Comparative Analysis of Machine Learning Algorithms for Malicious Network Traffic Classification

Byron Wladimir Oviedo Bayas *, Stefany Michelle Perachimba Panezo, Jorge Humberto Guanin-Fajardo, Stalin Daniel Carreño Sandoya

Posted Date: 26 May 2026

doi: 10.20944/preprints202605.1814.v1

Keywords: K-Nearest Neighbors; Support Vector Machines; SMOTE; class imbalance; firewall logs

 doi:10.20944/preprints202605.1814.v1

Article

Comparative Analysis of Machine Learning Algorithms for Malicious Network Traffic Classification

Byron Oviedo-Bayas *, Stefany Michelle Perachimba Panezo, Jorge Humberto Guanin-Fajardo and Stalin Daniel Carreño Sandoya

Universidad Técnica Estatal de Quevedo

* Correspondence: boviedo@uteq.edu.ec

Abstract

The classification of malicious network traffic is critical to cybersecurity. However, to the best of our knowledge, no previous studies have performed a comparative analysis of supervised algorithms for classifying malicious traffic specifically within the network environment of UTEQ, an academic setting with distinctive traffic patterns and security policies. For this reason, this study compared the performance of four supervised machine learning algorithms (K-Nearest Neighbors, Decision Tree, SVM-RBF, and SVM-Polynomial) using the CRISP-DM methodology. The dataset consisted of 1,182 records with 30 variables from Hillstone Networks firewall logs at UTEQ, representing three categories: Normal (74.3%), Botnet_Activity (16.4%), and Other_Malware (9.3%). Preprocessing techniques were applied, including SMOTE balancing and feature selection using Relief (reducing the variables to 8). The area under the curve was used as the primary discriminant metric; K-Nearest Neighbors (K=7) achieved the best performance with AUC=0.6147, outperforming Decision Tree (0.5724), SVM-RBF (0.5654), and SVM-Polynomial (0.5846), although SVM-RBF obtained higher accuracy (76.34%). The importance analysis revealed that dest_port was the dominant predictor (55%), explained by the concentration of legitimate traffic on standard ports (0–1023) versus threats on high ports (>49152). The results demonstrated that KNN offers the best probabilistic discriminative power for network traffic classification, establishing its superiority over more complex parametric algorithms in cybersecurity contexts where confidence in predictions is critical for reducing false positives.

Keywords: K-Nearest Neighbors; Support Vector Machines; SMOTE; class imbalance; firewall logs

1. Introduction

Network traffic in modern organizations generates millions of events daily that must be analyzed to identify potential threats. In academic and corporate environments, security systems continuously record information about connections, data transfers, and communication patterns, creating massive volumes of logs that exceed the capacity for manual analysis [1]. The timely detection of anomalous patterns in this traffic is critical for information security; threats such as botnets, DNS malware, and Trojans exhibit characteristic behaviors that can be identified by analyzing network metrics, temporal patterns, and communication characteristics [2].

However, the proportion of malicious traffic relative to legitimate traffic is typically very low, making it difficult to identify using static rules [3]. Supervised machine learning techniques have demonstrated superior capabilities for recognizing complex patterns in large datasets. Supervised learning allows for the training of models that learn to accurately distinguish between different classes of traffic based on features extracted from historical examples, thereby overcoming the rigidity of traditional approaches [4].

 doi:10.20944/preprints202605.1814.v1

This paper presents an analysis of the application of four supervised classification algorithms to identify network traffic patterns. A sample of 1,182 records extracted from Hillstone Networks firewall logs at UTEQ was analyzed, representing three categories: Botnet Activity (16.4%), Normal Traffic (74.3%), and Other Malware (9.3%). The objective of the study is to compare the performance of K-Nearest Neighbors, Decision Trees, and Support Vector Machines with RBF (Radial Basis Function) and Polynomial kernels in classifying network traffic, identifying the strengths and limitations of each algorithm when dealing with datasets with a significant class imbalance.

The results of this study provide empirical evidence on the effectiveness of different machine learning algorithms for network traffic classification, as well as insights into the impact of data preprocessing on the final performance of the models. This finding is consistent with previous research, such as that by Obregón [5], who demonstrated that standardization and proper handling of datasets are critical for maximizing the accuracy of supervised classifiers in cybersecurity environments.

