Intrusion Detection System - BlackWall

Emir Kızılçim
Ahmet Naim Torlak

Description of BlackWall

Inspiration

BlackWall is an intelligent, machine-learning–based network intrusion detection system inspired by the Blackwall concept from Cyberpunk 2077 — a powerful digital barrier designed to protect secure systems from hostile entities in the game.

What we aim

The system analyzes network flow–level traffic data and classifies each connection as BENIGN or ATTACK using supervised machine learning techniques. By extracting and processing statistical features from network flows, BlackWall learns behavioral patterns associated with cyber attacks such as abnormal packet rates, irregular flow timing, and suspicious traffic distributions.

Data Preprocessing

How artificial intelligence is revolutionizing network security

Our data, CIC-IDS-2017 has over 3.5 millions data. It is too much for us to study on it. We used only %10 of it to train. But number of attacks were too low.

This is the version of dataset after feature elimination:

```
(2522362, 80)
Attack Type
BENIGN          2096484
DoS              193748
DDoS             128016
Port Scan         90819
Brute Force        9152
Web Attack         2143
Bot                1953
Infiltration         36
Heartbleed           11
Name: count, dtype: int64
```

Data Preprocessing

Handling Missing & Infinite Values

- **Problem: Real-world network traffic data often contains missing (NaN) or infinite values from measurement errors**
- **Solution:**
  - **NaN values → Replaced with median of each feature**
  - **Infinite values → Replaced with 0**
- **Why median, not mean? Median is robust to outliers**

Data Preprocessing

Label Encoding & Binary Classification

- Original: Multi-class labels (14 attack types + benign)
- Transformed: Binary classification attack-type
  - BENIGN → 0 (Normal traffic)
  - Any Attack → 1 (Malicious traffic)
- Why binary?
  - Simplifies the intrusion detection problem
  - Enables StandardScaler compatibility
  - Focuses on primary goal: detect if traffic is malicious
  - Serves as foundation for multi-class detection later

Data Preprocessing

Data Deduplication

- Removed duplicate rows
- Why?:
  - Prevents model from memorizing repeated patterns
  - Reduces dataset bias toward frequent samples
  - Improves generalization to new, unseen traffic
- More robust model that learns patterns, not specific instances

Data Preprocessing

Memory Optimization

- **Technique: Downcasting numerical types**
  - **int64 → int8**
  - **float64 → float32**
- **Benefits:**
  - **Memory usage reduction**
  - **Faster data loading and processing**
- **Example: Original 1.2+GB dataset → ~500MB processed dataset**

Data Preprocessing

'BENIGN': 'BENIGN',
'DDoS': 'DDoS',
'DoS Hulk': 'DoS',
'DoS GoldenEye': 'DoS',
'DoS slowloris': 'DoS',
'DoS Slowhttptest': 'DoS',
'PortScan': 'Port Scan',
'FTP-Patator': 'Brute Force',
'SSH-Patator': 'Brute Force',
'Bot': 'Bot',
'Web Attack \ufffd Brute Force': 'Web Attack',
'Web Attack \ufffd XSS': 'Web Attack',
'Web Attack \ufffd Sql Injection': 'Web Attack',
'Infiltration': 'Infiltration',
'Heartbleed': 'Heartbleed'

