EAI Endorsed Transactions
on Industrial Network and Intelligent Systems
Research Article EAI.EU

Machine Learning in Cybersecurity: Advanced Detection and Classification Techniques for Network Traffic Environments

Samer El Hajj Hassan [1*], Nghia Duong-Trung[2,3]

[1] IU International University of Applied Sciences, Berlin campus, Frankfurter Allee 73A, 10247 Berlin, Germany
[2] German Research Centre for Artificial Intelligence (DFKI), Alt-Moabit 91 C,10559 Berlin, Germany
[3] Kien Giang University, 320 A - Highway 61 - Minh Luong Town, Chau Thanh District - Kien Giang Province, Vietnam

Abstract

In the digital age, the integrity of business operations and the smoothness of their execution heavily depend on cybersecurity and network efficiency. The need for robust solutions to prevent cyber threats and enhance network functionality has never been more critical. This research aims to utilize machine learning (ML) techniques for the meticulous analysis of network traffic, with the dual goals of detecting anomalies and categorizing network activities to bolster security and performance. Employing a detailed methodology, this study begins with data preparation and progresses through to the deployment of advanced ML models, including logistic regression, decision trees, and ensemble learning techniques. This approach ensures the accuracy of the analysis and facilitates a nuanced understanding of network dynamics. Our findings indicate a notable enhancement in identifying network inefficiencies and in the more accurate classification of network traffic. The application of ML models significantly reduces network delays and bottlenecks by providing a strong defence strategy against cyber threats and network shortcomings, thereby improving user satisfaction, and boosting the organizational reputation as a secure and effective service layer.
Conclusively, the research highlights the pivotal role of machine learning in network traffic analysis, offering innovative insights and fresh perspectives on anomaly detection and the identification of malicious activities. It lays a foundation for future explorations and acts as an evaluation benchmark in the fields of cybersecurity and network management.

Keywords: Machine Learning, Cybersecurity, Network Analysis, Anomaly Detection, Data Security, Traffic Classification, Network Optimization, Traffic Volume

Received on 29 February 2024, accepted on 05 May 2024, published on 01 July 2024

doi: 10.4108/eetinis.v11i3.5237

*Corresponding author. Email: samer.elhajjhassan@iu-study.org

1. Introduction

In the rapidly evolving digital landscape, cybersecurity has emerged as a critical concern for businesses and organizations worldwide [1][2]. Cybersecurity and network performance are pivotal elements in maintaining the integrity and efficiency of technological infrastructures [3][4]. The world we live in is increasingly interconnected, with vast amounts of data being transferred every second across networks. This scenario presents both opportunities and challenges, especially in the realms of data protection and network optimization. Network issues affect customer satisfaction and company reputation significantly [3]. Users demand fast, secure access to services; delays or

*Corresponding author: Samer El Hajj Hassan
Email: samerhajjhassan@gmail.com

security breaches cause frustration, reducing trust and loyalty [4]. This is critical in sectors like finance and healthcare, where real-time data is essential. Network problems can damage a company's image, making client retention and new customer acquisition challenging, thereby affecting market competitiveness.

The escalating complexity of sophisticated cyber threats necessitates innovative strategies to guarantee robust network security [5] [6]. One such approach is the application of machine learning techniques for anomaly detection in network traffic [1][2]. Machine learning, with its ability to learn from and make decisions based on data, offers significant potential in enhancing network security [1][2]. It can be used to analyse network traffic, identify patterns, and detect anomalies that may indicate potential security threats [1][2]. However, the effectiveness of these machine learning models is heavily dependent on the quality and diversity of the datasets used for training [5][6]. Historically, research in this area has been constrained by the limited availability and scope of network traffic datasets [5][6]. Many existing datasets do not adequately capture the complexity and evolving nature of cyber threats, leading to models that may not accurately detect anomalies or classify network traffic. This represents a significant gap in the research and underscores the need for more diverse and comprehensive datasets [5][6].

In this context, the “Starter: Labeled Network Traffic flows b42983c1-a” dataset from Kaggle [7] presents an exciting opportunity. This dataset provides a rich and diverse source of network traffic data that has not been extensively explored in previous studies [7], particularly not within the realms of cybersecurity and network performance.

The use of this dataset in our research allows us to uncover novel approaches and insights into network security threats and enhance the accuracy and adaptability of our machine learning models. By leveraging this innovative dataset, we aim to contribute to the advancement of network security management and pave the way for future research in this critical area. This exploration is crucial for advancing the field of network management and cybersecurity, providing valuable visions and tools for organizations to safeguard their digital assets and ensure optimal network operations.
In conclusion, this study underscores the importance of dataset variety in enhancing the relevance and effectiveness of machine learning approaches in network security. It contributes to the understanding of cybersecurity and paves the way for future advancements in network management and security. We believe that our study will serve as a valuable reference for future research in this area.

2. Related Work

2.1 Introduction to Literature Review

The landscape of cybersecurity and Network traffic analysis is ever evolving, with new threats emerging at an unprecedented pace. As businesses become increasingly reliant on digital networks, the importance of network performance and security has never been more critical [8]. This literature review aims to explore the current state of cybersecurity, focusing on network performance and security, the role of machine learning and data science techniques in cybersecurity, and the existing gaps in the literature.

2.2 Network Performance and Security

Recent studies have indeed focused on the application of Machine Learning (ML) for optimizing network performance while ensuring robust security. The study [9] in the IEEE Transactions on Network and Service Management gives a broad overview of new network security methods. It talks about the rising complexity of online threats like DDoS and malware and how artificial intelligence (AI) and machine learning (ML) can help fight these threats. But it mentions more studies are needed to fully understand AI and ML's power in cybersecurity. Another paper [10] suggests a method to improve network safety by monitoring real-time data for any unusual patterns. By predicting data that might show a threat, the approach aims to make online spaces safer. "Machine Learning for Traffic Analysis" [11] reviews how ML can assess traffic, a method needed to review network services and detect security risks. It showcases ML's role in tracking and handling online software and understanding web service quality. A further encouraging work [12] demonstrates that machine learning algorithms can forecast internet service performance, easing data handling. A study in Computer Communications [13] describes a new platform for classifying network data, displaying its power to identify problems or breaches, which is quite the task given the need to cut down data blunders. "QUIC Network Traffic Classification Using Ensemble Machine Learning Techniques" published in Applied Sciences [14], shows the potential of certain ML modes to optimize specific web data better, which underlines ML's purpose in recent podcasting.

In brief, recent writing supports ML's important place in improving the equipment and cyber defense, in line with this paper's set aims. By using ML, the view is that web management can make a big leap, ensuring top service and higher online safety.

2.3 Machine Learning in Cybersecurity

Machine Learning (ML) definitely occupies a crucial position within the scope of cybersecurity. Its significance is elaborated through:

1. **Anomaly Detection**: ML algorithms can be trained to learn what normal behaviour looks like within a network or system [15][16]. Once this baseline is established, the algorithms can then identify any deviations from this norm, flagging these anomalies for further investigation. A notable study [17] in this field is the uses supervised machine learning algorithms, which explores the effectiveness of various classification algorithms—including Stochastic Gradient Descent, Support Vector Machines, K-Nearest Neighbor, Gaussian Naive Bayes, Decision Tree, Random Forest, and AdaBoost —on the UNSW-NB15 [18][19] dataset for anomaly detection in network traffic. The study highlights the superior performance of the Random Forest classifier in identifying anomalies [20].
2. **Malicious Activity Classification** [20]: ML can classify activities as either normal or malicious based on learned patterns. This is particularly useful in identifying new threats that may not be detected by traditional, signature-based security systems.
3. **Predictive Analytics**: In the study [20] ML can also predict future attacks by analyzing trends and patterns in historical data. This allows organizations to proactively defend against potential threats.
4. **Automation and speed**: ML can analyze vast amounts of data much faster than a human could. This speed, combined with automation, allows threats to be detected and responded to more quickly, reducing potential damage. the research [21] provides an extensive view of machine learning algorithms and how they can be employed for intelligent data analysis and automation in cybersecurity.
5. **Adaptability**: As new data is fed into the ML model, it can adapt and improve its threat detection capabilities over time. This makes ML a powerful tool against evolving cyber threats. By combining machine learning's adaptability and big data analytics' data processing capabilities in a study [22], organizations gain a comprehensive, real-time view of their security posture. This approach ensures that historical data, real-time information, and predictive analytics converge to form a formidable defense against cyber threats

Thereby, machine learning has emerged as a powerful tool in cybersecurity, capable of analyzing large amounts of data and identifying patterns that can help detect attacks in their earliest stages.

However, the application of machine learning in cybersecurity is not without its challenges. The lack of standardized data exchange languages and interoperability among security tools can complicate the integration of machine learning models into existing security frameworks [23][24][25].

2.4 Data Science Techniques

In the realm of network analysis, data science techniques play a leading role in enhancing both network performance and security [26][27].

Data science in network analysis holds a wide array of techniques, ranging from basic statistical analysis to advanced machine learning algorithms [26][27]. The core objective is to extract meaningful insights from network traffic data, which can be vast and complex [28]. This data includes information about packet sizes, flow durations, traffic volume, and other characteristics essential for understanding network behavior [28].

Data science techniques, such as deep learning and predictive analytics, have shown promise in enhancing cybersecurity measures [5][21][29]. These techniques can help identify unseen patterns and draw meaningful insights from data, aiding in the detection and prevention of cyber threats [21][29]. However, the effectiveness of these techniques can be limited by the quality and diversity of the datasets used for model training [30].

2.5 Gaps in Existing Literature

Our comprehensive review in the cybersecurity domain, especially analyzing network traffic through machine learning, uncovers several critical gaps in existing literature, from dataset diversity to the real-time applicability of ML models. We address these issues as follows:

Dataset Diversity and Representation: Current research often relies on a limited range of datasets for different determinations, many of which may not fully represent the diverse and evolving nature of network environments. This limitation can impact the generalizability of machine learning models. Moreover, due to the shortage of sufficient datasets, ML approaches for network intrusion detection suffer from a lack of accurate deployment, analysis, and evaluation. Although researchers are using some datasets—such as DARPA [31] KDD CUP 99 [32], NSL-KDD [33], CAIDA [34], and UNSW-NB15 [18][19]—to evaluate the performance of their proposed IDS approaches, most of those datasets are out-of-date and thus inaccurate. They lack traffic diversity, and do not reflect current network traffic trends [35].

While not all gaps could be bridged due to time constraints, we emphasize that there is no one-size-fits-all dataset. We emphasize using a new dataset never been used in context of cybersecurity and network traffic analysis. This aids to improve model generalizability, enhancing accuracy and applicability.

Our approach seeks to utilize a more varied dataset selection to better adapt to the dynamic landscape of cybersecurity threats, highlighting the necessity for diversified datasets to enhance model accuracy and applicability.

