Hybrid Machine Learning Approach for Attack Classification and Clustering in Network Security

Castro A. Yoga
Department of Computer Science & Software Engineering
Jaramogi Oginga Odinga University of Science and Technology

Anthony J. Rodrigues
Department of Computer Science & Software Engineering
Jaramogi Oginga Odinga University of Science and Technology

Silvance O. Abeka
Department Information System Technology
Jaramogi Oginga Odinga University of Science and Technology

ABSTRACT
Due to the increasing complexity and diversity of threats, network security has become a critical concern. The application of machine learning (ML) methods has demonstrated potential in enhancing network security by effectively recognizing and classifying threats. A hybrid ML-based approach is presented in this study within the framework of the three-layer network security domain (TLNSD) to address the task of attack classification and clustering. The approach utilizes a stacking ensemble classifier, which employs a meta learner (Logistic Regression) to combine the predictions from multiple base learners (K-Nearest Neighbors, Random Forest, Gaussian Naive Bayes). To identify the most relevant features, the SelectKBest algorithm is employed. Additionally, the K-means clustering technique is utilized to group similar attack instances. The performance evaluation of the proposed technique is conducted using the UNSW-NB15 dataset. The results demonstrate that the proposed technique surpasses the performance of individual base learners, achieving a high level of accuracy. This underscores its effectiveness in detecting and categorizing attacks. The clustering analysis provides insights into the distribution and occurrence frequency of diverse threat types, enabling the development of tailored security strategies. By presenting a comprehensive and integrated approach to threat analysis and mitigation, this study contributes to the advancement of network security. The proposed methodology offers a unified framework to effectively address the challenges posed by evolving cyber threats.

Keywords
Network security, machine learning, ensemble learning, stacking ensemble classifier, feature selection, K-means clustering, attack classification, attack clustering, three-layer network security domain, UNSW-NB15 attacks dataset

1. INTRODUCTION
Networks are composed of users, software, and hardware elements that work together to meet the network objectives. Due to their diversity and intricate nature, they are susceptible to a wider array of attacks [1]. Attackers can compromise network security by exploiting any vulnerable element, be it a user, software application, or physical infrastructure. The extensive attack surfaces of networks can also facilitate multistage attacks, where various parts of a network are targeted simultaneously[2].

The challenge faced in network security is the lack of knowledge about forthcoming attacks in attack analysis. It is evident that many network administrators have not stayed updated with the latest advancements in attack knowledge[3]. As a result, their security implementations are often less effective and sometimes rendered ineffective.

"Machine learning" as the field of study enables computers to learn autonomously without explicit programming. This field focuses on using existing data to teach computers how to perform specific tasks [4]. Ensemble learning, a popular machine learning approach, involves using multiple learners (classification or regression models) to solve the same problem. Instead of constructing a single model from training data, ensemble methods create a set of models and combine their outputs [5]. Research has shown that combining classifiers in ensemble methods often leads to more accurate predictions than using a single classifier. Boosting, a specific ensemble method, demonstrates that weak learners can be enhanced to become strong learners[6].

The three commonly used ensemble techniques are bootstrap aggregating (bagging), boosting, and stacking[7]. Bagging trains each model on random subsets of the training set, while boosting incrementally builds an ensemble model by focusing on misclassified instances from previous models. Stacking, also known as stacked generalization, combines the predictions of multiple models using a specific algorithm[8]. This study specifically chooses stacking because it is a generalized form of other ensemble methods.

This study proposes an approach of classifying and clustering attacks according to the network layers based on the three-layer network security domain (TLNSD) [9] which involves the user layer, host layer and media layer. This approach makes the following contribution takes into account attacks targeting the three layers of the network TLNSD providing a holistic unified strategy in attack analysis. Proposes a hybrid ML-based classifier framework consisting of a combination of Kneighbors, random forest, gaussian NB and Logistic regression classification model to detect attacks and KMeans for clustering the attacks according to the layers targeted.

1.1. Three-layer network security domain (TLNSD).
The TLNSD adheres to a compartmentalized and holistic approach in identifying an attack and which layer is targeted as shown in figure 1 Once the network is hit by an attack, it should be categorized according to which surface it is targeting, which is either the organization, host, or media layer surfaces, which are modularized layers representing their respective layers of the OSI model. To achieve this, attack context checking is performed on the three-layer network security domain (TLNSD) by analyzing the network attack traffic and identifying the targets in relation to the three layers.