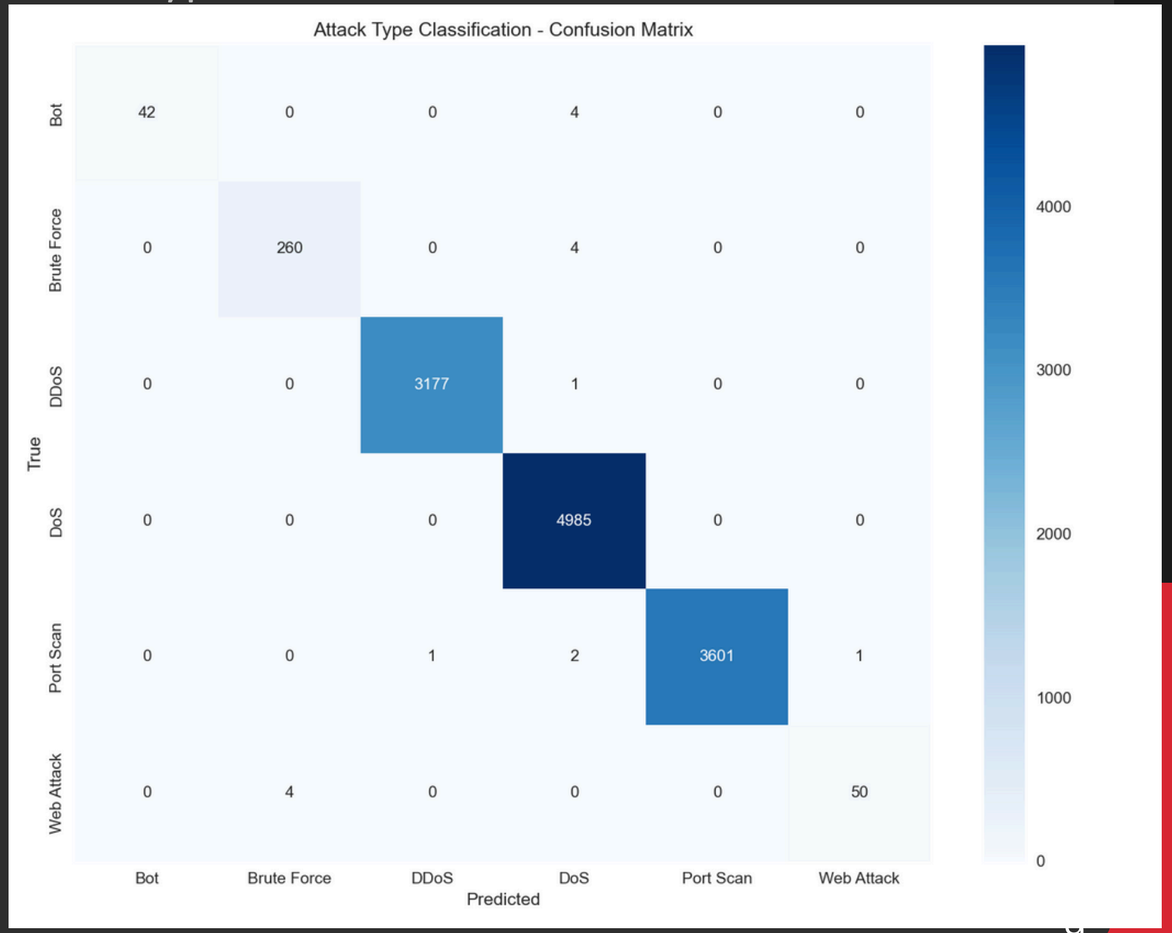
```
Scaler fitted
Selecting best features...
Feature selector fitted
PCA fitted with 35 components (explained variance: 0.999)
Final feature shape: (271146, 35)
Final feature shape: (271146, 35)

Dataset Info:
Training samples: 189802
Testing samples: 81344
Features: 35
Class balance: 0.180 attack ratio
```

15 Classes → 9 Classes

Multi-Classification Model

Classification Report (Random Forest) For Classifying Attack Types

	precision	recall	f1-score	support
Bot	1.000	0.913	0.955	46
Brute Force	0.985	0.985	0.985	264
DDoS	1.000	1.000	1.000	3178
DoS	0.998	1.000	0.999	4985
Port Scan	1.000	0.999	0.999	3605
Web Attack	0.980	0.926	0.952	54
accuracy			0.999	12132
macro avg	0.994	0.970	0.982	12132
weighted avg	0.999	0.999	0.999	12132



Feature Elimination

The Curse of Dimensionality Problem

- **79 features in CIC_IDS_2017**
- **Problem: High dimensionality leads to:**
  - **Overfitting: Models memorize noise instead of learning patterns**
  - **Computational cost: Exponential training time increase**
  - **Multicollinearity: Redundant features confuse models**
  - **Hughes phenomenon: Accuracy decreases with too many features**

Feature Elimination

Two-Stage Feature Selection Pipeline

Stage 1: Statistical Feature Filtering

- **Method: ANOVA F-test (SelectKBest)**
- **Selection: Top 50 most discriminative features**
- **How it works:**
- **For each feature:**
- **F-score = (Variance between attack/normal classes) / (Variance within classes)**
- **Higher F-score = Feature better separates attacks from normal traffic**
- **Result: Eliminated 30 irrelevant feature**

Feature Elimination

Stage 2: Dimensionality Reduction & Decorrelation

- **Method: Principal Component Analysis (PCA)**
- **Reduction: 50 features → 35 principal components**
- **How PCA works:**
  a. **Standardize features to zero mean, unit variance**
  b. **Compute covariance matrix (feature relationships)**
  c. **PCA transformed correlated network flow features into uncorrelated components, reducing multicollinearity and overfitting. (e.g. Total Fwd Packets, Total Backward Packets, Total Packets)**
  d. **Select top 35 components capturing 95% variance (Best scenario is 30)**