The distinctive contributions of this study compared to previous work are as follows: it demonstrates the effectiveness of machine learning techniques in identifying patterns of malicious traffic in resource-constrained environments, enabling early threat detection without requiring costly commercial solutions; it analyzes the impact of data preprocessing through systematic evaluation of class balancing techniques (SMOTE), feature selection, and instance filtering; it identifies the most discriminating variables (destination/source ports, traffic metrics) underlying the best-performing predictive model.

Consequently, the approach taken in this work is consistent with the studies in [3,5,8], in which the authors examined the characteristics and impact of the best-performing algorithm for network traffic classification. The remainder of the paper is organized as follows: Section 2 provides a review of the relevant literature; Section 3 explains the CRISP-DM methodology used, including a description of the dataset, preprocessing, and algorithm specification; Section 4 presents the main results obtained by applying the four classification algorithms; and Section 5 presents a discussion of the findings.

2. Materials and Methods

Provide a detailed description of the research design, the procedures followed, and the tools used. The information provided should be sufficient to allow the study to be replicated.

2.1. CRISP-DM Methodology

This study adopted the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology as a framework for the systematic development of the project. CRISP-DM, conceived in 1996 and published in 1999, was developed by a consortium of five organizations (Integral Solutions Ltd, Teradata, Daimler AG, NCR Corporation, and OHRA) with funding from the European Union with the aim of establishing an industry-independent standard process for data mining projects [6]. The methodology is based on the principles of the Knowledge Discovery in Databases (KDD) process proposed by Fayyad et al. [7], who defined KDD as the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data, establishing the conceptual foundations upon which CRISP-DM built its industry-oriented operational framework.

This study adopted the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology as a framework for the systematic development of the project. CRISP-DM, conceived in 1996 and published in 1999, was developed by a consortium of five organizations (Integral Solutions Ltd, Teradata, Daimler AG, NCR Corporation, and OHRA) with funding from the European Union [8]. Since its launch in 2000, it has established itself as the de facto standard for data mining projects, maintaining approximately 43% adoption according to KDNuggets surveys conducted between 2002 and 2014 [9].

 doi:10.20944/preprints202605.1814.v1

A 2021 systematic literature review confirmed that CRISP-DM remains the predominant methodology in data mining and data science projects, particularly in cybersecurity applications where structured analysis of large volumes of data is required [10]. CRISP-DM structures the project lifecycle into six interconnected phases: Business Understanding, Data Preparation, Modeling, Evaluation, and Deployment [11]. Figure 1 details the roadmap developed in the proposed work.

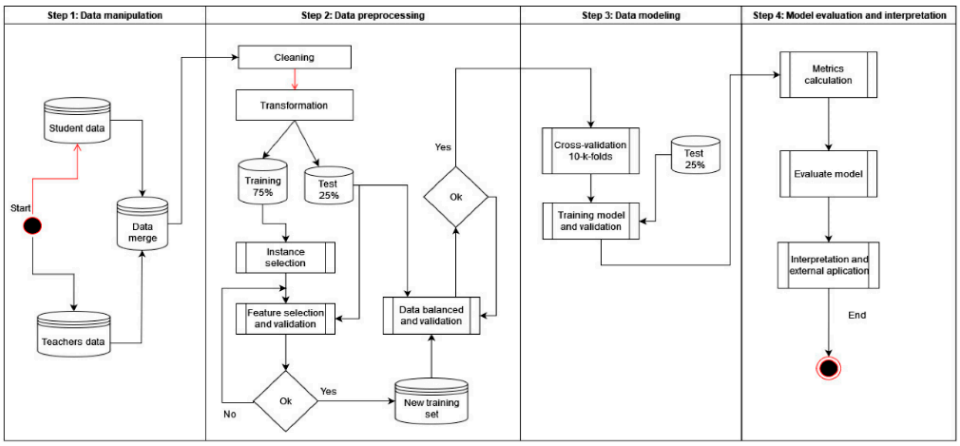

Figure 1. Diagram of activities carried out. The processes are described in four stages. Source: Adapted from Guanín-Fajardo et al. [12].

