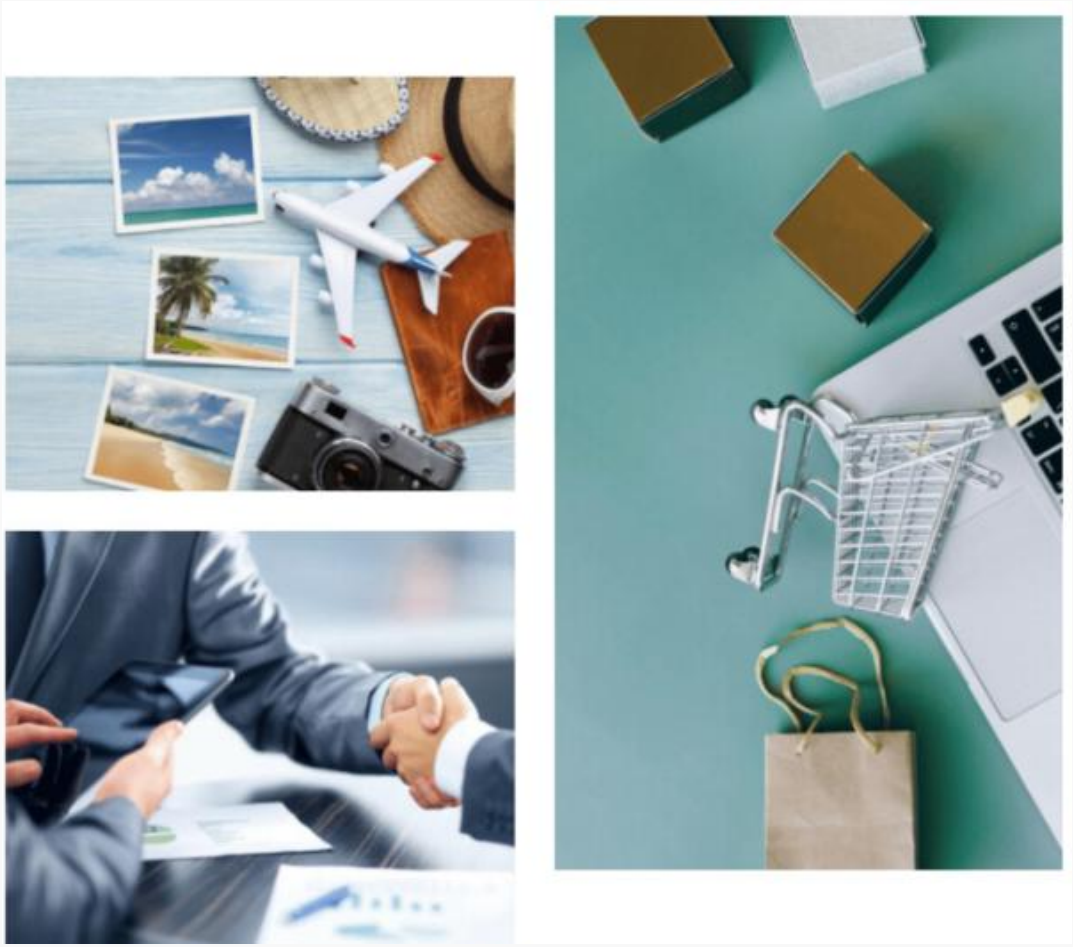


Cyber Security Case Study

Nikhil Kumar M

Problem Statement

Current Situation:

- The organization has access to multiple network traffic datasets captured across different working days and time windows.
- These datasets include normal traffic as well as labeled attack scenarios such as DDoS, Port Scan, Web Attacks, and Infiltration.
- Each dataset is stored separately by day and attack type, making it difficult to gain a consolidated view of network behavior and security threats.
- Currently, there is no automated system to intelligently analyze this data and detect intrusions.

Pain Points:

- Security teams do not have an intelligent, data-driven mechanism to automatically distinguish between benign and malicious traffic.
- Identifying attack characteristics, understanding traffic anomalies, and evaluating attack frequency across time periods is time-consuming and manual. This limits the ability to proactively detect and respond to cyber threats.

Problem Statement

Impact on Business:

- The lack of an automated intrusion detection capability increases the risk of undetected cyber attacks, delayed incident response, potential data breaches, system downtime, and financial loss.
- Inefficient threat detection can also lead to higher operational and security costs.
- This also impacts regulatory compliance and damages organizational reputation due to weak security posture.

Who is Affected:

- This impacts cybersecurity analysts, SOC teams, IT operations, risk management teams, and business leadership responsible for ensuring system availability and data security.

Business Objectives

The objective of this initiative is to analyze network traffic data to identify, understand, and monitor cybersecurity threats.

Key Objectives Include:

- Differentiate normal traffic from malicious traffic
- Analyze characteristics of various cyber attacks such as DDoS, Port Scan, Web Attacks, and Infiltration
- Identify traffic patterns and anomalies across different days and working hours
- Provide actionable insights for strengthening network security controls
- Design and develop a machine learning–based intrusion detection algorithm that can classify network traffic as benign or malicious based on extracted traffic features
- Evaluate the performance of the intrusion detection model using appropriate metrics to support future deployment decisions

Scope & Out of Scope

In Scope:

- Analysis of normal and attack traffic across all provided datasets
- Comparative analysis of different attack types
- Data preprocessing and feature analysis of network traffic datasets
- Classification ML Model of benign and malicious traffic
- Development and evaluation of machine learning models for intrusion detection
- Creation of security monitoring dashboards and KPIs

Out of Scope:

- Real-time deployment of the intrusion detection system
- Automated attack prevention or response mechanisms
- Integration with live network infrastructure

Stakeholders

- Cybersecurity and SOC Teams
- IT Infrastructure and Network Teams
- Risk and Compliance Teams
- Business Leadership
- Data Analytics and Data Science Teams

High – Level Requirements

- Ability to ingest and process large-scale network traffic data
- Feature extraction and selection suitable for machine learning models
- Classification of traffic into benign and attack categories
- Model evaluation using standard machine learning metrics
- Comparative analysis of attack detection performance
- Visualization of attack patterns and model results

Success Metrics

- Model accuracy
- Precision, recall, and F1-score for attack detection
- False positive and false negative rates
- Detection rate for each attack type
- Reduction in unidentified or misclassified traffic

Assumptions & Constraints

Assumptions:

- Network traffic data is correctly labeled for benign and attack classes
- Extracted features are relevant for intrusion detection
- Historical data is representative of real-world network behavior
- Sufficient data volume exists for training and evaluation

Constraints:

- Large dataset size may increase processing and training time
- Data is static and offline in nature
- Limited contextual information beyond network flow features
- Model performance is limited to the quality of available data

Dataset Overview

The intrusion detection model will be developed using the following network traffic datasets:

- Monday-WorkingHours.pcap_ISCX
- Tuesday-WorkingHours.pcap_ISCX
- Wednesday-workingHours.pcap_ISCX
- Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX
- Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX
- Friday-WorkingHours-Morning.pcap_ISCXFriday-WorkingHours-Afternoon-PortScan.pcap_ISCX
- Friday-WorkingHours-Afternoon-DDos.pcap_ISCX

Data Dictionary

Column Name	Description	Data Type
Destination Port	Port number of the destination service	Integer
Flow Duration	Total time duration of the flow in microseconds	Integer
Total Fwd Packets	Total packets sent from source to destination	Integer
Total Backward Packets	Total packets sent from destination to source	Integer
Total Length of Fwd Packets	Total bytes sent in forward direction	Integer
Total Length of Bwd Packets	Total bytes sent in backward direction	Integer
Fwd Packet Length Max	Maximum size of forward packets	Integer
Fwd Packet Length Min	Minimum size of forward packets	Integer
Fwd Packet Length Mean	Average size of forward packets	Float
Fwd Packet Length Std	Standard deviation of forward packet sizes	Float
Bwd Packet Length Max	Maximum size of backward packets	Integer
Bwd Packet Length Min	Minimum size of backward packets	Integer
Bwd Packet Length Mean	Average size of backward packets	Float
Bwd Packet Length Std	Standard deviation of backward packet sizes	Float
Flow Bytes/s	Bytes transferred per second	Float
Flow Packets/s	Packets transferred per second	Float