Traffic Flow Prediction and Evaluation: Our research advances traffic prediction models [36], crucial for network security and performance, by employing machine learning techniques for anomaly detection and traffic flow analysis, delivering valuable predictions of network behaviors.

Real-time Analysis and Scalability: Our methodology addresses the need for real-time threat detection by using historical data for training, setting a stage for future research on achieving full real-time scalability [5] in diverse networks.

Complexity and Interpretability of ML Models: While advanced machine learning models can be highly effective, they often lack interpretability, which is crucial for understanding model decisions in cybersecurity applications. Our study balanced the complexity and interpretability of machine learning models [37] by selectively using a smaller dataset sample (100,000 out of 2.7 million instances) and employing Google Colab for efficient computation. This approach facilitated the comparison of models across various metrics and running times, allowing for direct evaluation of their effectiveness. Through the application of confusion matrices, we gained deeper insights into model performance. This research makes the models more practical for real-world application.

Integration with Existing Network Systems [38]: Although exploring specific application programming interface and platform requirements for seamless model integration was beyond our scope, our research serves as a foundational layer for immediate incident response within multi-layered security frameworks, encouraging further development towards integration.

Adaptation to Evolving Threats [39]: Our flexible methodology allows for the integration of new rules, constraints and thresholds and even features whenever needed, enhancing the adaptability of models to evolving cybersecurity threats. This approach underscores the dynamic nature of cybersecurity and the need for models to evolve alongside threats.

Balancing Performance and Security [40]: We initiate discussions on optimizing security and network efficiency concurrently, addressing a gap in current research. our work paves the way for future comprehensive studies aimed at enhancing both aspects in tandem.

Cloud-Based Anomaly Detection: While our research does not directly tackle cloud-based anomaly detection as a service, the methodologies and codes developed lay the groundwork for future studies in applying and evaluating ML models in cloud environment [41].

In summary, our research bridges critical gaps in current literature, providing innovative solutions and methodologies for enhancing cybersecurity in the face of evolving threats.

3. Background

The section provides an overview of key machine learning models we employed, emphasizing their theoretical bases, practical applications, and core mechanics.

3.1 Logistic Regression [42][43]

Logistic Regression is a widely used statistical method for binary classification tasks. It models the probability of a binary outcome based on one or more predictor variables. The logistic function, also known as the sigmoid function, is used to transform the output of a linear combination of the predictor variables into a probability score between 0 and 1. The model learns the optimal coefficients for each predictor variable through the process of maximum likelihood estimation.

Principles: Logistic regression is based on the principle of fitting a linear decision boundary to the data to separate the two classes. It assumes that the relationship between the predictor variables and the log-odds of the outcome variable is linear. The model estimates the probabilities of the binary outcome by transforming the linear combination of predictor variables using the logistic function. The coefficients of the logistic regression model represent the change in the log-odds of the outcome variable for a one-unit change in the predictor variable, holding all other variables constant.

Key Equation: The logistic function, or sigmoid function, translates the linear combination of the predictor variables into a probability score. The logistic regression model can be represented by the following equation:

$$P(Y = 1 \,|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_p x_P)}}$$

Where:

- $P(Y=1/X)$ is the probability of the outcome variable Y being 1 given the predictor variables X.
- e is the base of the natural logarithm.

- $\beta_0, \beta_1, \beta_2,..., \beta_p$ are the coefficients of the predictor variables.
- $X_1, X_2,..., X_p$ are the predictor variables.

Relevance: Logistic regression is particularly useful when the outcome variable is binary and the relationship between the predictor variables and the outcome is assumed to be linear. It is interpretable, making it easy to understand the impact of each predictor variable on the probability of the outcome. Logistic regression is widely used in various fields, including healthcare, finance, and marketing, for predicting binary outcomes such as disease diagnosis, credit risk assessment, and customer churn prediction.

3.2 Decision Trees [44]

Decision Trees are a fundamental model in machine learning used for classification and regression tasks. They operate by recursively splitting data into subsets based on feature values, creating a tree-like structure of decisions. The process involves calculating the best split based on criteria like Gini impurity or entropy for classification, and variance reduction for regression, aiming to maximize the homogeneity of each subset.

Principles: The decision to split at each node is based on the criterion that results in the most significant information gain (IG) or the greatest reduction in impurity.

Key Equation: $IG(D, f) = I(D) - \sum_j \left(\frac{|Dj|}{|D|}\right) * I(D_j)$

Where:

- D is the dataset.
- f is the feature to split on.
- D_j is the subset of D after the split.
- I is the impurity measure.

Relevance: The relevance of decision trees in machine learning lies in their simplicity and interpretability. They easily handle feature interactions and can be visualized, making them accessible to non-technical stakeholders. However, they are prone to overfitting, especially with complex datasets, which necessitates techniques like pruning to enhance their generalizability.

3.3 Random Forest Model [45]

The Random Forest model is a versatile and powerful machine learning algorithm that belongs to the ensemble learning family. It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Renowned for its simplicity and robust performance across distinct types of data, Random Forest is widely used for both classification and regression tasks.

Principles: Random Forest builds upon the concept of bagging (Bootstrap Aggregating) to improve the stability and accuracy of machine learning algorithms. It involves the following key steps:

- **Bootstrap sampling**: Randomly selects samples from the training dataset with replacement to train each tree, ensuring diversity among the trees.
- **Random feature selection**: At each split in the training of individual trees, a random subset of features is chosen. This adds to the diversity, reducing the variance of the model without significantly increasing the bias.
- **Aggregation**: After training, predictions from all individual trees are aggregated to form a final prediction. For classification, this usually means taking a majority vote, and for regression, it means averaging the outcomes.

Equation: While Random Forest does not rely on a single equation like some algorithms, its performance metric, particularly for classification, can be represented by the the aggregation of predictions from individual decision trees:

For regression: By averaging the predictions of all decision trees in the Random Forest ensemble:

$$\hat{Y} = \frac{1}{N}\sum_{i=1}^{N}(f_i(X))$$

Where:

- $\hat{Y}$ is the predicted output (final prediction),
- N is the total number of decision trees,
- $f_i(X)$ represents the prediction of the *ith* decision tree for input X.

For classification: *Majority Vote=mode* $\{C_1, C_2, ..., C_n\}$ where C_i is the class predicted by the *ith* tree, and n is the number of trees. For regression, the prediction is the average of all trees' outputs.

Relevance: In cybersecurity, Random Forest is employed for various tasks such as intrusion detection, malware classification, and phishing attack detection. Its strength lies in its ability to handle large datasets with a high-dimensional feature space and provide insights into feature importance, helping to identify the most significant indicators of malicious or anomalous activity. The ensemble nature of Random Forest makes it robust against overfitting, a common challenge in complex security datasets.

3.4 AdaBoost [46][47]

AdaBoost, or Adaptive Boosting, is an ensemble learning method used for classification. It combines multiple weak classifiers into a strong one by fitting a sequence of weak learners on repeatedly modified versions of the data. The predictions from all weak learners are combined through a weighted majority vote to produce the final prediction. The

key is to increase the weights of incorrectly classified instances, making the algorithm focus more on difficult cases.

Principles: The principle behind AdaBoost centers on iteratively adjusting the weights of instances based on the previous classifier's performance, thereby focusing more on difficult-to-classify instances. This process helps in creating a sequence of models that specialize in correcting the mistakes of their predecessors, leading to a robust combined classifier that improves as more weak learners are added.

Key Equation: The final model, $F(x)$, is derived from the weighted sum of T weak classifiers ($f_t(x)$):

$$F(x) = \sum_{t=1}^{T} (\alpha_t f_t(x))$$

where α_t is the weight of the t-th classifier, influenced by its error rate ε_t , and $f_t(x)$ is the prediction of the weak learner. The weight α_t is calculated as:

$$\alpha_t = \frac{1}{2} * log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

This equation emphasizes the contribution of more accurate classifiers, enhancing classification performance by focusing on challenging instances.

Relevance: AdaBoost's relevance is highlighted by its robustness in enhancing classification performance, especially in scenarios where it is critical to focus on difficult-to-classify instances, making it widely applicable in fields such as image recognition, customer segmentation, and bioinformatics.

3.5 Gradient Boosting [48][49]

Gradient Boosting is a powerful ensemble learning technique that builds predictive models in a sequential manner, with each subsequent model correcting the errors made by the previous ones. The main principle behind Gradient Boosting is to fit a sequence of weak learners (typically decision trees) to the residuals of the preceding model. By iteratively minimizing the loss function, Gradient Boosting gradually improves the model's predictive accuracy.

Principles: The algorithm iteratively fits new models to the residuals of the previous models, optimizing the overall ensemble's performance by minimizing a specified loss function. By combining multiple weak learners, Gradient Boosting creates a strong learner capable of capturing complex relationships within the data.

Key Equation: The key equation governing the Gradient Boosting algorithm is:

$$F_m(x)=F_{m-1}(x)+\lambda\, h_m(x)$$

Where:

- $F_m(x)$ represents the current prediction or output of the ensemble at iteration mm,
- $F_{m-1}(x)$ denotes the prediction made by the ensemble at the previous iteration,
- λ is the learning rate, controlling the step size at each iteration,
- $h_m(x)$ is the weak learner (e.g., decision tree) fitted to the negative gradient of the loss function with respect to the previous model's output.

Relevance: Gradient Boosting has several advantages, including its ability to handle heterogeneous data types, automatic feature selection, and robustness to outliers. However, it may be sensitive to hyperparameters, such as the learning rate and tree depth, requiring careful tuning to achieve optimal performance.

In summary, Gradient Boosting is a versatile and effective machine learning technique widely used in various domains, including cybersecurity, due to its ability to produce accurate predictions by iteratively refining the model's predictions and reducing errors.

3.6 XGBoost (Extreme Gradient Boosting) [50]

XGBoost is an advanced implementation of gradient boosting algorithm that has gained widespread popularity in various machine learning tasks, including classification, regression, and ranking. It is renowned for its scalability, efficiency, and effectiveness in handling large datasets and complex feature spaces.

Principles: involves building an ensemble of decision trees sequentially, where each tree corrects the errors of the previous ones, using a gradient descent algorithm to minimize loss. Key equations include the optimization of an objective function that combines model predictions with a regularization term to prevent overfitting.

Key Equation: XGBoost extends the traditional gradient boosting framework by incorporating additional regularization terms into the objective function, effectively preventing overfitting, and enhancing model generalization. The objective function to minimize can be represented as: $Obj(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$
Where:

- θ represents the parameters of the model,
- $l(l(y_i, \hat{y}_i)$ is the loss function measuring the difference between the true target value y_i and the predicted value $\hat{y}_i$,
- $\Omega(f_k)$ denotes the regularization term penalizing the complexity of the individual trees f_k,
- K is the total number of trees in the ensemble.