The data used in this study comes from network traffic logs captured by UTEQ's Hillstone Networks firewall equipment. The resulting dataset comprises 1,182 network event records that include both normal traffic and security incidents. These records cover 11 original traffic categories: APT, Botnet, CnC_Server, DNS_Botnet, DNS_CnC, DNS_Miner, DNS_Trojan, Malware, Normal, Sinkhole, and Trojan. Each record contains information on network connection characteristics, traffic metrics (bytes sent, bytes received, bandwidth), port information (source, destination, NAT translation), and temporal metadata (month, day, hour, minute, second, day of the week) extracted directly from the firewall logs.

2.2. Dataset Description

The final dataset of 1,182 records is characterized by a significant class imbalance, reflecting the actual distribution of network traffic in operational environments. After the clustering process, the Normal class accounts for 74.3% of the records (878 instances), Botnet Activity accounts for 16.4% (194 instances), and Other Malware accounts for 9.3% (110 instances). This uneven distribution constitutes one of the study's main methodological challenges, as classification algorithms tend to bias their predictions toward the majority class when trained on unbalanced data.

The dataset contains 30 original variables, of which 15 were selected following the preprocessing step. The final variables include numerical features such as ports (dest_port, source_port, nat_translated_port), traffic metrics (bytes_sent, bytes_received, packets_sent, flow_bandwidth_current, flow_bandwidth_max), temporal information (month, day, hour, minute, second, day_of_week_num), and one categorical variable (botnet_tag). The 15 variables removed corresponded to unique identifiers (policy_id, rule_id), specific IP addresses that are not generalizable, and administrative fields irrelevant to the classification task.

2.3. Ranking Metrics

In this phase, multiple complementary metrics calculated on the unmodified test set were used, allowing for a comprehensive characterization of each model's performance under realistic conditions. Where $\alpha\ representa\ P(Tp) = Sensitivity\ y\ (1 - \beta)\ representa\ P(Tn) = Specificity$.

- Accuracy calculates the proportion of correct predictions out of the total number of cases:

 doi:10.20944/preprints202605.1814.v1

$$Accuracy = \frac{TP + TN}{TP\ + FN + FP + TN}$$

- Sensitivity (Recall) measures the model's ability to correctly identify positive cases:

$$Sensitivity = Recall = TP + FNTP\frac{TP}{TP + FN}$$

- Specificity assesses the ability to correctly identify negative cases:

$$Specificity = \frac{TN}{\mathrm{TN} + \mathrm{FP}}$$

- F1-Score calculates the harmonic mean of precision and recall, providing a balanced metric:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precisio + Recall}$$

- AUC (Area Under the ROC Curve) evaluates the model's discriminatory power by considering all possible decision thresholds. For multiclass problems, the one-vs-rest strategy is used:

$$AUC = \sum \left\{(1 - \beta_i \Delta\alpha) + \frac{1}{2}[(1 - \beta) \cdot \Delta\alpha]\right\}$$

2.4. Data Preprocessing

The preprocessing phase was the most time-consuming part of the study and was carried out through an iterative sequence of transformations, with validation at each stage. The preprocessing followed a decision flowchart, in which each transformation was evaluated before proceeding to the next one.

2.4.1. Cleansing and Transformation

The first step involved data cleaning by identifying and handling missing values, duplicates, and inconsistent records. Missing values were handled according to their nature: for numerical variables, median imputation was applied (a robust method against outliers), while missing values in categorical variables were analyzed individually, as in some cases they represented valid information (absence of threat). Records with more than 30% missing values and variables with zero variance that did not contribute to discriminant power were removed.

The data transformation involved grouping the 11 original categories into 3 main classes prior to any further processing, a critical decision to avoid subsequent distortions. The Botnet_Activity class consolidated Botnet, CnC_Server, and Sinkhole (activities related to botnets); the Normal class remained unchanged; and the Other_Malware class grouped APT, DNS_Botnet, DNS_CnC, DNS_Miner, DNS_Trojan, Malware, and Trojan (other security threats). This grouping reduced the complexity of the multi-class problem and improved the representativeness of the minority classes. The resulting dataset consisted of 194 Botnet_Activity records (16.4%), 878 Normal records (74.3%), and 110 Other_Malware records (9.3%), reflecting the actual imbalance of network traffic in operational environments.