Information Schema of Data

Column Name	Description	Data Type
Flow IAT Mean	Mean inter arrival time between packets	Float
Flow IAT Std	Standard deviation of packet inter arrival time	Float
Flow IAT Max	Maximum inter arrival time	Integer
Flow IAT Min	Minimum inter arrival time	Integer
Fwd IAT Total	Total inter arrival time for forward packets	Integer
Fwd IAT Mean	Mean inter arrival time of forward packets	Float
Fwd IAT Std	Standard deviation of forward inter arrival time	Float
Fwd IAT Max	Maximum inter arrival time forward	Integer
Fwd IAT Min	Minimum inter arrival time forward	Integer
Bwd IAT Total	Total inter arrival time for backward packets	Integer
Bwd IAT Mean	Mean inter arrival time of backward packets	Float
Bwd IAT Std	Standard deviation of backward inter arrival time	Float
Bwd IAT Max	Maximum inter arrival time backward	Integer
Bwd IAT Min	Minimum inter arrival time backward	Integer
Fwd PSH Flags	PSH flag in forward packets	Integer
Bwd PSH Flags	PSH flag in backward packets	Integer

Information Schema of Data

Column Name	Description	Data Type
Fwd URG Flags	URG flags in forward packets	Integer
Bwd URG Flags	URG flags in backward packets	Integer
Fwd Header Length	Total header bytes in forward direction	Integer
Bwd Header Length	Total header bytes in backward direction	Integer
Fwd Packets/s	Forward packets per second	Float
Bwd Packets/s	Backward packets per second	Float
Min Packet Length	Minimum packet size in the flow	Integer
Max Packet Length	Maximum packet size in the flow	Integer
Packet Length Mean	Average packet size in the flow	Float
Packet Length Std	Standard deviation of packet size	Float
Packet Length Variance	Variance of packet size	Float
FIN Flag Count	Count of FIN TCP flags	Integer
SYN Flag Count	Count of SYN TCP flags	Integer
RST Flag Count	Count of RST TCP flags	Integer
PSH Flag Count	Count of PSH TCP flags	Integer
ACK Flag Count	Count of ACK TCP flags	Integer

Information Schema of Data

Column Name	Description	Data Type
URG Flag Count	Count of URG TCP flags	Integer
CWE Flag Count	Congestion Window Reduced flag count	Integer
ECE Flag Count	Explicit Congestion Notification Echo flag count	Integer
Down/Up Ratio	Ratio of backward to forward packets	Integer
Average Packet Size	Average size of packets in flow	Float
Avg Fwd Segment Size	Average size of forward TCP segments	Float
Avg Bwd Segment Size	Average size of backward TCP segments	Float
Fwd Header Length	Total length of all packet headers in the forward direction	Integer
Fwd Avg Bytes/Bulk	Average bytes per forward bulk transfer	Integer
Fwd Avg Packets/Bulk	Average packets per forward bulk transfer	Integer
Fwd Avg Bulk Rate	Forward bulk transfer rate	Integer
Bwd Avg Bytes/Bulk	Average bytes per backward bulk transfer	Integer
Bwd Avg Packets/Bulk	Average packets per backward bulk transfer	Integer
Bwd Avg Bulk Rate	Backward bulk transfer rate	Integer
Subflow Fwd Packets	Number of forward packets in subflows	Integer
Subflow Fwd Bytes	Forward bytes in subflows	Integer

Information Schema of Data

Column Name	Description	Data Type
Subflow Bwd Packets	Number of backward packets in subflows	Integer
Subflow Bwd Bytes	Backward bytes in subflows	Integer
Init_Win_bytes_forward	Initial TCP window size forward	Integer
Init_Win_bytes_backward	Initial TCP window size backward	Integer
act_data_pkt_fwd	Forward packets carrying payload data	Integer
min_seg_size_forward	Minimum segment size forward	Integer
Active Mean	Mean duration of active periods	Float
Active Std	Standard deviation of active duration	Float
Active Max	Maximum active duration	Integer
Active Min	Minimum active duration	Integer
Idle Mean	Mean idle duration	Integer
Idle Std	Standard deviation of idle duration	Float
Idle Max	Maximum idle duration	Integer
Idle Min	Minimum idle duration	Integer
Label	Traffic class or attack type	String

Technology Stack

1. **Microsoft Excel:** Data Visualization
2. **MS SQL:** Data Storage, Data Cleaning
3. **Python:** Data Analysis, Machine Learning
4. **Streamlit:** Deploying ML Models and Dashboards

Model Evaluation Metrics

1. Confusion Matrix
2. Accuracy
3. Recall
4. Precision
5. F1 Score
6. ROC AUC
7. False Positive Rate
8. False Negative Rate

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

$$Precision = \frac{TP}{TP + FP}$$

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

$$F1\ Score = \frac{2 \times Presision \times Recall}{Precision \times Recall}$$

$$False\ Positive\ Rate = \frac{FP}{FP + TN}$$

$$False\ Negative\ Rate = \frac{FN}{FN + TP}$$

Data Discrepancies & Observations

- Data Size = 2,830,743 Rows, 79 Columns
- Duplicated Column: fwd_header_length
- Null Entries: flow_bytes_per_sec (1358 Null Entries)
- Duplicated Rows: 308381 Rows
- Infinite Value Entries: flow_packets_per_sec (2867), flow_bytes_per_sec (1509)

Data Processing

- Concatenated all the datasets.
- Cleaned the column names: Made sure no extra white space exists, no special characters other than “_”, all the characters are in lowercase.
- Cleaned the data labels and created multiclass and binary class labels (refer next slide)
- Replaced the infinite value entries with nulls.
- Removed the duplicated entries.
- Retained the nulls altogether.
- Nulls will be imputed using SimpleImputer or KNNImputer.
- After Data Cleaning we have: 2,522,362 Rows, 80 Columns

Data Labelling

Original Label	Cleaned Label	Multi-Class Label	Binary Class Label
BENIGN	benign	benign	0
FTP-Patator	ftp_patator	brute_force	1
SSH-Patator	ssh_patator		
Web Attack Brute Force	web_attack_brute_force		
DoS Slowhttptest	dos_slow_http_test	dos_ddos	
DoS Hulk	dos_hulk		
DoS GoldenEye	dos_golden_eye		
DoS slowloris	dos_slow_loriS		
DDoS	ddos		
Web Attack XSS	web_attack_xss	web_attack	
Web Attack Sql Injection	web_attack_sql_injection		
Infiltration	infiltration	infiltration	
Bot	bot	bot	
Heartbleed	heartbleed	heartbleed	
PortScan	portscan	portscan	

Label Definitions

Attack Type	Definition
benign	Normal and legitimate network traffic. No malicious activity
ftp_patator	A brute force attack targeting FTP servers. Attacker repeatedly tries different username and password combinations
ssh_patator	A brute force attack on SSH services. Attacker attempts many login credentials over SSH
web_attack_brute_force	Brute force login attempts on web applications. Targets login pages using automated credential guessing
dos_slow_http_test	Slow rate DoS attack. Attacker sends incomplete HTTP requests
dos_hulk	High volume Denial of Service attack. Attacker floods the server with massive HTTP requests. Causes CPU and memory exhaustion very quickly
dos_golden_eye	HTTP based DoS attack. Sends numerous requests with randomized headers. Mimics legitimate traffic but overwhelms the server
dos_slow_loriS	A Denial-of-Service attack. Attacker sends HTTP requests very slowly and keeps connections open. Server resources get exhausted, blocking legitimate users
ddos	Distributed Denial of Service attack. Multiple compromised systems attack a single target. Overwhelms network or application resources.
web_attack_xss	Cross Site Scripting attack. Malicious scripts are injected into web pages. Used to steal cookies, session tokens, or user data
web_attack_sql_injection	Attacker injects malicious SQL queries through user inputs. Allows unauthorized access to databases. Can read, modify, or delete database records

