

- Reviewed a couple of papers detailing models used for training
- Both supervised and unsupervised are possible
- Need to find different models to test out
- Majority of papers used v.4.2 of the dataset
- Same conclusion as EDA: use same version for train test or different versions?

Questions to Ask Prof during Consultation:

- Define fraud : does she agree w the scenario - Ethically wrongful or criminal deception intended to result in financial or personal gain; In the context of a company/corporate setting monitoring employee's device activity
- Our problem statement: prevent malicious users' activities in next step (does it fit w the task)
- update progress
- update data storage and can use docker? currently we are planning to upload the features extracted csv on git instead of raw data
- share w her the technical difficulties of our dataset, ask for advice

set a date with prof

8.3.3 Meeting 3

Date: 29 March 2024, 4pm - 5.30pm
Present: Melody, Shi An, Amruth, Micole, Jazlynn, Hannah
Agenda: Docker implementation, subsetting dataset
Things-To-Do: Feature extraction file Notes:
Subsetting the dataset:

- 4.2: 70 fraud
- 5.2: 100 fraud
- Non-fraud takes up the most
- Subset according to ratio?

Micole: feature extraction code took over 6 hours
Need to work on subset data to reduce extraction time

- how to subset (undersample) feature extracted data (remove instance by user) from data 5.2 only
- feature engineering https://github.com/lcd-dal/feature-extraction-for-CERT-insider-threat-test-datasets

How to run docker: build the ubuntu image (include docker instructions in report..?)
use parquet → read.parquet for raw data
Micole and amruth settle csv → parquet
Jaz - detects scenario 3&4 only, login, user & email csv only, Djikstra, weighted edges
Use weekr5.2.csv feature extracted file for all your models/eda everything. apply PCA on each category of columns, eg PCA on all colums starting with ”weekendfile”, PCA on all columns starting with ”afterhourfile”
Everyone → look through the feature extracted file and try to remove columns yourself (until Monday), try to reduce num of columns until below 100 (20 is ideal). decide scenario after evaluating the code

Meeting 4
Date: 1 April 2024, 5pm - 6pm
Present: Melody, Shi An, Amruth, Micole, Jazlynn, Hannah
Agenda: Hypothesis, chosen scenario, further steps

Things-To-Do:
OUR NEW APPROACH:

- output risk/confidence score
- 3 models: explain and evaluate where they fail/succeed and why we chose them (explainable ML) need to show which features are significant or not
- 3 or 4 best insights for presentation (only 10 mins anyway)
- Need heuristics for risk and confidence score

HOW TO PRESENT:

- don't go through any code at all, only insights charts findings
- system diagram only
- final results, no functions no python
- approach as business solution

Notes:

- Quite a simple solution to scenario 2
- IRL can just email the person that model identified as potential threat and let the person explain themselves (solution is meh)
- A lot of FP IRL when flagging people out $\rightarrow$ from off the shelf solutions e.g. if download exceeds limit 10gb then will trigger an alert/ flag
- But for our project can say we have considered etc.. so will reduce FP and increase TP (metrics of evaluation)
- **jamf**
- Explore how the FP comes about $\rightarrow$ interesting component
- But will it create a better model to reduce FP
- other models to build from this? hard to defend - subjective to company, very contextual for each email content,

- Maybe need to cross refer more research papers to determine how the FP happens
- meant for cybersecurity team to flag individuals who are suspicious then look through manually, to reduce man-hours, improve efficiency,
- create a priority list/ranked queue
- be very careful not to sound discriminatory
- Out of all the files flagged as suspicious, the proportion of TP/FP is?

Beyond FE + basic modelling:

- Point out the exceptions, where they arise from and how to take it a step further
- Beyond the if-else statements (rules for flagging suspicious activity)
- How else can we evaluate the instance
- return a risk score instead of binary classification: Risk score »> truth score
- CONFIDENCE SCORE: good for IRL implementation (relate back to IRL context and application)
- How to implement confidence risk score?
- One good model vs multiple models
- Do 3 models then evaluate why this model is the best
- In rubrics got say: “Model Ensemble - you may try model averaging or “poor man’s stacking” but explore other ensemble techniques as well. Your team’s creativity and improvement will be judged and you can potentially earn bonus points here.”
- need a baseline model
- our chosen model doesn’t have to be new
- Can replicate research paper ones

Models to explore:

- Baseline: SVM/ Linear SVM
- Model 1: Random Forest/ Decision Tree/ Boosted Decision Tree

- In contrast to black-box alternatives with obscure interpretative rules, random forests are ensembles of many decision trees and thus offer a deep but human-readable set of detection rules. Such explainable artificial intelligence is recognized as a growing need and the subject of intense investigations particularly in high-stakes decision realms like medicine, defense, and human resources.
- Model 2: Autoencoder?
- Neural Network?