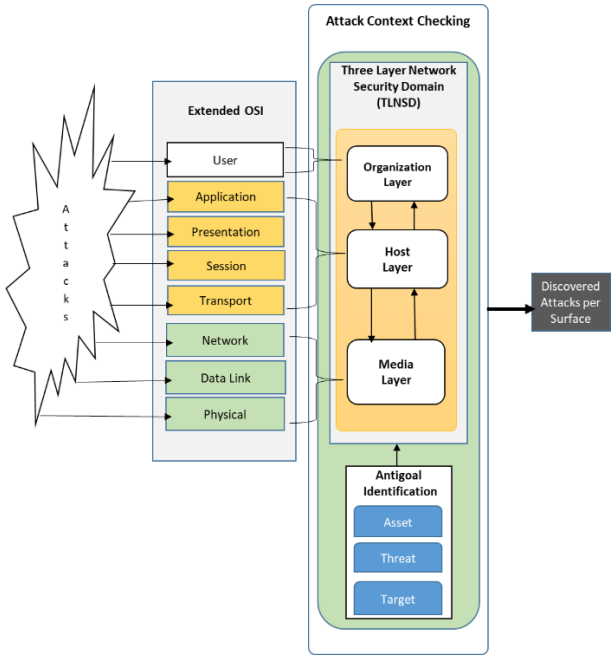

Figure 1. Three-layer network security domain (TLNSD). [9]

2. RELATED WORK

2.1 Machine Learning Network Traffic Analysis.

Due to the expanding digital world, network security is a critical concern for consumers, businesses, and governmental agencies. The network infrastructure must be protected since it is essential to many services. Network security managers must constantly resist these attacks since hackers exploit weaknesses despite the adoption of several security solutions [10].In this regard, sophisticated methods like machine learning have proven successful in a variety of sectors, including healthcare, banking, education, and energy. As a result, network security experts are investigating how these methods could improve the general state of network security [11].

Using the Random Forest (RF), [12] assessed how well different forms of attacks, such as shellcode, Intrusions and malwares performed at being recognized. They standardized the data and took into account 17 features in the experiment. F-Score performance was 99% overall. However, it was noted that the effectiveness of identifying various attacks varied. They suggest utilizing the same dataset and altering the training circumstances to further assess the effectiveness of attack type detection.

Rendall et al. [13] developed a two-layered detection system for phishing online attacks employing attributes extracted from domain and DNS packet-level data using four ML models: MLP, SVM, NB, and DT. The team looked into the method of classifying phishing domains more than once, with further classifications only occurring when a domain's score fell below a certain confidence threshold established by the system's owner. The team's own dataset, which included 5995 phishing records and 7053 benign records, was used to assess the model. MLP and DT were able to reach the greatest accuracy of 86% after using the models in the two-layered architecture.

To identify phishing web pages, [14] created a stacking model employing URL and HTML information. To enable real-time phishing detection, they made advantage of lightweight HTML and URL features, HTML string embeddings, and other techniques. they created and used the 50K-PD dataset, which comprised about 49,947 samples, and the 50K-IPD dataset, which contained 53,103 web page samples. By layering GBDT, XGBoost, and LightGBM, the stacking model was created. On the 50K-PD dataset, the model had an accuracy of 97.30%, while on the 50K-IPD dataset, it had an accuracy of 98.60%.

2.2. Malicious Traffic Classification and clustering

Network traffic must first be evaluated and categorized to identify anomalous and malicious attacks in order to safeguard from cyber-attacks. Given how crucial the categorization of harmful communications is, several academics have worked to enhance classification methods by utilizing Artificial Intelligence with Studies concentrating on both anomalous and unusual traffic.

[15] proposed a framework for hardware-assisted malware detection utilizing machine learning and memory access pattern categorization. In order to generate a virtual address trace, they suggested in-processor monitoring. To do this, they divided accesses into epochs, summed up the memory access patterns of each epoch into features, and then supplied the features to ML classifiers Random Forest (RF) and logistic regression (LR). It was determined that RF performed the best classifier for both memory corruption attacks and kernel rootkits. Its success rate for finding kernel rootkits was 100% TPR with less than 1% FPR. User-level memory corruption attacks were mitigated by the method with a 99.0% DR and less than 5% FPR.

It was noted that issues with feature selection crop up while developing ML models for the identification of legitimate or fraudulent communications. In light of this, [16] introduced a hybrid feature selection approach for machine learning termed weighted mutual information area under the curve (WMI_AUC), which aids in choosing the most useful features in traffic flow. The HIT Trace 1 database, which the authors obtained from WeChat messenger using Wireshark, and the NIMS dataset, which the authors gathered from their research-tested network, were the databases utilized in the study. The researchers utilized 11 distinct ML algorithms to create the final model. Using the HIT Trace 1 dataset, the model created using the partial decision tree (PART) technique has an accuracy of 97.88%. The accuracy of RF for the NIMS dataset was 100%.