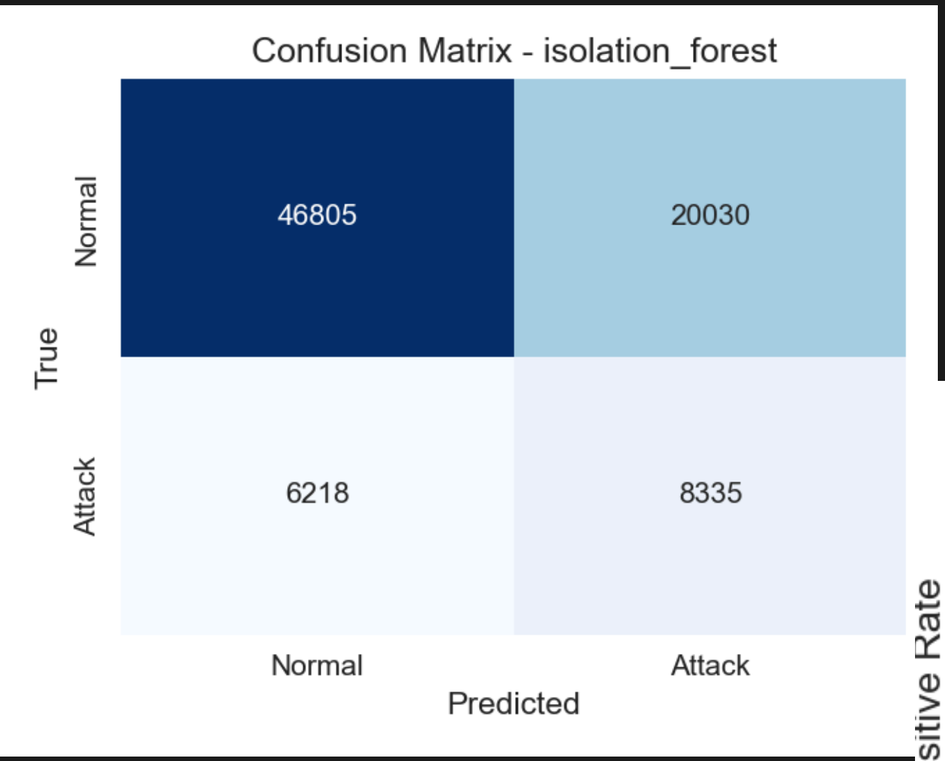
Feature Elimination

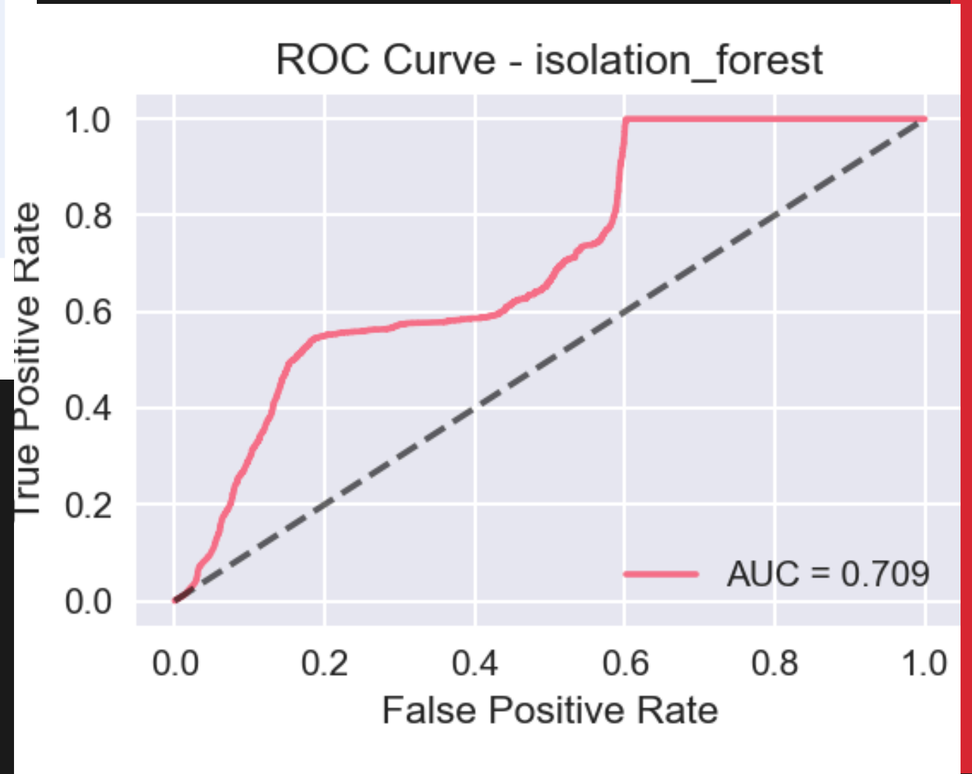
We still kept PCA even if it’s a tree model, why?:

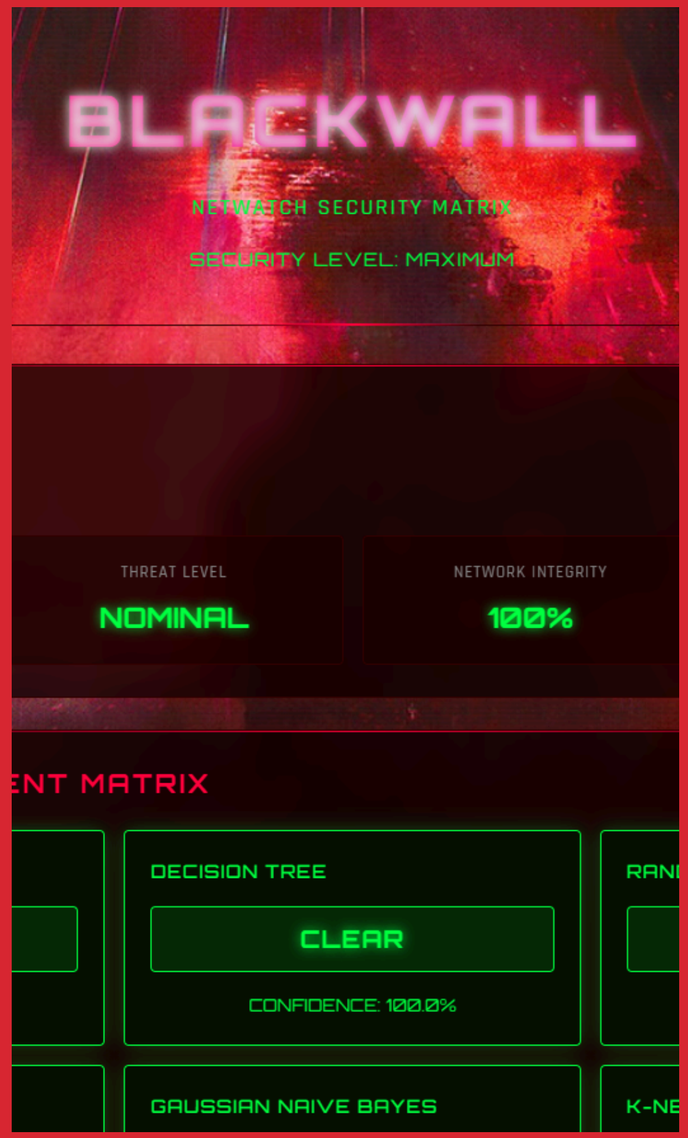
- In theory, trees can work without PCA.
- However, CIC-IDS-2017 contains many correlated flow features, and PCA helped reduce split instability and overfitting.
- Since our goal is real-time robustness rather than interpretability, PCA was beneficial.

