



Lightweight Machine Learning Models for Detecting DNS Data Exfiltration Attacks in Cloud and Enterprise Networks

Tolulope Onasanya

tdonasanya@aggies.ncat.edu

North Carolina Agricultural and
Technical State University, USA

John Aigberua

johnneya16@gmail.com

Obafemi Awolowo University, Nigeria

Hannah I. Tanimowo

tanim006@umn.edu

University of Minnesota Duluth

Oduwunmi Esther Odukoya

odukoyao@etsu.edu

East Tennessee State University

ABSTRACT

Cloud and Enterprise Networks are the foundation of today's digital age, facilitating frictionless communication and service delivery. Cloud and Enterprise Network attacks increasingly depend on trusted protocols, and DNS Data Exfiltration Attacks in Cloud and Enterprise Networks have evolved as a devious and powerful means to evade classic defences. Detecting DNS Data Exfiltration Attacks in Cloud and Enterprise Networks is therefore a pressing challenge that requires efficient and accurate solutions. This study investigates Machine Learning Models for Detecting DNS Data Exfiltration Attacks in Cloud and Enterprise Networks, focusing on lightweight approaches such as Random Forest, Decision Tree, Multi-Layer Perceptron, Logistic Regression, and Gaussian Naïve Bayes. Both Random Forest and Decision Tree achieved perfect evaluation scores (100%) across standard metrics, but closer inspection of confusion matrices revealed Random Forest as the superior model, misclassifying only two malicious instances while generating no false positives. The significance of this research lies in demonstrating that lightweight models, particularly Random Forest, can provide highly accurate, resource-efficient, and practical real-time protection against DNS exfiltration threats, ensuring the resilience of cloud and enterprise infrastructures.

Keywords: DNS Attack, Cloud, Machine Learning, Data Exfiltration.

1. INTRODUCTION

Cloud and business networks have now become indispensable in enabling worldwide connectivity, supporting contemporary services, and providing large-scale data exchange within organizational and technical contexts. Their pervasiveness and reliance on critical protocols, such as the Domain Name System (DNS), make them a lucrative target for nefarious actors willing to spread network disruption. As DNS is the root to translate readable domain names into IP addresses, its openness and generally unmanaged state in cloud and enterprise networks leave it vulnerable.

With an intent to leverage DNS for use in attacks, attackers leverage DNS to create stealthy communication channels to suck out information without avoiding standard security controls [1], [2]. The increasing threat indicates the need for effective, scalable, and efficient detection techniques best suited to the distinctive dynamics of DNS traffic.

Traditional detection methods, such as signature-based intrusion detection and statistical anomaly detection, have not kept up with the increasing sophistication of DNS exfiltration attacks [13]. Signature-based options heavily depend on predetermined patterns and, consequently, cannot cope with new or obfuscated attack patterns. Similarly, anomaly detection techniques incessantly generate high false positives, overwhelming security analysts and making operations ineffective [8]. With attackers now using low-throughput, covert tunneling techniques, default defenses are inadequate for deployment in real-world situations, particularly in cloud and large enterprise environments where accuracy and effectiveness of detection count [2], [5].

To counter the aforementioned problems, machine learning (ML) has also proven to be a robust paradigm capable of automatically learning complex traffic patterns and distinguishing between malicious and benign flows [23]. ML-based techniques are especially ideal for detecting DNS exfiltration because they have the capability to learn fine-grained non-linear patterns between high-dimensional traffic features that are difficult to model using conventional methods [7], [14].

Apart from that, light ML models are computation-friendly, such that they are ideally deployable on devices with limited resources or low-latency demands within networks [9]. The efficiency-accuracy trade-off is critical to real-time detection within enterprise and cloud setting. In this research, the DNS exfiltration dataset CIC-Bell-DNS-EXF-2021 [22] was employed as the primary data source. This dataset is specifically designed to capture realistic DNS traffic, encompassing both benign communication patterns and malicious exfiltration attempts.

Its comprehensive coverage and balanced representation of attack and normal traffic make it particularly suitable for training and evaluating machine learning models. The real-world adherence of the dataset to actual DNS behavior makes the models not only theoretically accurate but also of practical use, thereby increasing their potential for effective deployment in real-world environments [21].

Several lightweight ML models were tried out in this study, including Random Forest, Decision Tree, Multi-Layer Perceptron, Logistic Regression, and Gaussian Naïve Bayes. These models were selected for their complementary qualities of data structure handling, interpretability, and scalability [6], [16]. Random Forest and Decision Tree, for instance, are highly capable of non-linear patterns, with Logistic Regression providing interpretability as well as tractability. The application of models like the Multi-Layer Perceptron neural network enables the capturing of more subtle feature interactions, while that of Gaussian Naïve Bayes enables computational efficiency with the cost of simplifying assumptions. Evaluating this diverse set of models enables a comprehensive understanding of trade-offs between accuracy, interpretability, and resource efficiency [18].

The significance of this research lies in its potential to advance DNS security by demonstrating that lightweight ML models can provide both high detection accuracy and operational efficiency. Unlike heavyweight deep learning approaches that demand substantial computational resources, the lightweight models investigated here are deployable in real-time environments with minimal overhead [4], [17]. By systematically evaluating and comparing their performance on realistic datasets, this study contributes to the development of practical detection solutions capable of mitigating DNS data exfiltration threats in both cloud and enterprise networks. Ultimately, the findings aim to guide security practitioners toward deploying resilient and efficient models that strengthen defenses against one of the most covert and persistent attack vectors in modern networking [24], [25].

2. AIMS AND OBJECTIVES

The primary aim of this work is to apply lightweight machine learning models to secure cloud and enterprise networks by detecting and preventing DNS data exfiltration attacks using the DNS exfiltration dataset. To achieve this aim, the study is guided by the following specific objectives:

- i. To review existing research on DNS data exfiltration detection, with emphasis on the limitations of traditional detection techniques and the opportunities presented by machine learning approaches.
- ii. Applying light-weight machine learning algorithms like Random Forest, Decision Tree, Multi-Layer Perceptron, Logistic Regression, and Gaussian Naïve Bayes for detecting malicious DNS traffic.
- iii. To compare and test these algorithms in a systematic way against each other in terms of the detection of DNS exfiltration based on notable parameters like accuracy, precision, recall, F1-score, confusion matrix, and AUC-ROC score.
- iv. To recommend the most effective lightweight machine learning model for real-time deployment in securing cloud and enterprise networks against DNS data exfiltration threats.

3. LITERATURE REVIEW

DNS data exfiltration attacks have been a major focus of academic interest over the past few years, as DNS plays a crucial role in cloud and enterprise systems today. Signature-based intrusion detection, as well as statistical anomaly-based analysis, cannot effectively defend against sophisticated and stealthy exfiltration methods. So, they resorted to machine learning (ML) and deep learning (DL) methods, appreciating their capacity to recognize high-dimensional data, capture patterns, and learn in adversarial environments. This section presents a survey of some existing works in the area, with an emphasis on research that explores machine learning methods, ensembles of them, lightweight models, and detection schemes that evolve.

According to Sabir et al. [1], machine learning represents a paradigm shift in the detection of data exfiltration across diverse protocols, including DNS. Their comprehensive survey highlights how ML techniques provide improved adaptability, scalability, and efficiency compared to conventional rule-based methods. The authors categorized approaches into supervised, unsupervised, and hybrid detection models, noting that supervised learning remains the most widely adopted due to the availability of labeled datasets. They also highlighted the important issue of sacrificing detection precision for computational expense, which is typically the factor that determines whether a solution can be applied in real-world settings. This article lays a solid groundwork to explore further in future work, especially for lightweight models that are deployable in real-time use.

Building on these foundations, Žiža, Tadić, and Vuletić [2] examined DNS exfiltration detection in adversarial contexts where the adversary alters behavior to avoid standard defenses. Their experiment proved that adversarial techniques, such as modifying domain name lengths or varying query frequencies, considerably declined the performance of the majority of baseline detection approaches. They emphasized the need for robust ML models that are resilient to such dynamic patterns. Notably, their research points out that model robustness is no less crucial than accuracy, especially in adversarial environments where the attackers put concerted efforts into avoiding security solutions.

In another study that is very close to this one, Mahdavifar et al. [3] suggested a lightweight hybrid detection framework, which incorporates statistical feature extraction with ML classifiers. Their work concentrated on DNS-based data exfiltration and illustrated how lightweight ML models like decision trees and ensemble methods are able to yield useful detection performance without placing heavy computational burdens.

In striking a balance between speed and precision, their method responded to the realities of scale and resource-limited deployment situations. The study validates the use of lightweight and hybrid ML methods as effective alternatives to real-time detection.

Salat, Davis, and Khan [4] explored the broader context of DNS tunneling, exfiltration, and detection in cloud environments. Their contribution was to highlight the challenges faced by DNS threat detection and monitoring in cloud-based systems because they are elastic and distributed. They pointed out that DNS exfiltration attacks in the cloud will likely be fully integrated into typical traffic, thereby making them harder to detect. The authors presented machine learning-based methods that account for contextual and behavioral features for separating malicious from benign DNS queries. Their work highlights the significance of calibrating detection mechanisms to particular deployment scenarios, including cloud networks, for which flexibility and scalability are particularly critical.

In another interesting work, Açıkgözoğlu [5] compared several ML algorithms for the detection of DNS-based data exfiltration. The paper introduced algorithms systematically, such as Random Forest, Naïve Bayes, Logistic Regression, and Neural Networks, with differences in their accuracy, precision, and consumption of computational power. The experiments showed that tree models are better than straightforward statistical methods, but are still effective and therefore viable candidates for lightweight detection systems. The comparison viewpoint gives us valuable guidance to choose models according to the trade-off between performance and resources.

Continuing the research from this direction, Cai et al. [6] also studied applying ensemble deep learning tree models to detect advanced persistent threats (APTs) via DNS-based exfiltration. Their work brought forth new measures of detection that were meant not only to assess accuracy but also robustness and resilience to advanced attack tactics. Ensemble techniques, by combining several models, performed more effectively in detecting low-signal, stealthy exfiltration attacks.

While deep learning models are typically computationally expensive, Cai et al. suggested optimizations that render their method more efficient, showing how top-performing methods could be modified to work on the ground.

The reviewed literature collectively sets a series of repeating themes. First, traditional detection methods are no match for contemporary DNS exfiltration attacks, particularly when attackers dynamically alter their methods. Second, ML methods offer significant advantages in adaptability and detection precision, with tree-based methods often being beneficial and lightweight. Third, adversarial robustness and scalability within a cloud setting are major considerations that must be incorporated into the design of any potentially successful

detection system. Finally, contrasting hybrid and algorithmic methods indicates that no one model fits all; rather, success will rely upon finding a reasonable balance between accuracy, efficiency, and deployment environment.

The studies reviewed here generally support the growing evidence that lightweight machine learning models offer an attractive direction for advancing effective DNS exfiltration detection. Research works like Sabir et al. [1] and Mahdavifar et al. [3] point towards the possibilities of ML in general and light-weight methods in specific, whereas research works like Žiža et al. [2] and Cai et al. [6] indicate the need for adversarial robustness as well as ensemble learning. Together, these projects form the basis for long-term research in efficient and helpful ML-based detection systems for cloud and enterprise environments.

3.1 Research Gap

Although recent studies have improved the detection of DNS data exfiltration using machine learning (ML) and deep learning (DL), some inherent limitations remain. Sabir et al. [23] have recognized the flexibility of ML models but mentioned that the majority of models suffer from a loss of accuracy and computational efficiency, rendering real-time deployment an issue. Similarly, Žiža, Tadić, and Vuletić [24] highlighted how adversarial evasion strategies degrade the effectiveness of existing methods, underscoring the lack of robust lightweight models that can withstand modified exfiltration behaviors.