Relevance: XGBoost's relevance lies in its ability to handle large-scale and complex data sets, making it a popular choice for challenges in classification, regression, and ranking tasks due to its high accuracy and efficiency.

3.7 LightGBM [51][52][53]

LightGBM, short for Light Gradient Boosting Machine, is an advanced implementation of gradient boosting framework that uses tree-based learning algorithms. It is designed for speed and efficiency, and is widely used in various machine learning tasks, including classification, regression, and ranking. LightGBM is particularly favored for its ability to handle large-sized data and high-dimensional features with ease.

Principles: LightGBM improves traditional gradient boosting methods by using two novel techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). GOSS focuses on instances with larger gradients (i.e., those with higher errors), while EFB reduces the feature space by bundling mutually exclusive features. These innovations allow LightGBM to train faster and use less memory than other gradient boosting frameworks without compromising accuracy.

The algorithm builds the model in the following steps:

1. **Initiate**: Starts by making an initial prediction (usually the mean of the target variable) and calculating the loss.
2. **Iterate**: Sequentially adds weak learners (decision trees), each correcting its predecessor, by choosing the tree that minimizes the loss when added to the ensemble.
3. **Optimize**: Applies GOSS and EFB to focus on more informative instances and features, speeding up calculations and reducing memory usage.
4. **Update**: Updates the model with the addition of a new tree that reduces the overall prediction error.

Key Equation: The objective function in LightGBM, like other gradient boosting methods, combines a loss term and a regularization term. The objective function to minimize is typically defined as:

$$Obj(\theta) = \sum_{i=1}^{N} l(y_i, F(x_i)) + \sum_{k=1}^{K} \Omega(f_k)$$

Where:

- θ represents the parameters of the model,
- N is the number of instances in the training data,
- K is the number of trees,
- $l(y_i, F(x_i))$ is the loss function, which measures the difference between the actual and the predicted values,
- $\Omega(f_k)$ denotes the regularization term penalizing the complex models to prevent overfitting,
- $F(x_i)$ denotes the predicted value for sample x_i.

Relevance: In cybersecurity, LightGBM can be used for a variety of tasks, such as malware detection, phishing website identification, and intrusion detection. Its high efficiency and accuracy make it suitable for processing and analyzing large volumes of data characteristic of network environments. LightGBM's ability to handle imbalanced data sets and its feature importance output are particularly useful for identifying key indicators of malicious activity.

3.8 KMeans [42][54][55][56]

K-means clustering is an unsupervised machine learning algorithm widely used for grouping data into predefined numbers of clusters based on similarity. It is straightforward yet powerful, making it applicable across various domains, including cybersecurity for anomaly detection.

Principles: K-means iteratively assigns data points to clusters based on the nearest mean, with the goal of minimizing the variance within each cluster. The process involves:

1. Randomly initializing cluster centroids.
2. Assigning each data point to the closest centroid, forming clusters.
3. Recalculating the centroids as the mean of all points in a cluster.
4. Repeating the assignment and recalculating steps until convergence.

Key Equation: The algorithm's objective is to minimize the total intra-cluster variance, represented by:

$$J = \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2$$

where J is the objective function, k is the number of clusters, S_i contains the data points in cluster i, x is a data point, and μ_i is the centroid of cluster i.

Relevance: In cybersecurity, K-means is valuable for detecting anomalies that deviate from established patterns, signalling potential threats. By clustering normal behaviour, it highlights outliers as potential security risks. This capability is crucial for identifying novel attacks or breaches without predefined signatures.

3.9 Isolation Forest [57][58]

The Isolation Forest algorithm is an unsupervised learning method used primarily for anomaly detection. It isolates anomalies instead of profiling normal data points, operating under the assumption that anomalies are few and different, making them easier to isolate.

Principles: Isolation Forest, often abbreviated as iForest, isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. The logic is straightforward: anomalies require fewer random splits to be isolated compared to normal points, thus they have shorter paths in the tree structure.

The process involves:

1. Building multiple isolation trees (iTrees) on subsamples of the dataset.
2. Splitting the data recursively until instances are isolated or until a limit in tree depth is reached.
3. Assigning anomaly scores based on the path lengths to isolate observations; shorter paths indicate anomalies.

Key Equation: The anomaly score of an instance is calculated as:

$$s(x, n) = 2^{\frac{-E(h(x))}{c(n)}}$$

where *s(x,n)* is the anomaly score of instance *x* in a dataset of n instances, *E(h(x))* is the average path length of *x* over a collection of isolation trees, and *c(n)* is the average path length of unsuccessful search in a binary search tree.

Relevance: In cybersecurity, Isolation Forest excels in identifying data breaches, intrusions, and other security threats with minimal requirement for domain knowledge. Its ability to handle high-dimensional data and detect subtle anomalies makes it particularly useful for monitoring network traffic, identifying malicious activities, or spotting unusual behaviour in system logs.

4. Experiments:

A- Data Understanding

Introduction to the Dataset:

Our research utilizes a Kaggle dataset from Universidad Del Cauca, featuring 2.7 million network flow instances with 50 features. This dataset is central to our study, providing an invaluable rich source for analyzing and uncovering network traffic patterns, behaviors, and anomalies, using machine learning techniques to enhance network security and management [59]. It represents a significant leap forward in cybersecurity research. The dataset utilized in this study was sourced from an existing collection located on Kaggle [7]. Network packets were aggregated into flows, analyzed for detailed statistics, and labeled by application using the Flow Labeler application and the nDPI (Network Deep Packet Inspection) library. Further details can be found online [7].

Data Features and Significance:

The dataset includes features crucial for identifying network flows, traffic volume, packet sizes, and temporal features, essential for detecting anomalies and threats. Key features are explained in Appendix A.1 (Table 1. Unicauca-dataset-April-June-2019-Network-flows Dataset extracted features).

Dataset Origin and Uniqueness:

Sourced from Universidad Del Cauca, Popay´an, Colombia, the dataset's comprehensive coverage and detailed feature set offer fresh insights into network traffic analysis and cybersecurity, standing out in the domain of machine learning applications for detecting and classifying network threats.

Preliminary Observations:

Our initial examination reveals a dataset that contains 2,704,839 network flow entries with 50 features, including numerical, categorical, and identifier types, crucial for network traffic analysis. It is well-structured with no missing values, covering flow information, traffic statistics, temporal aspects, and bidirectional flow metrics. Features are mapped to OSI model layers, aiding in cybersecurity analysis. The diversity and completeness of the dataset make it suitable for machine learning applications in anomaly detection, traffic classification, and pattern recognition, necessitating normalization and encoding for effective model training. These observations are detailed in Appendix B (B.2).

OSI Layer Feature Distribution

As we delve deeper into network security and traffic analysis, it becomes imperative to understand the distribution of network features across the OSI model's layers. This understanding aids in pinpointing the layers most susceptible to security threats and the type of data most indicative of network behavior [60] [61] [62]. Our dataset, comprehensively categorized in Appendix A.2 Table 2., serves as a foundational pillar for this analysis.

The dataset predominantly encompasses features from the Network layer (Layer 3) upwards to the Application layer (Layer 7), as detailed in Appendix A.2 Table 2. This distribution is not coincidental but rather by design, reflecting the dataset's tailored focus towards high-level network traffic patterns and security analysis. Notably, the absence of features from the Physical (Layer 1) and Data Link (Layer 2) layers suggests an intentional abstraction away from hardware-specific details, allowing for a broader applicability across diverse network environments.

The significance of categorizing features according to the OSI model lies in the strategic insights it offers for cybersecurity measures. For instance, the presence of application layer features (flow_key, application_protocol, web_service) underscores the importance of monitoring software-level interactions for potential security breaches. Conversely, the absence of presentation and session layer specifics hints at the dataset's strategic prioritization of traffic flow and control over data transformation processes.

This layered approach not only facilitates a structured analysis of network traffic and potential vulnerabilities but also guides the development of machine learning models. By focusing on the layers most relevant to security threats, we can tailor feature engineering and model training processes to enhance threat detection and anomaly identification capabilities.

In summary, the OSI Layer Feature Distribution, as outlined in Appendix A.2 Table 2., provides a roadmap for navigating the complex landscape of network security analysis. It underscores the dataset's alignment with contemporary cybersecurity challenges [63], highlighting the critical layers and features that are pivotal for detecting and mitigating security threats.

B- Experiments

Introduction to Methodology

This methodology section lays the foundation for our investigative journey into network security using machine learning (ML). It delineates our analytical framework, steering our exploration toward identifying network anomalies and security threats. This blueprint encapsulates our approach to dissecting the intricate dynamics of network traffic, aiming to unearth underlying patterns indicative of security vulnerabilities.

Experiments Settings:

The platform utilized for the execution of our experiments was Google Colab, which provided a cloud-based environment conducive to high-intensity computational tasks. From Google Colab, we managed the code execution and subsequently uploaded the results to a GitHub repository, ensuring transparency and reproducibility of our research process.

Our experimental dataset comprised 100,000 instances, selected to represent a broad spectrum of scenarios within our study's scope. Initial tests were conducted on randomly selected subsets of the data, indicating consistent outcomes across varying sample sizes. Notably, we observed that when the dataset exceeded 50,000 instances, the results demonstrated a high level of consistency, suggesting that our models' performance stabilizes beyond this threshold. This observation underscores the robustness of our analytical approach and the reliability of our findings within the specified computational environment.

Research Design and Approach

Our research is methodically partitioned into three critical segments, each focusing on distinct aspects of network traffic analysis:

- **Traffic Classification and Security Analysis:** At the core of our study, we leverage a traffic flow function for labeling anomaly instances then a series of supervised learning techniques for evaluation and comparison. This dual approach facilitates the precise categorization of network flows, augmenting traditional analysis with ML-driven insights to distinguish between benign and potentially malicious traffic. Evaluative metrics such as accuracy, recall, precision, F1-score, ROC-AUC, process running time aid in refining model performance, ensuring robust threat detection capabilities.

- **General Anomaly Detection:** Venturing into the realm of unsupervised learning, we deploy algorithms like Isolation Forest and K-Means to detect outliers. This segment emphasizes the detection of atypical network behaviors without relying on pre-labeled data, highlighting the versatility of ML in identifying uncharted threats.

- **Volume Analysis:** Through time-series analysis and unsupervised learning methods, this segment scrutinizes traffic volume trends. The objective is to pinpoint fluctuations indicative of network issues, such as congestion or potential DDoS attacks, offering a proactive stance on network management.

Data Preprocessing

We initiated the data preprocessing phase, essential for network traffic analysis, with a focus on cyber threat detection. This phase involved comprehensive cleaning, normalization, and transformation techniques to refine the dataset for optimal algorithm performance.

