Network Traffic Anomaly Detection Using Machine Learning

Xin Hung Chan

July 2025

Abstract

This project presents a machine learning pipeline for detecting network traffic anomalies using the CICIDS2017 dataset. I explored both supervised and unsupervised learning approaches to classify benign and malicious network flows. XGBoost achieved an AUC score of 96.69%, while an autoencoder attained 87.14%, demonstrating the effectiveness of ensemble learning and deep learning for cybersecurity applications. The pipeline incorporates extensive data cleaning, feature engineering, and model evaluation, resulting in robust performance and cool visualizations.

1 Introduction

With the growing volume and sophistication of cyber threats, traditional signature-based intrusion detection systems are becoming insufficient. This project develops a machine learning framework for network anomaly detection, focusing on detecting attacks such as DDoS, PortScan, Infiltration, WebAttack, BruteForce, Heartbleed, and Botnet.

2 Dataset

The CICIDS2017 dataset contains 5.6 million network flow records, collected over several days to simulate realistic traffic. Each record consists of 79 features, capturing statistics like packet lengths, flow durations, and TCP flag counts. The dataset includes both benign traffic and multiple attack types.

The dataset structure is as follows:

Benign Traffic Days: Monday, Tuesday, Wednesday, Friday morning
Attacks: DDoS, PortScan, Infiltration, WebAttack, BruteForce, Heartbleed, Botnet
Attack Rate: 64.14% (very high)
Feature Count: 79 network flow characteristics

3 Data Preprocessing

3.1 Labeling

Labels were gotten from the filenames. Benign traffic was found on Monday, Tuesday, Wednesday, and Friday morning logs, while other files represented attacks.

3.2 Cleaning

I cleaned the data for missing and infinite values. Infinite values were replaced with NaNs and subsequently filled with feature-wise medians. Extremely large values were removed.

3.3 Feature Engineering

I created domain-specific features such as:

- Log and square root transformations of flow duration
- Bytes and packets per second
- Forward/backward packet and byte ratios

These features improved model performance by capturing flow dynamics.

4 Modeling Approaches

4.1 Supervised Learning

Two methods were used:

Random Forest: Used 100 trees with default splitting parameters. Achieved 96.33% AUC.

XGBoost: Tuned with 100 trees, learning rate 0.1, and max depth 6. Achieved the highest performance with an AUC of 96.69%.

4.2 Unsupervised Learning

Isolation Forest: Trained only on benign samples. Performed poorly (AUC 41.62%) probably due to high attack rate.

Autoencoder: Deep neural network with symmetric encoder-decoder structure. Trained to reconstruct benign flows and flagged high-reconstruction-error samples as anomalies. Achieved 87.14% AUC.

5 Results and Analysis

5.1 Supervised Model Performance

Model	Accuracy	AUC	Precision	Recall	F1
Random Forest	92.66%	96.33%	100.00%	88.58%	93.95%
XGBoost	92.82%	**96.69%**	99.91%	88.89%	**94.10%**

Table 1: Supervised model performance on test set

5.2 Unsupervised Model Performance

Model	Accuracy	AUC	Precision	Recall	F1
Isolation Forest	34.11%	41.62%	34%	3%	5%
Autoencoder	51.99%	**87.14%**	91%	28%	43%

Table 2: Unsupervised model performance

5.3 Feature Importance

Top features included:

- **Fwd URG Flags**, **RST Flag Count**, **PSH Flag Count** (XGBoost)
- **Subflow Fwd Packets**, **Destination Port**, **Total Fwd Packets** (Random Forest)

These revealed that packet-level behaviors are highly predictive of anomalies.