Label Definitions

Attack Type	Definition
attack Type	Attacker gains internal network access. Usually happens after initial compromise
Infiltration	Attacker gains internal network access. Usually happens after initial compromise
bot	Infected system controlled by a command-and-control server
heartbleed	Exploits a vulnerability in OpenSSL. Attacker reads sensitive server memory. Can leak passwords, private keys, and session data
portscan	Attacker scans a target system for open ports. Helps attackers identify vulnerable services

- **FTP Server:** An FTP (File Transfer Protocol) server is a system that allows users to upload, download, and manage files over a network.
- Used for: Uploading website files to a hosting server, Transferring , large files between systems Remote file management
- **SSH Server:** An SSH (Secure Shell) server allows secure remote login and command execution on another computer.
- Used For: Remote server administration, Secure file transfer using SCP or SFTP, Running commands on remote machines

Exploratory Data Analysis (EDA)

Attack Distribution:

Attack Type	Distribution
dos_ddos	75.5531%
portscan	21.3251%
brute_force	2.4941%
bot	0.4586%
web_attack	0.1580%
infiltration	0.0085%
heartbleed	0.0026%

Attack Distribution:

Attack	Distribution
0 (Benign)	83.1159%
1 (Attack)	16.8841%

Exploratory Data Analysis (EDA)

Label Distribution:

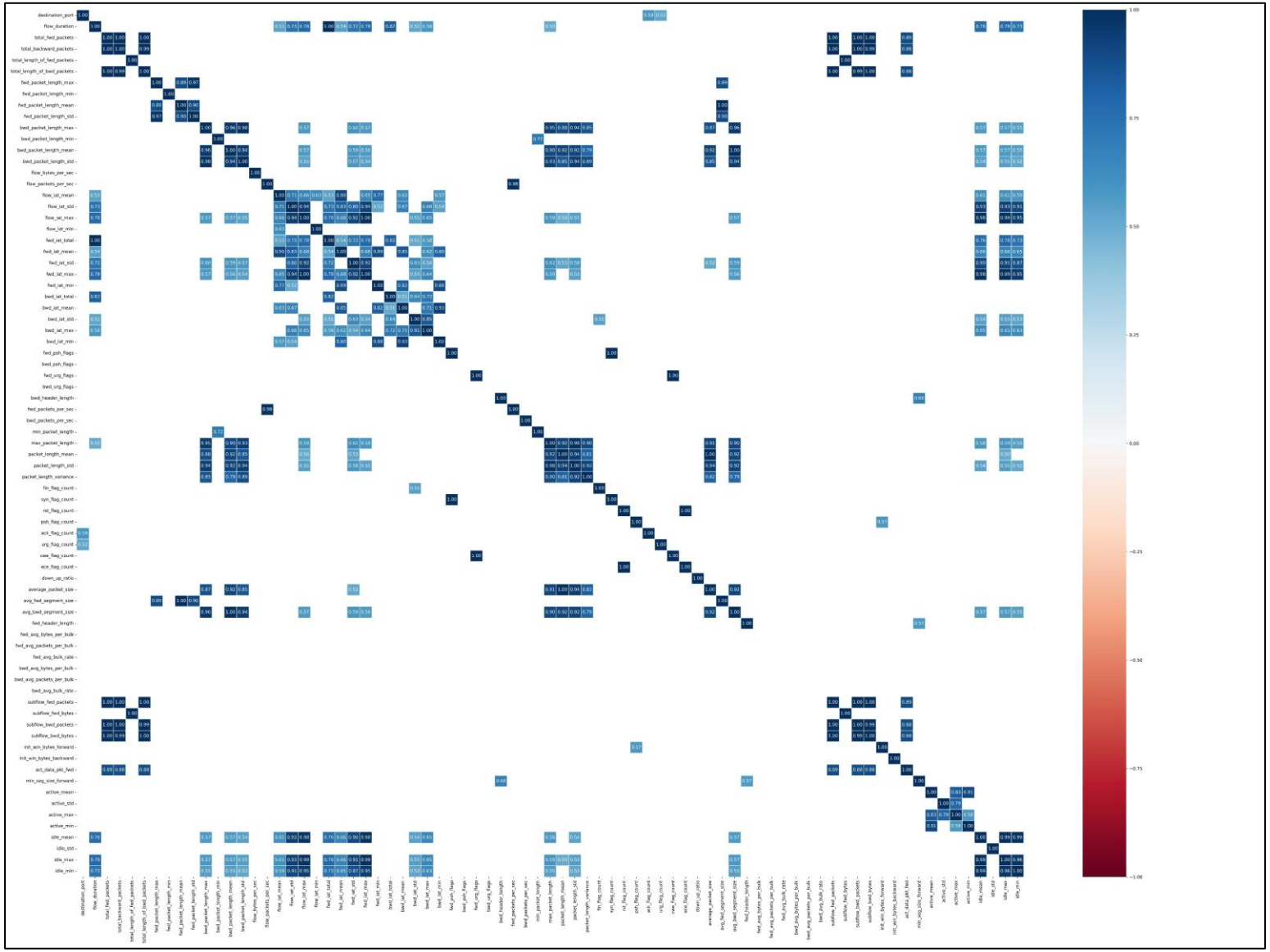
Label	Distribution
benign	83.1159%
dos_hulk	6.8527%
ddos	5.0752%
portscan	3.6006%
dos_golden_eye	0.4078%
ftp_patator	0.2352%
dos_slow_loris	0.2135%
dos_slow_http_test	0.2073%
ssh_patator	0.1276%
bot	0.0774%
web_attack_brute_force	0.0583%
web_attack_xss	0.0258%
infiltration	0.0014%
web_attack_sql_injection	0.0008%
heartbleed	0.0004%

Exploratory Data Analysis (EDA)

Attack Distribution:

Attack Type	Distribution
dos_hulk	40.5865%
ddos	30.0593%
portscan	21.3251%
dos_golden_eye	2.4152%
ftp_patator	1.3931%
dos_slow_loris	1.2644%
dos_slow_http_test	1.2276%
ssh_patator	0.7559%
bot	0.4586%
web_attack_brute_force	0.3452%
web_attack_xss	0.1531%
infiltration	0.0085%
web_attack_sql_injection	0.0049%
heartbleed	0.0026%

Multicollinearity



- Multicollinearity exists between multiple features.
- Tree based models are intelligent enough to deal with these features.

Model Building

Models Built:

1. Logistic Regression Model
2. Decision Tree
3. Random Forest
4. XGBoost

Models Results:

Model	Accuracy	Macro Average Recall	Macro Average Precision	Macro Average F1-Score
Logistic Regression	95 %	96 %	88 %	92 %
Decision Tree	99 %	98 %	99 %	99 %
Random Forest	99 %	99 %	99 %	99 %
XGBoost	99 %	99 %	98 %	98 %

- We need a model with less False Negatives (Attack Behaviour classified as Normal Behaviour).
- Recall metric is chosen to decide which model to choose, because high Recall gives us less false negatives.

Model Building

Why XGBoost?

