2025

Intrusion Detection Using Supervised Learning: A Big Data Approach

PREPARED BY
MD MUNIF AKANDA
3551133

INFT 6201 | Big Data

Executive Summary

This project investigates on how supervised machine learning models can detect cybersecurity incidents.

The project was motivated by the increasing frequency of cyber-attacks including credential-stuffing, brute-force attempts, and insider misuse in enterprise environments. There is a growing the need for scalable behavioural intrusion detection. Early detection and escalation are critical to reduce massive financial impacts. (IBM, 2023)

The project builds on a dataset sourced from Kaggle. This dataset contains 9,537 sessions of user activity. Key user session behaviour features include-

- failed login attempts,
- session duration,
- unusual access times,
- IP reputation scores

A comprehensive modelling techniques were developed using six supervised classifiers: Random Forest, Gradient Boosting, AdaBoost, Decision Tree, Logistic Regression, and Naïve Bayes.

Performance was evaluated using accuracy, precision, recall, F1-score, ROC curves, and precision-recall curves

Among all models, Gradient Boosting achieved the strongest performance (Accuracy = 0.895, F1 = 0.890), closely followed by Random Forest (Accuracy = 0.894)

The project contributes to secure digital infrastructure aligned with SDG 9 (Industry, Innovation and Infrastructure) and SDG 16 (Peace, Justice and Strong Institutions) by demonstrating an approach for early intrusion detection using scalable machine-learning methods. Future work may incorporate hyperparameter tuning, SMOTE balancing, and additional models such as XGBoost or LightGBM to further enhance detection performance.

Overall, this study demonstrates that analysing user behaviour with supervised machine learning provides a robust, practical, and scalable method for detecting cyber intrusions in modern enterprise systems.

Table of Contents

1. Introduction

Cybersecurity attacks continue to grow in scale and sophistication. Enterprise systems are heavily targeted. This places pressure on organisations to adopt scalable and intelligent detection systems. Traditional rule-based intrusion detection systems (IDS) often struggle to identify new or evolving attack patterns.

This project investigates how machine-learning models can support early-stage intrusion detection by analyzing user session behaviour. The project also aligns with broader digital-security goals and contributes to development of scalable and data-driven security solutions.

Six machine-learning models were implemented to understand which classifier can help identify intrusion more effectively. Key features such as failed login attempts, session length, IP reputation, and unusual access times were used to train the classifiers.

To evaluate the and identify the best performing classifier, Accuracy, Precision and F1 score was used to measure performance.

2. Dataset

The dataset used in this project was sourced from Kaggle's Cybersecurity Intrusion Detection Dataset (Kumar, 2023). It contained 9,537 login sessions with both numerical and categorical features.

The dataset includes 11 variables, such as login attempts, failed logins, session duration, IP reputation, encryption method, and browser type.

The target variable, attack_detected, labels each session as either normal (0) or an attack (1).

A missing-value check showed 1,966 rows without an encryption value. These were removed which left 7,571 clean observations for analysis. Overall, the dataset is suitable for supervised machine-learning models used in intrusion detection.

Diagram 1 provides the summary statistics for the numerical features, including mean, standard deviation, and quartiles.

3. Dataset Analysis

3.1 Descriptive Statistics Analysis

Descriptive statistics for numeric features:

	count	mean	std	min	25%	50%	75%	max
network_packet_size	7571.0	499.556598	198.458221	64.000000	364.000000	498.000000	632.000000	1285.000000
login_attempts	7571.0	4.031832	1.963112	1.000000	3.000000	4.000000	5.000000	13.000000
session_duration	7571.0	789.334671	789.993725	0.500000	230.590993	547.137302	1101.216078	7190.392213
ip_reputation_score	7571.0	0.330749	0.177953	0.002497	0.190350	0.313306	0.453525	0.924299
failed_logins	7571.0	1.523313	1.036751	0.000000	1.000000	1.000000	2.000000	5.000000
unusual_time_access	7571.0	0.150707	0.357786	0.000000	0.000000	0.000000	0.000000	1.000000
attack_detected	7571.0	0.443006	0.496774	0.000000	0.000000	0.000000	1.000000	1.000000

Figure 1: Summary statistics for the numerical features, including mean, standard deviation, and quartiles

The diagram above represents descriptive statistics for the numerical variables in the dataset summary statistics such as quartiles, standard deviation, mean, and count.