Exploratory Data Analysis (EDA):

Utilizing a correlation matrix, we identified relationships among dataset features, aiding in feature selection. This analysis enabled the elimination of half of the highly correlated feature pairs while retaining those most informative for classification purposes.

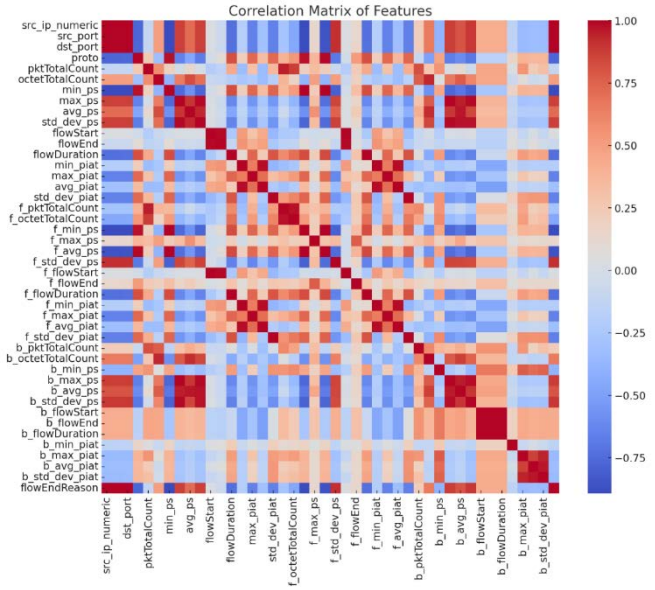

Figure 1 Correlation Matrix

Detailed Feature Analysis:

- The detailed feature analysis within the thesis categorizes the dataset's 50 features into 44 numerical and 6 categorical types, assessing their relevance and redundancy for network traffic analysis. Numerical features, quantifying aspects like packet and byte counts, alongside categorical features, identifying traffic types, form the basis of the analysis. The study identifies redundant features and suggests dropping them to streamline the dataset, such as overlapping information provided by IP address representations, as well as duplicated flow statistics.
- A significant portion of the analysis focuses on refining forward and backward flow metrics, emphasizing the retention of average and standard deviation values for packet sizes and inter-arrival times due to their high correlation, which suggests a redundancy in keeping all metrics. Additionally, the research advocates for the selection of key features indicative of network traffic behaviour, including packet counts, data volume, and packet size statistics, while recommending the elimination of perfectly correlated inter-arrival time features to avoid redundancy.

Cleaning the Data

With no missing values present, our focus shifted to handling missing values, removing duplicates, examining unique values, assessing categorical feature cardinality. We relied on the correlation matrix to guide our elimination of redundant features, ensuring a streamlined and informative feature set for model training.

Data Preprocessing Steps for Network Traffic Analysis

We followed a structured approach to prepare the data, which involved:

- **Labeling Traffic Data**: A labeling function categorizes traffic as 'normal' or 'malicious' based on predefined criteria, transforming raw data into a format suitable for supervised learning.
- **Encoding Categorical Labels [**64**]**: Applied one-hot encoding and LabelEncoder for categorical variables, facilitating their use in various machine learning models.
- **Feature Selection**: In our analysis, we conducted two distinct experiments to evaluate the impact of feature selection on model performance. In the first experiment, we utilized the complete set of features available in our dataset, aiming to establish a baseline for classification accuracy. This comprehensive approach (Appendix B (B.3.1) allowed us to assess the fundamental value of each feature in predicting network traffic categories. Conversely, in the second experiment (Appendix B (B.3.2)), we refined our feature set based on EDA findings by excluding specific features deemed redundant or less informative or highly correlated, such as 'flow_key', 'src_ip_numeric', 'min_ps', 'max_ps', ‘f_flowStart’, ‘f_flowEnd’, ‘f_flowDuration’, ‘f_min_piat’, ‘f_max_piat’, and all b_ prefixed features, to test the hypothesis that a reduced feature set could enhance model efficiency without compromising accuracy. These experiments underscore the critical role of feature selection [65] in optimizing machine learning models for network traffic classification.

This rigorous preprocessing ensured our used dataset was primed for the application of advanced machine learning techniques, setting a solid foundation for the subsequent analysis phases detailed in Appendices B.1 and B.2.

❖ In the first segment **“Traffic Classification and Security Analysis”** of our study, we embarked on a comprehensive analysis to classify network traffic into normal and malicious categories, leveraging a dataset of network flows. This methodology, encapsulated within the script (detailed in Appendix B.1 and B.2), is essential for enhancing cybersecurity measures.

1. Initial Setup and Library Import: We initiated our analysis by importing essential Python libraries, facilitating data manipulation, visualization, and machine learning. This foundational step ensures our workflow is equipped with the necessary tools for sophisticated data analysis.

2. Function Definitions: We defined key functions to streamline our workflow, including record_time for tracking the analysis duration, plot_confusion_matrix for visualizing classifier performance, and plot_roc_curve to

assess the trade-off in classifier decision thresholds. Notably, the label_traffic function was developed to categorize network traffic based on predefined criteria, enriching our dataset with crucial labels for supervised learning.

3. Data Labeling and Preprocessing: Following data importation, we applied the label_traffic function, systematically classifying each network flow. This step is critical for distinguishing between benign and malicious traffic patterns.

4. Handling Missing Values and Encoding: We addressed missing values by substituting them with mean values for numeric columns and placeholder strings for non-numeric columns. Additionally, categorical labels were encoded using LabelEncoder and One-Hot techniques [64] to numeric formats, rendering the dataset amenable to machine learning algorithms.

5. Network Traffic Labeling: The traffic label function systematically labels network traffic based on predefined criteria that include protocols, port ranges, packet counts, octet counts, packet sizes, flow durations, and packet inter-arrival times (PIAT). These criteria were meticulously selected to capture the essence of normal network behavior, with deviations from these norms flagged as malicious. This nuanced approach allows for a detailed exploration of network behavior, facilitating the identification of anomalies.

6. Data Splitting and Scaling: Train-Test Split: The dataset is split into features (X) and the target variable (y), followed by further splitting into training and testing sets (80% for training, 20% for testing) using train_test_split, with a fixed random seed for reproducibility. This ensures that the same split is achieved every time someone runs the code and enables other researchers to verify results, compare methods, and ensures transparency in our research. We used stratify=y which means the splitting process maintains the proportion of each class label in both the training and testing sets. It is especially useful when dealing with imbalanced datasets.

7. Feature Scaling: Normalizes numeric features using StandardScaler. It is applied to numeric features to rescale them, ensuring that their values have a mean of 0 and a standard deviation of 1, thus preventing any single feature from dominating the model training process and impacting the model performance [66].

8. Class Imbalance Handling [67]: We employed the Synthetic Minority Over-sampling Technique (SMOTE) to address class imbalance, a common challenge in network traffic datasets.

9. Model Preparation and Evaluation: Our research employed a meticulous model preparation and evaluation strategy, ensuring robustness and accuracy in classifying network traffic. This process involved two critical stages: cross-validation setup and hyperparameter tuning, followed by an exhaustive training and evaluation of classifiers.

10. Cross-Validation Setup [68]: We adopted the StratifiedKFold method for cross-validation, setting the number of folds to five. This approach guaranteed that each fold reflected the overall class distribution, allowing for a more balanced and comprehensive evaluation of model performance across various subsets of the data.

11. Hyperparameter Tuning [68]: Utilizing the RandomizedSearchCV technique, we conducted hyperparameter tuning for each of our classifiers, including Logistic Regression, Random Forest, AdaBoost, Gradient Boosting, Decision Tree, XGBoost, and LightGBM. This step was crucial for identifying the optimal configuration of parameters for each model, enhancing their predictive capabilities.

Table 3: Summary of Best Hyperparameter Tuning Results for Various Machine Learning Models

Model	Hyperparameters	Best Parameters
Logistic Regression	C, solver, max_iter	solver='liblinear', max_iter=500, C=10
Decision Tree	max_depth, min_samples_split, min_samples_leaf, criterion	min_samples_split=5, min_samples_leaf=1, max_depth=30, criterion='entropy'
Random Forest	n_estimators, max_depth	n_estimators=100, max_depth=None
AdaBoost	n_estimators, learning_rate	n_estimators=200, learning_rate=1
Gradient Boosting	n_estimators, learning_rate, max_depth	n_estimators=100, max_depth=7, learning_rate=1
XGBoost	n_estimators, max_depth, learning_rate	n_estimators=200, max_depth=6, learning_rate=0.1, use_label_encoder=False, eval_metric='logloss'
LightGBM	n_estimators, max_depth, learning_rate	n_estimators=200, max_depth=10, learning_rate=0.1

12. Evaluation: Following the tuning, we embarked on classifier training, utilizing the preprocessed data. Each model was rigorously trained and then evaluated using a separate testing dataset. The evaluation focused on various performance metrics, such as accuracy, F1-score, recall, precision, and the area under the ROC curve. Additionally,

confusion matrices and ROC curves played pivotal roles in interpreting each classifier's performance, providing insights into their ability to distinguish between normal and malicious traffic effectively.

Evaluation involves the following steps:
Predictions: The trained classifiers are used to make predictions on the testing data. These predictions are compared to the actual target labels in the testing data to assess how well the classifiers are performing.

Confusion Matrices: Confusion matrices are created for each classifier. These matrices provide a breakdown of true positive, true negative, false positive, and false negative predictions. They help in understanding the classifier's performance in terms of correct and incorrect predictions. The confusion matrix is typically generated using the predictions made by the classifier on the testing data 20% of the total number of instances.

Confusion Matrix Breakdown [69]:
True Positives (TP): Correctly predicted positive cases.
True Negatives (TN): Correctly predicted negative cases.
False Positives (FP): Incorrectly predicted positive cases.
False Negatives (FN): Incorrectly predicted negative cases.

ROC Curves [69]: Receiver Operating Characteristic (ROC) curves are plotted for each classifier. ROC curves illustrate the trade-off between the true positive rate and false positive rate at different thresholds. The area under the ROC curve (AUC-ROC) is a measure of a classifier's ability to distinguish between classes. measures the ability to distinguish between classes, with 1 being perfect and 0.5 no better than random. A higher AUC-ROC indicates better performance.

Performance Metrics: Several performance metrics [69] are calculated for each classifier, including:

- Accuracy: The proportion of correct predictions.
- F1-Score: A balanced metric that considers both precision and recall.
- Recall (Sensitivity): The proportion of actual positives correctly predicted.
- Precision: The proportion of predicted positives that are correct.
- AUC-ROC: The area under the ROC curve, as mentioned earlier.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \qquad \text{F1-Score} = 2 \times \frac{Precision\ x\ Recall}{Precision + Recall}$$

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

❖ In the second segment **"General Anomaly Detection,"** the study begins by addressing the challenge of detecting network traffic anomalies through the application of Isolation Forest and K-Means clustering algorithms. The primary objective is to identify deviations from standard network behavior patterns.

