Network Traffic Anomaly Detection using Machine Learning

Table of Contents

- Objective
- Problem Description
- About the Project
- How It Works
- Process Flow
- Technical Details
- Architecture
- Key Technologies and Libraries
- Dataset Used
- How to Run the Project
- Members

Objective

The main aim of this project is to use machine learning to detect anomalies in network traffic, especially focusing on embedded systems. The idea is to identify unusual patterns or behaviors that may indicate malicious activity.

AI/ML for Networking

Category: Network Security

Pre-requisites:

- Computer Systems Basics – CPU/Memory/Storage/NIC
- Good Hands-on Experience on Linux
- Programming Skills in Python and/or C
- Basics of AI/ML

Problem Statement

Description:
Modern networks face increasing challenges in monitoring and securing traffic due to the exponential growth of data, encrypted communication, and sophisticated cyber threats. Traditional rule-based security measures and deep packet inspection (DPI) techniques are becoming less effective in detecting and classifying threats, especially in encrypted traffic. Manual intervention in network traffic classification is inefficient, leading to delayed threat detection and security vulnerabilities. To address these issues, AI-driven solutions can analyze traffic patterns, detect anomalies, classify applications, and enhance security in real-time, ensuring adaptive and intelligent network defense.

Expected Outcome:

- **Automated Network Traffic Analysis** using AI/ML models to detect and classify traffic in real time.
- **Improved Threat Detection & Security**, identifying anomalies, malware, and encrypted attacks with higher accuracy.
- **Reduced False Positives & False Negatives**, enhancing the efficiency of network security operations.
- **Scalability & Performance Optimization**, ensuring AI models can handle high-traffic environments with minimal latency.
- **Privacy-Preserving Traffic Analysis**, leveraging AI for encrypted traffic analysis without decryption.

Problem Description

Traditional security systems don't always catch evolving or unknown threats in network traffic. With the rise of embedded and IoT devices, it's more important than ever to monitor communication effectively. This project addresses that by training ML models to recognize normal and abnormal traffic behavior using real data.

About the Project

We used a labeled dataset from Kaggle with various features representing network traffic. After cleaning and preparing the data, we trained multiple models (like Random Forest, XGBoost, SVM, etc.) and compared how well each performed.

Models we used:

- Random Forest
- XGBoost
- Logistic Regression
- Support Vector Machine (SVM)

- K-Nearest Neighbors (KNN)

We used metrics like accuracy, F1 score, and ROC AUC to evaluate them.

How It Works

1. Load the dataset
2. Clean and preprocess the data
3. Select key features
4. Train ML models
5. Evaluate each model
6. Display and compare results