Mahdavifar et al. [1] and Açıkgözoğlu [18] demonstrated the promise of lightweight models such as decision trees and ensemble methods, yet their studies were limited in scope and did not comprehensively compare multiple lightweight algorithms under large-scale traffic conditions. Furthermore, while Salat, Davis, and Khan [5] examined detection in cloud environments, they stressed the difficulty of distinguishing malicious DNS flows from legitimate queries in dynamic, distributed infrastructures. Lastly, Cai et al. [28] investigated ensemble deep learning methods for advanced persistent threats; however, these methods are typically computationally intensive and unsuitable for enterprise deployment. Collectively, these works acknowledge the need for a thorough assessment of lightweight ML models in detecting DNS exfiltration, with a focus on adversarial robustness, computational resource efficiency, and suitability in enterprise and cloud deployments.

This research directly addresses these gaps by conducting a comparative evaluation of multiple lightweight ML models, Random Forest, Decision Tree, Logistic Regression, Multi-Layer Perceptron, and Gaussian Naïve Bayes, using a large-scale DNS dataset [22]. Unlike previous work that focused primarily on either accuracy [18][23] or adversarial robustness [24], this study integrates both aggregate evaluation metrics and confusion matrix analysis to reveal nuanced trade-offs between false positives and false negatives. The results show that while Decision Tree and Random Forest initially appeared to achieve perfect performance, only Random Forest consistently demonstrated minimal misclassifications, with zero false positives and only two false negatives across over 1.5 million flows.

This finding directly extends prior work by Mahdavifar et al. [1] and Açıkgözoğlu [18], validating the effectiveness of lightweight ensemble models while providing new empirical evidence of their operational reliability in large-scale environments. Moreover, by situating the analysis within the context of cloud and enterprise networks as emphasized by Salat et al. [5], this research bridges the gap between theoretical experimentation and practical deployment. In doing so, it demonstrates how Random Forest, as a lightweight yet robust model, can effectively mitigate DNS exfiltration attacks in real-world scenarios, filling a critical void in existing scholarship and practice.

3.2 Novelty of the Study

The novelty of this study lies in its systematic and practical evaluation of lightweight machine learning models for detecting DNS data exfiltration attacks in cloud and enterprise environments.

While previous works such as those by Mahdavifar et al. [1] and Açıkgözoğlu [18] explored individual lightweight algorithms, their analyses did not comprehensively compare multiple models under realistic traffic conditions using large-scale datasets. This study fills that gap by examining five widely adopted lightweight algorithms side by side—Random Forest, Decision Tree, Logistic Regression, Multi-Layer Perceptron, and Gaussian Naïve Bayes, through both aggregate performance metrics and detailed confusion matrix analysis.

Unlike prior research that often emphasized accuracy alone [23][28], this study highlights the operational implications of model performance, including false positives and false negatives, which are critical in security operations centers where analyst workload and alert fatigue directly affect response effectiveness. By demonstrating that Random Forest achieves near-perfect detection with negligible misclassifications, the research establishes a practical and scalable pathway for real-time deployment in enterprise and cloud infrastructures.

Furthermore, by aligning its findings with challenges noted in cloud-based DNS monitoring [5] and adversarial attack resilience [24], this work introduces a novel contribution: the identification of Random Forest as a lightweight yet robust solution capable of balancing accuracy, efficiency, and operational reliability in modern network environments.

4. METHODOLOGY

The research methodology of this study aims to formulate and test lightweight machine learning models for detecting DNS data exfiltration attacks in cloud and enterprise networks. The focus is on implementing efficient and interpretable models that can operate under resource constraints without sacrificing detection accuracy. The approach follows a systematic pipeline that encompasses exploratory data analysis, data cleaning, feature engineering, model training, and evaluation against multiple performance metrics. The systematic process ensures that the models are trained on clean representative data and vigorously tested to determine the best solution for real-world deployment.

4.1 Research Design

This research adopts a quantitative experimental design, focusing on the application of supervised machine learning algorithms for binary classification. The design was selected because DNS exfiltration detection inherently involves categorizing traffic as either benign or malicious [1], [3]. The workflow involves preprocessing the dataset to eliminate noise and redundancies, engineering features that enhance model learning, and implementing machine learning classifiers to differentiate between normal DNS traffic and exfiltration attempts. The experimental nature of the study allows systematic comparisons of multiple algorithms, Random Forest, Decision Tree, Multi-Layer Perceptron, Logistic Regression, and Gaussian Naïve Bayes, within the same framework, ensuring objective evaluation of their relative performance [2], [5].

4.2 Research Method

The study employs supervised machine learning as the primary method. Supervised learning is appropriate because the dataset used is labeled, with clear distinctions between benign and malicious DNS traffic [6]. Data preprocessing and feature engineering were performed to optimize the dataset for model training. The cleaned and engineered dataset was split into training (70%) and testing (30%) subsets to prevent overfitting and to evaluate the generalizability of the models [4]. All the models were trained via the Python scikit-learn library for reproducibility and uniform application. Accuracy, precision, true positive rate (TPR), true negative rate (TNR), F1-score, and area under the receiver operating characteristic curve (AUC-ROC), the metrics used for evaluation, were selected to yield a balanced assessment of model performance [1], [7].

Confusion matrices were also analyzed to capture the distribution of misclassifications, offering deeper insights into practical detection reliability.

4.3 Dataset Description

The study utilized the DNS Exfiltration Dataset [22], a publicly available benchmark dataset hosted on Kaggle. This dataset was selected due to its comprehensive coverage of both benign and malicious DNS traffic, making it highly suitable for supervised machine learning tasks in this domain [1], [2]. The dataset contains 1,583,163 rows and 45 columns, including 797,626 benign samples and 785,537 malicious samples. These samples capture diverse attributes of DNS queries and responses, including character distributions, resource record types, and temporal information, which are critical for distinguishing exfiltration attempts from normal traffic [3], [4].

After a comprehensive cleaning process, the dataset was refined to 38 columns while retaining all rows to preserve label distribution and dataset integrity. Redundant, single-valued, and irrelevant features, such as flow identifiers and fixed port numbers, were removed, while new features were engineered to improve model interpretability. For instance, the character distribution column was transformed into quantitative features like query length and unique character count, while the sending-to-receiving byte ratio was derived to capture communication asymmetry often present in malicious DNS flows [5]. Temporal features were also introduced by extracting the query hour from timestamps, enabling models to detect time-based patterns in attack behaviors [6].

The dataset contains two categorical variables, including the `dns_top_level_domain`, which was converted into numeric form using label encoding, while all other features were numeric. The target variable, identified as either "Benign" or "Malicious," was also translated into binary values (0 and 1, respectively) to facilitate model training and evaluation. This created an evenly matched, comprehensible, and ideally optimized dataset for the employed machine learning algorithms. Through the combination of rigorous cleaning, feature engineering, and encoding, the dataset was prepared to optimally represent authentic DNS traffic patterns and take advantage of the models' learning capabilities [3], [5].

In summary, this methodology leverages a structured pipeline, cleaning, exploration, feature engineering, supervised learning, and multi-metric evaluation, to develop lightweight yet highly accurate machine learning models for detecting DNS data exfiltration. The dataset’s comprehensiveness and preprocessing ensure that the models are trained on realistic and discriminative features, while the comparative evaluation framework allows identification of the most effective algorithm for deployment in resource-constrained cloud and enterprise environments.

4.4 DATA PREPROCESSING

This study utilizes the Data Exfiltration dataset, which is publicly available at Kaggle (<https://www.kaggle.com/datasets/daumel/dns-tunneling-dataset>). This dataset contains labelled samples of benign DNS traffic and malicious traffic generated through data exfiltration techniques. The dataset comprises 45 columns capturing various attributes of DNS queries and responses, and 1,583,163 rows (797,626 benign samples and 785,537 malicious samples).

Following the data cleaning process, the dataset was refined to 38 columns. Redundant features were removed, and more informative features were created, but all rows were retained. This ensured that the label distribution and overall dataset integrity remained intact.

The dataset’s features can be broadly classified into numeric and categorical types. Numeric features represent measurable data points that can may be continuous or discrete, while categorical features capture qualitative information and consist of discrete categories. In the cleaned dataset, there are only two categorical features; all other features are numerical.

The explanatory variables consist of the various attributes of the dataset relevant to network traffic and DNS requests, which serve as inputs to the machine learning models, enabling them to learn discriminative patterns that separate normal DNS activity from data exfiltration attacks.

The target variable is the outcome to be predicted. In this study, the target variable is the column named label, which is a categorical variable with two values: ‘Benign’ and ‘Malicious’. This variable indicates whether a given DNS query flow is considered normal traffic (Benign) or associated with DNS data exfiltration attack (Malicious). Since there are just two possible outcomes, the issue is essentially a binary classification one.

The column names of the dataset are listed in Table 1, which reports the data type, count of unique values, and range of observed values for all columns, providing a clear indication of the dataset's structure and variability.

Table 1: Overview of Dataset Columns

S/N	Features	Data type	Unique	Range
1.	src_port	Numeric	64504	1025 ... 65535
2.	Duration	Numeric	221158	0 ... 17.58328
3.	total bytes	Numeric	658	144 ... 1737
4.	packets_rate	Numeric	501555	0.11374 ... 699050.7
5.	packets_len_rate	Numeric	1156695	9.2133 ... 85366420
6.	mean_packets_len	Numeric	658	72 ... 868.5
7.	dns_domain_name_length	Numeric	230	5 ... 253
8.	dns_subdomain_name_length	Numeric	62	0 ... 63
9.	numerical_percentage	Numeric	2548	0 ... 0.7442922
10.	character_entropy	Numeric	340679	0.887656 ... 5.932563
11.	max_continuous_numeric_len	Numeric	46	0 ... 53
12.	max_continuous_alphabet_len	Numeric	61	0 ... 63
13.	max_continuous_consonants_len	Numeric	33	0 ... 57

14.	max_continuous_same_alphabet_len	Numeric	25	0 ... 57
15.	vowels_consonant_ratio	Numeric	3166	0 ... 7
16.	conv_freq_vowels_consonants	Numeric	2753	0 ... 0.9
17.	distinct_ttl_values	Numeric	7	1 ... 7
18.	ttl values mean	Numeric	77567	0 ... 655360
19.	distinct_A_records	Numeric	27	0 ... 29
20.	Hour	Numeric	16	3 ... 18
21.	character_length	Numeric	230	5 ... 253
22.	unique_characters	Numeric	65	2 ... 66
23.	resource_record_type_A	Numeric	27	0 ... 29
24.	resource_record_type_NS	Numeric	9	0 ... 10
25.	resource_record_type_CNAME	Numeric	7	0 ... 6
26.	resource_record_type_NULL	Numeric	2	0 ... 1
27.	resource_record_type_HINFO	Numeric	2	0 ... 1
28.	resource_record_type_TXT	Numeric	4	0 ... 3
29.	resource_record_type_KEY	Numeric	2	0 ... 1
30.	resource_record_type_AAAA	Numeric	11	0 ... 15
31.	resource_record_type_NAPTR	Numeric	2	0 ... 3
32.	resource_record_type_DNAME	Numeric	2	0 ... 1
33.	resource_record_type_RRSIG	Numeric	3	0 ... 2
34.	resource_record_type_PRIVATE_USE_65399	Numeric	2	0 ... 1
35.	ans_resource_record_class_count	Numeric	27	1 ... 29
36.	sending_receiving_ratio	Numeric	14271	0.073758 ... 0.9179331
37.	dns_domain_name_length	Categorical	522	'com' ... 'zone'
38.	label	Categorical	2	'Benign' & 'Malicious'

Figure 1 illustrates the distribution of flow durations for DNS traffic categories. Benign DNS traffic spans a broad range of flow durations, reflecting the variability in legitimate network activity. This variation is expected, as normal DNS queries can be influenced by factors such as server response times, caching, and the diversity of requested domains.