Model Overviews

ISOLATION FOREST

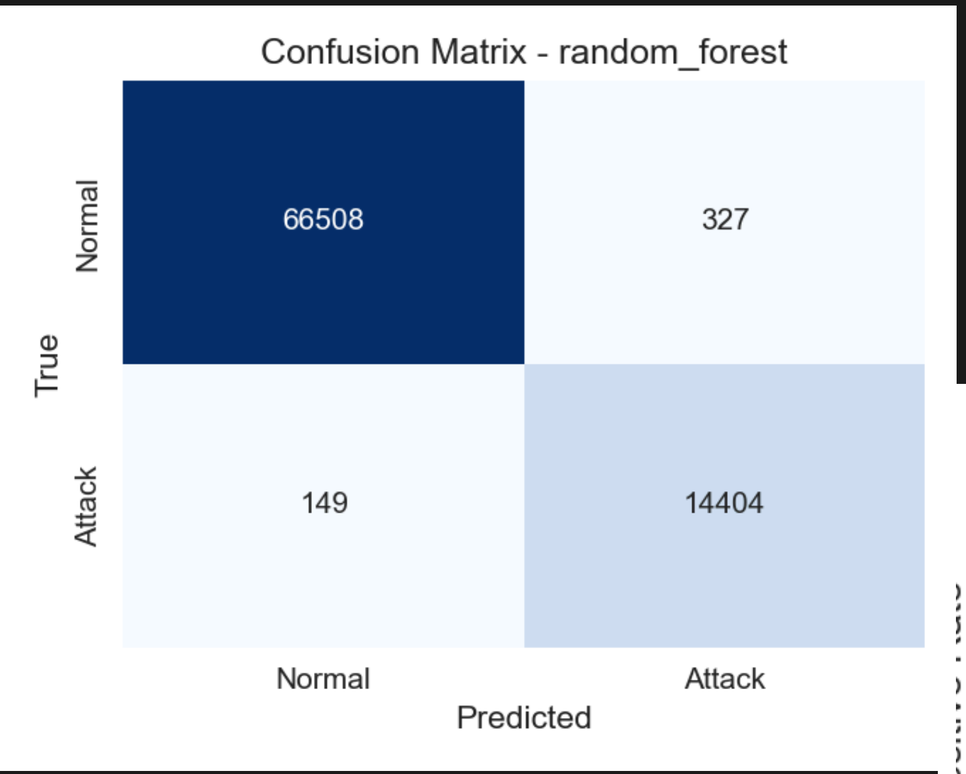


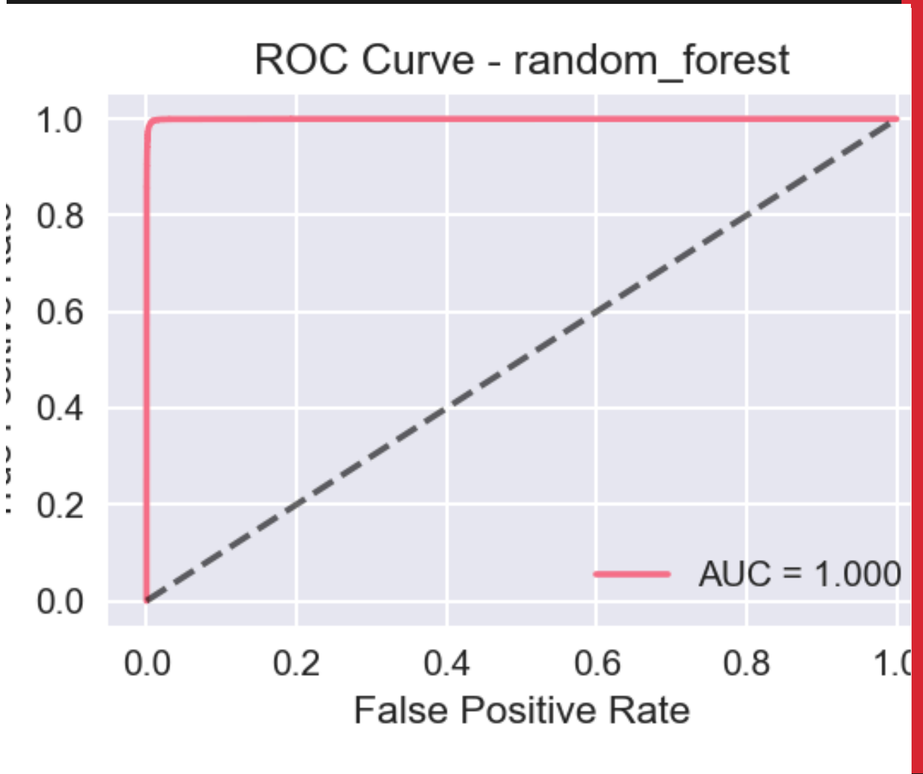




Model Overviews

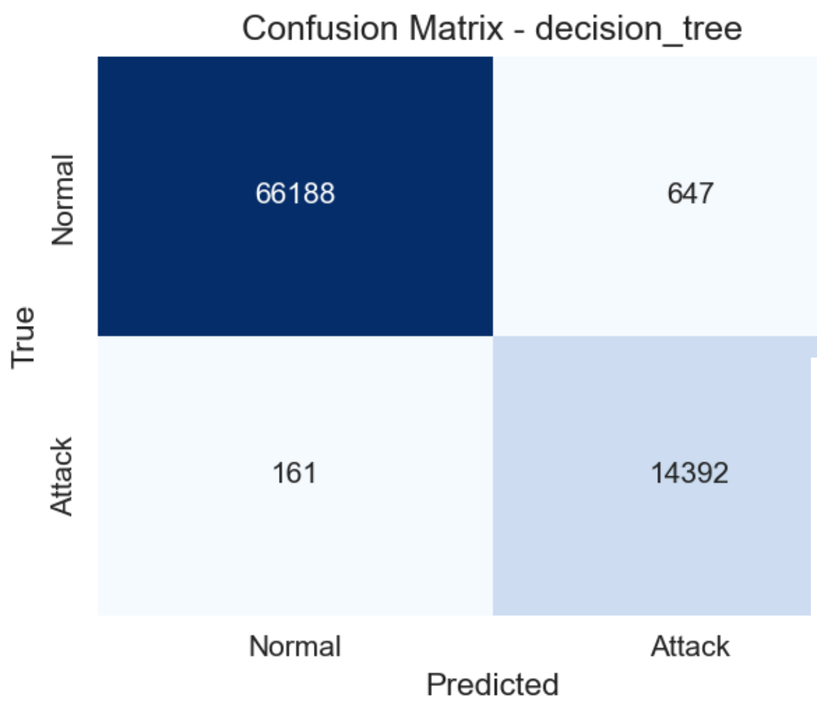