- We selected **XGBoost** as the final model due to its high computational efficiency, native handling of missing values, and strong performance on complex datasets.
- In our experiments, XGBoost demonstrated better generalization compared to other tree-based models, making it well suited for network traffic classification where patterns are highly non linear and dynamic.

Why Not Logistic Regression, Even Though It Generalizes Well?

- Although Logistic Regression often generalizes well, it was not chosen because the primary requirement in this use case is high predictive accuracy.
- Network attack detection is a highly sensitive and high priority problem where misclassification can have serious consequences.
- Logistic Regression has limited capability in modeling non linear relationships and complex feature interactions, which are common in network traffic data. Therefore, a more robust model like XGBoost was preferred.

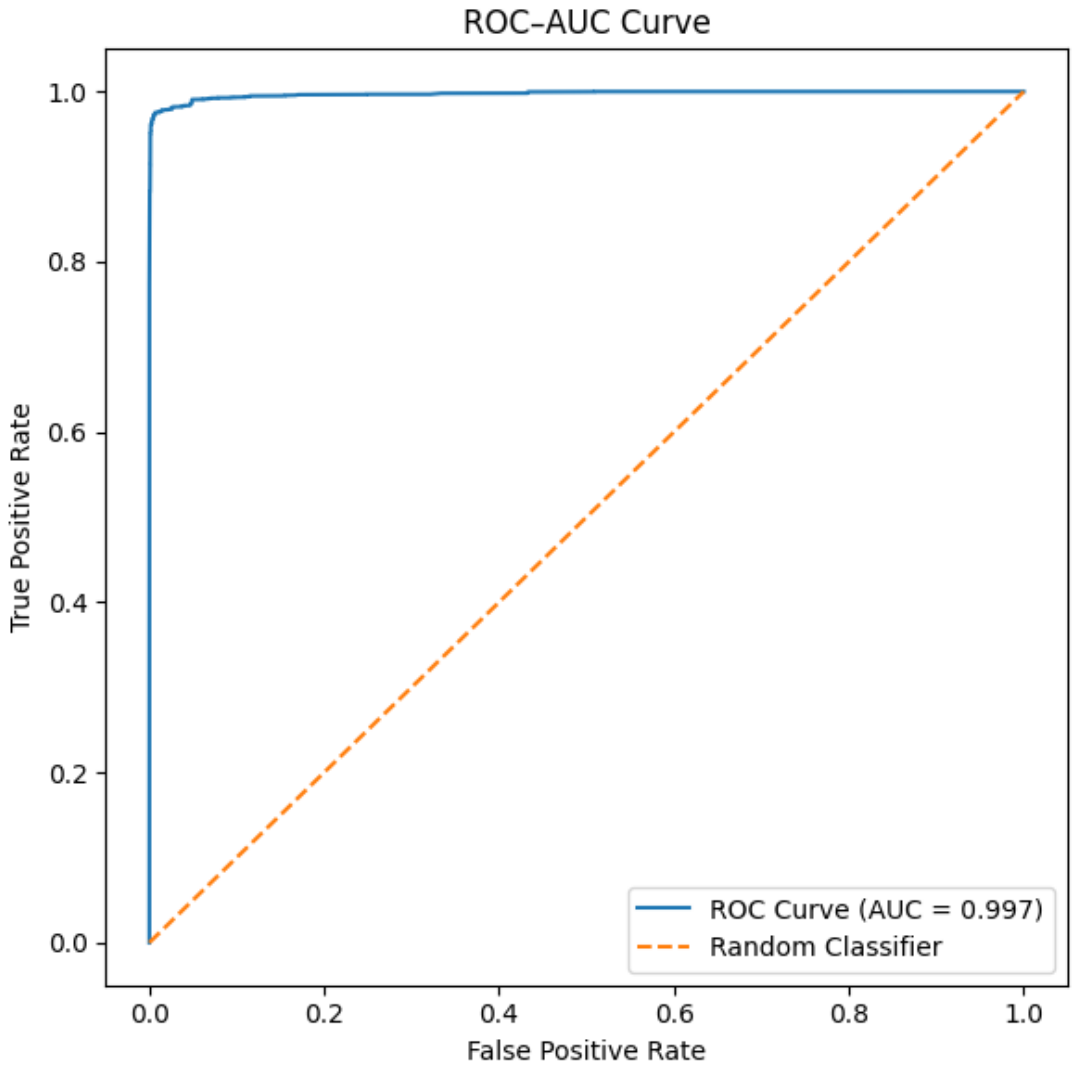
Model Building

XGBoost Model Hyperparameters (Binary Classification Model):

Hyperparameter	Definition	Value
N Estimators	Number of boosting trees built sequentially to improve model performance.	500
Learning Rate	Controls how much each tree contributes to the final model, with smaller values improving	0.01
Max Depth	Limits the depth of each tree to prevent overfitting by keeping trees simple.	3
Min Child Weight	Sets the minimum sum of instance weights required in a child node, helping reduce overfitting.	6
Subsample	Fraction of training data randomly sampled for each tree to improve robustness.	0.7
Colsample Bytree	Fraction of features randomly sampled for each tree to reduce correlation among trees	0.7
Gamma	Minimum loss reduction required to make a further split, controlling tree complexity.	0.3
Reg Alpha	L1 regularization term that encourages sparsity in feature usage.	0.5
Reg Lambda	L2 regularization term that penalizes large weights to improve model stability.	2.0
Tree Method	Uses histogram-based optimization for faster training on large datasets.	Hist
Max Delta Step	Limits the maximum weight change per tree to improve convergence in imbalanced datasets.	1

Model Building

ROC-AUC Curve (Binary Classification):



		Predicted	
		False	**True**
Actual	**False**	82.59% (TN)	0.52% (FP)
	True	0.40% (FN)	16.48% (TP)

Tuned to Custom Threshold = 0.3399

Multiclass Model

- In the multiclass model, we have 6 distinct attack classes.
- Class imbalance was handled using the SMOTE oversampling technique.
- An XGBoost multiclass classifier was trained to predict the specific type of network attack, and the model receives input only from attack traffic, excluding benign traffic.
- This XGBoost Multiclass Classification Model has **Accuracy of 99%,** Macro Average **F1-Score of 98%,** Average **Recall of 95%**
- The Recall is lower due to class imbalance.

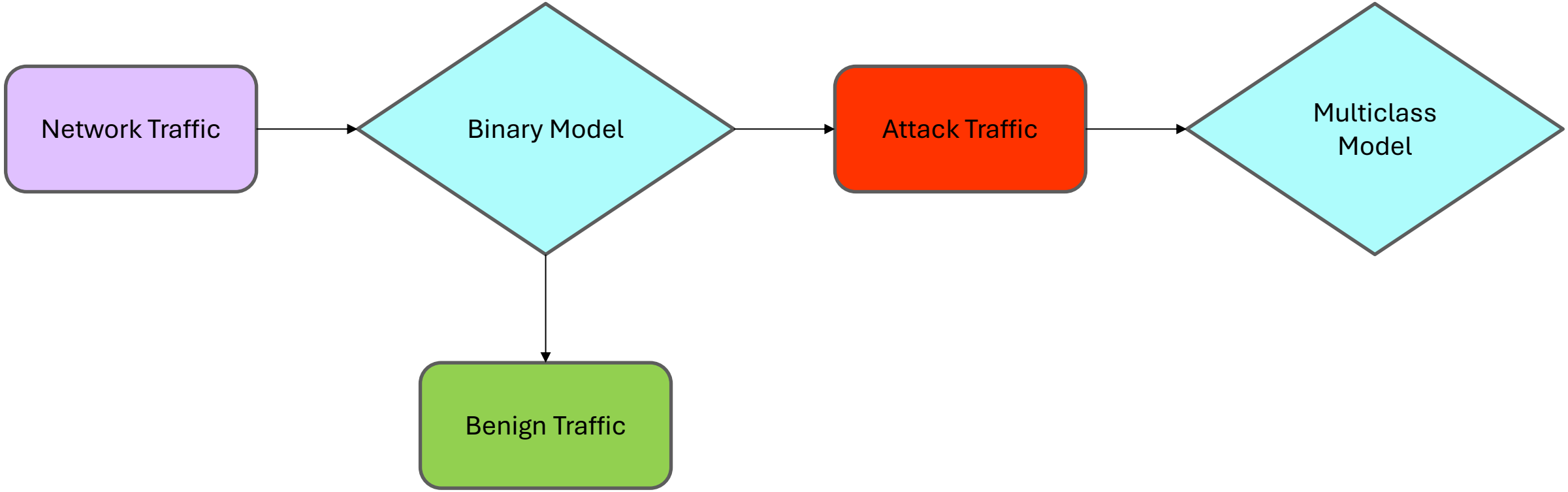
Model Building

XGBoost Model Hyperparameters (Multiclass Classification Model):