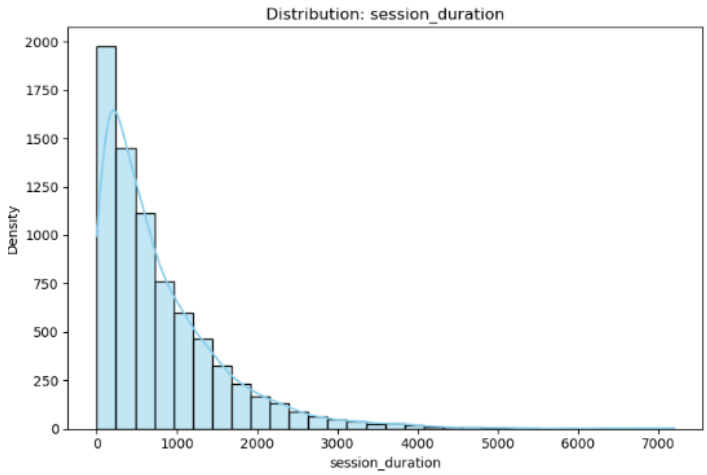


Figure 2 :Histogram: session_duration

The histogram of session_duration shows a highly right-skewed distribution. Majority of the sessions lasts less than 1000 seconds. The long tail, that extends more than 7000 seconds, indicates a small number of sessions last significantly longer which can represent abnormal or automated (DDoS) activity. The high variance observed here supports the idea that session length can be an important behavioral indicator for intrusion detection.

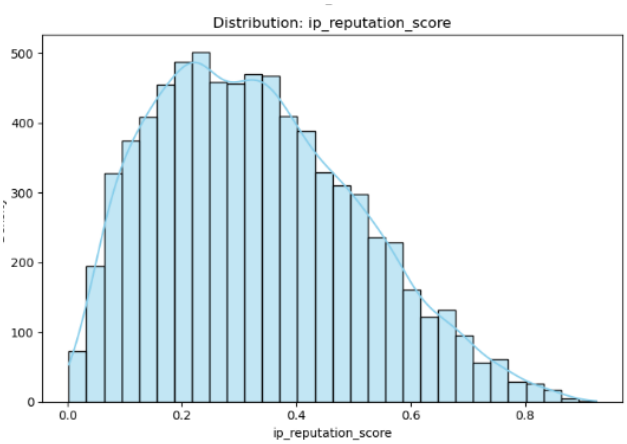


Figure 3: IP Reputation histogram

The ip_reputation_score distribution shows a moderately right-skewed pattern. Most IP addresses that interact with the system fall within the low to medium reputation range (0.1–0.5). This is typical in an enterprise environment where legitimate users dominate the traffic (Scarfone & Mell, 2007). However, there is a noticeable tail extending toward higher scores (0.7–0.9). This demonstrates a small subset of sessions associated with suspicious or high-risk IP addresses. These outliers make ip_reputation_score an important feature because they help distinguish normal traffic from potentially malicious activity during modelling.