On the other hand, malicious DNS traffic demonstrates a far more constrained pattern. The majority of flows are concentrated within a relatively narrow and shorter duration range, suggesting that data exfiltration attempts in DNS communication are executed in brief, consistent bursts. Such behaviour is likely intentional, as attackers may apply tunnelling tools to keep flows compact and repetitive, thereby reducing the likelihood of triggering anomalies in conventional monitoring systems. The distinct difference in flow duration between benign and malicious traffic underscores its value as a discriminative feature for classification.

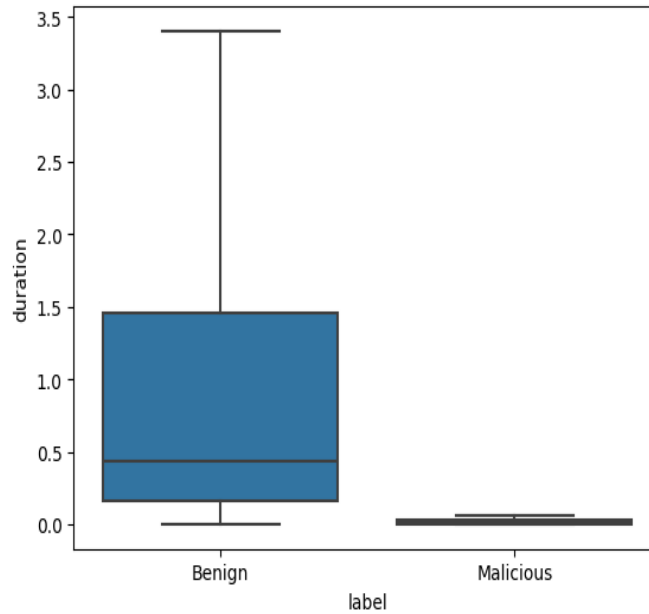


Figure 1: Distribution of Flow Durations in DNS Traffic

Figure 2 shows the average sending/receiving ratio for both benign and malicious DNS traffic. Malicious DNS traffic exhibits a higher ratio, indicating that data exfiltration attempts typically involve sending more data relative to what is received. This asymmetry is characteristic of tunnelling activity, where the attacker’s primary objective is to push information outward rather than to request and retrieve data.

By contrast, benign DNS traffic shows a lower and more balanced ratio, which aligns with the standard query–response pattern of normal DNS communication. This disparity highlights the sending-to-receiving ratio as a useful feature for detecting DNS data exfiltration attacks.

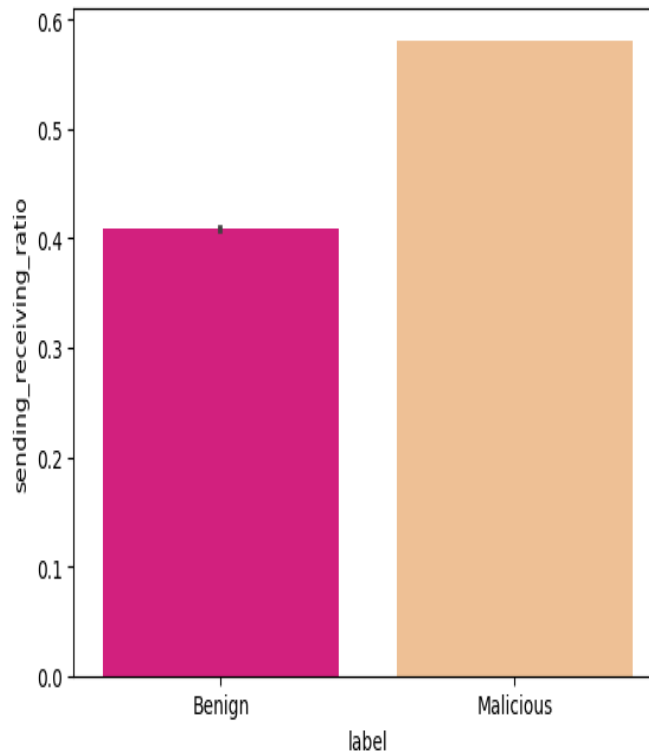


Figure 2: Average Sending-to-Receiving Byte Ratio in DNS Traffic

Figure 3 illustrates the average packet size for the two types of DNS traffic. Malware DNS traffic typically employs larger packets on average than ordinary traffic, which means that data exfiltration attempts are more likely to be successful by sending more data per packet. Benign DNS traffic, on the other hand, is smaller packets in accordance with the usual query-response behavior in regular network traffic.

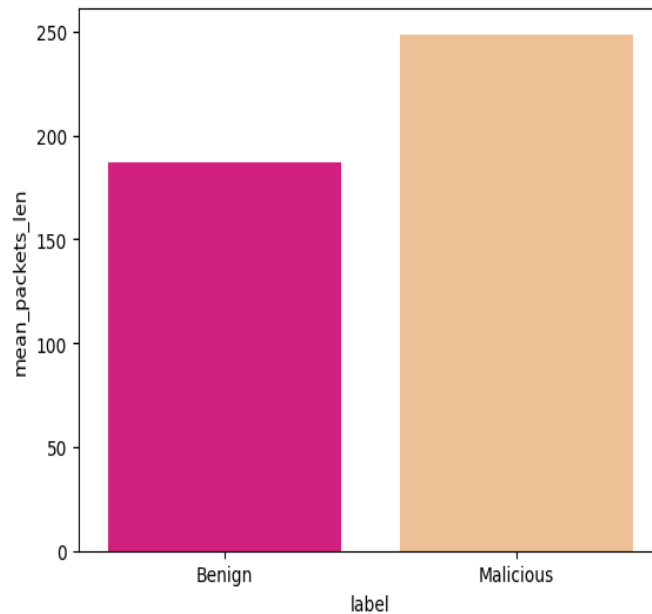


Figure 3: Average Packet Length in DNS Traffic

Figure 4 displays the average packet rate of the DNS traffic categories. This refers to the rate of DNS packets transmitted or received within a flow.

Benign DNS traffic will exhibit a higher packet rate, roughly four times higher than malicious traffic, for normal high-rate query-response traffic.

In contrast, malicious DNS traffic has a significantly lower packet rate, which may be caused by stealthy data exfiltration techniques for evasion or by transmitting larger packet sizes, thereby reducing the overall packet rate. The distinction between the two classes highlights packet rate as a prominent behaviour characteristic in separating benign and malicious DNS traffic.

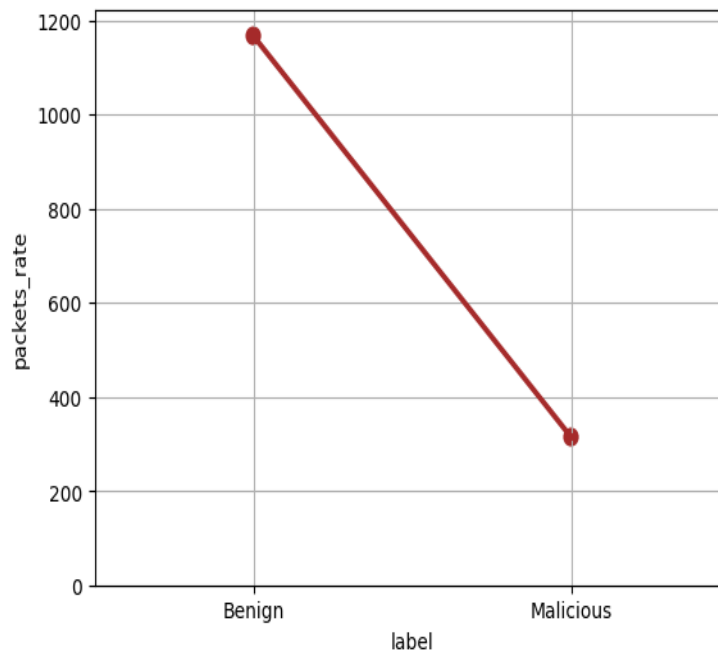


Figure 4: Average Packet Rate in DNS Traffic

Figure 5 illustrates the lengths of subdomain names and DNS domain names for malicious and legitimate DNS traffic. In both instances, malicious traffic has longer name lengths compared to legitimate traffic, which indicates that data exfiltration attempts tend to use long or complicated domain formats to hide and carry data.

The most significant difference lies in the example of DNS domain name length, whose mean for benign traffic is almost four times that of malicious traffic.

Such elongated domains are unlikely to occur in typical user or application-driven queries, making them a strong indicator of suspicious activity.

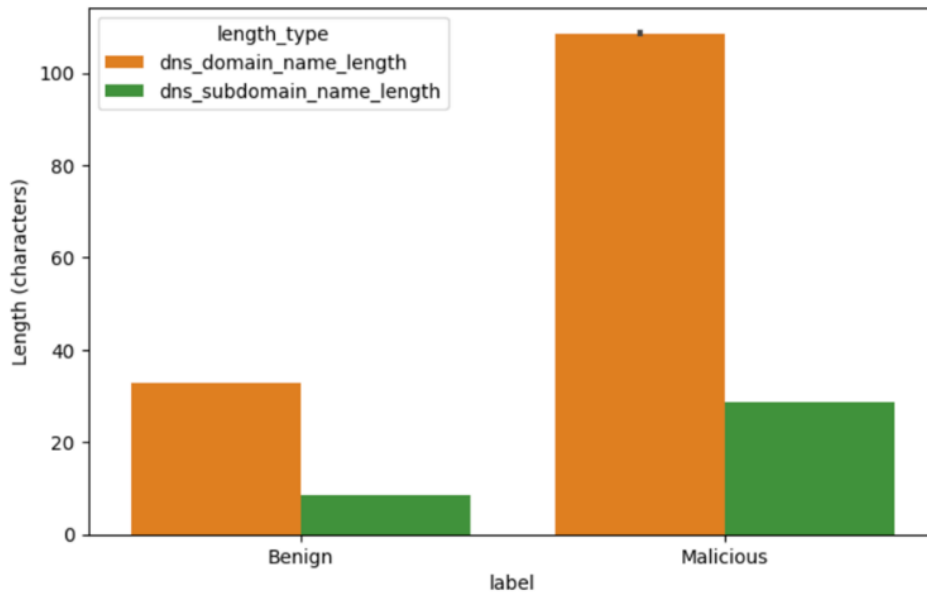


Figure 5: Average DNS Domain Name and Subdomain Lengths in DNS Traffic

Figure 6 shows the occurrence of distinct Time-to-Live (TTL) values in benign and malware DNS traffic. There is a maximum of three distinct TTL values in benign traffic, but some are observed up to seven, which is indicative of the range of typical DNS responses for several queries.

Conversely, malicious DNS traffic always takes on one TTL value, indicating strongly consistent responses, which may be a characteristic of automated data exfiltration or DNS tunnelling methods. It is high contrast, indicating that the count of unique TTL values is a suitable feature to differentiate between regular and exfiltration-induced DNS activity in cloud and enterprise network environments.