Hyperparameter	Definition	Value
Num Class	Specifies the total number of target classes in a multiclass classification problem	6
N Estimators	Number of boosting trees built sequentially to improve model performance.	250
Learning Rate	Controls how much each tree contributes to the final model, with smaller values improving	0.01
Max Depth	Limits the depth of each tree to prevent overfitting by keeping trees simple.	3
Min Child Weight	Sets the minimum sum of instance weights required in a child node, helping reduce overfitting.	20
Subsample	Fraction of training data randomly sampled for each tree to improve robustness.	0.6
Colsample Bytree	Fraction of features randomly sampled for each tree to reduce correlation among trees	0.6
Gamma	Minimum loss reduction required to make a further split, controlling tree complexity.	2.0
Reg Alpha	L1 regularization term that encourages sparsity in feature usage.	1.0
Reg Lambda	L2 regularization term that penalizes large weights to improve model stability.	2.0
Tree Method	Uses histogram-based optimization for faster training on large datasets.	Hist

Classification Model Architecture

Model Flow Chart:



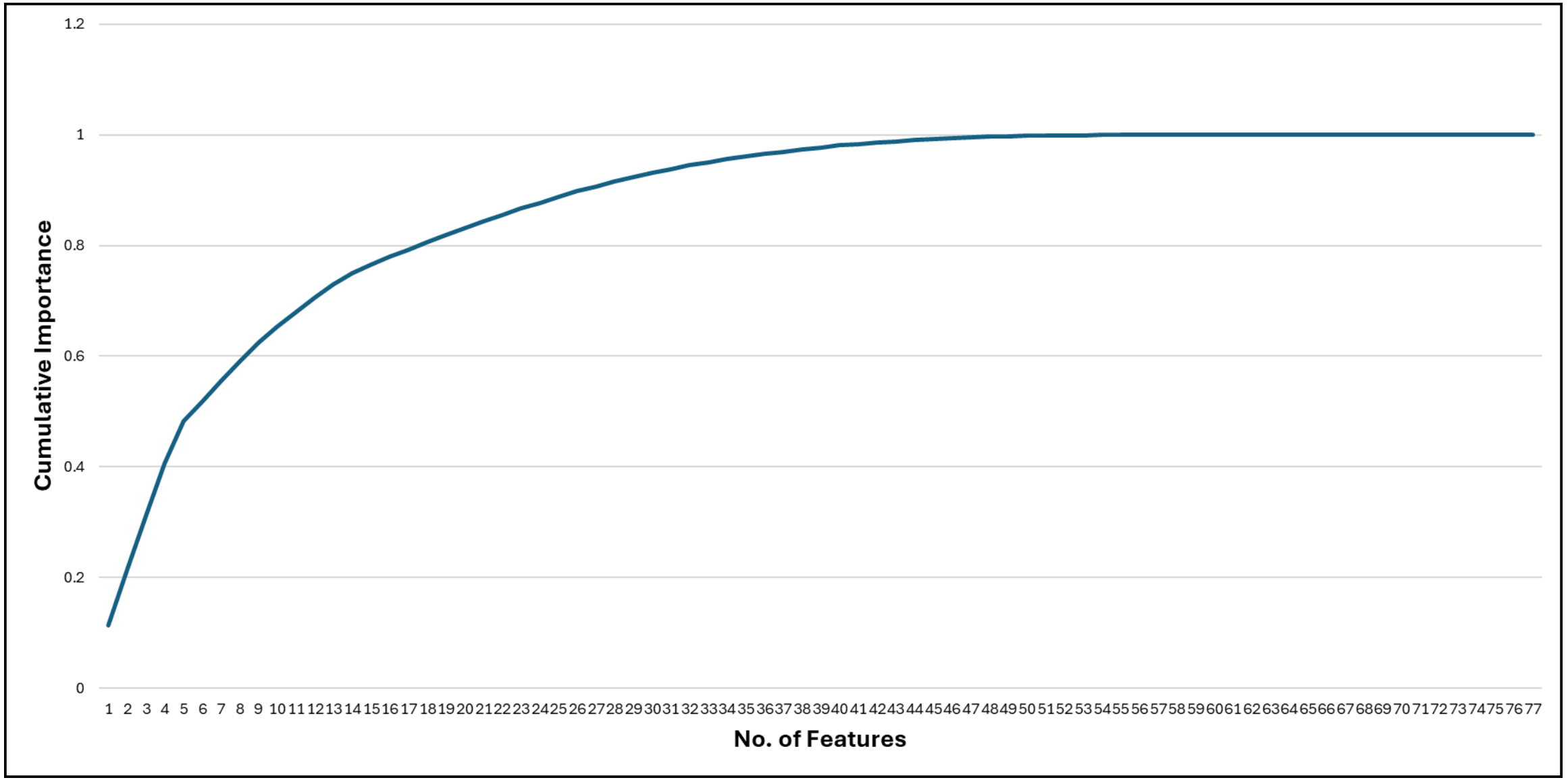
Feature Importance

Top 10 Features:

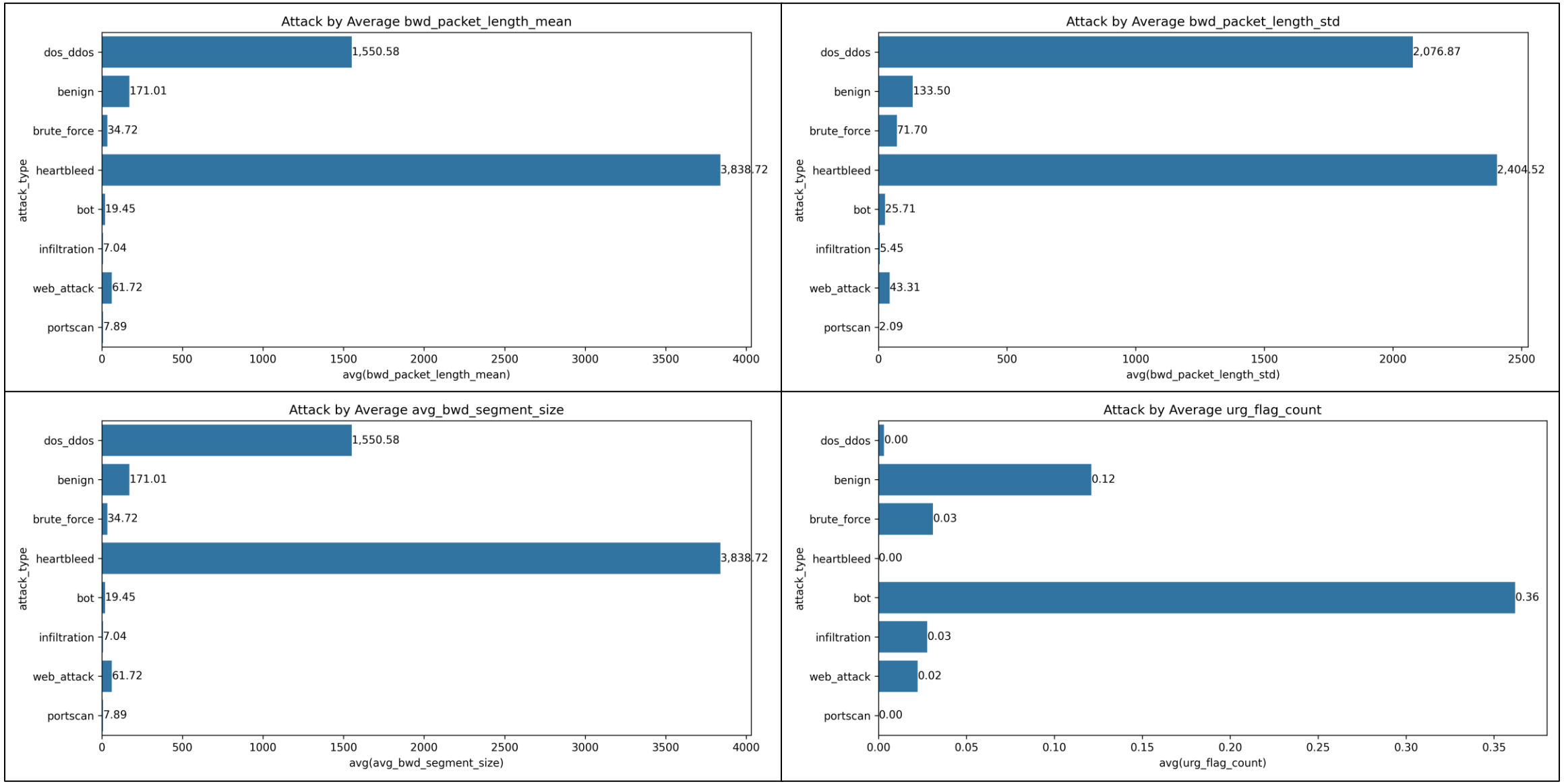
Feature	Importance	Cumulative Importance
bwd_packet_length_mean	11.34%	11.34%
bwd_packet_length_std	10.16%	21.50%
avg_bwd_segment_size	9.92%	31.42%
urg_flag_count	9.16%	40.58%
average_packet_size	7.69%	48.27%
max_packet_length	3.64%	51.91%
bwd_header_length	3.60%	55.51%
flow_bytes_per_sec	3.56%	59.07%
fwd_packet_length_max	3.44%	62.52%
packet_length_variance	2.80%	65.32%

- The Top 10 Features Account for About 65% of the ML Model's Total Importance.