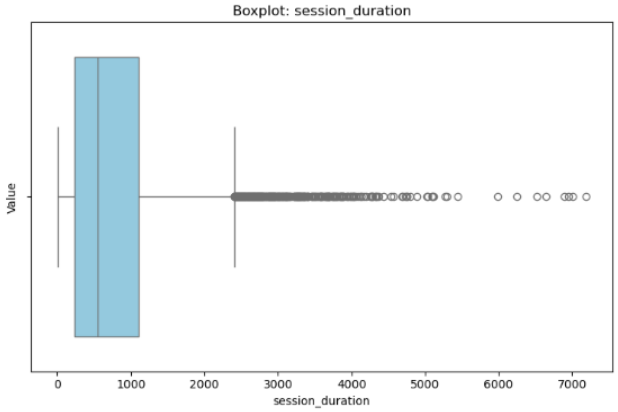


Figure 4: Boxplot Session Duration

The boxplot for session_duration highlights a substantial number of long-session outliers. The central box shows that most sessions fall within a lower end of the scale, under 1,500 seconds but the whiskers extend far beyond this range indicating a cluster of outliers stretching 2500-7000 seconds.

These long and unusual sessions are important feature from security perspective as it indicates automated processes, brute-force attempts, or abnormal user behavior.

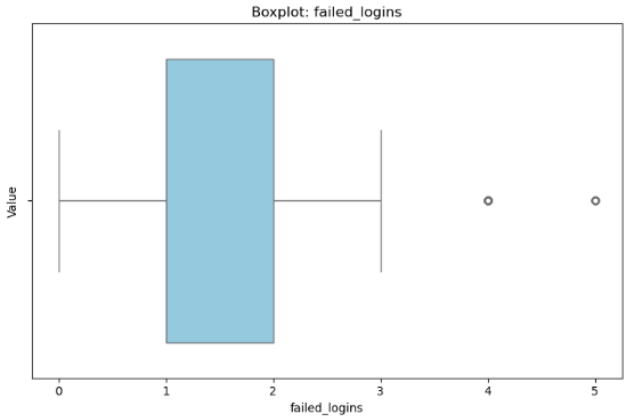


Figure 5: Box Plt failed logins

The failed_logins boxplot shows most people need up to 2 login attempts which is typical normal user behaviour in an enterprise environment (Sommer & Paxson, 2010). The outliers, where 4 or 5 failed attempts are recorded, can represent unusual activity. These higher counts are strong indicators of potential cyberattacks such as brute-force or dictionary attacks, making failed_logins an important feature for machine-learning models to detect malicious behaviour.

3.2 Correlation Analysis

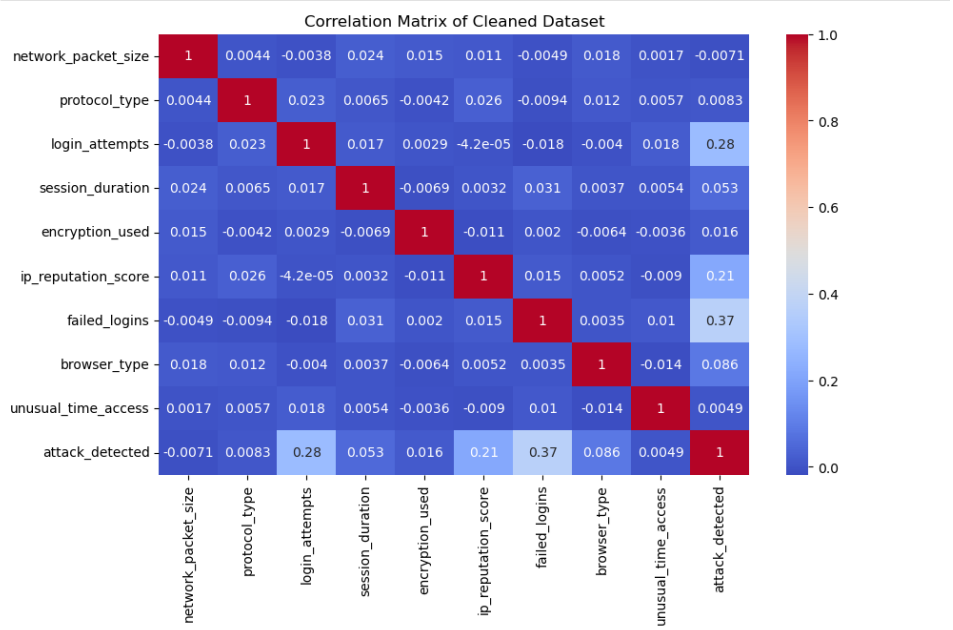


Figure 6: Correlation Matrix

The correlation matrix shows that most variables in the dataset have weak linear relationship. This is expected in real world security log (Scarfone & Mell, 2007) as user behavior can vary.