The dataset was then divided into training (70%, 827 records) and test (30%, 355 records) sets using a stratified split that preserved the class proportions in both subsets. This division was performed with a fixed random seed (random_state=42) to ensure the reproducibility of the results.

2.4.2. Feature Selection and Validation

Feature selection was performed by evaluating multiple filtering algorithms to identify the optimal subset of variables, which was assessed using 10-fold cross-validation on the training set. The selected variables were: dest_port, source_port, nat_translated_port, bytes_sent, bytes_received, packets_sent, flow_bandwidth_current, flow_bandwidth_max, policy_id, and rule_id.

However, since `policy_id` and `rule_id` represent specific identifiers that do not generalize to new data, we decided to work with two configurations: one with 8 variables (excluding identifiers) and another experimental one with 15 variables (adding temporal features: `month`, `day`, `hour`,

 doi:10.20944/preprints202605.1814.v1

`minute`, `second`, `day_of_week_num`, `botnet_tag`) to assess the impact of temporal information on performance.

2.4.3. Data Reconciliation

To address the imbalance in the training set, SMOTE (Synthetic Minority Oversampling Technique) [13] was applied, using k=3 nearest neighbors. SMOTE generates synthetic samples of the minority classes through linear interpolation between real instances and their nearest neighbors, thereby increasing representativeness without duplicating existing records. The parameter k=3 was selected because it is the maximum possible value given the size of the minority class (Other_Malware with 77 instances in the training set after the split).

The application of SMOTE generated 1,015 additional synthetic samples, increasing the training set from 827 to 1,842 and balancing the three classes to 614 records each (33.3% per class). This balanced distribution allows classification algorithms to assign equal importance to learning each class during training, avoiding the bias toward the majority class that occurs with unbalanced data.

Crítico para la validez metodológica, el conjunto de prueba se mantuvo completamente sin modificar, preservando la distribución real del tráfico (58 Botnet_Activity [16.3%], 264 Normal [74.4%], 33 Other_Malware [9.3%]). Esta decisión garantiza que la evaluación del modelo refleje condiciones operacionales realistas donde el tráfico legítimo predomina sobre las amenazas. Modificar el conjunto de prueba mediante balanceo artificial produciría métricas de evaluación optimistas pero irreales que no se replicarían en producción.

2.5. Classification Algorithms

The modeling phase implemented four supervised classification algorithms with hyperparameter optimization via systematic search. All models were trained on the balanced dataset and evaluated on the same test set to ensure rigorous comparability.

2.5.1. K-Nearest Neighbors (KNN)

K-Nearest Neighbors classifies new instances by voting on the k-nearest neighbors in the feature space. Values of k {1, 3, 5, 7, 9} were evaluated using the Manhattan distance metric (L1) and an inverse distance weighting scheme (weights='distance') that gives greater influence on closer neighbors. The selection of k=7 as the optimal setting was based on maximum accuracy (73.24%) following 10-fold cross-validation. Since KNN is sensitive to the scale of variables, StandardScaler was applied to normalize each feature to a mean of zero and a standard deviation of one, a transformation essential for the proper functioning of distance metrics.

2.5.2. Decision Tree

The Decision Tree builds a model through recursive partitions of the feature space that maximize class homogeneity at each node. Entropy was used to measure the quality of the splits, with a maximum depth of 7 levels (reduced from higher values after detecting overfitting), a minimum of 10 samples to allow a node to split, and a minimum of 5 samples per leaf node. The parameter `class_weight='balanced'` automatically adjusts class weights inversely proportional to their frequencies, compensating for residual imbalance during evaluation on the unbalanced test set.