In order to better detect threats to mobile devices, [17] concentrated on merging three algorithms—RF, JRIP, and PART. The researchers utilized Wireshark to gather 600 samples from the virtual computer and included them in the dataset. The researchers employed bidirectional flow export utilizing the IP flow information export technique for feature extraction. Overfitting issues and concept drift conditions, which are brought on by selecting poor performance providing features, were a difficulty to the researchers. The ensemble model has a 98.2% accuracy rate and could distinguish between good and bad traffic. The researchers want to merge ML with traditional NIDS in their upcoming work and lessen concept drift by using cutting-edge techniques.

Similar to this, to increase DR and lower FPR and FNR alarms, [18] developed a hybrid ML approach for classifying network traffic as normal or invasive by combining K-means clustering with SVM classification. The NSL-KDD dataset was used to test the suggested approach. The dataset underwent pre-processing in order to remove ambiguity and give the detection engine correct data. Both the classifiers—K-means and SVM—were then put to the test and had their performance assessed after using the classifier subset evaluator and best-first search methods. The findings of the hybrid ML approach indicated that they achieved a DR and FNR of 96.26% and 3.7%, respectively. The model demonstrated much improved DoS, PROBE, and R2L attack detection.

3. PROPOSED APPROACH

3.1 Stacking Ensemble Classifier

Figure 2. the shows the stacking ensemble approach to classifying of the malicious traffic consisting of the select Kbest

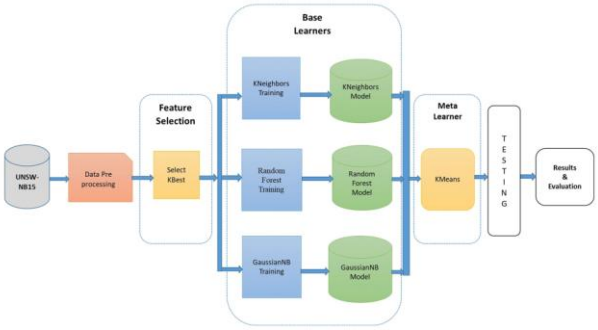

Figure 2. proposed stacking ensemble

3.1.1. Feature selection

To use machine learning algorithms efficiently, it is becoming increasingly important to perform feature selection [19]. Features election, also referred to as attribute selection or variable selection, is a process of selecting more relevant attributes, and removing irrelevant or less relevant attributes or noisy data or features that do not add additional value to a machine learning algorithm.[20]Using only relevant features for machine learning algorithms allows for faster processing and more accurate predictability. [21]A lot of work has been done on feature selection using traditional techniques like Information Gain, the Gini Index, uncertainty, and correlation coefficients. In this paper, this study used the SelectKBest algorithm for feature selection. The SelectKBest method chooses the top *k* best features from the input dataset based on a scoring measure. The feature selection equation is represented as:

$$fs(x) = x\_selected \quad (1)$$

where *fs is the feature selection, x* is the input feature matrix and *X_selected* comprises the *k* best features that were chosen. The parameter *k* for SelectKBest defines the number of characteristics to be chosen.

3.1.2. Base Learners

The KNearest Neighbors (KNN) method classifies or predicts the target variable using the feature space's *k* nearest neighbors. The KNearest Neighbors training equation is represented as follows

$$b1(x\_train, y\_train) = model\_b1 \quad (2)$$

where *b1* is the first base learner KNN, *X_train* is the training feature matrix and *y_train* is the associated labels. The argument for KNearest Neighbors is *k*, which specifies the number of neighbors to take into account.

The Random Forest (RF) method constructs a decision tree ensemble and generates predictions based on the majority vote or average of the individual trees. The Random Forest training equation can be represented as

$$b2(x\_train, y\_train) = model\_b2 \quad (3)$$

where *b1* is the second base learner RF, *X_train* is the training feature matrix and *y_train* is the associated labels. Random Forest settings include the number of trees, the maximum depth, and other tree-specific characteristics.

To classify or forecast the target variable, the Gaussian Naive Bayes (GNB) method assumes that features are independent and uses a Gaussian distribution. The Gaussian NB training equation is represented as:

$$b3(x\_train, y\_train) = model\_b3 \quad (4)$$

where *b3* is the third base learner GNB, *X_train* is the training feature matrix and *y_train* is the associated labels. There are no settings to adjust for Gaussian NB.

3.1.3. Meta Learner

Logistic Regression (*LR*) employs the logistic function to represent the association between the chosen characteristics and the intended variable. A representation of the Logistic Regression training equation is:

$$l(x\_meta, y\_meta) = model\_l \quad (5)$$

where *l* is the *LR*, where *X_meta* is the meta-features (predictions from *b1, b2*, and *b3*) and *y_meta* is the associated labels. The regularization term, solver technique, and convergence criteria are some of the variables in logistic regression.