However, the matrix displays some meaningful association with the target variable, attack_detected. For example, failed_logins has a moderate positive correlation (0.37) with attack_detected. Login_attempts also shows a positive correlation (0.28). IP_reputation_score has a mild positive correlation (0.21) with attacks. Other features such as network_packet_size, session_duration, browser_type, and protocol_type show correlations close to zero, meaning they contribute less direct predictive power individually.

3.3 Categorical Patterns

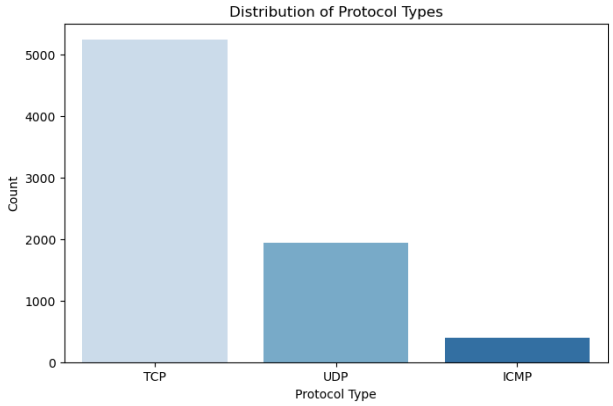


Figure 7: Protocol Type Bar Chart

This bar chart shows that TCP (Transmission Control Protocol) is the dominant protocol type for packet transfer. While this protocol type is not a strong predictor feature for attack, documenting its distribution provides useful context.

4. Data Modelling and Results

4.1 Preprocessing

Preprocessing dataset is a necessary step to make sure the dataset is clean and ready for modelling. The original dataset contained 9,537 rows and 11 columns. The dataset consists of numeric, categorical and binary features. Following steps has been taken for data modelling-

- After finding missing values in the encryption_used, rows with missing values were removed from the dataset. Session_ID field was removed as it is an identifier data.
- Categorical variables; protocol_type, browser_type, encryption_used; were transformed using Label Encoding, converting string categories into numerical labels suitable for machine-learning algorithms.

- The dataset was then split into training (75%) and testing (25%) sets using the attack/no-attack class distribution.
- Numerical data was standardised and fitted on the training set and applied to the test set

4.2 Models Trained

A diverse set of supervised classification algorithms were trained

- Random Forest Classifier (200 trees, class_weight=balanced)
- Decision Tree Classifier (class_weight=balanced)
- AdaBoost Classifier
- Gradient Boosting Classifier
- Logistic Regression (max_iter=1800)
- Gaussian Naive Bayes

We have evaluated their effectiveness on predicting the target variable **attack_detected**.

Model Selection Justification

Random Forest Classifier: An ensemble of multiple decision trees. The advantage of this classifier that it reduces overfitting and improves generalization making it suitable for network log which can be messy.

Gradient Boosting: works by combining multiple weak learners to create a highly accurate model. It is known to perform well on tabular data (network and system log).

Decision Tree Classifier: Provides a non-linear method that captures rule-based patterns in user behavior

AdaBoost Classifier: A boosting method that works by iteratively focuses on correcting misclassified samples. Generally, performs better than Decision Tree Classifier.

Logistic Regression: Chosen because it's a simple classifier and can be used as benchmark.

Gaussian Naïve Bauyes: Works by assuming each variable is independent. Works well in practice than theory.

4.3 Evaluation Metrics and Performance Summary

The models were evaluated using accuracy, precision, recall and F1-Score.

