An Explainable Approach for Network Intrusion Detection Using Machine Learning and Deep Neural Networks

Presented by

Sajan Kumer Sarker – 2111131642
Arifa Akbor Srity – 2021149642
Mahzabeen Rahman Meem – 2021300642

CSE499B.17

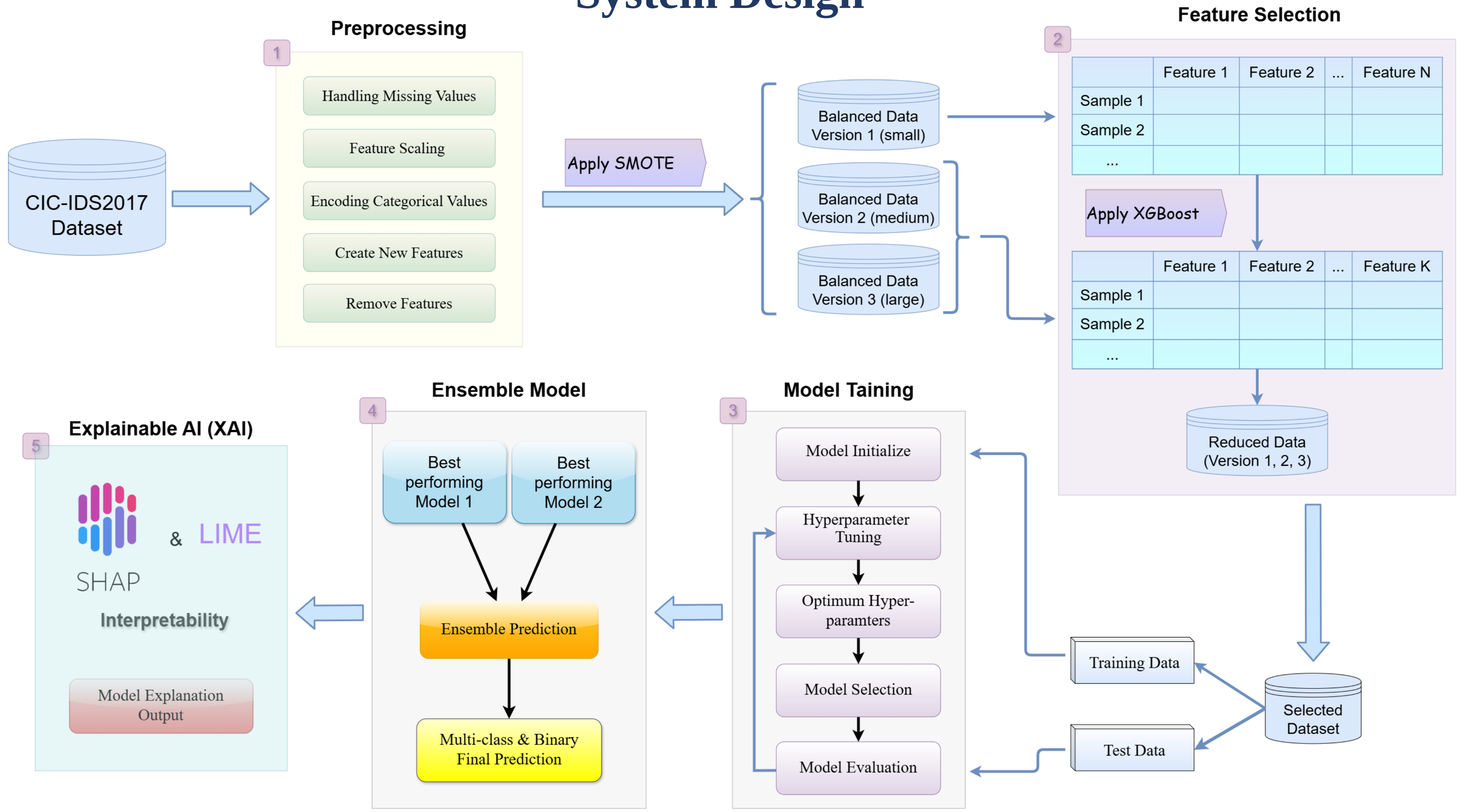
Faculty Advisor

Dr. Mohammad Ashrafuzzaman Khan (AZK)
Associate Professor
ECE Department
Summer, 2024