- **Data Preparation**: Network traffic data is prepared by selecting significant features such as 'pktTotalCount,' 'octetTotalCount,' 'flowDuration,' 'min_ps,' 'max_ps,' 'avg_ps,' and 'std_dev_ps,' which collectively provide a detailed snapshot of network activity.
- **Handling Missing Values:** The integrity of the analysis is maintained by filling missing values with the median of corresponding features, ensuring no data point is overlooked due to incomplete information.
- **Isolation Forest Application:** This algorithm is applied to isolate anomalies by evaluating how data points diverge from the majority, utilizing the Interquartile Range (IQR) method for defining what constitutes an anomaly.
- **K-Means Clustering Application:** K-Means is used to further investigate anomalies by grouping data points into clusters and identifying outliers based on their distance from cluster centers.

❖ The focus of the third segment **"Volume Analysis,"** is a comprehensive examination of network traffic volume to understand data transfer patterns, which is crucial for network management tasks such as performance monitoring and capacity planning. This analysis employs data processing and clustering techniques, specifically focusing on two primary features: octetTotalCount (total bytes transferred) and pktTotalCount (total packets exchanged), analyzed over various time intervals.

- **K-Means Clustering for OctetTotalCount and Time Interval**: The analysis begins with applying K-Means clustering to segment network flows by bytes transferred (octetTotalCount) and the time interval since the dataset's start. This approach aims to uncover patterns or anomalies in the volume of network traffic.
- **Clustering Based on pktTotalCount**: A similar clustering strategy is used for packet count (pktTotalCount) and flow duration, offering insights into the packet-level behavior of network traffic.

C- Results

The evaluation revealed that certain models, notably Random Forest and XGBoost, demonstrated superior performance, showcasing their efficacy in network traffic classification. These findings underscore the importance of a systematic approach to model preparation and evaluation in machine learning-driven cybersecurity research.

Table 4: Performance Metrics Summary of Machine Learning Models on Test Data

Total running Process Time: 2:16:01.453190 hours

Classifier	Accuracy	F1-Score	Recall	Precision	AUC-ROC	Running Time (s)
Logistic Regression	0.7011	0.7398	0.7797	0.7038	0.7800	1.3030
Random Forest	0.9404	0.9458	0.9541	0.9377	0.9845	39.1685
AdaBoost	0.8241	0.8446	0.8771	0.8145	0.9010	18.8132
Gradient Boosting	0.8601	0.8766	0.9113	0.8444	0.9382	92.7662
Decision Tree	0.9098	0.9169	0.9128	0.9211	0.9095	4.5506
XGBoost	0.9275	0.9338	0.9374	0.9302	0.9795	2.4863
LightGBM	0.9033	0.9122	0.9210	0.9035	0.9696	3.4357

Comparison:

- Random Forest shows exceptional performance with the highest accuracy (0.9404) and AUC-ROC (0.9845), indicating its strong predictive power and ability to distinguish between classes.
- Gradient Boosting excels in recall (0.9113), highlighting its effectiveness in identifying true positives, with significant computational time (92.7662s).
- AdaBoost and Gradient Boosting demonstrate a good balance across metrics, suggesting robustness, with AdaBoost being faster (18.8132s).
- Decision Tree offers a good speed (4.5506s) with high precision (0.9211), making it a quick and fairly accurate option.
- XGBoost and LightGBM provide a solid mix of high accuracy and AUC-ROC with relatively low running times, indicating efficiency and effectiveness.
- Logistic Regression, while not leading in any metric, requires the least running time (1.3030s), offering a fast but less accurate option.

Overall, the Random Forest Classifier appears to be the best-performing model based on the provided metrics, with high accuracy, F1-Score, and AUC-ROC. However, the choice of the best model may also depend on the specific requirements of an application and the trade-offs between precision and recall. The significance of this analysis is in checking and comparing various machine learning classifiers to decide if an instance is 'normal' or 'malicious'. By considering metrics like accuracy, F1-Score, recall, precision, and AUC-ROC, as well as the running time, we can make informed decisions about which classifier is most suitable for this specific task.

Key takeaways from the performance metrics summary are:

1. **Performance Variation Across Models:** Models show diverse accuracy, with Random Forest and XGBoost leading in performance, while Logistic Regression lags behind.
2. **Discrimination Capability:** Random Forest excels in distinguishing between classes (highest AUC-ROC), indicating superior predictive power compared to others.
3. **Computational Efficiency:** Logistic Regression and Decision Tree are efficient, requiring significantly less time. Gradient Boosting is the most resource intensive.
4. **Balancing Precision and Recall:** Decision Tree and XGBoost demonstrate a good balance, with high scores in both precision and recall, indicating fewer false positives and negatives.
5. **Choosing the Right Model:** Random Forest is ideal for scenarios prioritizing accuracy and discrimination capability but is computationally intensive. Logistic Regression suits rapid, less resource-intensive tasks but at the cost of lower accuracy.

In selecting the optimal model for network traffic classification, XGBoost emerges as a preferred choice, closely followed by LightGBM. These models blend high accuracy, robust discrimination capability, and relative computational efficiency, making them suitable for complex classification tasks where precision and scalability are paramount. The selection underscores a strategic compromise between computational demands and predictive performance, positioning XGBoost and LightGBM as leading solutions in scenarios requiring depth of analysis and efficiency.

Table 5: Performance Metrics Summary of ML Models on Test Data (Reduced Features)

Total running Process Time: 1:36:25.757937 hours

Classifier	Accuracy	F1-Score	Recall	Precision	AUC-ROC	Running Time (s)
Logistic Regression	0.6962	0.7418	0.8007	0.6910	0.7733	1.4601
Random Forest	0.9311	0.9374	0.9462	0.9287	0.9813	35.1135
AdaBoost	0.8167	0.8372	0.8651	0.8111	0.8931	12.3459
Gradient Boosting	0.8495	0.8683	0.9105	0.8299	0.9282	58.3760
Decision Tree	0.9036	0.9108	0.9032	0.9185	0.9036	2.5394

XGBoost	0.9146	0.9220	0.9253	0.9187	0.9733	1.5698
LightGBM	0.8945	0.9049	0.9212	0.8892	0.9615	2.2330

With reduced features, the performance of the classifiers remains consistent with the previous analysis. Random Forest still achieves the highest accuracy, F1-Score, and AUC-ROC score, indicating strong predictive performance. However, Logistic Regression and Decision Tree classifiers continue to offer faster predictions with slightly lower accuracy.

Computational Efficiency and Running Time Analysis

Our analysis on machine learning models for cybersecurity reveals key insights into computational efficiency and its impact on threat detection:

Comparative Efficiency: We noted a significant improvement in computational efficiency, with the overall processing time decreasing from 2:16:01.453190 hours to 1:36:25.757937 hours after feature reduction. This observation points to the benefit of utilizing a streamlined feature set, which not only maintains model accuracy but also enhances processing speed, a vital factor in real-time cybersecurity threat detection.

Operational Implications: The variance in running times across models highlights the operational trade-offs between computational efficiency and predictive accuracy. Models such as Logistic Regression, which consistently showed the lowest running times, offer practical solutions for real-time or near-real-time applications, where rapid threat response is paramount. Conversely, models like Gradient Boosting, despite their longer running times, offer depth in analysis, suitable for comprehensive threat assessment where time is not the primary constraint.

Model Selection Considerations: Our findings illustrate the importance of model selection based on specific operational needs. For instance, Logistic Regression and XGBoost emerge as preferable choices for scenarios demanding swift threat detection due to their minimal running times. On the other hand, Gradient Boosting, with the longest running time, may be more suited for in-depth, periodic analysis rather than real-time threat detection, given its computational demand.

Practical Recommendations: To strike a balance between accuracy and computational efficiency for scalable models, we suggest the following:

- For Real-Time Threat Detection: Prioritize models with shorter running times, such as Logistic Regression or XGBoost, to facilitate swift response capabilities.
- For Comprehensive Analysis: When time constraints are less stringent, opt for models that emphasize accuracy over speed, like Gradient Boosting, which are well-suited for in-depth evaluations.
- Regular Assessment: It's essential to continuously evaluate the trade-off between feature complexity and operational efficiency to ensure cybersecurity operations are both effective and efficient.

In the second segment- General Anomaly Detection (Appendix B (B.4)), the work is to delve into identifying deviations from expected network behavior, leveraging Isolation Forest and K-Means clustering algorithms. We initiated our analysis by selecting key features that encapsulate network traffic dynamics. Our approach utilized Isolation Forest to pinpoint outliers based on data isolation criteria and applied K-Means clustering to detect anomalies through data segmentation.

A comprehensive statistical analysis was conducted on the identified key features, employing box plots and distribution plots to explore their characteristics thoroughly. This analysis helped in understanding the data's distribution, central tendency, and variability.

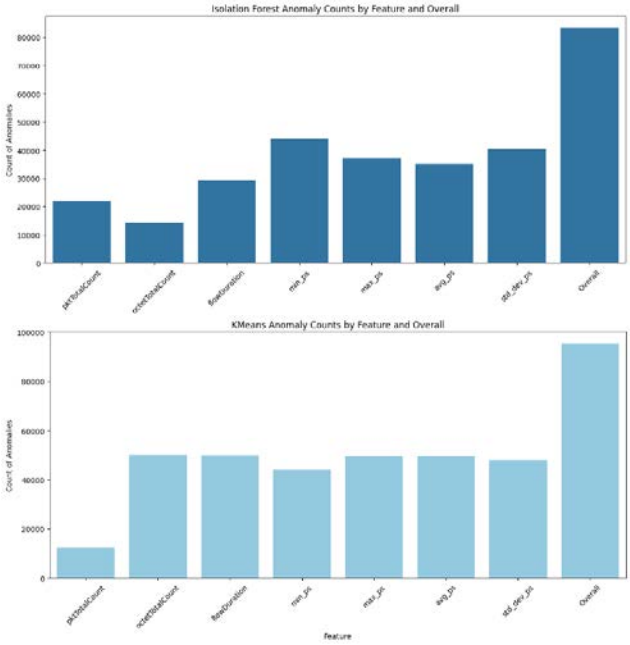

Figure 2. Detecting Anomalies with Isolation Forest and K-Means Clustering Techniques

Additionally, the Interquartile Range (IQR) method was employed to quantify anomalies, providing a clear picture of their prevalence across different features. Visualizations, including bar charts of anomaly percentages shown in (Appendix B (B.4)), offered insightful views into which features were more prone to anomalous behaviors, highlighting potential areas of concern in network traffic.