6 Visualizations

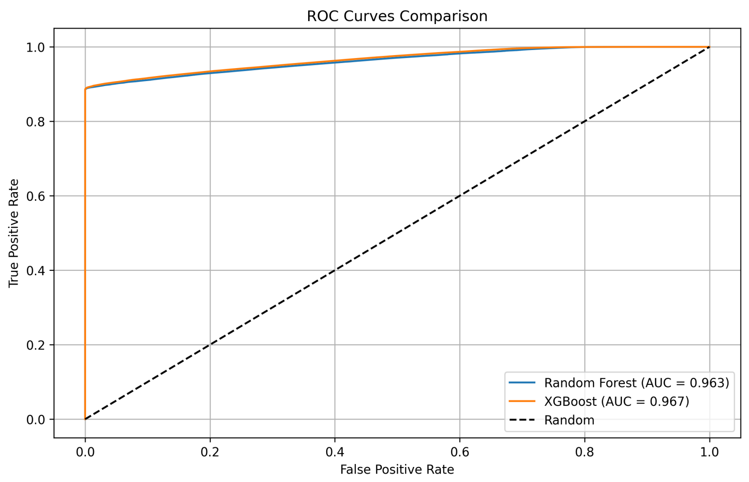

Figure 1: ROC curves for Random Forest and XGBoost

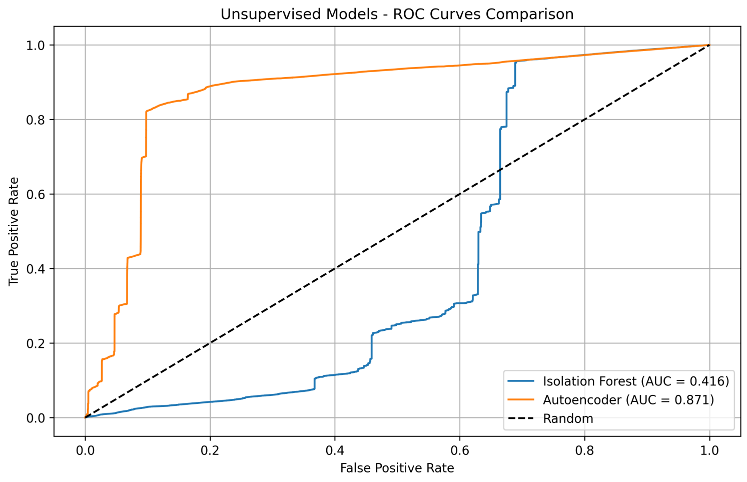

Figure 2: ROC curves for unsupervised models

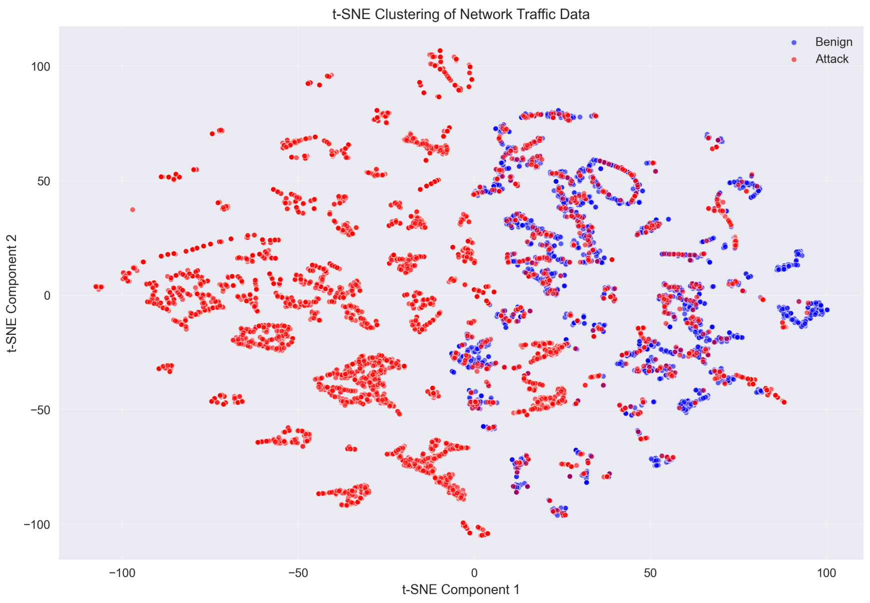

Figure 3: t-SNE Clustering of Network Flows

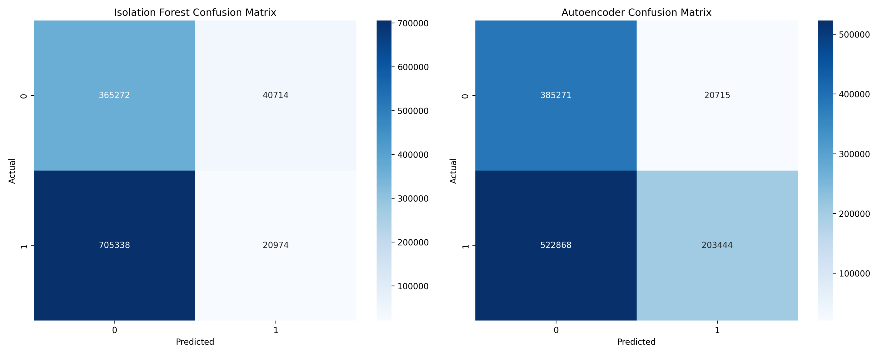

Figure 4: Confusion Matrix for Isolation Forest and Autoencoder

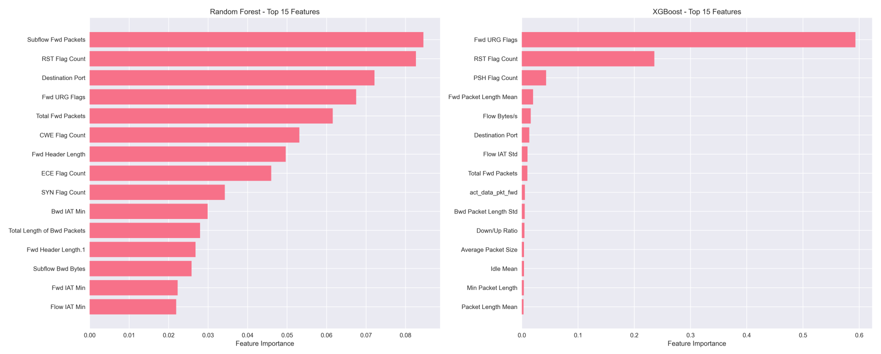

Figure 5: Feature importance of Random Forest and XGBoost

7 Discussion

Supervised models significantly outperformed unsupervised ones due to the availability of labeled data and the high attack rate (64%). XGBoost had excellent detection capabilities with low false positives.

8 Conclusion

This was an interesting project and I learned alot from it, I felt that many parts could be improved upon, such as tuning the hyperparameters for XGBoost, and trying to find ways to make the unsupervised models perform better, but in general I think the anomaly detection part was quite successful and the comparison of the two methods was also pretty comprehensive.

References

1. Sharafaldin, Iman & Habibi Lashkari, Arash & Ghorbani, Ali. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. 108-116.

2. Chen, Tianqi, Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." 2016.

3. Breiman, L. (2001). Random Forests. Machine Learning. 45. 5-32.

4. F. T. Liu, K. M. Ting and Z. -H. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 413-422,

5. Hinton, G.E. & Salakhutdinov, R.R.. (2006). Reducing the Dimensionality of Data with Neural Networks. Science (New York, N.Y.). 313. 504-7.