Introduction

An Intrusion Detection System (IDS) is a vital cybersecurity tool that monitors network traffic for sings of malicious activity. It analyzes data packets to detect anomalies or known attack patterns, such as malware or unauthorized access. IDS can be either signature-based or anomaly-based, where anomaly-based system use machine learning techniques to spot unusual behavior. These system serve as second line of defense, complementing firewalls that might miss sophisticated threats. With the increasing complexity of cyberattacks, IDS helps to protect critical infrastructures and sensitive data by providing real-time alerts and enabling swift responses. Modern IDS solutions leverage AI to improve detection accuracy while minimizing the false positives. Deploying an effective IDS is essential to a strong cybersecurity strategy.

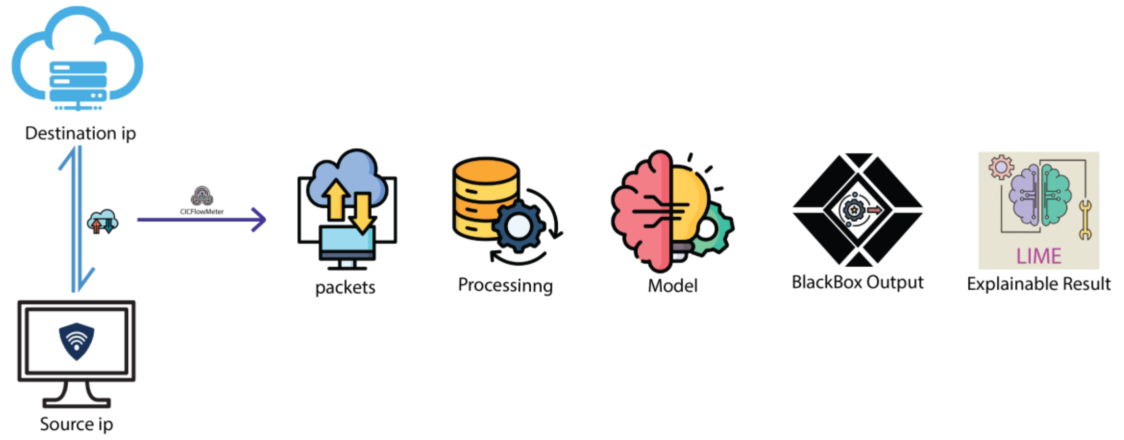
System Design



Usability, Manufacturability, Sustainability

- **Enhanced usability through adaptability & Explainability :** The system adapts seamlessly across datasets and network environments with minimal configuration. LIME-based explainable AI improves user trust and allows even non-technical users to understand and act on alerts with confidence.
- **Modular Manufacturability & Flexible Deployment :** A modular pipeline—from XGBoost feature selection to SMOTE balancing and ensemble modeling—allows independent development, tuning, and testing. The container-ready architecture supports cloud deployment and reuse of optimized components.
- **Sustainable Resource Utilization :** The machine learning models handle basic network traffic for complex anomalies. This intelligent model routing reduces energy consumption and hardware strain, promoting eco-friendly operations in data centers.
- **Reduced False positive & Efficient Performance** : Higher prediction accuracy and fewer false alerts lessen the burden on analysts and systems alike, saving both human and computational resources and extending system lifespan.