Stacking Ensemble's prediction *(e)* combines the predictions of each model *(B1, B2, B3, and L)* after they have all been trained to provide the final prediction. The test feature matrix *X_test* is used in the equation for the stacking ensemble prediction, which is written as

$$e(x\_test) = l\left(b1\left(fs(x\_test)\right), b2\left(fs(x\_test)\right), b2\left(fs(x\_test)\right)\right) \quad (6)$$

where *X_test* is the test feature matrix. To apply the equations and parameters for training and prediction, it would typically follow the following steps:

BEGIN:

1. Apply feature selection *(fs)* on the training dataset *(x_train)* to select the top k features.
2. Train the base learners *(b1, b2, b3)* on the selected features from step 1 using their respective algorithms and parameters.
3. Generate predictions from each base learner on the meta-features *(x_meta)* of the training dataset.
4. Train the meta learner *(m)* using the meta-features and the corresponding labels *(y_meta)*.
5. For prediction, apply feature selection *(fs)* on the test dataset *(x_test)* to select the same k features as in step 1.
6. Generate predictions from each base learner on the selected features from step 5.
7. Combine the base learners' predictions as meta-features and input them into the trained meta learner *(m)* to obtain the final prediction.

END

3.2 Clustering Algorithm based on KMeans

The primary goal of the K-means algorithm is to minimize the total sum of squared distances between the data points and the centroids of their respective assigned clusters. The representation is as follows.

$$J(C, \mu) = \sum ||xi - \mu cj||^2 \quad (7)$$

The objective function, denoted as *J (C, μ)*, is defined as the sum of the squared Euclidean distances between each data point xi and its corresponding cluster center μcj, summed over all data points. The objective function that requires minimization is denoted as *J (C, μ)*. The set *C*, denoted as *C = {c1, c2, …, cn}*, represents the cluster assignments for each individual data point in the dataset X. On the other hand, the set *μ*, denoted as *μ = {μ1, μ2, μ3}*, represents the cluster centroids. The objective function computes the squared Euclidean distance between every data point *xi* and its corresponding cluster centroid *μcj*. The objective is to minimize the distance between each data point and its corresponding centroid, thereby indicating a closer proximity.

The K-means algorithm is subject to two primary constraints: (1) It is imperative that every individual data point is allocated to a single cluster. (2) The centroid of each cluster is calculated as the average of the data points assigned to that cluster, denoted by the equations *x* and *y*, respectively. The summation of *cj* equals *k* is equal to *1*, for all *j* ranging from *1* to *N*, and *k* ranging from *1* to *3*.

$$\sum (cj = k) = 1, for\ all\ j = 1, 2, \ldots, N, and\ k = 1, 2, 3. \quad (8)$$

Constraint 1; guarantees that each data point is exclusively assigned to a single cluster. The expression *(cj = k)* yields a value of *1* when the data point *xj* is assigned to cluster *k*, and *0* otherwise. The total of this term across all clusters should be equivalent to *1*, signifying that each data point possesses a distinct cluster assignment. The formula for calculating the mean of a set of values *xi*, where each value is associated with a category *cj*, is given by

$$\mu k = (1/|\{j: cj = k\}|) \sum (xi, cj = k), for\ all\ k = 1, 2, 3. \quad (9)$$

Constraint 2; involves the updating of cluster centroids in accordance with the assigned data points. The centroid μk for each cluster k is determined by calculating the average of the data points xi that are assigned to that specific cluster. The expression (*xi, cj = k*) denotes the summation over the data points xi, subject to the condition that their cluster assignment *cj* is equal to *k*. Additionally, *| {j: cj = k} |* represents the count of data points that have been assigned to cluster *k*.

The K-means algorithm iteratively optimizes the cluster assignments and cluster centroids by minimizing the objective function *J (C, μ)* while ensuring that the constraints are satisfied, until convergence is achieved. The algorithm employs a two-step process, wherein it iteratively updates the assignments by considering the closest centroid and recalculates the centroids based on the assigned data points. The objective is to identify the configuration that minimizes the total sum of squared distances. Below is Implementation algorithm for the K-means clustering algorithm based on *k=3.* To implement the clustering, it would typically follow the following give steps

Input:
- Dataset X = {x1, x2, ..., xn} with N data points in d-dimensional space.
- Number of clusters K.

Output:
- Cluster assignments C = {c1, c2, ..., cn} indicating the cluster assignment for each data point.
- Cluster centroids μ = {μ1, μ2, ..., μK} representing the centroids of the clusters.