RANDOM FOREST

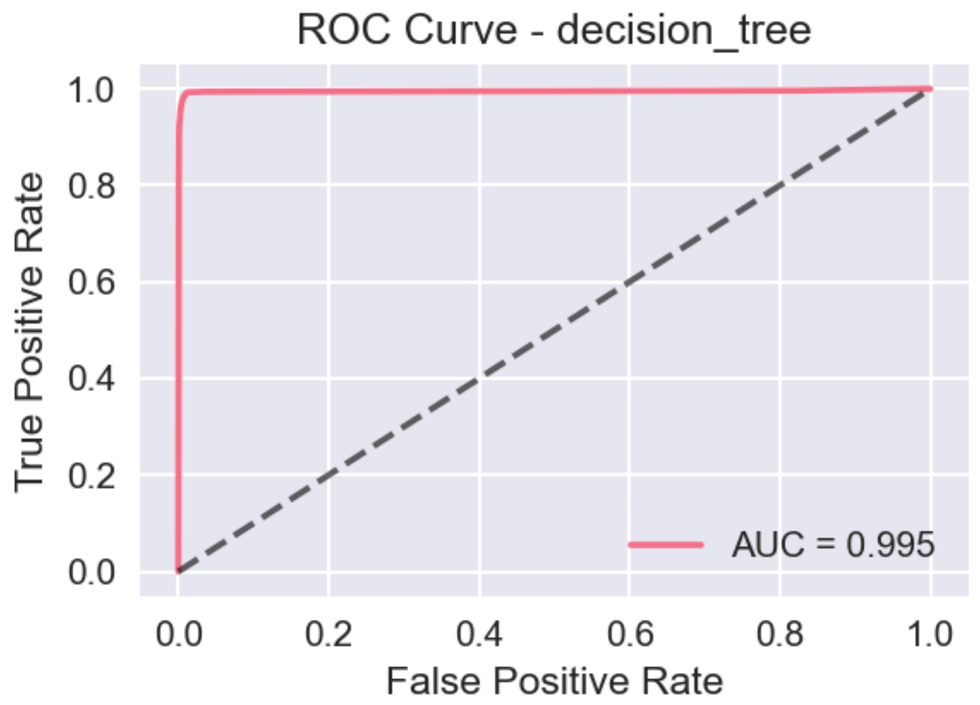


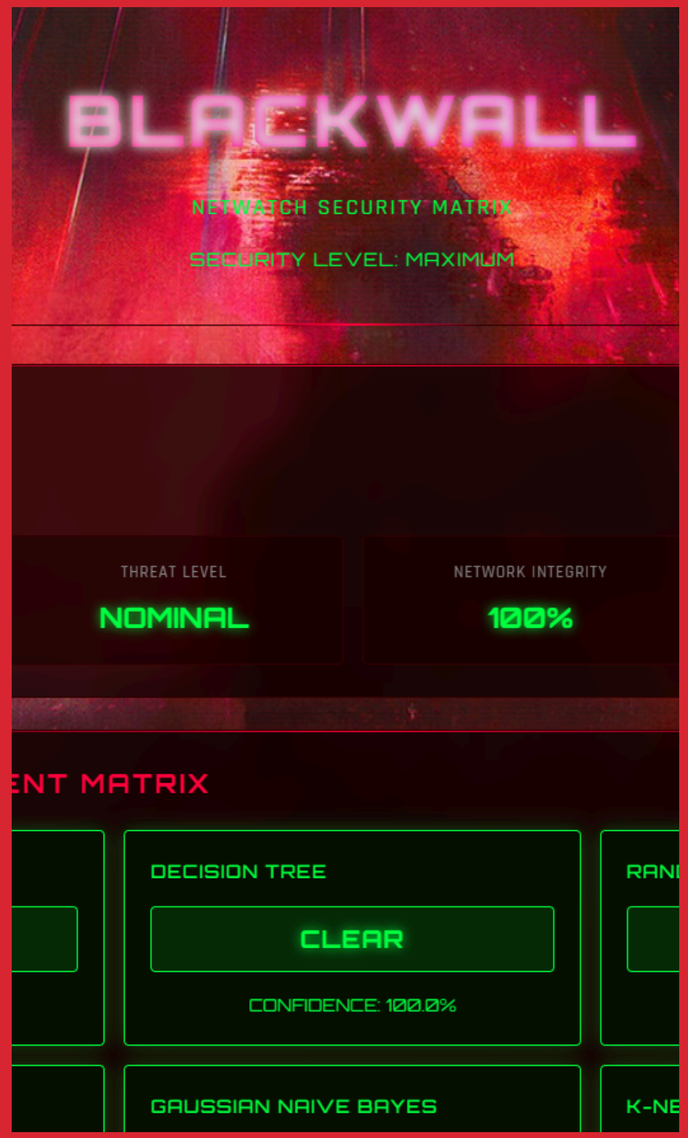


Model Overviews

DECISION TREE - 2ND BEST MODEL*