Implementations



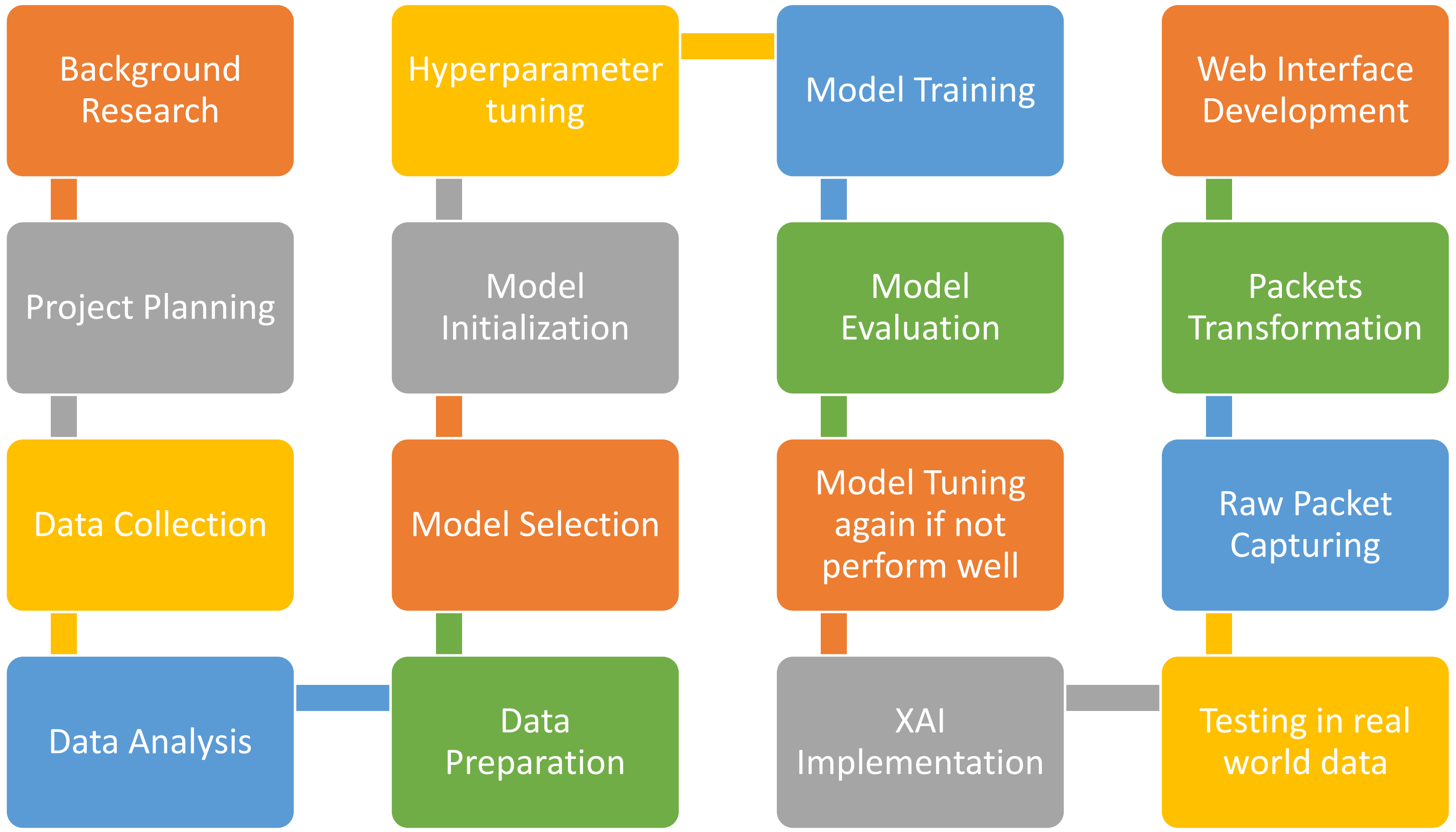
Steps:

- **Background Research:** Analyze methods for classifying network attacks by examining network traffic data.
- **Dataset Collection:** Use the CIC-IDS2017 network traffic dataset.
- **Preprocessing:** Handle missing values, encode categorical variables, and perform necessary data cleaning.
- **SMOTE:** Apply SMOTE to generate three dataset variants: **small**, **medium**, and **large**.
- **Feature Selection:** Select features based on importance scores from an XGBoost model.
- **Final Datasets:** Prepare small, medium, and large datasets with the top **28 features**.