BEGIN:

1. Initialize the cluster centroids μ = {μ1, μ2, ..., μK}:
   - Randomly select or initialize K centroids in the feature space.
2. Repeat until convergence:
   - Assign data points to the nearest centroid:
     - For each data point xi, calculate the Euclidean distance to each centroid μk.
     - Assign xi to the cluster corresponding to the nearest centroid.
   - Update the cluster centroids:
     - For each cluster k, calculate the mean of the data points assigned to that cluster.
     - Update the centroid μk by setting its value to the computed mean.
   - Check for convergence:
     - If the cluster assignments remain unchanged from the previous iteration, terminate the algorithm.
3. Return the final cluster assignments C and cluster centroids μ.

END

3.3 Dataset

For the experimental processes the study utilized the UNSW-NB15 attacks dataset [22]. In its clean format, the UNSW-NB15 contains 49 features as shown in Table 1. Out of the 49 features, 2 instances are non-numeric (categorical) features and 47 are numeric in nature. The UNSW-NB15 is subdivided in the following main datasets: UNSW-NB15- TRAIN, which is employed for training various models and the UNSW-NB15-TEST (100%) which is employed for testing the trained models. A further split was done to the UNSW-NB15-TRAIN in the following two partitions: the UNSW-NB15-TRAIN-1 (70% of the full training set) for training and the UNSW-NB15-VAL (30% of the full training set) for validation before testing. The UNSW-NB15 contains instances with the following categories of network attacks: Backdoor, Shellcode, Reconnaissance, Worms, Fuzzers, DoS, Generic, Analysis, Shellcode and Exploits.

3.4 Methodology

First The SelectKBest algorithm is applied to the new data to select the same K features as in the training phase. The selected features, along with the new data, are fed into the trained base learners (KNN, Random Forest, GNB). The base learners generate their predictions based on the selected features. The meta-learner (Logistic Regression) takes the predictions of the base learners as input and produces the final prediction. The stacking ensemble's final prediction is obtained by combining the individual predictions of the base learners using the learned weights from the meta-learner.

Secondly the predictions from the meta-learner becomes the set of input samples. Scaling of the numerical features is done to ensure they have the same range and prevent any bias in the clustering process. Identification of the clusters is done in this case three clusters. K-means clustering algorithm is applied on the normalized and feature-represented data to partition it into the specified number of clusters the process is Iterated until convergence by assigning data points to the nearest cluster centroid and updating the centroids is achieved. Interpretation of the clusters is done by analyzing the characteristics and patterns within each cluster to understand the similarities in the meta-learner's predictions and also to Gain insights about the different groups in the data based on the cluster finally the trained K-means model is used to predict the cluster membership of new data points by applying the same feature representation and normalization techniques.

4. EXPERIMENTS RESULTS AND DISCUSSION

The experiment presented in this work were conducted in Kaggle, The ML models are built, trained, evaluated and tested on the Sklearn ML Python framework [23]. Sklearn is constructed on top of matplotlib, NumPy and Scipy Python libraries. Moreover, Classification, Regression and Clustering tasks can all be conducted using SkLearn

4.1 Performance metric

There exist a number of metrics to evaluate ML based IDS systems; however, this research aims to maximize the correct predictions of instances in the test dataset. The main measure to look at is the *Accuracy (AC)* defined as follows:

$$AC = \frac{TN + TP}{FP + FN + TP + TN} \quad (10)$$

whereby the *TP* stand for True Positive and is the rate of examples correctly identified as attacks. *TN*, True Negative, is the rate of legitimate traffic classified as legitimate. *FP*, False Positive, sometimes referred to as Type I error, is the rate of legitimate traffic classified as attacks. *FN*, sometimes referred to

as Type II error, is the rate of legitimate traffic classified as intrusions. Additional metrics considered in this paper are the Recall (R), the Precision (P)and *F1*score defined as follows

$$P = \frac{TP}{TP + FP} \quad (11)$$

$$R = \frac{TP}{TP + FN} \quad (12)$$

$$F1_{score} = 2\frac{P.R}{P + R} \quad (13)$$

training of the classifiers to make predictions on the testing set, and calculate the evaluation metrics, table 1 show output of the accuracy, precision, recall, and F1 score of the SelectKBest feature selection algorithm. the performance metrics for each base learner (K-Nearest Neighbors, Random Forest, and Gaussian Naive Bayes) and the meta learner (Logistic Regression) using the predictions from the base learners in the stacking ensemble.