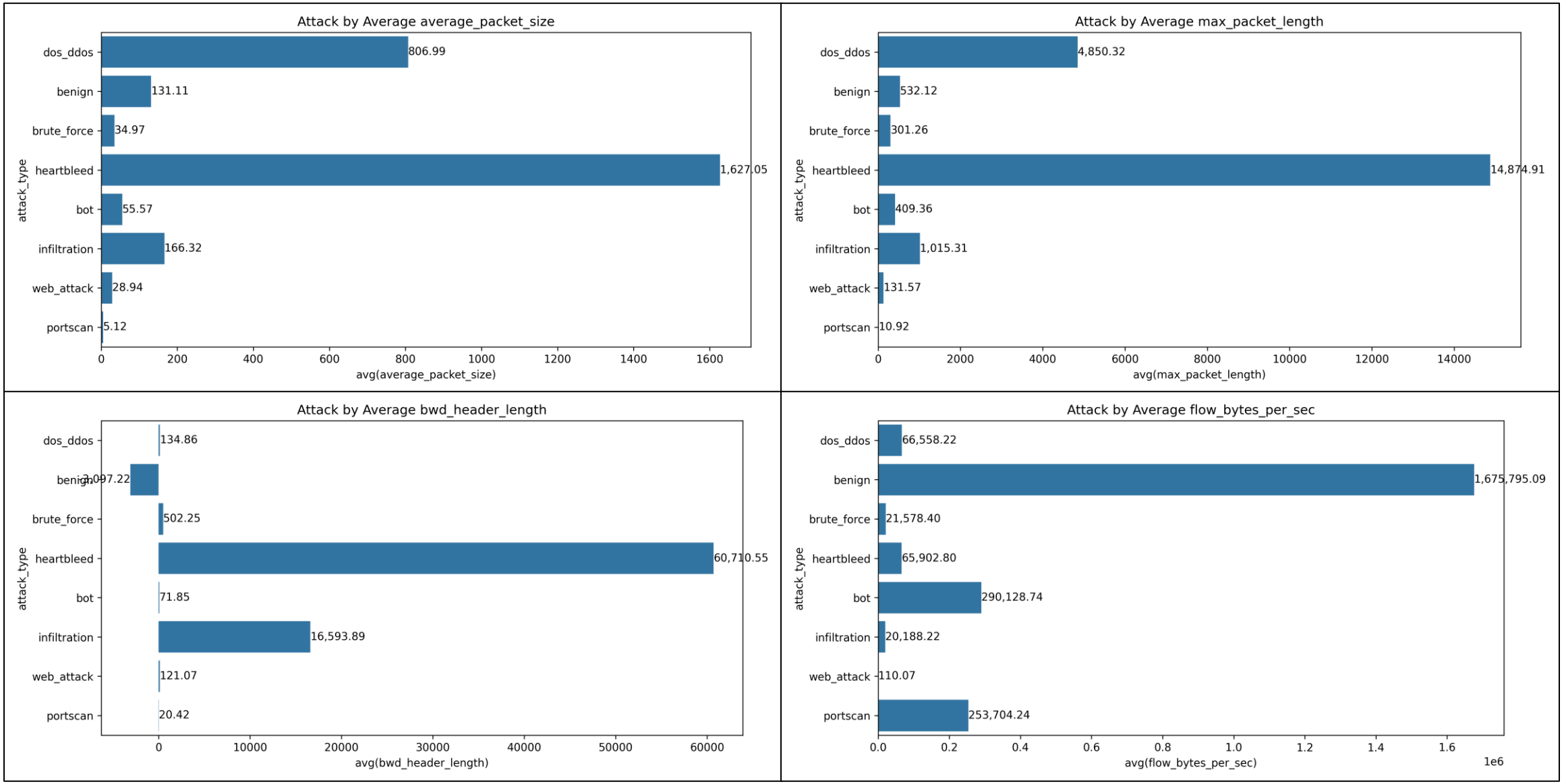
Feature Importance



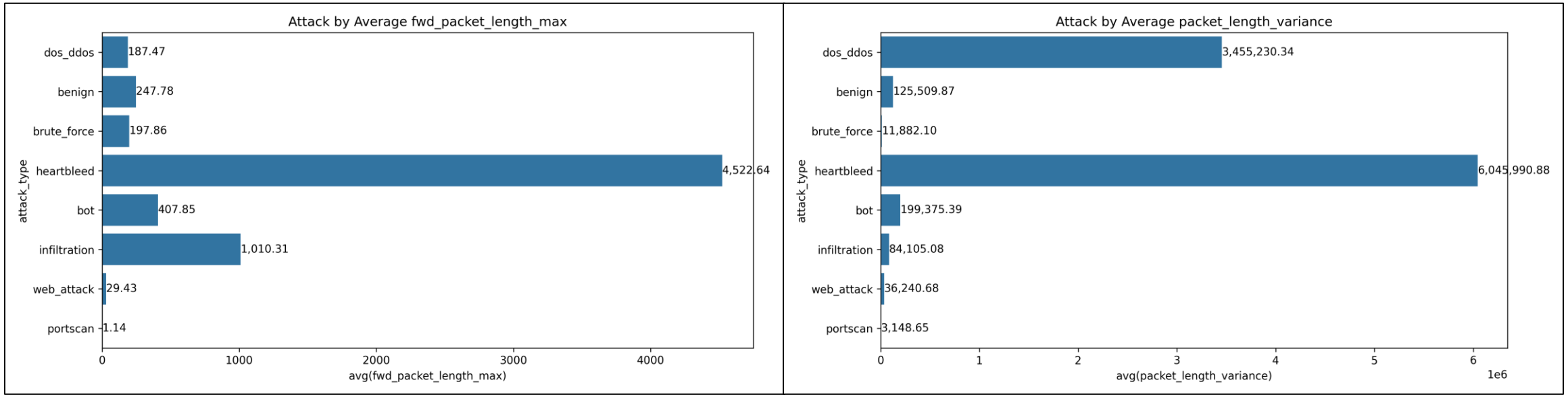
Top 10 Features



Top 10 Features



Top 10 Features

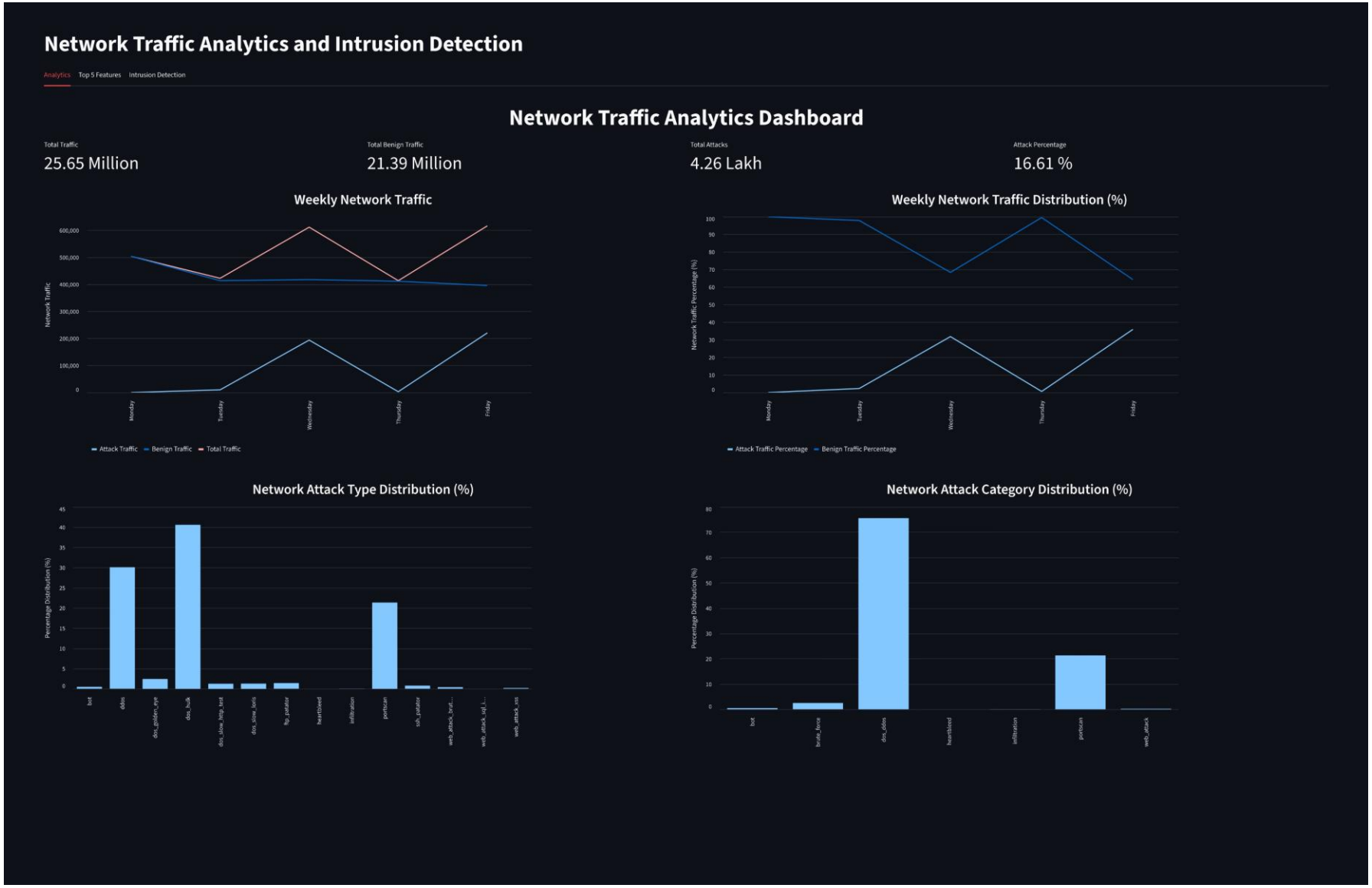


- There is a clear distinction between attack types across the top features.
- XGBoost effectively captures these differences by learning non linear relationships and feature interactions through tree-based splits.
- It also handles multicollinearity effectively and automatically ignores irrelevant features, as evidenced by many features having zero or negligible importance in the feature importance plots.