Implementations

- **Model Selection:** Selected the following models: Random Forest, Decision Tree, Naïve Bayes, AdaBoost, Multi-Layer Perceptron (MLP)
- **Base Model Training:** Train all models to establish baseline performance.
- **Hyperparameter Tuning:** Use **GridSearchCV** to tune model parameters for optimal performance.
- **Evaluation:** Evaluate models on the test set using the following metrics: Accuracy, Precision, Recall, F1-score, False Positive Rate (FPR), Confusion Matrix, Precision-Recall Curve.
- **Ensemble Model:** Build an ensemble using the top two performing models to improve overall performance.
- **Explainable AI:** Incorporate explainability techniques to ensure transparent and interpretable results.
- **Packet Captures:** Capture raw network traffic using **Wireshark**.
- **Transform Data:** Convert raw packet data into **flow-based statistical features** suitable for model input using CICFlowMeter .
- **Prediction with Real World Data:** Generate final predictions on real-world data, along with interpretability insights or model explanations.

Development Process



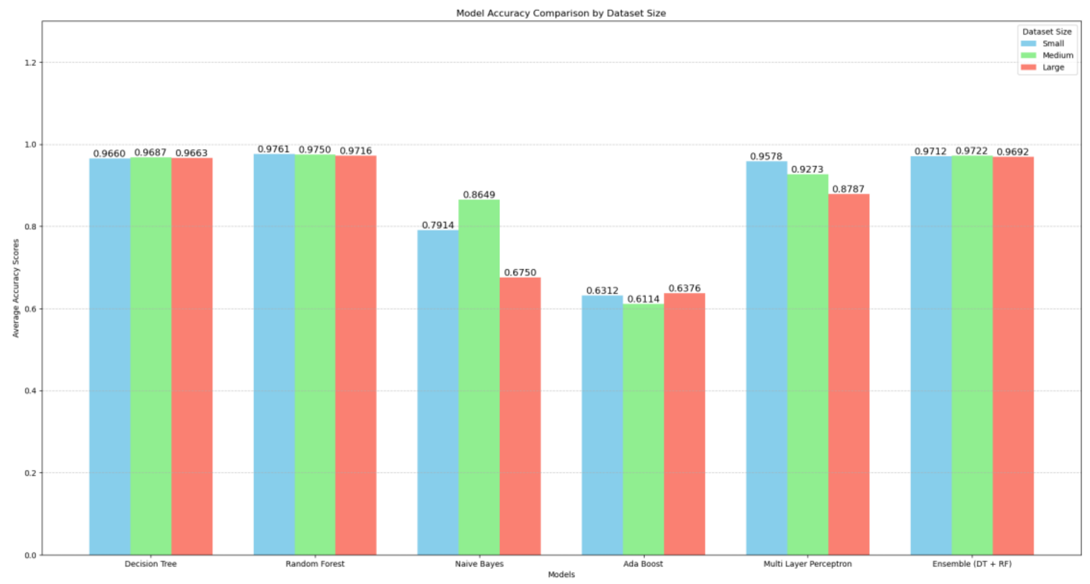
Environment Considerations & Sustainability