Table 1. Performance metrics scores

Classification Technique	Accuracy	Precision	Recall	F1 Score
Select Kbest	0.93466	0.94393	0.93466	0.95387
KNN	0.85767	0.86776	0.85765	0.85734
Random forest	0.92335	0.91254	0.96667	0.95667
Gaussian Naive Bayes	0.88899	0.82785	0.80956	0.79676
*Proposed ensemble	0.96667	0.96967	0.96667	0.96998

The stacking ensemble method outperformed other classification techniques in the evaluation. This method combines multiple models' strengths, excelling in various performance metrics. The "Select Kbest" technique achieved high accuracy (93.47%) and balanced precision (94.39%) and recall (93.47%), emphasizing its ability to discriminate between classes effectively and strike a balance between false positives and true positives. KNN had decent accuracy (85.77%), precision (86.77%), and recall (85.77%), but its F1 score (85.73%) indicated room for improving precision-recall balance. The Random Forest model stood out with remarkable accuracy (92.34%), solid precision (91.25%), and high recall (96.67%), resulting in an impressive F1 score (95.67%) and proficiency in correctly identifying positive cases. Gaussian Naive Bayes had moderate accuracy (88.90%) but struggled with precision (82.79%) and recall (80.96%), leading to a lower F1 score (79.68%), highlighting challenges in minimizing false positives and capturing true positives. The stacking ensemble achieved the highest accuracy (96.67%), exceptional precision (96.97%), and recall (96.67%), along with a remarkable F1 score (96.99%), demonstrating its superior predictive performance by optimizing precision and recall. In conclusion, the stacking ensemble's comprehensive approach with exceptional accuracy, precision, recall, and F1 score presented a compelling solution for the classification task.

4.2 Results

Performing feature selection using the SelectKBest algorithm with chi-squared test as the scoring function 10 appropriate features were selected as shown in table 2 which were the basis of clasification and clustrering of the attacks in the UNSW-NB15 dataset

Table 2. UNSW-NB15 selected features using select Kbest

	ct_flw_http_mthd	is_ftp_login	ct_ftp_cmd	ct_srv_src	ct_srv_dst	ct_dst_ltm	ct_src_ltm	ct_src_dport_ltm	ct_dst_sport_ltm	ct_dst_src_ltm
19	0	0	0	1	1	1	1	1	1	1
20	1	0	0	3	2	2	1	1	1	1
21	1	0	0	5	2	2	1	1	1	1
38	0	0	0	1	1	1	1	1	1	1
39	1	0	0	3	1	1	1	1	1	1

The 10 features were subjcted to the stacking ensemble and the generated classification output was subjected to the kmeans clustering where the data was split into three clusters k=3 according to the three surfaces being atatcked that is user, host and media. To ensure that evertime the same output was achieved when running the clustering code the random state was set to 0. Table 3 shows the sample output of the clustering process.

Table.3 sample clustering process output for the attack surfaces

	0	1	2	3	4	5	6	7	8	KMeans_cluster	results	ys
12917	0	0	6	6	2	2	2	2	2	1	1	Users
12778	0	0	1	4	1	0	0	0	0	0	0	Hosts
9059	0	0	6	6	1	5	1	1	1	1	1	Users
19977	0	0	2	2	2	2	2	2	2	0	0	Hosts
13128	0	0	7	2	0	0	0	0	0	0	0	Hosts
18121	0	0	0	5	1	0	0	0	0	0	0	Hosts
21411	0	0	6	4	0	0	0	0	0	0	0	Hosts
10297	0	0	17	17	10	10	10	2	10	2	2	Media
18457	0	0	11	11	2	2	2	2	2	1	1	Users
3297	0	0	0	0	0	0	0	0	0	0	0	Hos

To visualize the attacks on the three surface layers as a result of clustering, a count of attack distribution per surface linked to the specific nine attacks that is exploits, generic, Fuzzers, reconnaissance, dos, backdoor, analysis shellcode and worms within the UNSW-NB15 dataset was generate as shown in figure 3.

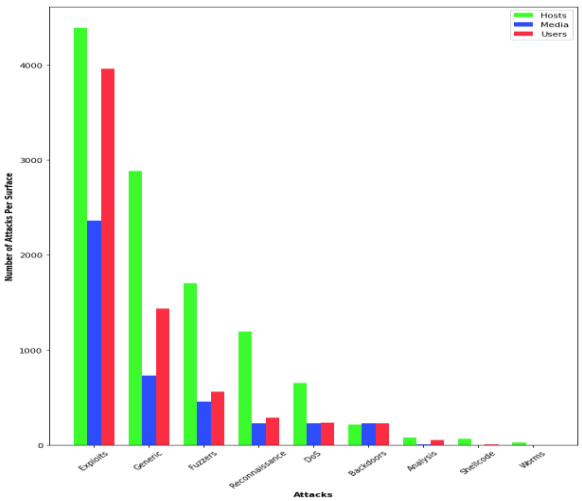

Figure 3. Attack Distribution for Each Surface of Attack.

4.3 Clustering analysis

Table 4. shows the analysis of the distribution of attacks on each surface of the TLNSD from the clustering process.