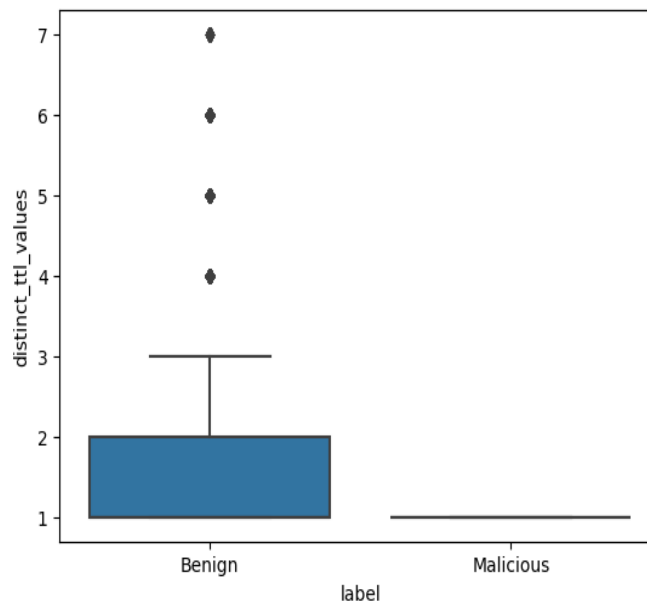


Figure 6: Distribution of Distinct TTL Values in DNS Traffic

Figure 7 illustrates the average Time-To-Live (TTL) values of DNS traffic categories. TTL is a DNS response field that specifies the lifetime of the DNS record, indicating how long a resolver or client will cache a DNS record before it requests an update again. Benign DNS traffic has far larger average TTLs – over 35 times larger than what is seen in malicious DNS traffic. This large disparity indicates that malicious DNS traffic is linked with extremely short-lived records, likely to assist in enabling stealthy data exfiltration or enabling continuous communication with compromised servers without lengthy caching.

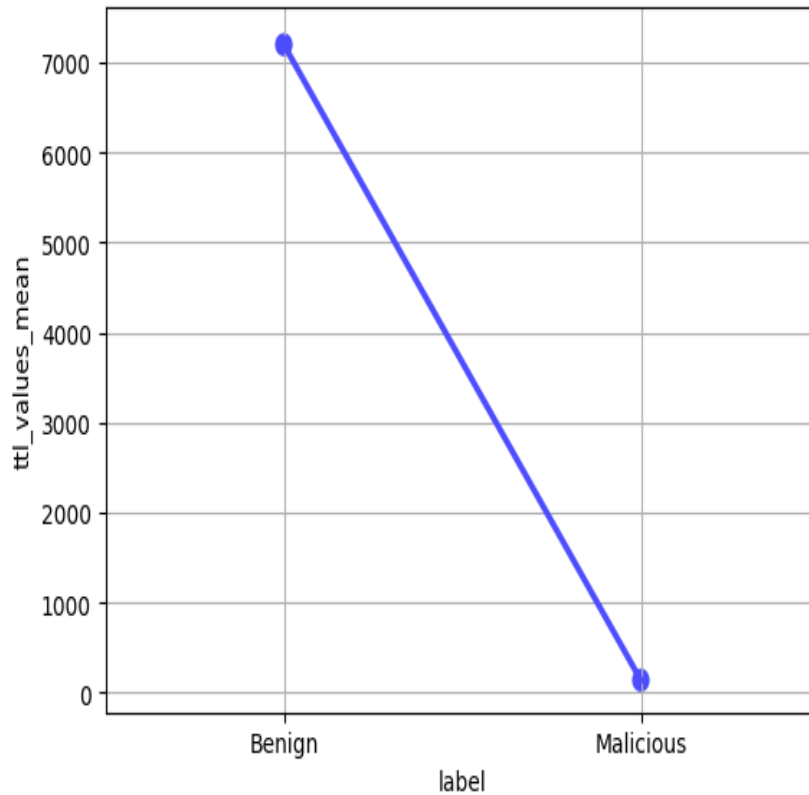


Figure 7: Average TTL Values in DNS Traffic

Figure 8 depicts the average unique characters per DNS queries for benign and malicious traffic. Benign DNS traffic can have fewer unique characters, which is in line with typical domain naming conventions.

On the other hand, malicious DNS traffic exhibits almost twice as many unique characters. This can be attributed to the use of data encoding or obfuscation techniques commonly employed in DNS data exfiltration and tunnelling. Attackers attempt to encrypt the sensitive data into domain names, thereby increasing the character variation beyond their typical occurrence in benign traffic. This variability is what makes the number of unique characters an appropriate feature for detecting DNS data exfiltration attacks.

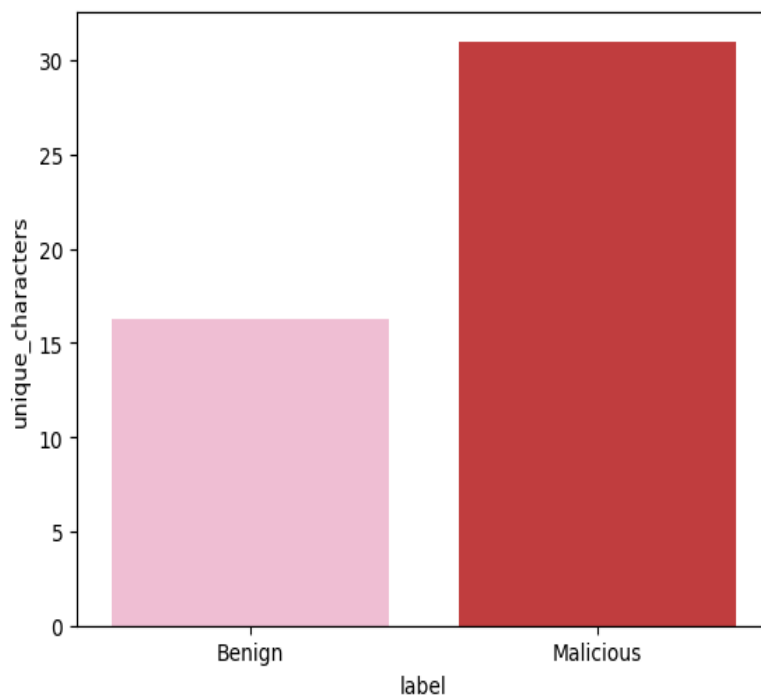


Figure 8: Average Number of Unique Characters in DNS Queries

Figure 9 illustrates the benign and malicious DNS query character length distribution. Benign DNS traffic overall has a less wide range of character lengths, with most of the queries being relatively short and consistent with typical internet traffic of expected and relatively short domain names.

In contrast, ill-behaved traffic encompasses a much wider scope and includes a much longer query string, indicative of the use of longer or more complex domain names utilized to encode sensitive information in attempts to exfiltrate data.

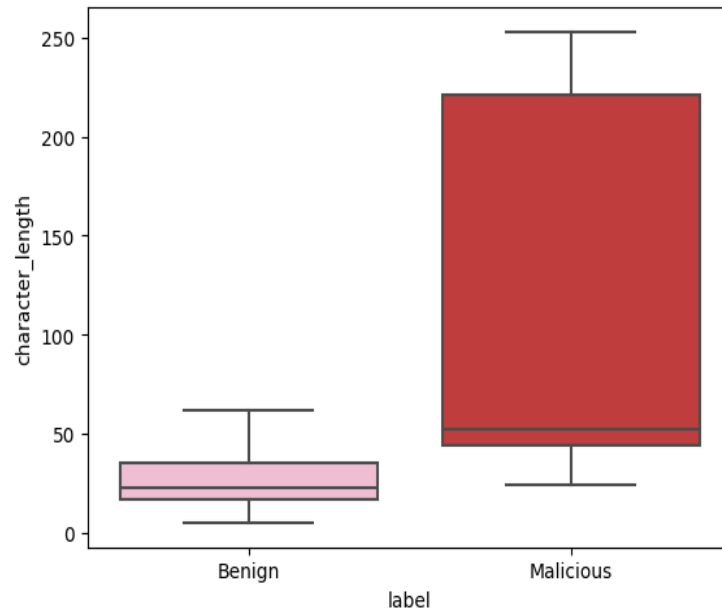


Figure 9: Distribution of Character Lengths in DNS Queries

Figure 10 shows the proportion of the top five DNS top-level domains (TLDs) of benign and malicious DNS traffic. In both instances, the .com TLD is used the most often, as would be expected of its popularity and the vast number of global internet domain registrations. Apart from this shared characteristic, independent tendencies in the employment of TLDs emerge in benign and malicious traffic. Malicious DNS traffic is highly localized to only three TLDs: .com, .tk, and .xyz, indicating that attackers are employing a very small set of domains. Limited options could be a result of cost-effectiveness, convenience in domain acquisition, or the susceptibility to manipulation of less-regulated domain registries to avoid detection.

On the other hand, benign DNS traffic reflects off much larger numbers of TLDs, such as .com, .net, .in, .org, and .br, indicating the widespread use of the internet across geographic locations, organizations, and services.

This stark contrast in TLD distribution underscores its utility as a discriminative feature for distinguishing between normal and malicious DNS traffic.

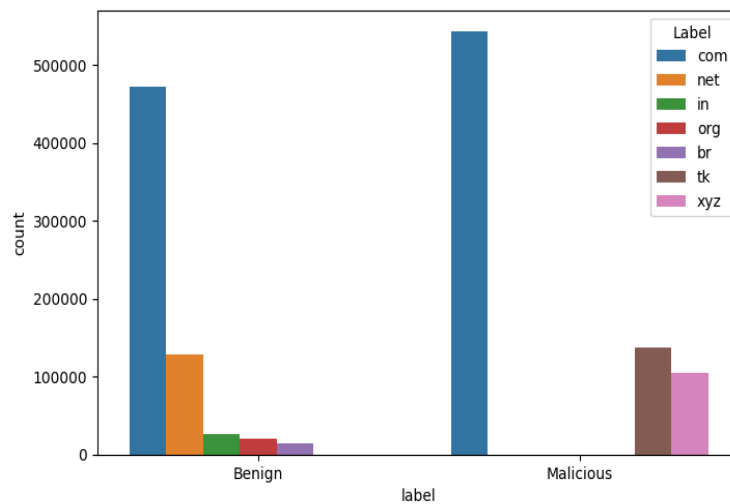


Figure 10: Top Five DNS Top-Level Domains in DNS Traffic

4.5 FEATURE ENGINEERING

This stage entails the modification and transformation of raw features into a more useful set of input variables for the machine learning models.

First, to incorporate temporal information into the dataset, the timestamp column was converted into a standardized datetime format. From the cleaned timestamps, a new column named hour was extracted, capturing the hour of each event. This time-based feature provides additional analytical value by enabling more granular, temporal analysis of DNS traffic.

Second, the character_distribution column, which originally contained string representations of character frequency dictionaries, was transformed from its original textual distribution into quantitative features. From this column, two new features were derived: character_length, representing the total number of characters in each DNS query; and unique_characters, capturing the count of distinct

characters per query. These features are particularly informative because they are interpretable indicators of DNS query structure, which are often higher in malicious DNS traffic due to data encoding or obfuscation techniques.

In addition, several features were engineered from the `ans_resource_record_type` column, which contains the DNS response resource record types associated with each query. The resource record types were mapped from numeric codes to human-readable categories and for each type, new columns were created to record the frequency of their occurrence per query. This transformation converted raw list-based data into numeric, model-ready features that capture the composition of resource records for each DNS flow.

Similarly, a new column was engineered from the `ans_resource_record_class` column, which represents the DNS response record classes for each query and indicates the protocol class of the resource record. In this dataset, all entries consisted of lists containing only the value 1, representing the standard Internet class (IN). Although this value is uniform, it was leveraged in a more meaningful way. To extract useful information, a new column named `ans_resource_record_class_count` was created, representing the number of 1s in each list—that is, the total number of resource records returned per DNS query, offering a compact yet informative indicator of DNS response structure.