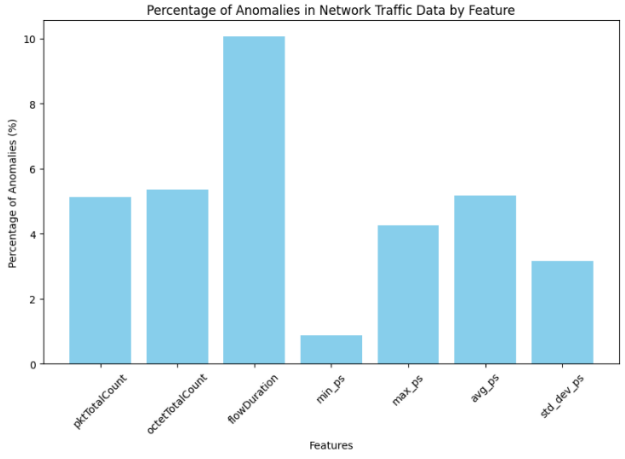

Figure 3. Percentage of Anomalies in Network Traffic Data by Feature

By leveraging advanced algorithms and statistical analyses, we could uncover hidden anomalies within network traffic data, offering valuable insights for proactive network management and security enhancement.

The segment of "Volume Analysis" (Appendix B (B.5)) presents an in-depth examination of network traffic volume, crucial for understanding data transfer patterns and aiding in network management. Utilizing K-Means clustering, we analyzed key features like 'octetTotalCount' and 'pktTotalCount' to identify traffic patterns and potential anomalies.

K-Means Clustering for OctetTotalCount and Time Interval: Applied K-Means clustering revealed distinct patterns in data transfer volumes, as visualized in Figure 4, demonstrating the clustering based on octetTotalCount.

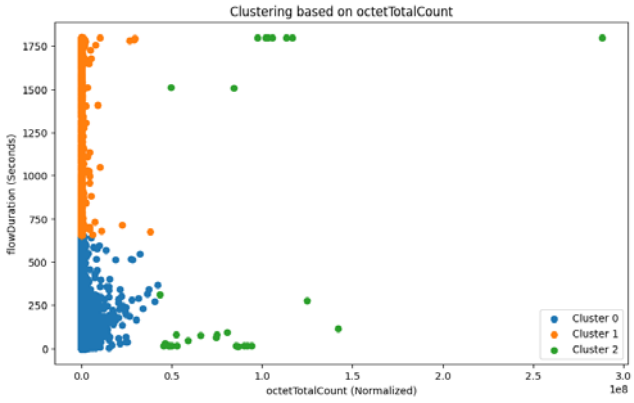

Figure 4. Clustering based on octetTotalCount

Clustering Based on pktTotalCount: This analysis, shown in Figure 5, provided insights into packet-level traffic behavior, highlighting how different network flow characteristics contribute to traffic volume.

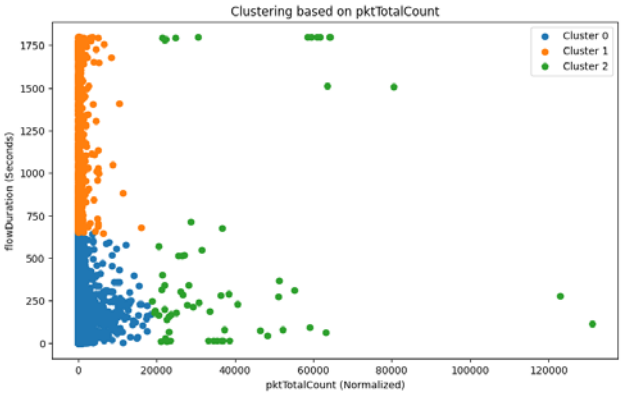

Figure 5. Clustering based on pktTotalCount

Visualization of Data Volume: Through visualizations in 30-minute intervals (Figure 6), we identified peak data transfer periods and assessed the impact of various categories on network traffic.

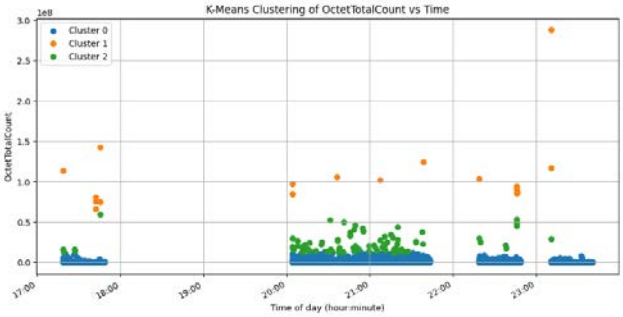

Figure 6. K-Means Clustering of octetTotalCount vs Time

Identification of Most Impactful Categories: By analyzing data volume across time intervals, we determined the categories with significant contributions to data transfer volume, enhancing the understanding of network load distribution.

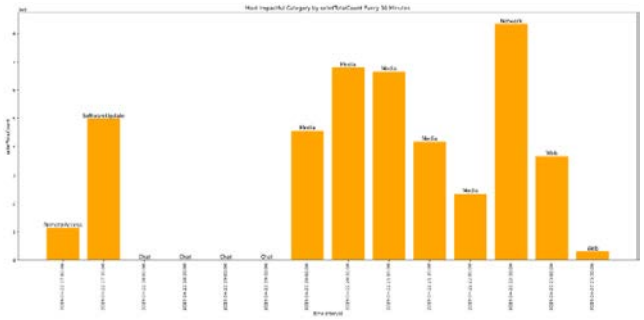

Figure 7. Most Impactful Category by octetTotalCount vs Time

The clustering and visualization techniques employed not only uncovered traffic patterns but also facilitated performance monitoring and capacity planning. These findings, supported by Figures 4 to 7, underscore the importance of volume analysis in network management and anomaly detection, guiding data-driven decisions to optimize network performance and security.

D. Remarks:

The results obtained firmly align with the initial hypotheses and the theoretical framework established in the "Research Experiments" section. The outcomes from Traffic Classification, Anomaly Detection, and Volume Analysis not only verify the objectives outlined but also demonstrate the study's alignment with its core aims, highlighting the effectiveness of our chosen methodologies.

Our study introduces an innovative approach by analyzing network traffic flow through various methods, adding a novel perspective to the field. This multifaceted analysis, leveraging both supervised and unsupervised learning, along with statistical calculations and visual plotting, offers deep insights into the network's capacity and behavior. Such comprehensive analysis is a significant advancement, providing nuanced understandings that were previously unexplored.

The excellent performance metrics obtained from our experiments underscore the robustness and efficiency of the models and techniques employed. Our findings resonate with existing studies, as mentioned in the "Related Work" section, and contribute to the growing body of evidence within our research domain. Where our results differ from previous works, we attribute these variations to methodological, dataset, or analytical technique differences, ensuring our research's context is well understood.

Statistical tests conducted to assess the significance of our results support their validity, forming a solid foundation for our conclusions. The meticulous approach in data preprocessing, feature analysis, and evaluation metrics guarantees methodological consistency, enhancing the reliability and robustness of our findings.

The practical implications of our study are significant, addressing the identified real-world problem in the "Introduction." By offering insights into network security and management, our research demonstrates its practical utility, providing administrators with crucial information for decision-making. This real-world applicability underscores the relevance of our work, bridging theoretical research and practical needs.

Furthermore, the diversity of analytical approaches, including the outcomes of supervised and unsupervised learning, enriches the research landscape. This innovative exploration into network traffic analysis presents new avenues for understanding and managing network dynamics, marking a notable contribution to the field.

Our high-quality dataset, reflective of real-world network flows and security threats, underpins the credibility of our results. While network dynamics can vary widely, the methodologies and findings of this study offer a solid foundation for adaptation and validation across different network environments, highlighting the versatility and applicability of our research.

Acknowledging the continuous evolution of network security challenges, we advocate for future research directions that extend our findings, such as applying our models to various datasets or real-world scenarios. These efforts will further ascertain the generalizability and practicality of our work, ensuring its ongoing relevance and contribution to the field of network security research.

In conclusion, our study not only aligns with its set objectives but also exceeds expectations by delivering excellent performance and introducing innovative analytical perspectives. This dual achievement—adherence to research goals and innovation—provides valuable insights for network administrators and contributes significantly to the scholarly and practical understanding of network traffic analysis.

5. Discussion and Outlook

5.1 Contribution to Understanding the Research Problem

Our study significantly advances the understanding of network traffic analysis, demonstrating the efficacy of machine learning in enhancing cybersecurity frameworks. Through rigorous model performance evaluation, we have shown that machine learning models, particularly the

Random Forest classifier, are robust in classifying network traffic, detecting anomalies, and analyzing traffic volume trends. Notably, feature reduction, while streamlining the model, does not compromise its performance, thereby providing a solid foundation for future research and practical applications in network security and management.

5.2 Discussion: Interpretation of Results

Our research delves into the application of machine learning to network traffic data, aiming to bolster network security and performance. We have addressed several key questions:

1. **Identifying Security Threats:** Machine learning significantly improves the detection of security threats, pinpointing unusual activities indicative of breaches.
2. **Classifying Network Activities:** Random Forest and Gradient Boosting emerge as efficient techniques for classifying network activities.
3. **Detecting Network Inefficiencies:** Machine learning, especially Gradient Boosting, excels in identifying and predicting network inefficiencies.
4. **Enhancing Security and Traffic Management:** The deployment of machine learning models contributes to more robust security measures and efficient traffic management.
5. **Continuous Evaluation and Adaptation:** We've implemented strategies for continuous model evaluation and adaptation, ensuring effectiveness under dynamic network conditions.
6. **Impacting Customer Satisfaction and Business Reputation:** Machine learning's role in network analysis positively impacts customer satisfaction and business reputation, especially in cybersecurity contexts.

Our findings not only extend theoretical understanding but also offer practical insights, marking significant advancements in network security and management through machine learning.

6. Conclusion

This study has established a comprehensive framework for network security enhancement through the application of advanced machine learning techniques. Our investigation into network traffic behavior, leveraging both supervised and unsupervised learning algorithms, has led to the development of a novel benchmark for evaluating network anomalies and security threats. The primary contributions of our research can be summarized as follows:

1. **Innovative Benchmark Creation**: We introduced a new benchmark based on our experimental dataset, which significantly advances the methodologies for identifying malicious network activities. This benchmark provides a robust foundation for future comparative studies and algorithmic development in the field of cybersecurity.
2. **Enhanced Network Security**: Our findings demonstrate the efficacy of machine learning in differentiating between benign and malicious traffic, thereby significantly improving threat detection capabilities. The utilization of algorithms like Isolation Forest and KMeans has been instrumental in detecting anomalies, highlighting the practical utility of our research for network administrators and cybersecurity professionals.
3. **Valuable Insights into Network Traffic**: Through detailed analysis, our research has uncovered intricate patterns in network traffic behavior and volume trends. These insights are crucial for preempting and mitigating potential network issues, including congestion and DDoS attacks.
4. **Optimization of Machine Learning Models**: The study emphasizes the importance of tailored feature selection in enhancing the accuracy and efficiency of machine learning models for network security. Our approach ensures that these models are both effective and practical for real-world application.
5. **Future Directions**: Acknowledging the rapid evolution of network threats and the limitations posed by our dataset's scope, future research should focus on expanding the diversity and representativeness of datasets. This expansion will facilitate the development of more generalized and robust models. Moreover, exploring real-time analysis capabilities and integrating machine learning solutions into existing network infrastructures remain paramount for advancing network security methodologies.