Process Flow

Load

↓

Preprocess

↓

Feature Selection

↓

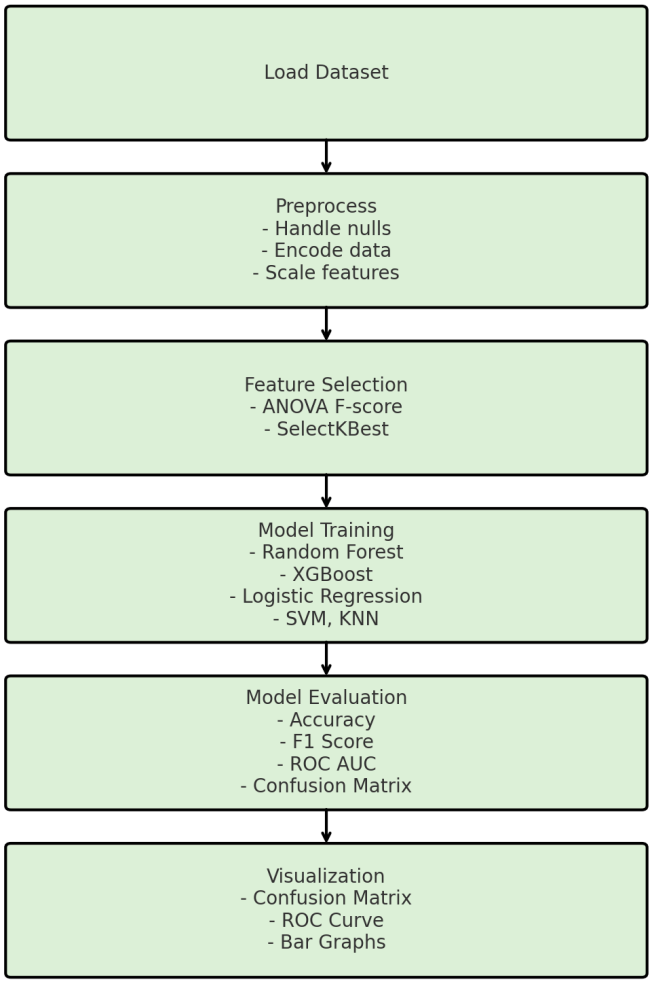
Train

↓

Evaluate

↓

Visualize



Technical Details

- Language: Python
- Used Jupyter Notebook

- Task Type: Classification (normal vs anomalous)
- Evaluation Metrics: Accuracy, F1 Score, ROC AUC, Confusion Matrix

Architecture

- Input: CSV dataset
- Processing: Cleaning, encoding, feature selection
- Models: Multiple classifiers
- Output: Accuracy scores, confusion matrices, ROC curves

Key Technologies and Libraries

- `pandas`, `numpy` – for data handling
- `scikit-learn` – for ML models and metrics
- `xgboost` – gradient boosting
- `matplotlib`, `seaborn` – for graphs and visualizations

Dataset Used

Dataset Overview

This dataset contains real network traffic data collected from embedded systems, labeled with whether the traffic is normal or abnormal. It includes features that describe various properties of network packets, such as packet size, duration, protocol, and port usage.

- Total Records: 100,000+ rows

- Features: Includes multiple numerical and categorical features representing traffic characteristics

- Target Column: `Label` — indicating "Normal" or "Anomalous"

Key Findings

- Feature Importance:
  Using feature selection techniques like ANOVA F-score and SelectKBest, we identified the top contributing features influencing classification (e.g., `Protocol`, `PacketLength`, `Duration`, etc.).

- Anomaly Distribution:
  The dataset is imbalanced, with significantly more normal traffic entries than anomalies. Techniques like stratified train-test splits were used to ensure fair model evaluation.

- Model Performance:
  Random Forest and XGBoost performed the best, achieving over 96% accuracy and ~0.97 ROC AUC, suggesting strong separability between normal and anomalous traffic.

Actionable Insights

- Anomaly Detection Pipelines:
  Machine learning models can be effectively used in real-time embedded systems to monitor and classify network traffic without relying on predefined rule sets.

- Security Applications:
  This approach could be integrated into firewalls or network monitoring tools to flag suspicious behavior automatically, reducing dependency on manual rule configuration.

- Data-Centric Development:
  Future improvements could include real-time streaming data analysis and deploying these models on edge devices for on-device inference.

How to Run the Project

1. Upload:
Users upload the network traffic dataset (CSV format) through the notebook interface or specify the file path manually.

2. Process:
The data is cleaned — missing values, duplicates, and non-numeric features are handled. Features are also scaled for better model performance.

3. Analyze:
Multiple machine learning models (Random Forest, XGBoost, Logistic Regression, etc.) are trained on the data to detect anomalous traffic patterns.

4. Visualize:
Key results like confusion matrices, ROC curves, and accuracy/F1/ROC scores are visualized using Matplotlib and Seaborn to help understand model performance.

5. Interact:
Although the current version doesn't use an AI chatbot or SQL, users can manually test the models with different data samples by modifying inputs in the notebook.

6. Predict:
Once trained, models can be used to make predictions on new or unseen network traffic data, helping identify whether the incoming traffic is normal or anomalous.

VideoExplanation

https://drive.google.com/file/d/1j-DnT9iK9v5aZ8dPJQQ_ymCLyHocnDkH/view?usp=sharing

Members

Meesala Eesha - BU22EECE0100424

B.Bharath Narasimha Sai - BU22EECE0100184

L.Mokshagna Reddy - BU22EECE0100089