Another engineered feature was the `sending_receiving_ratio`, created by dividing the number of sending bytes by the number of receiving bytes for each DNS flow. The ratio measures the proportion or discrepancy of outgoing to incoming communication. In normal, benign DNS traffic, the give-and-take is fairly equal, as small requests are reciprocated by proportionally small responses. However, in malicious data exfiltration attacks, far more data is sent than received, resulting in these larger ratios. By injecting this imbalance into one numeric feature, the model has to learn an interpretable signal of suspicious communication that it can apply to separate malicious traffic from regular DNS traffic.

After deriving the new features, the original columns used to create them were removed. Retaining both the engineered and raw versions would have introduced redundancy without contributing additional information, while also increasing the dimensionality of the dataset unnecessarily. Dropping them not only simplified the dataset to handle but also contributed to noise reduction, where the machine learning models focused on the most interpretable and informative representations of DNS traffic behavior.

Furthermore, the categorical attribute `dns_top_level_domain`, which was the top-level domain in every DNS query, was also converted into a numerical format for machine learning algorithm supportability, which mostly needs it for numeric input. The variable was encoded using the scikit-learn Label Encoder, which maps every unique TLD to a unique numeric value, allowing the dataset to be processed without errors caused by non-numeric input.

In addition, the target variable also required conversion into a numerical format for classification. A new column named `Target` was created by mapping the original labels into binary values: 0 for Benign DNS traffic and 1 for Malicious DNS traffic.

Figure 11 shows the distribution of incidents across the two DNS traffic classes. The incidents are well distributed, which is optimal for training machine learning algorithms because it does not bias towards one class and allows the models to be trained on normal behavior from both benign and malicious traffic. While there are fewer benign instances than malicious, such an imbalance is not significant and does not constitute a severe imbalance. Instead, it reflects natural benign traffic predominance within real-world network environments.

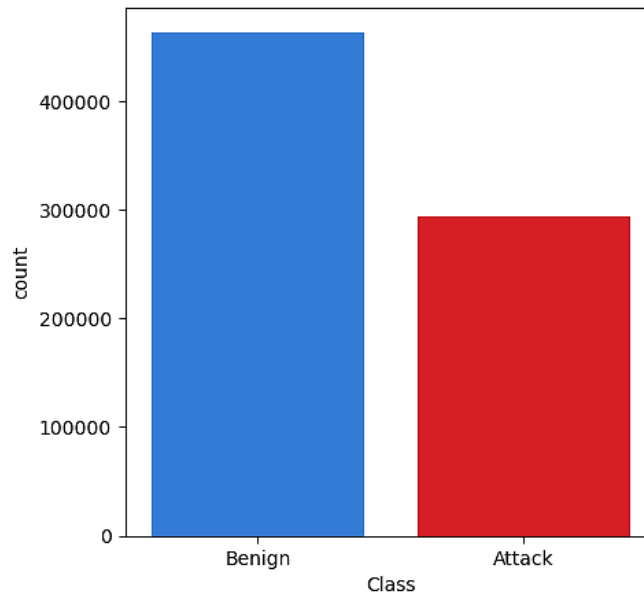


Figure 11: DNS Network Traffic Categories

5. MODEL DEVELOPMENT AND EVALUATION

For replicability and consistency of results, a constant random seed value of 21 was defined before any machine learning process was executed. This ensures the random number generator returns the same sequence of values when repeated execution is performed, thus ensuring consistency in results while training, testing, and using other stochastic algorithms.

The data was split into two sets: 70% for training the machine learning models and 30% for testing how well they perform.

Five machine learning models were used: Random Forest, Decision Tree, Multi-Layer Perceptron, Logistic Regression, and Gaussian Naïve Bayes. The models' performance was then evaluated with the test set and many classification metrics after training. This multi-metric assessment provided strict measurement of each algorithm's capacity to classify benign and malicious DNS traffic. The results of these evaluations are summarized in Table 2, enabling a comparative analysis of the models and the identification of the most effective algorithm for detecting DNS data exfiltration attacks in cloud and enterprise networks.

Table 2: Evaluation Results

Model	Accuracy	Precision	TPR	TNR	F1 Score	AUC ROC
Random Forest	1.000	1.000	1.000	1.000	1.000	1.000
Decision Tree	1.000	1.000	1.000	1.000	1.000	1.000
Multi-Layer Perceptron	0.999	0.999	1.000	0.999	0.999	0.999
Logistic Regression	0.964	0.945	0.985	0.943	0.965	0.964
Gaussian Naïve Bayes	0.822	0.741	0.986	0.661	0.846	0.823

The figure 12 below illustrates the accuracy scores of the different machine learning models. Decision Tree and Random Forest models both achieved 100% accuracy, demonstrating their ability to classify benign and malicious DNS traffic perfectly in this dataset. Multi-Layer Perceptron came close with 99.9% accuracy, confirming its strength in learning intricate feature correlations. Logistic Regression also performed strongly with an accuracy score of 96.4%, while Gaussian Naïve Bayes recorded the lowest accuracy at 82.2%, reflecting the challenges of its independence and distributional assumptions.

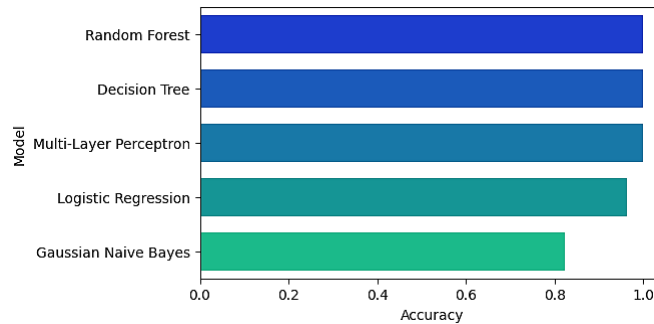


Figure 12: Accuracy Scores of Models

Figure 13 presents the precision scores of the models. Random Forest and Decision Tree achieved the highest precision of 100%, indicating that these models are highly effective in correctly identifying DNS data exfiltration attacks. In contrast, the Gaussian Naïve Bayes model recorded the lowest precision at 74.1%, indicating a relatively higher incidence of false positives compared to the other models.

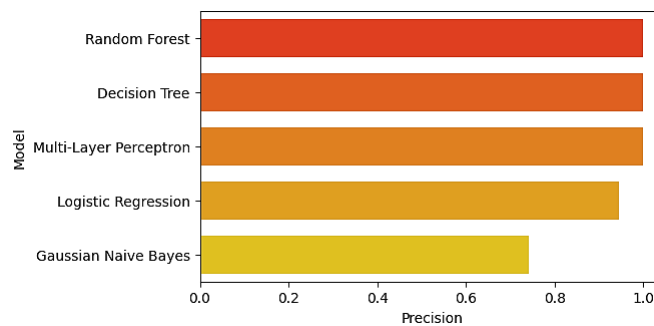
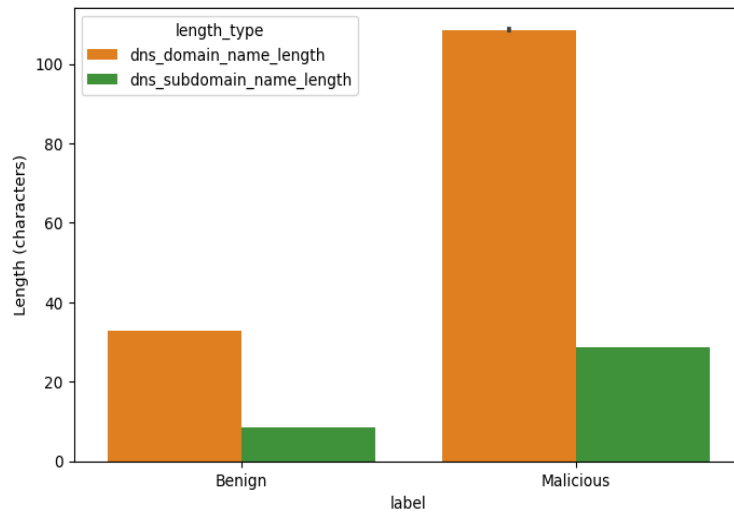


Figure 13: Precision Scores of Models

Figure 14 depicts the true positive rates of the models. This metric measures the proportion of actual DNS data exfiltration attacks that were correctly identified by each model. All the models achieved very high TPRs. Random Forest, Decision Tree, and Multi-Layer

Perceptron reached 100% TPR, while Logistic Regression and Gaussian Naïve Bayes achieved slightly lower T



PR values of 98.5% and 98.6%, respectively, still reflecting strong performance but with a small margin of missed detections compared to the tree-based and neural network models.

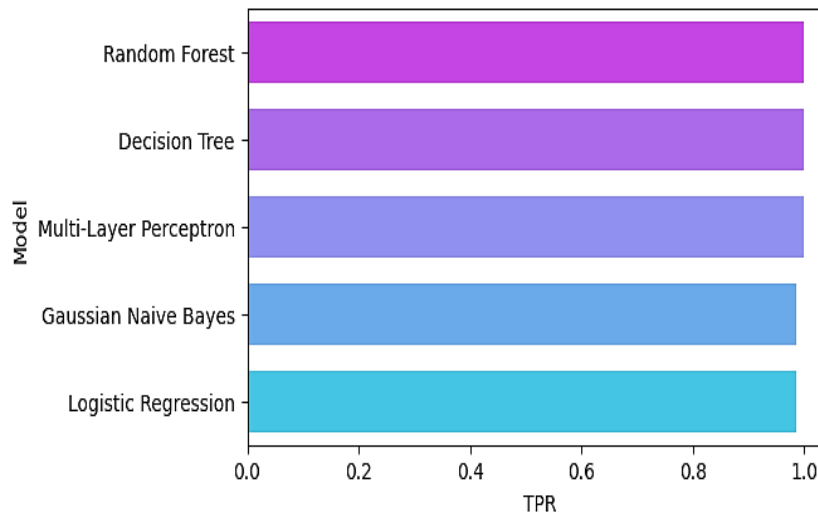


Figure 14: True Positive Rates of Models

Figure 15 illustrates the true negative rates of the models. This metric represents the proportion of actual benign DNS traffic that were correctly identified. Both Random Forest and Decision Tree achieved the highest TNR of 100%. On the other hand, Gaussian Naïve Bayes had the lowest TNR at 66.1%, indicating significant difficulty in correctly identifying benign traffic, and a tendency to mislabel some benign flows as malicious, which could result in unnecessary alerts.

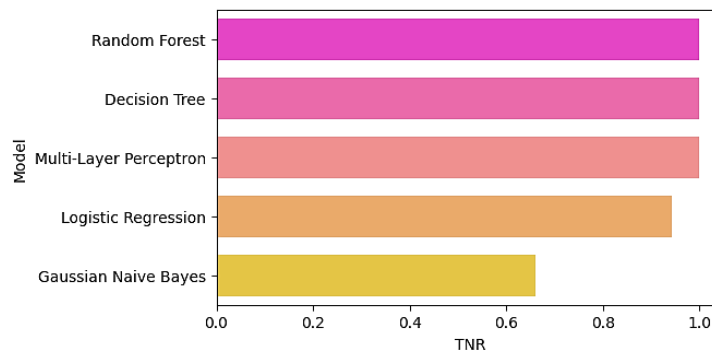


Figure 15: True Negative Rates of Models

Figure 16 presents the f1-scores of models, which balance precision and recall. Random Forest and Decision Tree both achieved 1.000, confirming their excellent ability to balance detecting malicious traffic and avoiding misclassifications, whereas Gaussian Naïve Bayes recorded the lowest F1 Score at 0.846, suggesting reduced consistency in balancing true positive detection with the avoidance of false positives.