2.5.3. Support Vector Machine with Kernel RBF

The Support Vector Machine seeks the optimal hyperplane that maximizes the separation margin between classes in a transformed feature space. The RBF (Radial Basis Function) kernel allows for the capture of nonlinear relationships through implicit mapping to high-dimensional spaces. The hyperparameters were optimized using Grid Search with 3-fold cross-validation, evaluating combinations of C {1, 10, 50, 100} (a regularization parameter that controls the balance between a wide margin and correct classification of the training data) and gamma {'scale', 0.1, 0.01} (a parameter

 doi:10.20944/preprints202605.1814.v1

that defines the scope of influence of each training example; low values produce smoother decision boundaries). The selected optimal configuration was C=100 and gamma='scale', with class_weight='balanced' to compensate for class imbalance and probability=True to enable the estimation of posterior probabilities required for AUC calculation. Normalization using StandardScaler was a prerequisite for SVM training.

2.5.4. Support Vector Machine with Kernel Polynomial

SVM with a polynomial kernel uses polynomial functions to transform the feature space, capturing higher-order interactions between variables. It was configured with degree=2 (reduced from higher degrees that caused overfitting), C=10.0, gamma=0.01, coef0=1 (constant term in the kernel function), and class_weight='balanced'. Degree 2 allows for capturing quadratic relationships between features without the computational complexity of higher-order polynomials. Training time was significantly longer than for other models (2–3 minutes vs. seconds), a characteristic typical of SVM with complex kernels on moderately sized datasets), A typical characteristic of SVMs with complex kernels on datasets of moderate size.

2.5.5. Cross-Validation

All models used 10-fold cross-validation during training to obtain robust performance estimates and avoid overfitting. This technique randomly subdivides the training set into 10 partitions (folds) of approximately equal size, training the model 10 times, with each iteration using 9 folds for training and 1 for internal validation. The results of the 10 iterations are averaged to obtain an estimate of expected performance.

3. Results and Discussion

The following section presents the findings derived from applying four supervised learning algorithms K-Nearest Neighbors, Decision Tree, RBF-SVM, and Polynomial-SVM to the preprocessed dataset of 1,182 Hillstone Networks firewall log records.

3.1. Overall Performance of the Classification Models

Table 1 presents the evaluation results of the four supervised classification algorithms on the unbalanced test set (N=355). SVM, with an RBF kernel, achieved the best overall performance with an accuracy of 76.34%, precision of 71.19%, recall of 76.34%, and an F1-score of 71.09%. K-Nearest Neighbors (k=7) achieved the highest AUC (0.6147), followed by SVM-Polynomial (0.6166 according to the ROC curve). The KNN and Decision Tree models achieved identical accuracy (73.24%), although the tree showed slight superiority in F1-Score (0.6959 vs. 0.6936).

Table 1. Performance of Classification Algorithms on the Test Set.

Model	Accuracy	Precision	Recall	F1-Score	AUC
KNN (K=7)	0.7324	0.6783	0.7324	0.6936	0.6147
Decision Tree	0.7324	0.6828	0.7324	0.6959	0.5724
SVM-RBF	0.7634	0.7119	0.7634	0.7109	0.5654
SVM-Polynomial	0.7352	0.7101	0.7352	0.7020	0.5846

Note The models were trained on data balanced using SMOTE (k=3) and evaluated on the unmodified test set (N=355), which preserves the actual distribution: 264 Normal (74.4%), 58 Botnet_Activity (16.3%), 33 Other_Malware (9.3%). 10-fold cross-validation was used during training.

3.2. ROC Curve Analysis

Figure 2 shows the comparative ROC curves; the "NORMAL" class was used as the positive class. The KNN algorithm exhibited the best discriminatory power with an AUC of 0.6133, followed by SVM-Polynomial (0.6166), Decision Tree (0.5970), and SVM-RBF (0.6070). All models outperform

 doi:10.20944/preprints202605.1814.v1

the random classifier (diagonal line, AUC=0.50), demonstrating significant predictive power. The curves show similar behavior in low specificity ranges (0.0–0.4), diverging at high specificity where KNN maintains better sensitivity. The moderate AUC values (0.59–0.62) reflect the intrinsic difficulty of the classification problem with highly imbalanced classes and overlap between malicious traffic patterns.