In conclusion, our research contributes a significant benchmark to the domain of network security, offering a new perspective on the use of machine learning techniques for network traffic analysis. By addressing the current limitations and following the outlined recommendations, future work can build upon our findings to further enhance network resilience against an ever-evolving threat landscape. Our study not only advances the theoretical understanding of network behavior and anomaly detection but also provides practical insights for improving network security protocols and strategies.

References

[1] Bierbrauer DA, Chang A, Kritzer W, Bastian ND. Cybersecurity anomaly detection in adversarial environments. arXiv preprint arXiv:2105.06742. 2021 May 14.

[2] Sarker IH, Furhad MH, Nowrozy R. Ai-driven cybersecurity: an overview, security intelligence modeling and research directions. SN Computer Science. 2021 May;2(3):173.

[3] Makridis CA. Do data breaches damage reputation? Evidence from 45 companies between 2002 and 2018. Journal of Cybersecurity. 2021 Jan 1;7(1):tyab021.
[4] Kim J, Lennon SJ. Effects of reputation and website quality on online consumers' emotion, perceived risk and purchase intention: Based on the stimulus-organism-response model. J Res Interact Mark. 2013;7(1):33-56. https://doi.org/10.1108/17505931311316734.
[5] Sarker IH, Kayes AS, Badsha S, Alqahtani H, Watters P, Ng A. Cybersecurity data science: an overview from machine learning perspective. Journal of Big data. 2020 Dec;7:1-29.
[6] Alani MM. Big data in cybersecurity: a survey of applications and future trends. Journal of Reliable Intelligent Environments. 2021 Jun;7(2):85-114.
[7] Rojas JS. Labeled network traffic flows 114 applications [Internet]. [cited 2020 Apr 7]. Available from: https://www.kaggle.com/jsrojas/labeled-network-traffic-flows-114-applications.
[8] Mvula PK, Branco P, Jourdan GV, et al. A systematic literature review of cyber-security data repositories and performance assessment metrics for semi-supervised learning. Discov Data. 2023;1:4. https://doi.org/10.1007/s44248-023-00003-x
[9] Badonnel R, Fung C, Scott-Hayward S, Li Q, Valenza F, Hesselman C. Guest editors introduction: special section on recent advances in network security management. IEEE Trans Netw Serv Manag. 2022;19(3):2251-2254. https://doi.org/10.1109/TNSM.2022.3202426
[10] Zakroum M, Francois J, Chrisment I, Ghogho M. Monitoring Network Telescopes and Inferring Anomalous Traffic Through the Prediction of Probing Rates. IEEE Trans Netw Serv Manag. 2022; pp.1-1. https://doi.org/10.1109/TNSM.2022.3183497.
[11] Alqudah N, Yaseen Q. Machine Learning for Traffic Analysis: A Review. Procedia Comput Sci. 2020;170:911-916. https://doi.org/10.1016/j.procs.2020.03.111
[12] Kumar P, Pandey D, Srivastav RK, Pandey PK. Network Traffic Analysis and Prediction Using Machine Learning. Int J Res Publ Rev. 2023;4(8):2071-2075.
[13] Bozkır R, Cicioğlu M, Çalhan A, Toğay C. A new platform for machine-learning-based network traffic classification. Comput Commun. 2023;208:1-14. https://doi.org/10.1016/j.comcom.2023.05.010
[14] Almuhammadi S, Alnajim A, Ayub M. QUIC Network Traffic Classification Using Ensemble Machine Learning Techniques. Appl Sci. 2023;13(8):4725. https://doi.org/10.3390/app13084725.
[15] Girubagari N, Ravi TN. Methods of anomaly detection for the prevention and detection of cyber attacks. Int J Intell Eng Inform. 2023;11(4):299-316. https://doi.org/10.1504/IJIEI.2023.136097.
[16] Nour M, Slay J. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective. 2016;25(3):18-31.
[17] Fosić I, Žagar D, Grgić K, Križanović V. Anomaly detection in NetFlow network traffic using supervised machine learning algorithms. Journal of industrial information integration. 2023 Apr 20:100466.
[18] Zoghi Z, Serpen G. UNSW-NB15 computer security dataset: Analysis through visualization. Security and Privacy. 2024 Jan;7(1):e331.
[19] Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In2015 military communications and information systems conference (MilCIS) 2015 Nov 10 (pp. 1-6). IEEE.
[20] Ahmed Y, Azad MA, Asyhari T. Rapid Forecasting of Cyber Events Using Machine Learning-Enabled Features. Information. 2024 Jan 11;15(1):36.
[21] Sarker IH. Machine Learning for Intelligent Data Analysis and Automation in Cybersecurity: Current and Future Prospects. Ann Data Sci. 2023;10:1473–1498. https://doi.org/10.1007/s40745-022-00444-2
[22] Nassar A, Kamal M. Machine Learning and Big Data analytics for Cybersecurity Threat Detection: A Holistic review of techniques and case studies. Journal of Artificial Intelligence and Machine Learning in Management. 2021 Feb 6;5(1):51-63.
[23] Amit I, Matherly J, Hewlett W, Xu Z, Meshi Y, Weinberger Y. Machine Learning in Cyber-Security - Problems, Challenges and Data Sets. In: The AAAI-19 Workshop on Engineering Dependable and Secure Machine Learning Systems; 2019. Available from: https://arxiv.org/abs/1812.07858v3.
[24] Guo K, Tan Z, Luo E, Zhou X. Machine learning: The cyber-security, privacy, and public safety opportunities and challenges for emerging applications. Security and Communication Networks. 2021 Dec 3;2021:1-2.
[25] Shaukat K, Luo S, Varadharajan V, Hameed IA, Chen S, Liu D, Li J. Performance comparison and current challenges of using machine learning techniques in cybersecurity. Energies. 2020 May 15;13(10):2509.
[26] Sarker IH. Data Science and Analytics: An Overview from Data-Driven Smart Computing, Decision-Making and Applications Perspective. SN Comput Sci. 2021;2:377. https://doi.org/10.1007/s42979-021-00765-8
[27] Wu HY, Klein K, Yan D. Effective Network Analytics: Network Visualization and Graph Data Management. IEEE Computer Graphics and Applications. 2023 May 17;43(3):10-1.
[28] Aouedi O, Piamrat K, Hamma S, Perera JM. Network traffic analysis using machine learning: an unsupervised approach to understand and slice your network. Annals of Telecommunications. 2022 Jun;77(5):297-309.
[29] Sarker IH. Deep Cybersecurity: A Comprehensive Overview from Neural Network and Deep Learning

Perspective. SN Comput Sci. 2021;2:154. https://doi.org/10.1007/s42979-021-00535-6.
[30] DeCastro-García N, Pinto E. A Data Quality Assessment Model and Its Application to Cybersecurity Data Sources. In: Herrero Á, Cambra C, Urda D, Sedano J, Quintián H, Corchado E, editors. 13th International Conference on Computational Intelligence in Security for Information Systems (CISIS 2020). Advances in Intelligent Systems and Computing, vol 1267. Springer, Cham; 2021. https://doi.org/10.1007/978-3-030-57805-3_25.
[31] Bhuyan MH, Bhattacharyya DK, Kalita JK. Towards Generating Real-life Datasets for Network Intrusion Detection. Int J Netw Secur. 2015;17:683–701
[32] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
[33] Deshmukh DH, Ghorpade T, Padiya P. Improving classification using preprocessing and machine learning algorithms on NSL-KDD dataset. In: Proceedings of the 2015 International Conference on Communication, Information and Computing Technology (ICCICT), Mumbai, India, 15–17 January 2015.
[34] Nehinbe JO. A critical evaluation of datasets for investigating IDSs and IPSs researches. In Proceedings of the 2011 IEEE 10th International Conference on Cybernetic Intelligent Systems (CIS); 2016 Sep 1-2; London, UK.
[35] Sharafaldin I, Lashkari AH, Ghorbani AA. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In ICISSP; 2018; Fredericton, NB, Canada.
[36] Sayed A, Abdel-Hamid Y, Hefny HA. Artificial intelligence-based traffic flow prediction: a comprehensive review. J Electr Syst Inf Technol. 2023;10(13).
[37] Parisineni SRA, Pal M. Enhancing trust and interpretability of complex machine learning models using local interpretable model agnostic shap explanations. Int J Data Sci Anal. 2023.
[38] Aloqaily M, Kanhere S, Bellavista P, Nogueira M. Special Issue on Cybersecurity Management in the Era of AI. J Netw Syst Manag. 2022;30(39).4.
[39] Kerkdijk R, Tesink S, CISA CISM CISSP, Fransen F, Falconieri F. Evidence-Based Prioritization of Cybersecurity Threats. ISACA J. 2021;65.
[40] Chaudhary S, Gkioulos V, Katsikas S. Developing metrics to assess the effectiveness of cybersecurity awareness program. J Cybersecur. 2022;8(1).6.
[41] Shahzad F, Mannan A, Javed AR, Almadhor AS, Baker T, Al-Jumeily D. Cloud-based multiclass anomaly detection and categorization using ensemble learning. J Cloud Comput. 2022;11(74).7.
[42] Sarker IH. Machine learning: Algorithms, real-world applications and research directions. SN computer science. 2021 May;2(3):160.
[43] Jurafsky D, Martin JH. Logistic Regression. In: Speech and Language Processing. Stanford University.
[44] Kotsiantis SB. Decision trees: a recent overview. Artificial Intelligence Review. 2013 Apr;39:261-83.
[45] Breiman L. Random Forests. Machine Learning. 2001;45:5–32. https://doi.org/10.1023/A:1010933404324.
[46] Ding Y, Zhu H, Chen R, Li R. An efficient AdaBoost algorithm with the multiple thresholds classification. Applied sciences. 2022 Jun 9;12(12):5872.
[47] Ramakrishna MT, Venkatesan VK, Izonin I, Havryliuk M, Bhat CR. Homogeneous Adaboost Ensemble Machine Learning Algorithms with Reduced Entropy on Balanced Data. Entropy. 2023;25:245. https://doi.org/10.3390/e25020245
[48] Natekin A, Knoll A. Gradient boosting machines, a tutorial. Front Neurorobot. 2013 Dec 4;7:21. Available from: https://doi.org/10.3389/fnbot.2013.00021.
[49] He Z, Lin D, Lau T, Wu M. Gradient boosting machine: a survey. arXiv preprint arXiv:1908.06951. 2019 Aug 19.
[50] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). ACM, New York, NY, USA. 2016. pp. 785-794. doi: 10.1145/2939672.2939785.
[51] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y. LightGBM: a highly efficient gradient boosting decision tree. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17. Red Hook, NY, USA: Curran Associates Inc.; 2017. p. 3149–3157.
[52] Gan M, Pan S, Chen Y, Cheng C, Pan H, Zhu X. Application of the Machine Learning LightGBM Model to the Prediction of the Water Levels of the Lower Columbia River. J Mar Sci Eng. 2021;9(5):496. https://doi.org/10.3390/jmse9050496
[53] McCarty DA, Kim HW, Lee HK. Evaluation of light gradient boosted machine learning technique in large scale land use and land cover classification. Environments. 2020 Oct 3;7(10):84.
[54] Jain AK. Data clustering: 50 years beyond K-means. Pattern Recognit Lett. 2010;31(8):651-666.
[55] Ahmed M, Seraj R, Islam SM. The k-means algorithm: A comprehensive survey and performance evaluation. Electronics. 2020 Aug 12;9(8):1295.
[56] Zubair M, Iqbal MA, Shil A, et al. An Improved K-means Clustering Algorithm Towards an Efficient Data-Driven Modeling. Ann Data Sci. 2022. https://doi.org/10.1007/s40745-022-00428-2
[57] Liu FT, Ting KM, Zhou Z-H. Isolation Forest. 2008 Eighth IEEE International Conference on Data Mining. doi:10.1109/ICDM.2008.17.