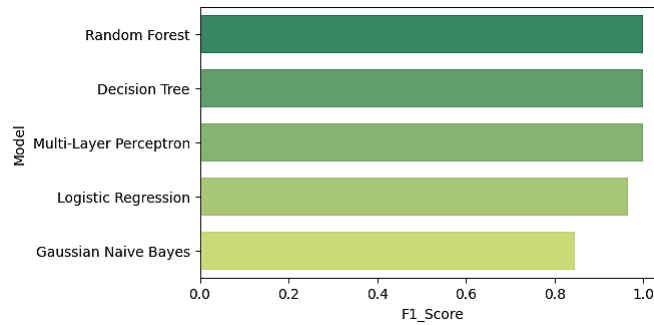


Figure 16: F1-Scores of Models

Figure 17 displays the AUC ROC Curve for all the models. This curve measures a model’s ability to discriminate between two classes across different classification thresholds. The Random Forest, Decision Tree and Multi-Layer Perceptron models each achieved AUC ROC scores of approximately 1.000, indicating that these models almost perfectly differentiate between benign and malicious DNS traffic. Logistic Regression achieved the second-best AUC-ROC of 0.964; although it fell short of the top models, it demonstrated very high discriminative power. Gaussian Naïve Bayes recorded the least AUC-ROC of 0.823, which reflects low discrimination power to differentiate malicious traffic from normal traffic among the models.

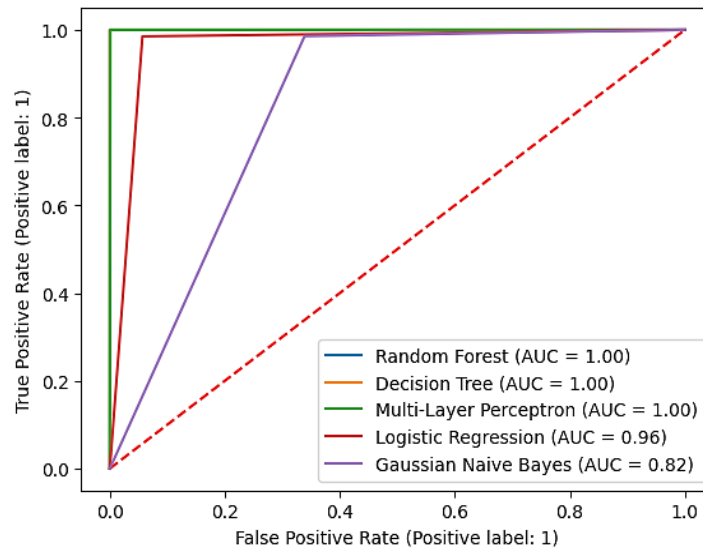
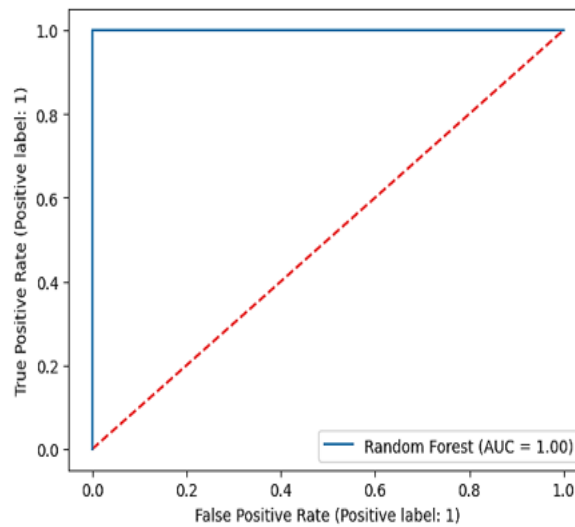
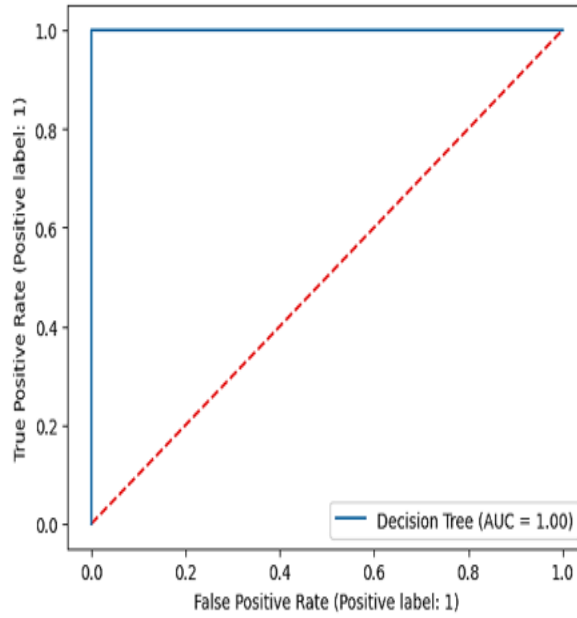


Figure 17: Joint ROC Curve of Models

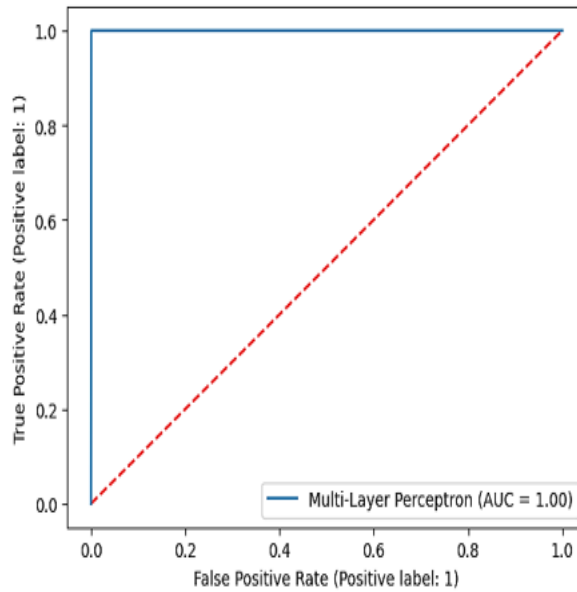
Figures 18 a-e show the individual AUC ROC curve for each machine learning model.



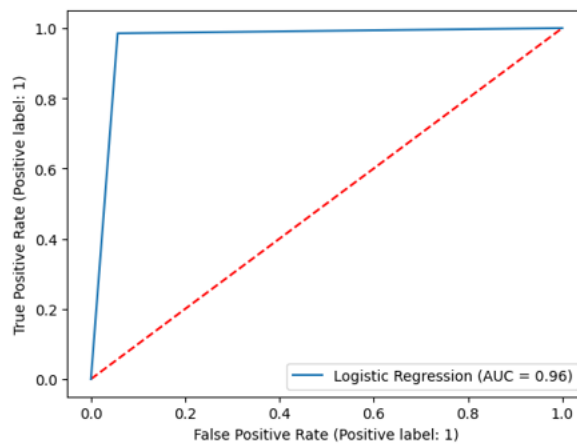
a) Random Forest



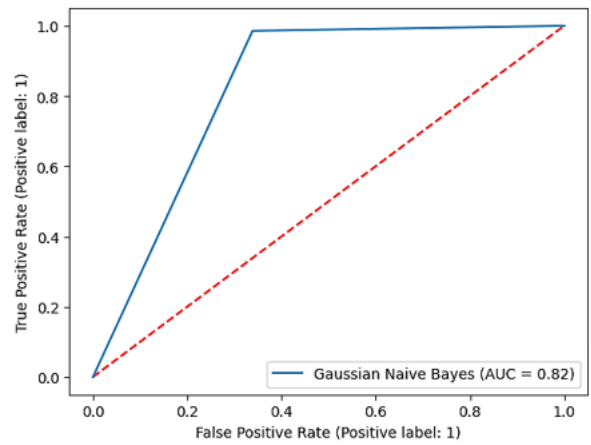
b) Decision Tree



c) Multi-Layer Perceptron



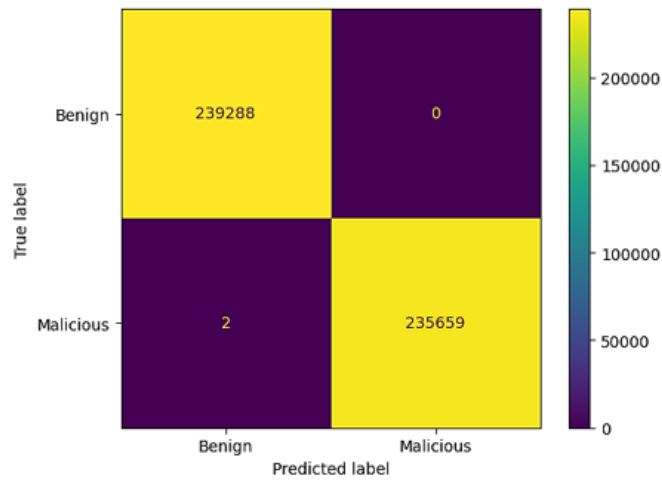
d) Logistic Regression



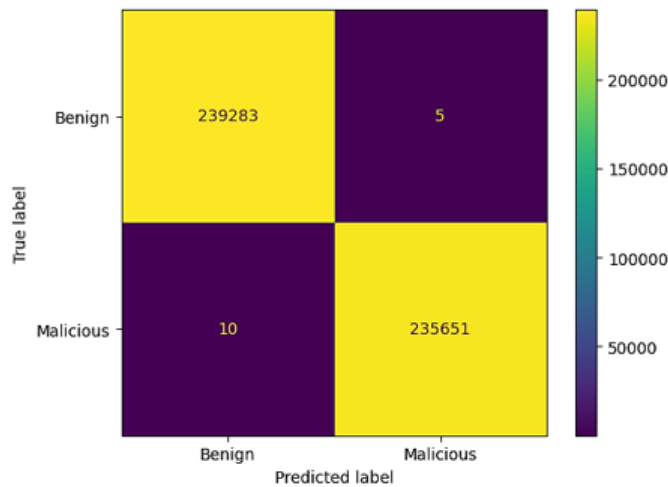
e) Gaussian Naïve Bayes

Figure 18: Individual ROC Curves for Classification Models

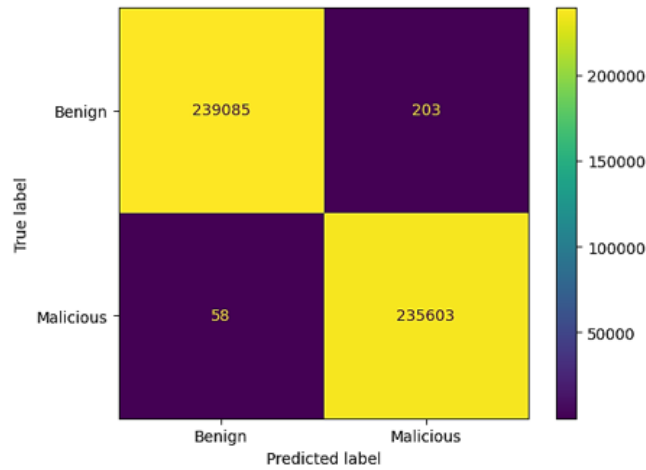
Figures 19 a-e present the confusion matrices of the five models. These represent the predictive outcomes of the models in distinguishing between benign DNS traffic and malicious DNS traffic.