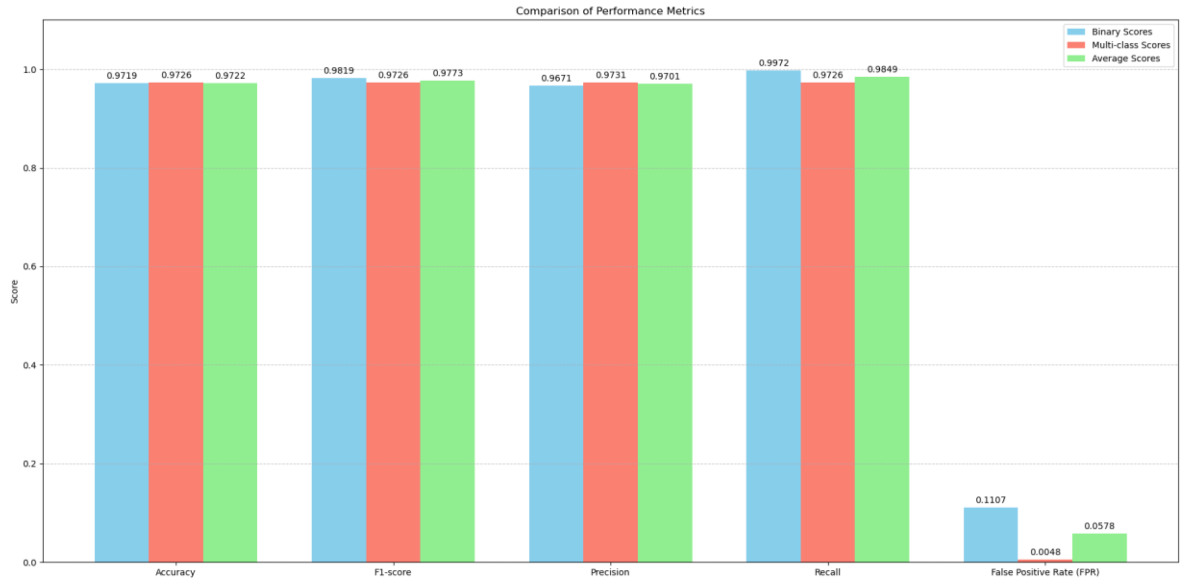
- **Energy-Efficient Model Allocation :** The system uses machine learning models for simple patterns and complex anomalies, reducing power consumption, training time, and overall CPU/GPU usage—ideal for scalable or cloud-based environments.
- **Modular & Reusable Architecture :** Separate models and feature sets enable reuse and faster experimentation without starting from scratch, reducing infrastructure strain and supporting eco-conscious deployment in enterprise and IoT settings.
- **Explainability for Targeted Refinement :** LIME-based explainable AI helps developers understand predictions, making debugging and model tuning more focused. This reduces the need for frequent retraining and extends hardware lifespan, lowering e-waste.
- **Sustainable Development Practices:** Modular training and ensemble structures promote clean, maintainable code and long-term model sustainability, reducing digital waste and encouraging responsible AI practices.
- **Portable, Lightweight & Open-Source :** The use of open-source datasets (like CIC-IDS2017) and container-ready tools ensures system portability, compatibility, and minimal environmental impact compared to legacy infrastructure.

Tools & Technologies

- **Programming Language:**
  - Python3.11
- **Machine Learning Library:**
  - Scikit-Learn
  - PyTorch
- **Machine Learning Algorithms:**
  - Decision Tree
  - Random Forest
  - Extreme Gradient Boosting (XGBoost)
  - Naïve Bayes
  - Ada Boost
  - Multi Layer Perceptron
- **Data Visualization:**
  - Matplotlib
  - Seaborn
- **Model Evaluation & Metrics:**
  - Scikit-Learn Metrics (Accuracy Score, Precision, Recall, F1-Score, False Positive Rate, Confusion Matrix, Precision-Recall Curve)
- **Web Framework:**
  - Flask
- **Version Control:**
  - Git/GitHub
- **Programming Environment:**
  - VS Code
  - Google Colab
- **Networking Tools:**
  - Wireshark
  - CICFlowMeter