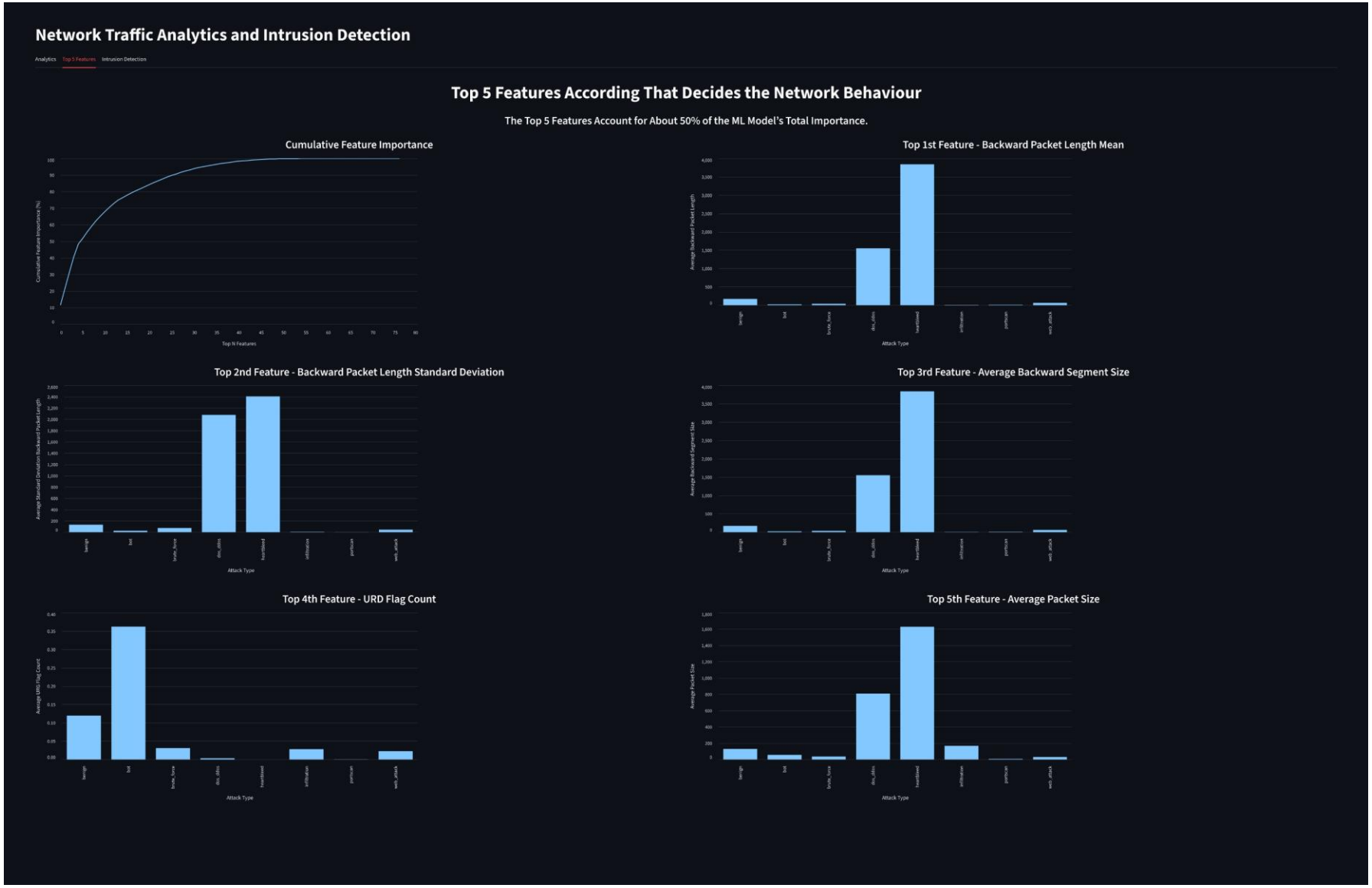
Streamlit Dashboard – 1

- The dashboard is deployed using Streamlit and GitHub, enabling easy access and seamless updates directly from the repository.
- The application consists of three tabs, each serving a specific purpose.
  1. Analytics: Basic charts like distributions.
  2. Top 5 Features: Top 5 features in ML model.
  3. Intrusion Detection: Upload a .csv file and predict and download the results.
- **Dashboard Link:** https://network-intrusion-detection-123.streamlit.app/
- **GitHub Repo Link:** https://github.com/thecodefather77/network-intrusion-detection
- **User Name:** admin
- **Password:** admin123

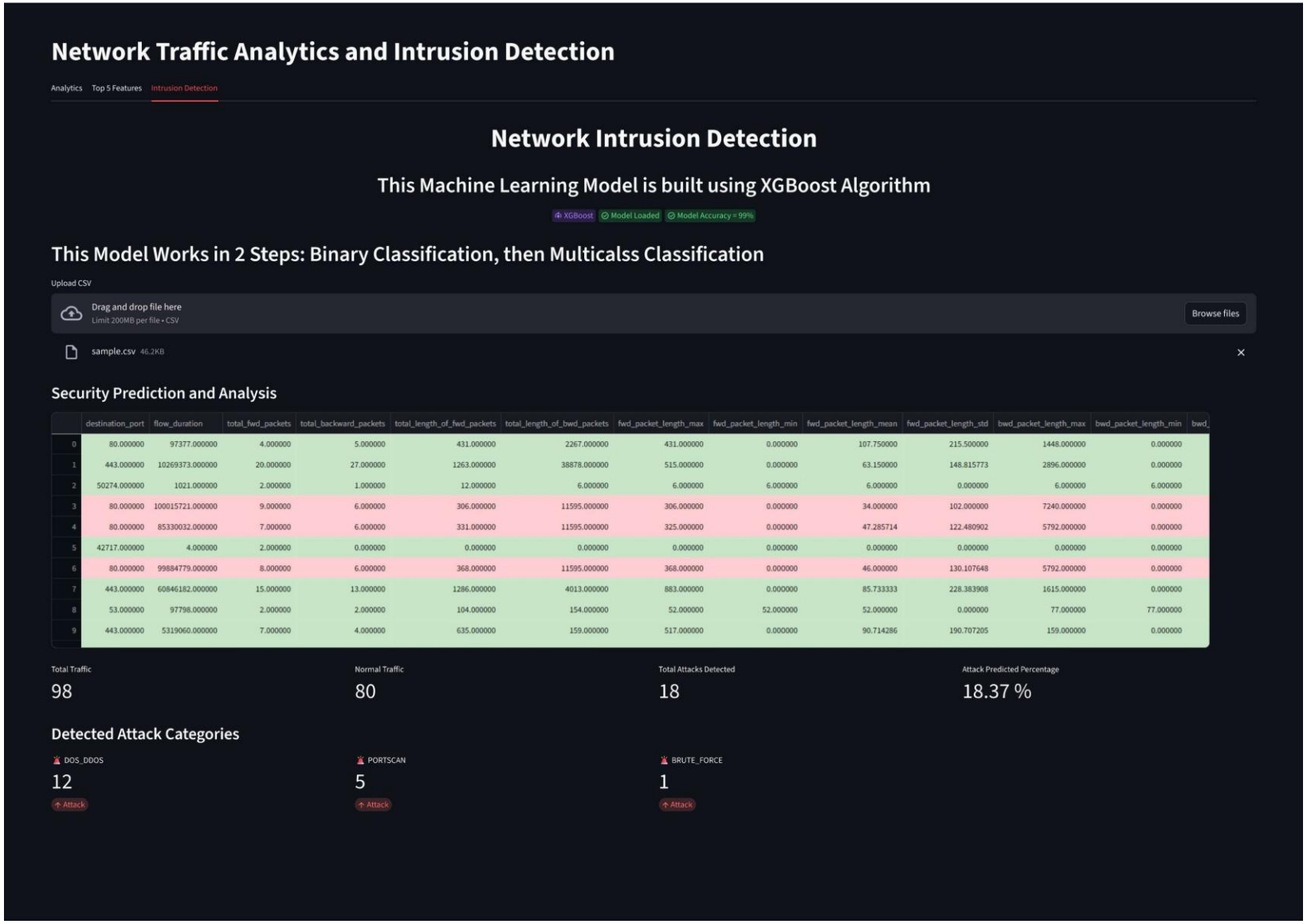
Streamlit Dashboard – 1



Streamlit Dashboard – 2



Streamlit Dashboard – 3



	destination_port	flow_duration	total_fwd_packets	total_backward_packets	total_length_of_fwd_packets	total_length_of_bwd_packets	fwd_packet_length_max	fwd_packet_length_min	fwd_packet_length_mean	fwd_packet_length_std	bwd_packet_length_max	bwd_packet_length_min	bwd_
0	80.000000	97377.000000	4.000000	5.000000	431.000000	2267.000000	431.000000	0.000000	107.750000	215.500000	1448.000000	0.000000	
1	443.000000	10269373.000000	20.000000	27.000000	1263.000000	38878.000000	515.000000	0.000000	63.150000	148.815773	2896.000000	0.000000	
2	50274.000000	1021.000000	2.000000	1.000000	12.000000	6.000000	6.000000	6.000000	6.000000	0.000000	6.000000	6.000000	
3	80.000000	100015721.000000	9.000000	6.000000	306.000000	11595.000000	306.000000	0.000000	34.000000	102.000000	7240.000000	0.000000	
4	80.000000	85330032.000000	7.000000	6.000000	331.000000	11595.000000	325.000000	0.000000	47.285714	122.480902	5792.000000	0.000000	
5	42717.000000	4.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
6	80.000000	99884779.000000	8.000000	6.000000	368.000000	11595.000000	368.000000	0.000000	46.000000	130.107648	5792.000000	0.000000	
7	443.000000	60846182.000000	15.000000	13.000000	1286.000000	4013.000000	883.000000	0.000000	85.733333	228.383908	1615.000000	0.000000	
8	53.000000	97798.000000	2.000000	2.000000	104.000000	154.000000	52.000000	52.000000	52.000000	0.000000	77.000000	77.000000	
9	443.000000	5319060.000000	7.000000	4.000000	635.000000	159.000000	517.000000	0.000000	90.714286	190.707205	159.000000	0.000000	

THANK YOU!