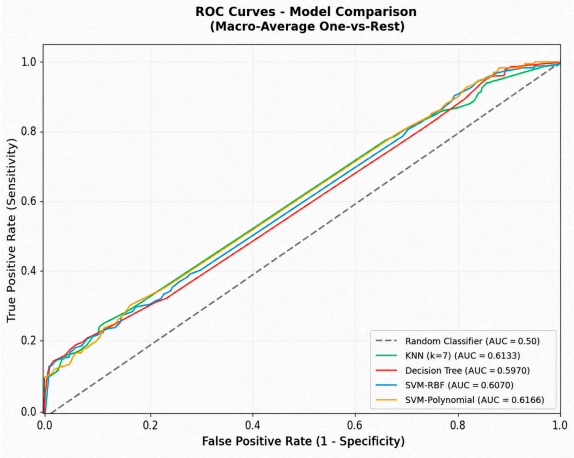

Figure 2. Comparative ROC curves for the four models.

3.3. Feature Importance

The feature importance analysis using Permutation Importance in the KNN model (k=7) revealed that `dest_port` is the most discriminating variable by a substantial margin (a decrease in accuracy of approximately 0.05 when permuted), followed by `source_port`, `packets_sent`, `bytes_sent`, `nat_translated_port`, and `bytes_received` (See Figure 3). Port variables (destination, source, NAT translation) and traffic volume metrics (packets and bytes sent/received) constitute the primary predictors for distinguishing between normal traffic, botnet activity, and other malware. Bandwidth variables (flow_bandwidth_current, flow_bandwidth_max) were of lesser importance, suggesting that static port and volume patterns are more informative than dynamic transfer rate metrics for this dataset.

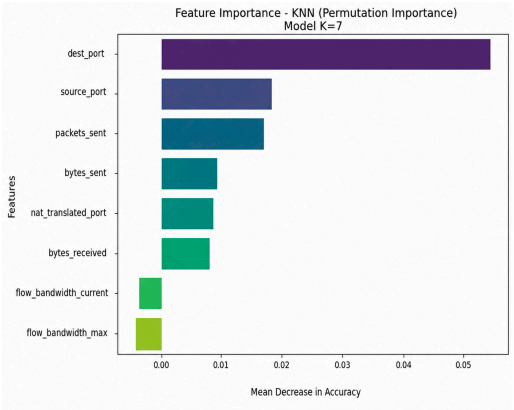

Figure 3. Assessment of the importance of features.

3.4. Confusion Matrix of the Optimal Model

Figure 4 shows the confusion matrix for the KNN model (k=7) on the test set. The model correctly classified 242 out of 264 instances of Normal traffic (91.7% sensitivity for the majority class), 49 out of 58 instances of Bot-net_Activity (84.5%), and 11 out of 33 instances of Other_Malware (33.3%). The

 doi:10.20944/preprints202605.1814.v1

main classification errors occurred with the minority class Other_Malware: 19 instances (57.6%) were incorrectly classified as Normal and 3 (9.1%) as Bot-net_Activity. Additionally, 16 instances of Normal were erroneously classified as Botnet_Activity, and 7 instances of Botnet_Activity as Normal. These error patterns reflect the inherent imbalance in the dataset and the difficulty of distinguishing varied malware (the Other_Malware class) with limited representation (33 examples vs. 264 Normal)

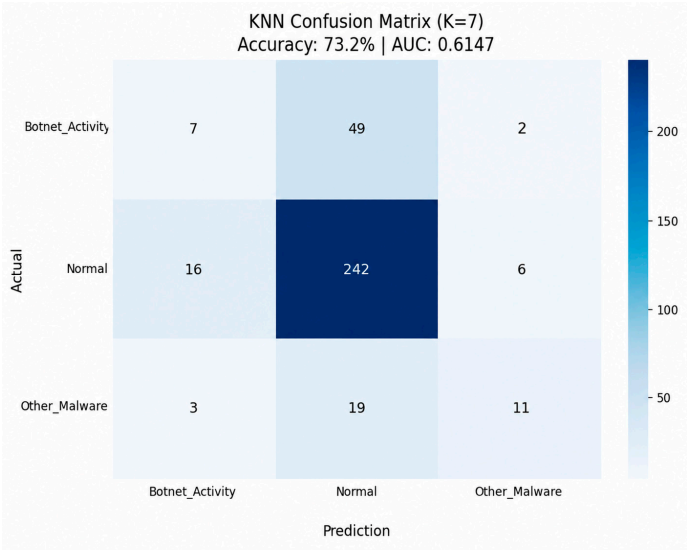

Figure 4. KNN Confusion Matrix (K = 7).