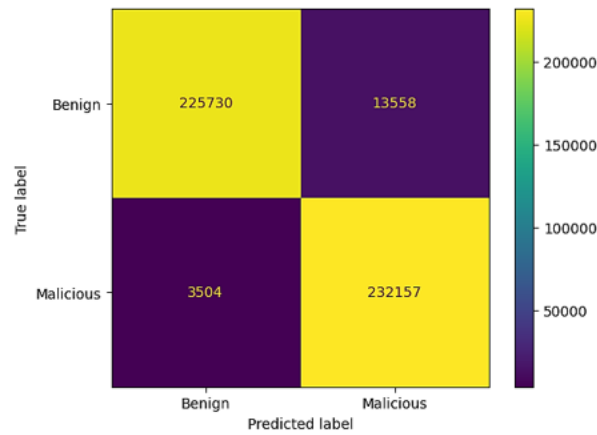
a) Random Forest



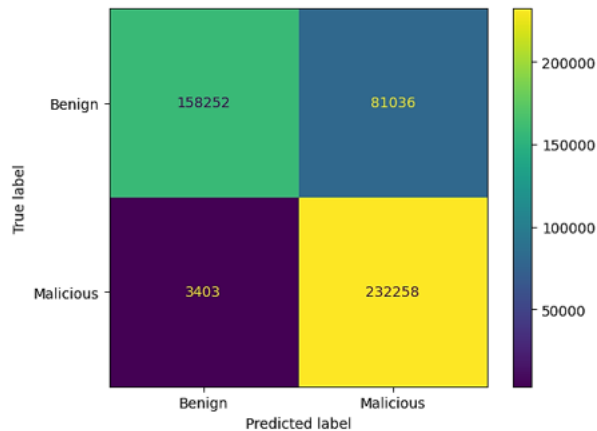
b) Decision Tree



c) Multi-Layer Perceptron



d) Logistic Regression



e) Gaussian Naïve Bayes

Figure 19: Confusion Matrices for Classification Models

The confusion matrices above provide a more granular view of each model’s performance by explicitly showing the number of instances that were correctly and incorrectly classified across both benign and malicious DNS traffic. In this confusion matrix analysis, each model’s effectiveness is determined by the number of instances that are inaccurately classified as either benign network traffic or DNS exfiltration attack.

Random Forest performed exceptionally well, with no benign instances misclassified, though it incorrectly labelled 2 instances of DNS exfiltration attack as benign.

Decision Tree made slightly more errors, misclassifying 5 benign samples as DNS exfiltration attack, and 10 DNS exfiltration attack samples as benign DNS traffic.

The Multi-Layer Perceptron misclassified 203 benign DNS traffic instances as DNS exfiltration attack and 58 DNS exfiltration attacks as benign DNS traffic.

Logistic Regression exhibited larger error counts, misclassifying 13558 normal DNS traffic instances as DNS exfiltration attack, and 3504 instances of DNS exfiltration attack as normal DNS traffic.

The Gaussian Naïve Bayes model had the weakest performance, with a substantial 81036 benign instances incorrectly flagged as DNS exfiltration attacks and 3403 instances of DNS exfiltration attack missed and classified as benign.

The confusion matrices analysis shows that earlier metrics such as accuracy, precision, recall, and F1-score, which suggested near-perfect performance for some models, did not reveal the full picture. This is because aggregate statistics can obscure small but vital misclassifications in large collections. A confusion matrix circumvents this limitation by offering a high-resolution, class-level breakdown of predictions, allowing one to better understand how models perform in real-world applications, where even a few misclassifications can make significant differences.

6. DISCUSSION OF RESULTS

The model performance used for this binary classification task of distinguishing between benign DNS traffic and DNS data exfiltration attack is discussed in detail as follows.

Both Decision Tree and Random Forest models performed seemingly flawlessly, with all the evaluation metrics' scores equal to 1.000: Accuracy, Precision, True Positive Rate, True Negative Rate, F1 Score, and AUC ROC. Both models apparently mark DNS traffic perfectly with no mistakes in marking benign traffic and DNS exfiltration attacks. However, when viewed in the context of the confusion matrix, the situation becomes slightly more complex. Although their evaluation scores are practically perfect, the two models did make a small number of misclassifications, nevertheless. Random Forest correctly classified all benign DNS traffic, i.e., had zero false positives. However, it misclassified 2 instances of DNS exfiltration attack as benign traffic (false negatives). Decision Tree, on the other hand, showed slightly more errors. It had 5 instances of benign DNS traffic misclassified as an attack (false positives) and 10 instances of DNS data exfiltration attacks misclassified as benign (false negatives). This means that while its general performance looks perfect according to summary metrics, it has a somewhat higher tendency to mislabel both benign and malicious traffic than Random Forest.

The Multi-Layer Perceptron performed very well too, with all evaluation measures being very close to 100%, i.e., 99.9% Accuracy, 99.9% Precision, 100% TPR, 99.9% TNR, 0.999 F1 Score, and 0.999 AUC ROC score. This implies that the Multi-Layer Perceptron performed very well in identifying malicious DNS traffic. The model's good F1 score also indicates that it has a good trade-off between correctly classifying DNS exfiltration attacks and incorrectly classifying benign flows. Its AUC-ROC value also suggests exceedingly high separability of the two classes. However, the confusion matrix has a more compelling story to tell, in that 203 benign samples were incorrectly classified as malicious, and 58 malicious samples were incorrectly classified as benign. Although the figures are numerically small compared to the dataset size, they do suggest that the Multi-Layer Perceptron still generates false positives and false negatives.

The logistic regression model performed relatively worse than tree models and neural models, achieving 96.4% Accuracy, 94.5% Precision, 98.5% True Positive Rate, 94.3% True Negative Rate, 0.965 F1 Score, and 0.964 AUC ROC score. Although results are good overall as well, Logistic Regression had the worst TPR among all the models, meaning it was less efficient at correctly identifying DNS data exfiltration attacks than others. This lack is also reflected in the confusion matrix, where 13,558 benign DNS flows were incorrectly labeled as malicious (false positives) and 3,504 malicious flows were incorrectly labeled as benign (false negatives). These inaccuracies demonstrate that while Logistic Regression learns overall trends from the data, its linearity hinders it from capturing complex, non-linear patterns that are prevalent in DNS traffic. Thus, it generates far more errors than the more advanced models and is thus less suitable for highly sensitive detection issues where one needs to minimize both false negatives and false positives.

The lowest performance among the models was achieved by the Gaussian Naïve Bayes model, which was also the lowest in almost all measures—82.2% Accuracy, 74.1% Precision, 66.1% True Negative Rate, 84.6% F1 Score, and 82.3% AUC ROC. Only in its True Positive Rate was it not the lowest, at a very high 98.6%, which shows that the model, in general, was very effective at identifying malicious DNS traffic. However, this came at the expense of a poor TNR, showing difficulty in correctly identifying benign traffic. Its precision was also the lowest, indicating a higher rate of false positives. The confusion matrix further emphasizes its shortcomings: Gaussian Naïve Bayes misclassified 81,036 benign DNS traffic instances as malicious (false positives) and 3,403 malicious instances as benign (false negatives) – the largest number of misclassifications across all models. These findings illustrate the model's limitation due to the strong independence assumption, as it compromises the intricate dependencies in DNS traffic features and produces common errors and lower dependability for real applications in DNS data exfiltration detection.

These findings are a valuable reminder: aggregate measures can overestimate model performance at times, especially when there are few misclassifications compared to the size of the dataset. Here, scores were high because predictions were generally accurate, so the comparatively small number of failures was trivial in overall measures. Nevertheless, such aggregated metrics might hide significant shortcomings, especially in security contexts where even a small number of false negatives might allow attacks to slip undetected through or too many false positives overwhelm analysts with useless alerts. Therefore, it is also imperative to complement aggregated metrics by offering a detailed examination of the confusion matrices, which show the precise number of benign and malicious DNS flows correctly and incorrectly classified. This approach provides a more transparent and reliable assessment of each model's practical utility. Based on the findings, the Random Forest model emerge as the best-performing model for the task of detecting DNS data exfiltration attack. Although Random Forest and Decision Tree both produced supposedly ideal evaluation metrics, the confusion matrix revealed that Random Forest misclassified fewer instances. That is, Random Forest incorrectly labeled only two malicious DNS traffic as benign, and correctly marked all benign DNS traffic, i.e., it generated zero false positives. Decision Tree, though, not only misclassified more malicious ones but also created false positives, reducing its effective reliability. Therefore, in contrast to all the other models, Random Forest performed the best in all the measures, better capturing the nonlinear and complex patterns of DNS traffic with much fewer misclassifications. Hence, the application of the Random Forest model in real-time DNS monitoring systems for detecting DNS data exfiltration attacks in cloud and enterprise networks is recommended. With the assistance of this model, network administrators can

tighten security protocols, thereby effectively mitigating the threat posed by DNS data exfiltration attacks and securing networks more effectively against novel vulnerabilities.

7. PRACTICAL IMPLICATIONS OF MODEL PERFORMANCE

While DNS serves a critical purpose in today's enterprise and cloud networks by resolving IP addresses and enabling persistent communication between dispersed systems, its exposure also renders it a tempting target for attackers involved in exfiltrating sensitive information. In the real world, information is split into small pieces, encrypted as DNS requests, and transmitted over to attacker domains. Since DNS traffic is not typically filtered, such exfiltration activity often evades firewalls and intrusion detection systems. This presents a large cloud-reliant organization and enterprise infrastructure security blind spot.

When applying lightweight machine learning models to this problem, the results of this research show that Random Forest stands out as the most effective option. In real-world terms, this means that if an enterprise were to implement Random Forest in its DNS monitoring pipeline, it would be able to detect nearly all exfiltration attempts without generating unnecessary alerts for normal traffic. The confusion matrix results confirm that Random Forest introduced zero false positives and only two false negatives out of more than 1.5 million flows. For a security operations center (SOC), this translates into a significant reduction in "alert fatigue," ensuring that analysts are only alerted when there is a real risk, rather than chasing down false alarms.

Meanwhile, while Decision Tree was flawless on test metrics, its behavior in the real world is a different story. Misclassifying malicious as well as benign traffic would make it such that an enterprise using Decision Tree would have risk doubled: the attacks would sail past defenses unnoticed, and innocent requests would be hit with false positives, consuming precious analyst time. For high-bandwidth environments like cloud providers or large hospitals that use IoMT devices, even minimal misclassified DNS flows can open doors for severe breaches. This makes Decision Tree less trustworthy for production deployment compared to Random Forest.

The Multi-Layer Perceptron (MLP) showed excellent performance as well, but its higher rate of false positives raises concerns for practical use. In a real deployment, 203 benign flows being flagged as malicious may sound negligible, but when scaled to millions of queries per day in a busy enterprise, this could snowball into thousands of false alerts per week. While MLP has the capability to capture complex traffic patterns, its cost in operational overhead makes it less suitable as a lightweight model, especially in environments where response speed and efficiency are critical.

Logistic Regression, though efficient, falls short when deployed in real DNS traffic monitoring. Its inability to handle complex non-linearities in DNS patterns means attackers using slightly modified or obfuscated exfiltration methods would likely evade detection. With more than 13,000 benign queries flagged as malicious in this study, an enterprise SOC would quickly become overwhelmed. This makes Logistic Regression unsuitable for real-time deployment in cloud and enterprise networks where minimizing both false positives and false negatives is crucial.