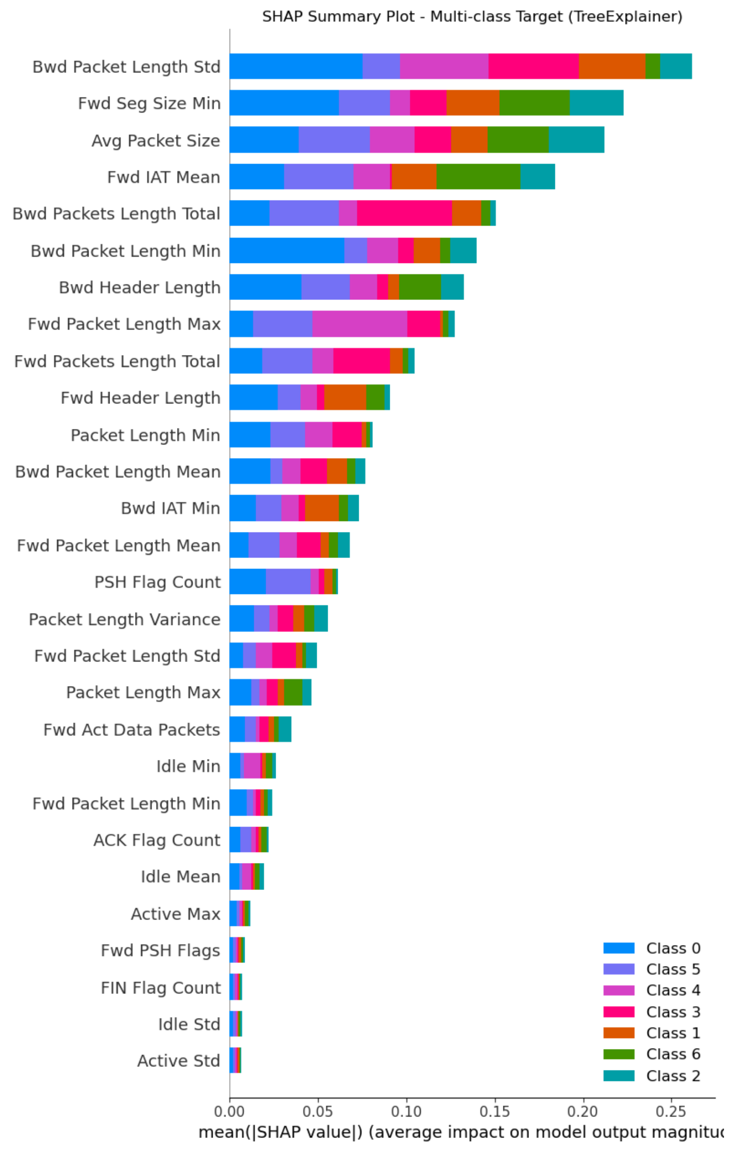
Results



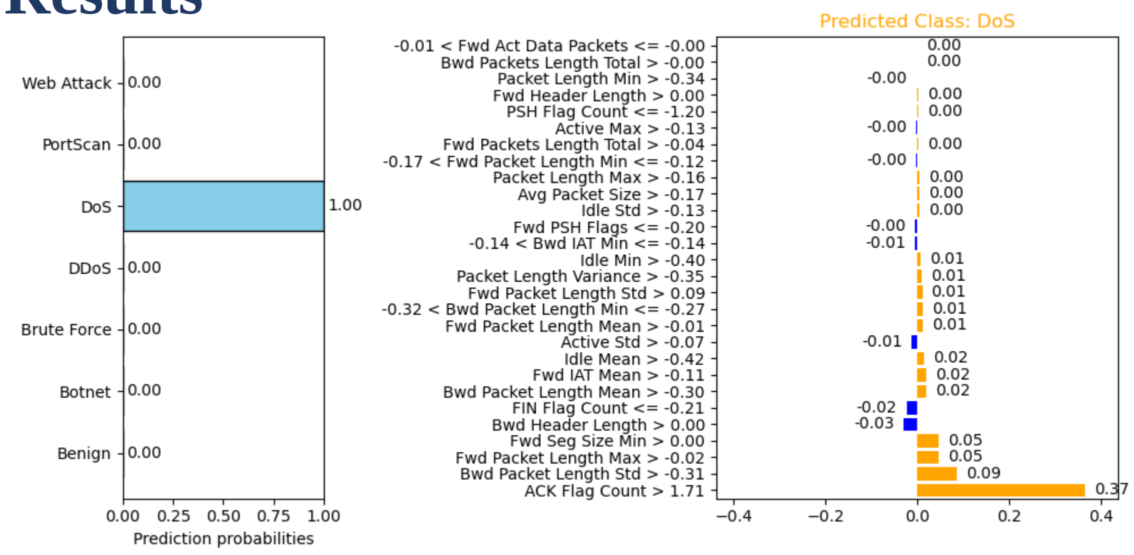


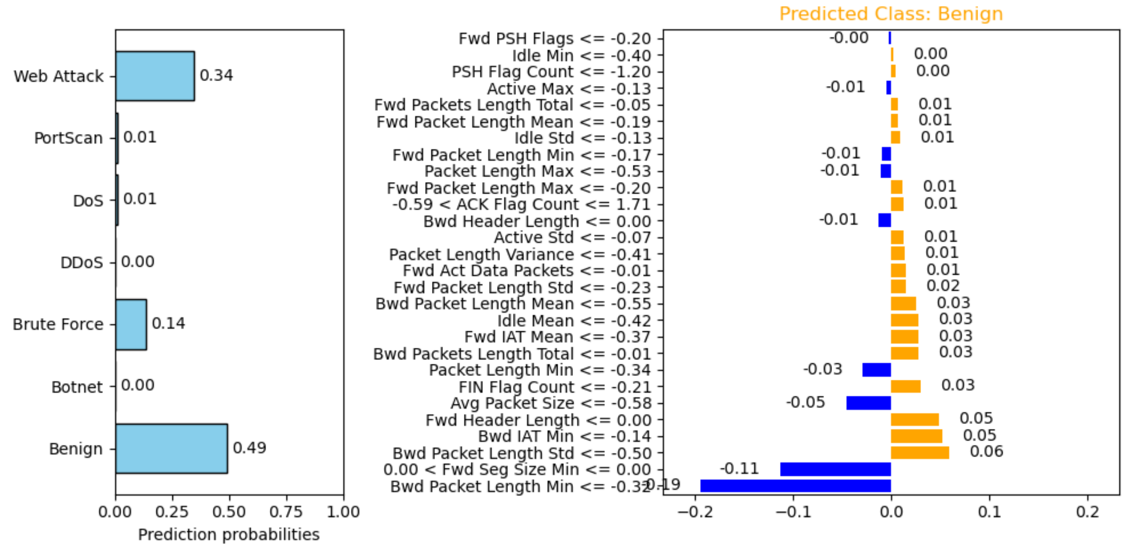
Observations:

- The Decision Tree and Random Forest models achieved the highest individual accuracy, up to 97.6%, when trained on the medium-sized dataset (version 2).
- The Ensemble Model (Decision Tree + Random Forest) performed better overall than any single model. It reached up to 97.2% average accuracy.
- The Ensemble model significantly reduced false positives, achieving a false positive rate of 0.0048, making it more effective for multi-class traffic classification.