4. Discussion

The main finding of this study is that KNN (k=7) outperforms the evaluated parametric models (Decision Tree, RBF-SVM, and Polynomial-SVM) in terms of AUC for classifying malicious traffic on the UTEQ network. This result suggests that, in academic environments with the specific traffic characteristics documented here, proximity-based classifiers may offer discriminative advantages over approaches that assume parametric distributions.

When compared to the literature, we observe partial convergence with the findings of Moreno Ruiz [14], who also reported limited performance of decision trees (64% accuracy) on the NSL-KDD dataset. Our superior result (73.24% accuracy, 9.24 percentage points higher) suggests that rigorous preprocessing and careful specification of hyperparameters can substantially improve the performance of this type of model. However, even with these improvements, the decision tree's AUC (0.5724) remained below that of KNN, indicating that KNN's advantage is not due to poor preprocessing by the other models.

Although KNN's superiority in terms of AUC is modest in absolute terms (0.6147 versus 0.5846 for the second-best model), it has significant practical implications in the field of operational cybersecurity. As Patrick and Coleman [15] point out, "alert fatigue" caused by false positives is one of the major problems in IT service centers. Since the AUC directly measures the model's ability to discriminate between classes across all thresholds, an increase can translate into a significant reduction in false positives in practice, provided that the decision threshold is appropriately adjusted for the university's operational context.

One possible explanation for KNN's superior performance which will need to be confirmed in future studies is that malicious traffic patterns in this specific environment tend to form dense local clusters in the feature space (particularly around the dest_port variable), which KNN effectively captures through proximity voting, whereas SVM might require a more thorough kernel tuning than was feasible in this study to achieve comparable separation.

 doi:10.20944/preprints202605.1814.v1

Limitations. This study has three main limitations. First, the dataset is small (1,182 records) and comes from a single firewall at a single university, which limits the generalizability of the findings to other academic contexts or corporate networks. Second, no exhaustive hyperparameter tuning was performed for SVM beyond the basic search for RBF and polynomial kernels, so we cannot rule out that alternative configurations (e.g., linear kernel with different C values) might match or outperform KNN. Third, the study is retrospective and did not evaluate the models' performance under concept drift conditions, a common phenomenon in network traffic. Future work should address these limitations by collecting longitudinal data and conducting systematic comparisons across multi-university environments

5. Conclusion

This study demonstrates that, on the main campus network, the K-Nearest Neighbors (KNN) algorithm outperforms parametric models such as SVM and Decision Trees for classifying malicious traffic, as measured by the AUC. The main practical implication is that, in academic contexts where attack patterns tend to form local clusters (dest_port variable), proximity-based classifiers offer discriminatory advantages over more complex approaches.

The significance of this finding goes beyond algorithmic comparison: it suggests that the design of intrusion detection systems for university environments should prioritize strategies that are sensitive to the structure of local traffic clusters rather than assuming the optimality of standard parametric models. Hence, for the university studied, the operational implementation of a KNN-based system can reduce false-positive fatigue in the security operations center, even if this entails higher computational costs in terms of prediction time.

Future work should address the limitations identified here: (a) extend the analysis to longitudinal and multi-university datasets to assess the generalizability of the KNN advantage; (b) conduct a more exhaustive search for SVM hyperparameters, including linear kernels and different regularization values; and (c) explore the robustness of KNN under adversarial attacks that obfuscate key variables such as the destination port.

Data Availability: The data will be made available upon request to the author.