Accuracy: provides overall percentage of correct predictions

Recall: demonstrates how many actual attacks the model detected

Precision: indicates how many predictions were correct

F-1 score: harmonic mean of precision and recall, indicates overall detection quality

Model	Accuracy	Precision	Recall	F1
Gradient Boosting	0.895	0.921	0.881	**0.890**
Random Forest	0.894	0.908	0.872	**0.889**
AdaBoost	0.875	0.899	0.840	0.868
Naive Bayes	0.821	0.825	0.807	0.816
Decision Tree	0.811	0.833	0.788	0.809
Logistic Regression	0.733	0.750	0.702	0.726

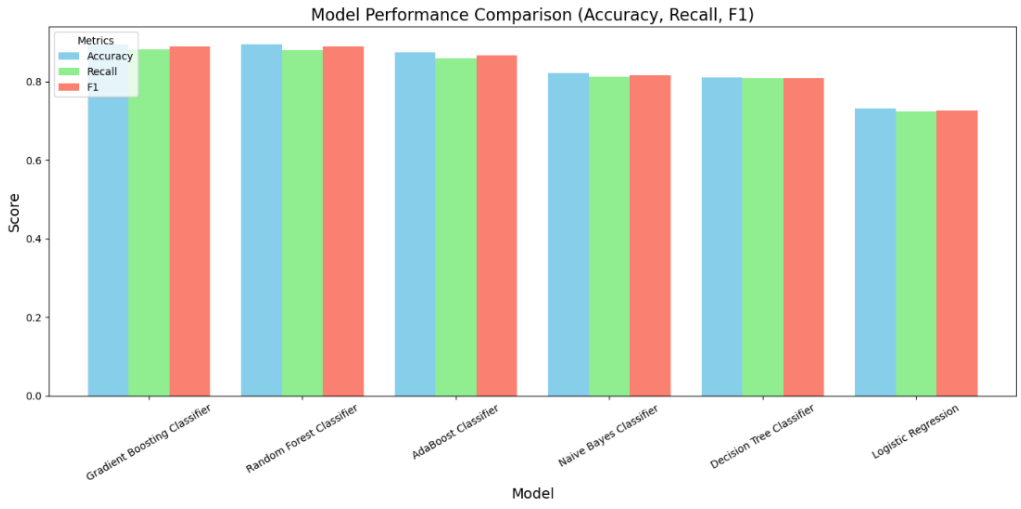
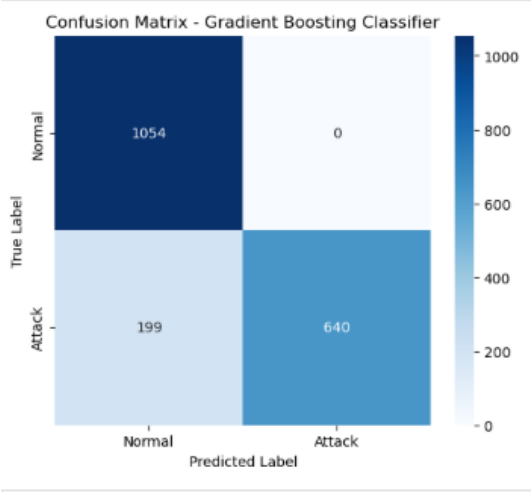


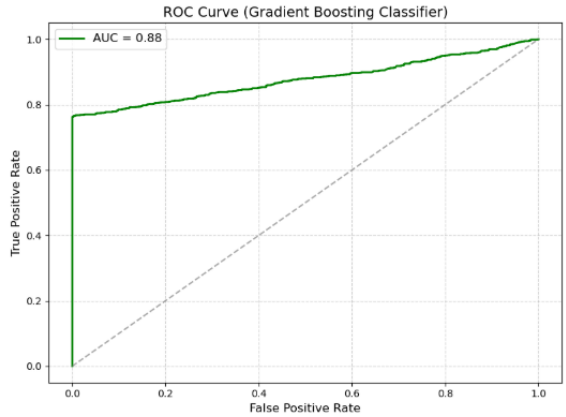
Figure 8: Model Performance Comparison

Overall, the performance comparison, using the metrics mentioned, shows that ensemble model Gradient Boosting and Random Forrest delivered the strongest result. Both of these models performed well but Gradient Boosting classifier was the most reliable model for intrusion detection.