SHAP Summary Plot - Multi-class Target (TreeExplainer)
Bwd Packet Length Std
Fwd Seg Size Min
Avg Packet Size
Fwd IAT Mean
Bwd Packets Length Total
Bwd Packet Length Min
Bwd Header Length
Fwd Packet Length Max
Fwd Packets Length Total
Fwd Header Length
Packet Length Min
Bwd Packet Length Mean
Bwd IAT Min
Fwd Packet Length Mean
PSH Flag Count
Packet Length Variance
Fwd Packet Length Std
Packet Length Max
Fwd Act Data Packets
Idle Min
Fwd Packet Length Min
ACK Flag Count
Idle Mean
Active Max
Fwd PSH Flags
FIN Flag Count
Idle Std
Active Std
Class 0
Class 5
Class 4
Class 3
Class 1
Class 6
Class 2
0.00
0.05
0.10
0.15
0.20
0.25
mean(|SHAP value|) (average impact on model output magnitu

Results


Predicted Class: DoS
Web Attack 0.00
PortScan 0.00
DoS 1.00
DDoS 0.00
Brute Force 0.00
Botnet 0.00
Benign 0.00
0.00 0.25 0.50 0.75 1.00
Prediction probabilities
-0.01 < Fwd Act Data Packets <= -0.00 0.00
Bwd Packets Length Total > -0.00 0.00
Packet Length Min > -0.34 -0.00
Fwd Header Length > 0.00 0.00
PSH Flag Count <= -1.20 0.00
Active Max > -0.13 -0.00
Fwd Packets Length Total > -0.04 0.00
-0.17 < Fwd Packet Length Min <= -0.12 -0.00
Packet Length Max > -0.16 0.00
Avg Packet Size > -0.17 0.00
Idle Std > -0.13 0.00
Fwd PSH Flags <= -0.20 -0.00
-0.14 < Bwd IAT Min <= -0.14 -0.01
Idle Min > -0.40 0.01
Packet Length Variance > -0.35 0.01
Fwd Packet Length Std > 0.09 0.01
-0.32 < Bwd Packet Length Min <= -0.27 0.01
Fwd Packet Length Mean > -0.01 0.01
Active Std > -0.07 -0.01
Idle Mean > -0.42 0.02
Fwd IAT Mean > -0.11 0.02
Bwd Packet Length Mean > -0.30 0.02
FIN Flag Count <= -0.21 -0.02
Bwd Header Length > 0.00 -0.03
Fwd Seg Size Min > 0.00 0.05
Fwd Packet Length Max > -0.02 0.05
Bwd Packet Length Std > -0.31 0.09
ACK Flag Count > 1.71 0.37
−0.4 −0.2 0.0 0.2 0.4


Predicted Class: Benign
Web Attack 0.34
PortScan 0.01
DoS 0.01
DDoS 0.00
Brute Force 0.14
Botnet 0.00
Benign 0.49
0.00 0.25 0.50 0.75 1.00
Prediction probabilities
Fwd PSH Flags <= -0.20 -0.00
Idle Min <= -0.40 0.00
PSH Flag Count <= -1.20 0.00
Active Max <= -0.13 -0.01
Fwd Packets Length Total <= -0.05 0.01
Fwd Packet Length Mean <= -0.19 0.01
Idle Std <= -0.13 0.01
Fwd Packet Length Min <= -0.17 -0.01
Packet Length Max <= -0.53 -0.01
Fwd Packet Length Max <= -0.20 0.01
-0.59 < ACK Flag Count <= 1.71 0.01
Bwd Header Length <= 0.00 -0.01
Active Std <= -0.07 0.01
Packet Length Variance <= -0.41 0.01
Fwd Act Data Packets <= -0.01 0.01
Fwd Packet Length Std <= -0.23 0.02
Bwd Packet Length Mean <= -0.55 0.03
Idle Mean <= -0.42 0.03
Fwd IAT Mean <= -0.37 0.03
Bwd Packets Length Total <= -0.01 0.03
Packet Length Min <= -0.34 -0.03
FIN Flag Count <= -0.21 0.03
Avg Packet Size <= -0.58 -0.05
Fwd Header Length <= 0.00 0.05
Bwd IAT Min <= -0.14 0.05
Bwd Packet Length Std <= -0.50 0.06
0.00 < Fwd Seg Size Min <= 0.00 -0.11
Bwd Packet Length Min <= -0.32 -0.19
−0.2 −0.1 0.0 0.1 0.2

Thank You!