Gaussian Naïve Bayes performed the weakest and demonstrated why simplifying assumptions can be dangerous in practice. By misclassifying over 81,000 benign queries as malicious, the model would generate an unmanageable volume of false alerts in a real network. Simultaneously, it still permitted over 3,000 harmful flows to slip through undetected. In a case of cloud hosting, where businesses might be held accountable for safeguarding customers' data, this degree of inaccuracy could prove disastrous. Clearly, this model should not be considered for practical deployment in DNS exfiltration detection.

The practical implications of these results are straightforward: Random Forest produces the optimal tradeoff between detection capability and operational robustness for DNS data exfiltration detection within enterprise and cloud environments. Its ensemble nature allows it to learn subtle patterns of attack while keeping computational costs low, thereby making it even feasible for hardware-constrained machines, such as edge routers or inexpensive intrusion detection appliances. Companies that use Random Forest in their DNS monitoring pipelines can reliably detect and disrupt attempts at exfiltration without overwhelming analysts with noise, thereby improving their overall security position against emerging threats.

8. CONCLUSION

This work demonstrates the practicality of employing light machine learning models in detecting DNS data exfiltration attacks on cloud and enterprise network environments. By comparing Random Forest, Decision Tree, Multi-Layer Perceptron, Logistic Regression, and Gaussian Naïve Bayes, the research established that although a majority of the models yielded high performance values, Random Forest was consistently the more preferable option in terms of real-world deployability. Its potential to generate near-perfect detection with minimal false alarms and no false negatives makes it very reliable in real-time DNS traffic inspection. These results highlight the importance of combining computing power with high-detection capabilities in modern security solutions. Random Forest, in this case, is an ideal model for strengthening defenses against stealth exfiltration methods that exploit the openness of DNS in cloud and enterprise networks.

9. LIMITATIONS

Although the results are overwhelmingly strong, this research is not without fault. Initially, the evaluation was done on a specific dataset, which may not fully reflect the diversity of traffic behaviors observed in different industries or regions. Relying on supervised learning methods also translates to reliance on the representativeness and quality of annotated data, whose biases may be injected if attack samples are not representative. In addition, the experiments failed to include adversarial attack contexts, i.e., tampered DNS tunneling attacks that are engineered to get around machine learning frameworks. Lastly, whereas the models were tested on accuracy and performance, their scalability in extremely high-throughput contexts was not tested through actual field deployments.

10 FUTURE WORK

Based on these findings, future work needs to move in several directions. First, applying these models to different datasets, such as real-time traffic from other cloud and enterprise environments, would lead to better generalization. Second, including adversarial machine learning attacks could make it more resilient to attackers who specifically modify their behavior to evade detection. Third, combining models such as Random Forest with deep learning models like LSTM or ensemble-based neural networks may be explored to enhance adaptability against adaptive DNS exfiltration techniques. Lastly, deploying the resultant models into production-level intrusion detection systems and validating them in real-time would yield indicative proof concerning scalability, latency, and resource consumption, thereby closing the research-applied cybersecurity gap.

REFERENCES

- [1] S. Mahdavi et al., "Lightweight Hybrid Detection of Data Exfiltration using DNS based on Machine Learning," 2021 the 11th International Conference on Communication and Network Security, Dec. 2021, doi: <https://doi.org/10.1145/3507509.3507520>.
- [2] A. Nadler, A. Aminov, and A. Shabtai, "Detection of malicious and low throughput data exfiltration over the DNS protocol," *Computers & Security*, vol. 80, pp. 36–53, Jan. 2019, doi: <https://doi.org/10.1016/j.cose.2018.09.006>.
- [3] M. Zhan, Y. Li, G. Yu, B. Li, and W. Wang, "Detecting DNS over HTTPS based data exfiltration," *Computer Networks*, vol. 209, p. 108919, May 2022, doi: <https://doi.org/10.1016/j.comnet.2022.108919>.
- [4] Orieb Abualghanam, Hadeel Alazzam, Basima Elshqeir, M. Qatawneh, and Mohammed Amin Almaiah, "Real-Time Detection System for Data Exfiltration over DNS Tunneling Using Machine Learning," *Electronics*, vol. 12, no. 6, pp. 1467–1467, Mar. 2023, doi: <https://doi.org/10.3390/electronics12061467>.
- [5] L. Salat, M. J. Davis, and N. Khan, "DNS Tunnelling, Exfiltration and Detection over Cloud Environments," *Sensors*, vol. 23, no. 5, pp. 2760–2760, Mar. 2023, doi: <https://doi.org/10.3390/s23052760>.
- [6] S. Chen, B. Lang, H. Liu, D. Li, and C. Gao, "DNS covert channel detection method using the LSTM model," *Computers & Security*, vol. 104, p. 102095, May 2021, doi: <https://doi.org/10.1016/j.cose.2020.102095>.
- [7] J. Zhang, L. Yang, S. Yu, and J. Ma, "A DNS Tunneling Detection Method Based on Deep Learning Models to Prevent Data Exfiltration," *Network and System Security*, pp. 520–535, 2019, doi: https://doi.org/10.1007/978-3-030-36938-5_32.
- [8] E. Joback, L. Shing, K. Alperin, S. R. Gomez, S. Jorgensen, and G. Elkin, "A Statistical Approach to Detecting Low-Throughput Exfiltration through the Domain Name System Protocol," *DSPACE@MIT (Massachusetts Institute of Technology)*, Dec. 2020, doi: <https://doi.org/10.1145/3477997.3478007>.
- [9] M. Hamidouche, B. F. Demissie, and B. Cherif, "Real-time Threat Detection Strategies for Resource-constrained Devices," *arXiv.org*, 2024. <https://arxiv.org/abs/2403.15078> (accessed Sep. 27, 2025).
- [10] D. Petrov, P. Ruffing, S. Zillien, and S. Wendzel, "Domainator: Detecting and Identifying DNS-Tunneling Malware Using Metadata Sequences," *arXiv.org*, 2025. <https://arxiv.org/abs/2505.22220> (accessed Sep. 27, 2025).
- [11] F. Palau, C. Catania, J. Guerra, S. Garcia, and M. Rigaki, "DNS Tunneling: A Deep Learning based Lexicographical Detection Approach," *arXiv:2006.06122 [cs]*, Jun. 2020, Accessed: Jun. 17, 2020. [Online]. Available: <https://arxiv.org/abs/2006.06122>
- [12] K. Jerabek, K. Hynek, O. Rysavy, and I. Burgetova, "DNS Over HTTPS Detection Using Standard Flow Telemetry," *IEEE Access*, vol. 11, pp. 50000–50012, Jan. 2023, doi: <https://doi.org/10.1109/access.2023.3275744>.
- [13] S. Adiwali, B. Rajendran, P. S. D., and S. D. Sudarsan, "DNS Intrusion Detection (DID)—A SNORT-based solution to detect DNS amplification and DNS tunneling attacks," *Franklin Open*, p. 100010, Feb. 2023, doi: <https://doi.org/10.1016/j.fraope.2023.100010>.
- [14] M. A. ALTUNCU et al., "Deep Learning Based DNS Tunneling Detection and Blocking System," *Advances in Electrical and Computer Engineering*, vol. 21, no. 3, pp. 39–48, 2021, doi: <https://doi.org/10.4316/aeece.2021.03005>.
- [15] W. K. Jung and B. I. Kwak, "MTL-DoHTA: Multi-Task Learning-Based DNS over HTTPS Traffic Analysis for Enhanced Network Security," *Sensors*, vol. 25, no. 4, pp. 993–993, Feb. 2025, doi: <https://doi.org/10.3390/s25040993>.
- [16] Q. Abu Al-Haija, M. Alohaly, and A. Odeh, "A Lightweight Double-Stage Scheme to Identify Malicious DNS over HTTPS Traffic Using a Hybrid Learning Approach," *Sensors*, vol. 23, no. 7, p. 3489, Jan. 2023, doi: <https://doi.org/10.3390/s23073489>.
- [17] D. Willems, K. Kohls, B. van der Kamp, and H. Vranken, "Data Exfiltration Detection on Network Metadata with Autoencoders," *Electronics*, vol. 12, no. 12, p. 2584, Jan. 2023, doi: <https://doi.org/10.3390/electronics12122584>.
- [18] E. Açıkgozöglü, "COMPARISON OF MACHINE LEARNING ALGORITHMS FOR DETECTION OF DATA EXFILTRATION OVER DNS," *Yalvaç Akademi Dergisi*, Aug. 2024, doi: <https://doi.org/10.57120/yalvac.1507402>.
- [19] C. Qi, X. Chen, C. Xu, J. Shi, and P. Liu, "A Bigram based Real Time DNS Tunnel Detection Approach," *Procedia Computer Science*, vol. 17, pp. 852–860, Jan. 2013, doi: <https://doi.org/10.1016/j.procs.2013.05.109>.
- [20] A. Das, M.-Y. Shen, M. Shashanka, and J. Wang, "Detection of Exfiltration and Tunneling over DNS," 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Dec. 2017, doi: <https://doi.org/10.1109/icmla.2017.00-71>.
- [21] Mouhammd Alkasasbeh and M. Almseidin, "Machine Learning Techniques for Accurately Detecting the DNS Tunneling," *Lecture notes in networks and systems*, pp. 352–364, Jan. 2023, doi: https://doi.org/10.1007/978-3-031-37717-4_24.
- [22] H. Ghani, "CIC-Bell-DNS-EXF-2021," *Kaggle.com*, 2021. <https://www.kaggle.com/datasets/humeral1/cicbelldnsexf2021> (accessed Sep. 28, 2025).
- [23] B. Sabir, F. Ullah, M. A. Babar, and R. Gaire, "Machine Learning for Detecting Data Exfiltration," *ACM Computing Surveys*, vol. 54, no. 3, pp. 1–47, Jun. 2021, doi: <https://doi.org/10.1145/3442181>.

- [24] Kristijan Žiža, Predrag Tadić, and Pavle Vuletić, “DNS exfiltration detection in the presence of adversarial attacks and modified exfiltrator behaviour,” *International Journal of Information Security*, vol. 22, no. 6, pp. 1865–1880, Jul. 2023, doi: <https://doi.org/10.1007/s10207-023-00723-w>.
- [25] N. Bykov and Y. Chernyshov, “Detecting DNS Tunnels Using Machine Learning,” pp. 92–94, May 2024, doi: <https://doi.org/10.1109/usbereit61901.2024.10584043>
- [26] R. Tang et al., “A Practical Machine Learning-Based Framework to Detect DNS Covert Communication in Enterprises,” *Security and Privacy in Communication Networks*, pp. 1–21, Jan. 2020, doi: https://doi.org/10.1007/978-3-030-63095-9_1.
- [27] Matee Witawasiri and Pongsarun Boonyopakorn, “Evaluating Machine Learning Techniques for DNS Traffic Classification Using Three Publicly Available Datasets,” pp. 192–197, Aug. 2024, doi: <https://doi.org/10.1109/ri2c64012.2024.10784355>.
- [28] X. Cai, H. Zhang, C. M. Ahmed, and H. Koide, “Detecting Advanced Persistent Threat Exfiltration with Ensemble Deep Learning Tree Models and Novel Detection Metrics,” *IEEE Access*, pp. 1–1, Jan. 2025, doi: <https://doi.org/10.1109/access.2025.3567772>.
- [29] L. Mahmoud, S. Pillai, and S. Gangopadhyay, “Interpretable Ensemble Learning Model for Enabling an IDS to Detect DNS Attacks,” *Communications in computer and information science*, pp. 85–101, Jan. 2025, doi: https://doi.org/10.1007/978-3-031-82153-0_7.
- [30] Daumel, “DNS exfiltration dataset,” Kaggle.com, 2016. <https://www.kaggle.com/datasets/daumel/dns-tunneling-dataset> (accessed Sep. 28, 2025).