4.4 Confusion Matrix Analysis

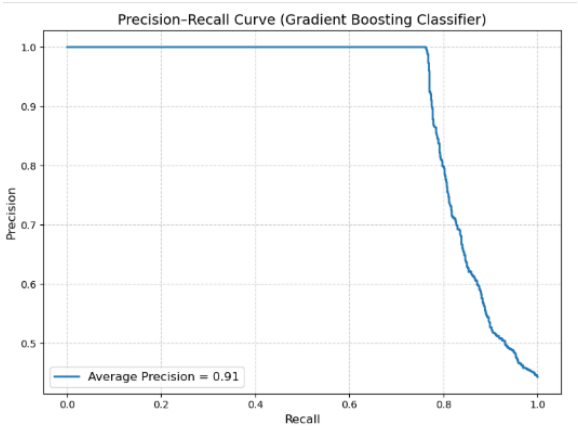


4.5 ROC Analysis



The ROC curve shows that the Gradient Boosting Classifier maintains a strong true positive rate across various thresholds and keeps false positives low, resulting in an AUC of 0.88. This is an indication that the model is highly effective at distinguishing normal and attack sessions.

4.6 Precision-Recall Analysis



The precision–recall curve demonstrates that the model maintains high precision across a wide range of recall values. The average precision score of 0.91. This is very important for intrusion detection as identifying attacks correctly while minimizing false alarms is critical.

5. Discussions & Conclusion

5.1 Strengths of the Analysis

A well structured and systematic approach to intrusion detection has been undertaken using features that can be found on real world network logs. The major strength of the analysis is its robust comparison across multiple supervised learning models. Moreover, the evaluation used macro-averaged metrics ensured fair assessment despite class imbalance. This is a common challenge in security data

5.2 Limitations

Despite its strong methodology, the analysis faces several limitations. Such as-

- Real enterprise network logs are messy and far more complex. They generate high volume of data at high velocity. This needs further testing and amend changes before rollout
- The project mostly focused on enterprise environment. Some features (such as login patterns or protocol usage) may differ significantly in other context.

5.3 Future Work Recommendation

Areas that can be improve are-

- Further analysis using larger and more diverse dataset

- Applying advanced techniques such as SMOTE, hyperparameter tuning
- Experimenting with other classifiers such as XGBoost or LightGBM

5.4 Conclusion

This project showed how machine-learning models can play a meaningful role in detecting cyber intrusions. Analyzing behavior patterns using six different classifier and comparing them with clear metrices, the study found that ensemble method, especially Gradient Boosting is the most effective in separating normal and malicious activity.

6. References

- Kumar, D. (n.d.). *Cybersecurity Intrusion Detection Dataset* [Data set]. Kaggle. https://www.kaggle.com/datasets/dnkumars/cybersecurity-intrusion-detection-dataset
- Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: Methods, systems and tools. IEEE Communications Surveys & Tutorials, 16(1), 303–336. https://doi.org/10.1109/SURVEY.2013.052213.00046
- Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security, 28(1–2), 18–28. https://doi.org/10.1016/j.cose.2008.08.003
- Scarfone, K., & Mell, P. (2007). Guide to intrusion detection and prevention systems (IDPS) (NIST Special Publication 800-94). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-94
- Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In 2010 IEEE Symposium on Security and Privacy (pp. 305–316). IEEE. https://doi.org/10.1109/SP.2010.25
- Zhang, Y., Chen, X., Xiang, Y., & Zhou, J. (2019). A survey on machine learning-based intrusion detection systems. Computer Networks, 151, 147–168.

Appendix

- Used Generative AI tool (ChatGPT-OpenAI) to assist with code generation, debugging and clarify data analytics and statistics concepts