Table 4. Attack Distribution for Each Surface of Attack.

Attacks	Host Attacks	Media Attacks	User Attacks	Cumulative attack
Exploits	4393(40.98 %)	2364(22.05 %)	3963(36.97 %)	10720(100 %)
Generic	2884(57.1%)	730(14.45%)	1437(28.45 %)	5051(100%)

Fuzzers	1705(62.66 %)	454(16.69%)	562(20.65%)	2721(100%)
Reconnaissance	1195(69.96 %)	227(13.29%)	286(16.74%)	1708(100%)
DoS	654(58.65%)	226(20.27%)	235(21.08%)	1115(100%)
Backdoors	215(32.14%)	226(33.78%)	228(34.08%)	669(100%)
Analysis	80(59.7%)	3(2.24%)	51(38.06%)	134(100%)
Shellcode	65(89.04%)	1(1.37%)	7(9.59%)	73(100%)
Worms	23(95.83%)	0(0%)	1(4.17%)	24(100%)

4.3.1 Distribution of Attack Types

The analysis demonstrates differences in the distribution of attack types across categories. Exploits are the most common type of host attacks, accounting for 40.98% of all attacks. Generic attacks account for 57.1% of total attacks, with a somewhat equal distribution among host, media, and user attacks. Fuzzers typically attack hosts, whereas reconnaissance assaults cover all attack vectors. DoS assaults are mostly directed at hosts, whereas backdoors are distributed equally across hosts, media, and users. Analysis attacks are mostly directed at hosts, whereas shellcode attacks are primarily directed at hosts. Worms, on the other hand, predominantly attack hosts, with just a tiny proportion impacting users.

4.3.2 Prevalence and Impact

The analysis also reveals the incidence and effect of different attack types within the dataset. Exploits, being one of the most common forms of attacks, represent a substantial risk to the security of hosts, media, and users. Generic attacks, while not as common, have a significant impact due to their spread across numerous attack vectors. While Fuzzers and reconnaissance attacks are less common, they nevertheless offer a significant danger due to their potential for information collection and vulnerability detection. DoS attacks, while not as common, can cause significant disruption to targeted hosts. While backdoors, analysis assaults, shellcode, and worms are less common overall, they nevertheless pose distinct threats depending on the attack vector.

4.3.3 Implications for Security Measures

Understanding the incidence and distribution of various attack types is critical for building effective security solutions. The findings point to the necessity for powerful host security systems to reduce the frequency of exploits and shellcode assaults. A comprehensive security plan should also concentrate on combating generic threats across all attack channels. To prevent such vulnerabilities from being exploited, network defenses against Fuzzers and reconnaissance attacks must be strengthened. To mitigate the impact of DoS attacks, proactive techniques such as traffic filtering and load balancing are required. To avoid unwanted access and data breaches, it is also critical to identify and eliminate backdoors, analysis assaults, and worms.

5. CONCLUSION

In this study, a hybrid machine learning-based technique was proposed for the classification and clustering of network security attacks. The approach employed a stacking ensemble classifier that amalgamated findings from several base learners, including K-Nearest Neighbors, Random Forest, and Gaussian Naive Bayes, with the assistance of a meta learner, namely Logistic Regression. To identify the most relevant characteristics, the SelectKBest algorithm was utilized in conjunction with the K-means clustering technique for grouping similar attack instances.

The experimental evaluation, conducted using the UNSW-NB15 dataset, substantiated the superiority of the proposed technique over individual base learners. Exceptional accuracy was achieved, underscoring the proficiency of the stacking ensemble classifier in detecting and categorizing network security breaches. The feature selection process, guided by the selection of the most informative attributes, enhanced prediction precision. Furthermore, the clustering analysis shed light on the prevalence and distribution of diverse threat types, offering valuable insights for tailoring security measures.

This study contributes significantly to the realm of network security by presenting a comprehensive and unified approach to threat analysis and mitigation. By leveraging machine learning methodologies and considering the multifaceted nature of the network security domain, the method furnishes a profound understanding of threats and their impact on various network components. Consequently, network administrators and security experts are empowered to fortify their defenses against potential attacks and implement more efficacious security protocols.

Nonetheless, there exist avenues for improvement. Subsequent research endeavors could focus on exploring alternative Machine learning algorithms, refining feature selection techniques, and incorporating more advanced clustering methodologies. Expanding the scope of evaluation to encompass larger and more diverse datasets would facilitate a deeper comprehension of the method's generality and scalability. In summary, the hybrid machine learning-based technique for classifying and clustering attacks in network security presented herein yields promising outcomes. Leveraging the potency of ensemble learning and accounting for network layers holds the potential to enhance threat detection and comprehension, ultimately leading to more robust network security measures.