[58] Ripan RC, Sarker IH, Anwar MM, Furhad MH, Rahat F, Hoque MM, et al. An Isolation Forest Learning Based Outlier Detection Approach for Effectively Classifying Cyber Anomalies. In: Hybrid Intelligent Systems. Advances in Intelligent Systems and Computing. Springer; 2021. p. 270-279. doi: 10.1007/978-3-030-73050-5_271.
[59] Sari FA, Alrammahi AA, Hameed AS, Alrikabi HM, Abdul–Razaq AA, Nasser HK, AL-Rifaie MF. Networks Cyber Security Model by Using Machine Learning Techniques. International Journal of Intelligent Systems and Applications in Engineering. 2022 Dec 31;10(3s):257-63.
[60] Mughal AA. Cyber Attacks on OSI Layers: Understanding the Threat Landscape. Journal of Humanities and Applied Science Research. 2020 Jan 15;3(1):1-8.
[61] Mayukha S, Vadivel R. Various Possible Attacks and Mitigations of the OSI Model Layers Through Pentesting – An Overview. In: Srivastava R, Pundir AKS, editors. New Frontiers in Communication and Intelligent Systems. SCRS, India; 2023. p. 799-809. doi:10.52458/978-81-95502-00-4-78.
[62] Aslan Ö, Aktuğ SS, Ozkan-Okay M, Yilmaz AA, Akin E. A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. Electronics. 2023 Mar 11;12(6):1333.
[63] Himmat M, Ibrahim MA, Hammam N, Eldirdiery HF, Algazoli G. The Current Trends, Techniques, and Challenges of Cybersecurity. European Journal of Information Technologies and Computer Science. 2023 Oct 30;3(4):1-5.
[64] Poslavskaya E, Korolev A. Encoding categorical data: Is there yet anything 'hotter' than one-hot encoding?. arXiv preprint arXiv:2312.16930. 2023 Dec 28.
[65] Pudjihartono N, Fadason T, Kempa-Liehr AW, O'Sullivan JM. A review of feature selection methods for machine learning-based disease risk prediction. Frontiers in Bioinformatics. 2022 Jun 27;2:927312.
[66] Ahsan MM, Mahmud MP, Saha PK, Gupta KD, Siddique Z. Effect of data scaling methods on machine learning algorithms and model performance. Technologies. 2021 Jul 24;9(3):52.
[67] Johnson JM, Khoshgoftaar TM. Survey on deep learning with class imbalance. J Big Data. 2019;6:27. Available from: https://doi.org/10.1186/s40537-019-0192-5
[68] Elgeldawi E, Sayed A, Galal AR, Zaki AM. Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis. Informatics. 2021;8(4):79. Available from: https://doi.org/10.3390/informatics8040079
[69] Erickson BJ, Kitamura F. Magician's Corner: 9. Performance Metrics for Machine Learning Models. Radiol Artif Intell. 2021 May 12;3(3):e200126. doi: 10.1148/ryai.2021200126.

Appendix A. Tables

A.1. Table 1. Unicauca-dataset-April-June-2019-Network-flows Dataset extracted features [7].

Feature Name	Description
flow_key	Unique identifier for each network flow.
src_ip_numeric	Numeric IP of source device.
src_ip	IP address of source device.
src_port	Port number on source device.
dst_ip	IP address of destination device.
dst_port	Port number on destination device.
proto	Transmission protocol.
pktTotalCount	Total packets in flow.
octetTotalCount	Total bytes transmitted in flow.
min_ps	Smallest packet size observed.
max_ps	Largest packet size observed.
avg_ps	Average size of packets.
std_dev_ps	Packet size variability.
flowStart	Flow start time.
flowEnd	Flow end time.
flowDuration	Duration of flow.
min_piat	Shortest packet interval.
max_piat	Longest packet interval.
avg_piat	Average packet interval.
std_dev_piat	Packet interval variability.
f_pktTotalCount	Forward packets count.
f_octetTotalCount	Forward bytes count.
f_min_ps	Smallest forward packet size.
f_max_ps	Largest forward packet size.
f_avg_ps	Average forward packet size.
f_std_dev_ps	Forward packet size variability.
f_flowStart	Forward flow start time.
f_flowEnd	Forward flow end time.
f_flowDuration	Forward flow duration.
f_min_piat	Minimum forward packet interval.
f_max_piat	Maximum forward packet interval.
f_avg_piat	Average forward packet interval.
f_std_dev_piat	Forward packet interval variability.
b_pktTotalCount	Backward packets count.
b_octetTotalCount	Backward bytes count.
b_min_ps	Smallest backward packet size.
b_max_ps	Largest backward packet size.
b_avg_ps	Average backward packet size.
b_std_dev_ps	Backward packet size variability.
b_flowStart	Backward flow start time.
b_flowEnd	Backward flow end time.
b_flowDuration	Backward flow duration.
b_min_piat	Minimum backward packet interval.
b_max_piat	Maximum backward packet interval.
b_avg_piat	Average backward packet interval.
b_std_dev_piat	Backward packet interval variability.
flowEndReason	Reason for flow ending.
category	Flow classification.
application_protocol	Application protocol identified.
web_service	Detected web service or application.

A.2. Table 2.: OSI Layer Feature Distribution [60] [61] [62].

OSI Layer	Predominant Features	Attack Type
Application (Layer 7) Dealing with software applications	flow_key, application_protocol, web_service	**DDoS**: Overloads services via application_protocol, causing congestion or shutdown. **Injection**: Inserts malicious code through flow_key or application_protocol, causing breaches. **XSS**: Uses web_service to inject scripts, stealing data or redirecting users. **Malware**: Disrupts or damages systems through web_service.
Presentation (Layer 6) This layer is responsible for the translation, encryption, and compression of data	Not specifically represented by the provided features. This layer suggests the dataset is focused on network traffic and behaviors rather than the processing or presentation of data for applications.	

Session (Layer 5) Establishes, manages, and terminates connections (sessions) between applications	flowStart, flowEnd, flowDuration, f_flowStart, f_flowEnd, f_flowDuration, b_flowStart, b_flowEnd, b_flowDuration	**Session Hijacking:** Takes over a user's session using start/end times, accessing private info without permission. **Session fixation:** Tricks a user into using a specific session ID.
Transport (Layer 4) Responsible for end-to-end communication and data flow control over a network	src_port, dst_port, min_ps, max_ps, avg_ps, std_dev_ps, f_min_ps, f_max_ps, f_avg_ps, f_std_dev_ps, b_min_ps, b_max_ps, b_avg_ps, b_std_dev_ps	**Port Scanning:** Finds open src_port/dst_port to break into systems. **SYN Flood:** Sends too many requests to open connections, making services slow or unavailable.
Network (Layer 3) Handles the routing and forwarding of data packets across different networks	src_ip_numeric, src_ip, dst_ip, proto, pktTotalCount, octetTotalCount, f_pktTotalCount, f_octetTotalCount, b_pktTotalCount, b_octetTotalCount	**IP Spoofing**: Alters src_ip to impersonate others, misleading systems. **Amplification**: Uses pktTotalCount and octetTotalCount to overload networks, causing disruptions. **Man-in-the-Middle**: Intercepts and alters data between src_ip and dst_ip. **Routing Attacks**: Manipulates proto to alter data paths.
Data Link (Layer 2) Controls data flow and transfer across the network, managing addressing, error detection, and frame sync.	Layer 2 features are missing, indicating the dataset is gathered from a higher network level (network devices or software), rather than MAC addresses or frame sequencing.	
Physical (Layer 1) The OSI model's lowest layer deals with sending and receiving raw bits over a physical medium.	Layer 1 features are missing. This layer suggests that the dataset does not include hardware-specific information, such as electrical signals, data rates, or physical connectors, which are typically beyond the scope of network flow analysis focused on higher-level traffic patterns and security analysis.	

Appendix B. Code Base

We have made our datasets and experiment codes publicly available on our GitHub repository at:

GitHub Repository Link:

https://github.com/samer-glitch/Navigating-Network-Complexity-Innovative-Strategies-for-Traffic-Analysis-and-Security-Optimization

GitHub Repository Notebook structure:

	File Name	Description	Related Section
B.1	README.md	Repository documentation and overview.	-
B.2	Data Cleaning and Exploratory Data Analysis EDA.ipynb	Initial data cleaning and exploratory analysis.	4- A, B
B.3.1	Network_Traffic_Analysis_and_Classification_with_6_ML_models.ipynb	Analysis and classification of network traffic using six different machine learning models.	4- B
B.3.2	Network_Traffic_Analysis_and_Classification_with_6_ML_models_(Reduced_Features).ipynb	Analysis and classification of network traffic using six different machine learning models with reduced features.	4- B
B.4	General Anomaly Detection.ipynb	Notebook for detecting anomalies in network traffic.	4- C
B.5	Volume Analysis.ipynb	Analysis of network traffic volumes.	4- C