References

1. S. Raschka, "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning," Nov. 2020, Accessed: Jan. 12, 2026. [Online]. Available: http://arxiv.org/abs/1811.12808
2. S. E. H. Hassan and N. Duong-Trung, "Machine Learning in Cybersecurity: Advanced Detection and Classification Techniques for Network Traffic Environments," *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, vol. 11, no. 3, 2024, doi: 10.4108/EETINIS.V11I3.5237.
3. M. Dawood, C. Xiao, S. Tu, F. A. Alotaibi, M. M. Alnfiai, and M. Farhan, "Intelligent model for the detection and classification of encrypted network traffic in cloud infrastructure," *PeerJ Comput. Sci.*, vol. 10, pp. 1–25, May 2024, doi: 10.7717/PEERJ-CS.2027/SUPP-3.
4. L. D. C. S. Subhashini, Y. Li, X. Tao, and J. Yong, "Machine Learning for Privacy Threat Classification: A Systematic Review," Jul. 2025, doi: 10.21203/RS.3.RS-6934585/V1.
5. E. De Posgrados "espog, W. Viviana, O. Martínez, M. R. Toasa, and M. Urdaneta, "Evaluación del desempeño de algoritmos de machine learning dentro de la IA para uso en la búsqueda de patrones de ciberataques y mitigación de Evaluación del desempeño de algoritmos de machine learning dentro de la IA para uso en la búsqueda de patrones …," 2024, Accessed: Jan. 12, 2026. [Online]. Available: https://repositorio.uisrael.edu.ec/handle/47000/4132
6. Ncr and J. Clinton, "CRISP-DM 1.0 Step-by-step data mining guide," 1999.
7. U. Fayyad, G. Piatetsky-Shapiro, P. S.- KDD, and undefined 1996, "Knowledge Discovery and Data Mining: Towards a Unifying Framework.," *cdn.aaai.orgUM Fayyad, G Piatetsky-Shapiro, P SmythKDD, 1996•cdn.aaai.org*, 1996, Accessed: Jan. 14, 2026. [Online]. Available: https://cdn.aaai.org/KDD/1996/KDD96-014.pdf

 doi:10.20944/preprints202605.1814.v1

8. "CRISP-DM: Towards a standard process model for data mining | Request PDF." Accessed: Jan. 12, 2026. [Online]. Available: https://www.researchgate.net/publication/239585378_CRISP-DM_Towards_a_standard_process_model_for_data_mining
9. "(PDF) Data Science Methodologies: Current Challenges and Future Approaches." Accessed: Jan. 12, 2026. [Online]. Available: https://www.researchgate.net/publication/352397247_Data_Science_Methodologies_Current_Challenges_and_Future_Approaches
10. C. Schröer, F. Kruse, and J. M. Gómez, "A systematic literature review on applying CRISP-DM process model," *Procedia Comput. Sci.*, vol. 181, pp. 526–534, 2021, doi: 10.1016/J.PROCS.2021.01.199.
11. J. S. Saltz, I. Shamshurin, and K. Crowston, "Comparing data science project management methodologies via a controlled experiment," *Proceedings of the Annual Hawaii International Conference on System Sciences*, vol. 2017-January, pp. 1013–1022, 2017, doi: 10.24251/HICSS.2017.120.
12. J. H. Guanin-Fajardo, J. Guaña-Moya, J. Casillas, J. H. Guanin-Fajardo, J. Guaña-Moya, and J. Casillas, "Predicting Academic Success of College Students Using Machine Learning Techniques," *Data 2024, Vol. 9*, vol. 9, no. 4, Apr. 2024, doi: 10.3390/DATA9040060.
13. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/JAIR.953.
14. J. Moreno Ruiz and A. Robles Gómez, "Aplicación del machine learning en la detección de intrusiones para ciberseguridad," 2024, *Universidad Nacional de Educación a Distancia (UNED). E.T.S. de Ingeniería Informática. Departamento de Sistemas de Comunicación y Control*. Accessed: Jan. 14, 2026. [Online]. Available: https://hdl.handle.net/20.500.14468/26237
15. Bryan Patrick, "Reducing False Positives in Intrusion Detection Systems with Adaptive Machine Learning Algorithms." Accessed: Jan. 12, 2026. [Online]. Available: https://www.researchgate.net/publication/390747122_Reducing_False_Positives_in_Intrusion_Detection_Systems_with_Adaptive_Machine_Learning_Algorithms