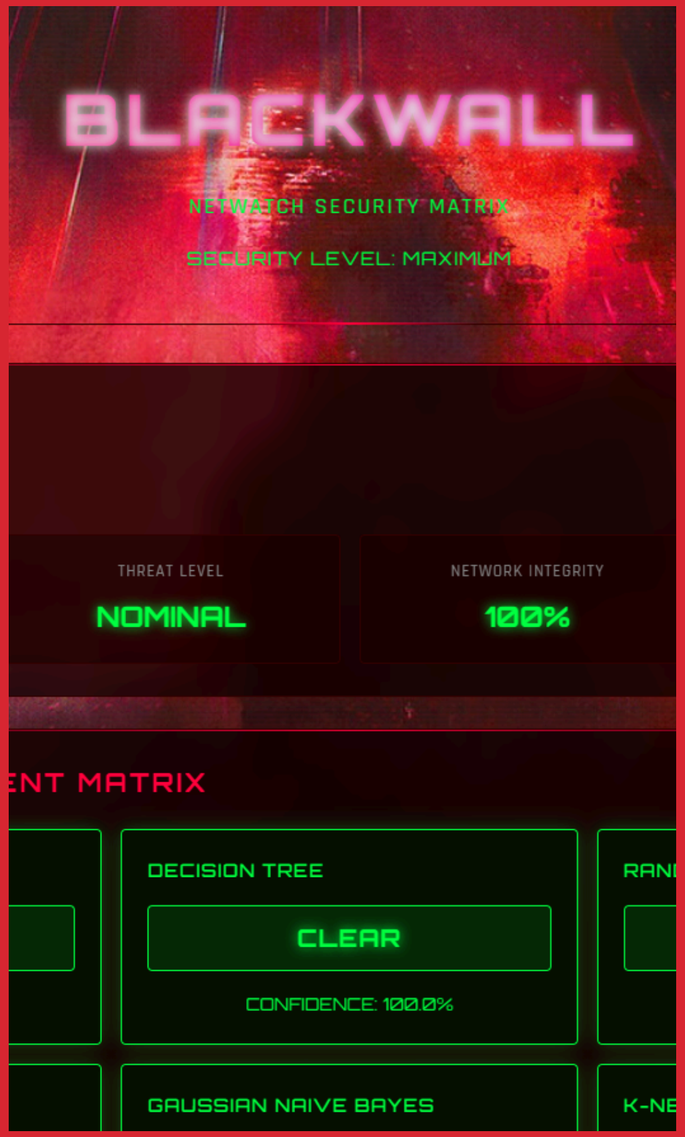


Model Overviews

DECISION TREE

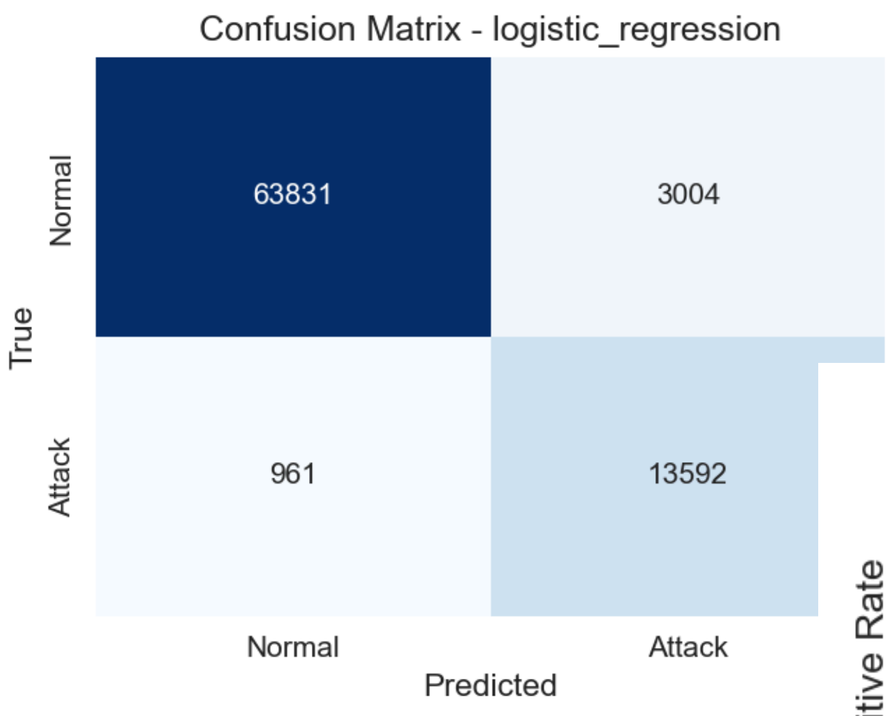
	precision	recall	f1-score	support
Normal	0.998	0.990	0.994	66835