6. REFERENCES

[1] M. Furdek *et al.*, “An overview of security challenges in communication networks,” in *2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM)*, IEEE, 2016, pp. 43–50.

[2] Q. Li *et al.*, “A comprehensive survey on DDoS defense systems: New trends and challenges,” *Computer Networks*, p. 109895, 2023.

[3] K.-K. R. Choo, “The cyber threat landscape: Challenges and future research directions,” *Computers & security*, vol. 30, no. 8, pp. 719–731, 2011.

[4] A. Moubayed, M. Injadat, A. B. Nassif, H. Lutfiyya, and A. Shami, “E-learning: Challenges and research opportunities using machine learning & data analytics,” *IEEE Access*, vol. 6, pp. 39117–39138, 2018.

[5] M. A. Ganaie, M. Hu, A. K. Malik, M. Tanveer, and P. N. Suganthan, “Ensemble deep learning: A review,” *Engineering Applications of Artificial Intelligence*, vol. 115, p. 105151, 2022.

[6] C. B. C. Latha and S. C. Jeeva, “Improving the accuracy of prediction of heart disease risk based on ensemble classification techniques,” *Informatics in Medicine Unlocked*, vol. 16, p. 100203, 2019.

[7] R. Odegua, “An empirical study of ensemble techniques (bagging, boosting and stacking),” in *Proc. Conf.: Deep Learn. IndabaXAt*, 2019.

[8] P. Casas, M. Seufert, N. Wehner, A. Schwind, and F. Wamser, “Enhancing machine learning based qoe prediction by ensemble models,” in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2018, pp. 1642–1647.

[9] C. Yoga, A. Rodrigues, and S. Abeka, "Holistic Security Pattern-Based Model to Protect Network Architecture," *International Journal of Research Publications*, vol. 130, pp. 268–278, Aug. 2023, doi: 10.47119/IJRP1001301820235295.

[10] Y. Li and Q. Liu, "A comprehensive review study of cyber-attacks and cyber security; Emerging trends and recent developments," *Energy Reports*, vol. 7, pp. 8176–8186, 2021.

[11] Z. Zhang, H. Al Hamadi, E. Damiani, C. Y. Yeun, and F. Taher, "Explainable artificial intelligence applications in cyber security: State-of-the-art in research," *IEEE Access*, 2022.

[12] K. Park, Y. Song, and Y.-G. Cheong, "Classification of attack types for intrusion detection systems using a machine learning algorithm," presented at the 2018 IEEE fourth international conference on big data computing service and applications (BigDataService), IEEE, 2018, pp. 282–286.

[13] K. Rendall, A. Nisioti, and A. Mylonas, "Towards a multi-layered phishing detection," *Sensors*, vol. 20, no. 16, p. 4540, 2020.

[14] X. Li, K. Li, D. Qiao, Y. Ding, and D. Wei, "Application research of machine learning method based on distributed cluster in information retrieval," in *2019 International Conference on Communications, Information System and Computer Engineering (CISCE*, IEEE, Jul. 2019, pp. 411–414.

[15] Z. Xu, S. Ray, P. Subramanyan, and S. Malik, "Malware detection using machine learning based analysis of virtual memory access patterns," presented at the Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, IEEE, 2017, pp. 169–174.

[16] M. Shafiq, X. Yu, A. K. Bashir, H. N. Chaudhry, and D. Wang, "A machine learning approach for feature selection traffic classification using security analysis," *The Journal of Supercomputing*, vol. 74, pp. 4867–4892, 2018.

[17] S. Kumar, A. Viinikainen, and T. Hamalainen, "Evaluation of ensemble machine learning methods in mobile threat detection," presented at the 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST), IEEE, 2017, pp. 261–268.

[18] H. Mohamad Tahir *et al.*, "Hybrid machine learning technique for intrusion detection system," 2015.

[19] V. Kumar and S. Minz, "Feature selection: a literature review," *SmartCR*, vol. 4, no. 3, pp. 211–229, 2014.

[20] P. Yang and Q. Zhu, "Finding key attribute subset in dataset for outlier detection," *Knowledge-based systems*, vol. 24, no. 2, pp. 269–274, 2011.

[21] G. Forman, "An extensive empirical study of feature selection metrics for text classification.," *J. Mach. Learn. Res.*, vol. 3, no. Mar, pp. 1289–1305, 2003.

[22] N. Moustafa, "The UNSW-NB15 dataset." UNSW, Sydney, 2019. doi: 10.26190/5d7ac5b1e8485.

[23] Scikit-learn, "scikit-learn: machine learning in Python — scikit-learn 1.2.2 documentation," 2023. https://scikit-learn.org/stable/ (accessed May 30, 2023).