Attack	0.957	0.989	0.973	14553
accuracy			0.990	81388
macro avg	0.977	0.990	0.983	81388
weighted avg	0.990	0.990	0.990	81388

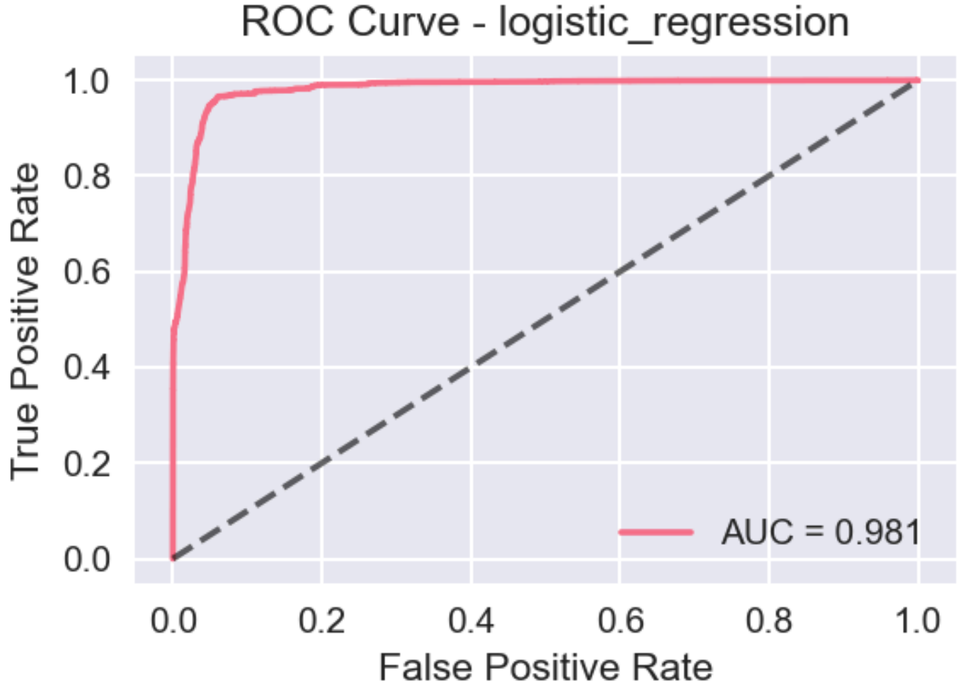
ROC-AUC: 0.995

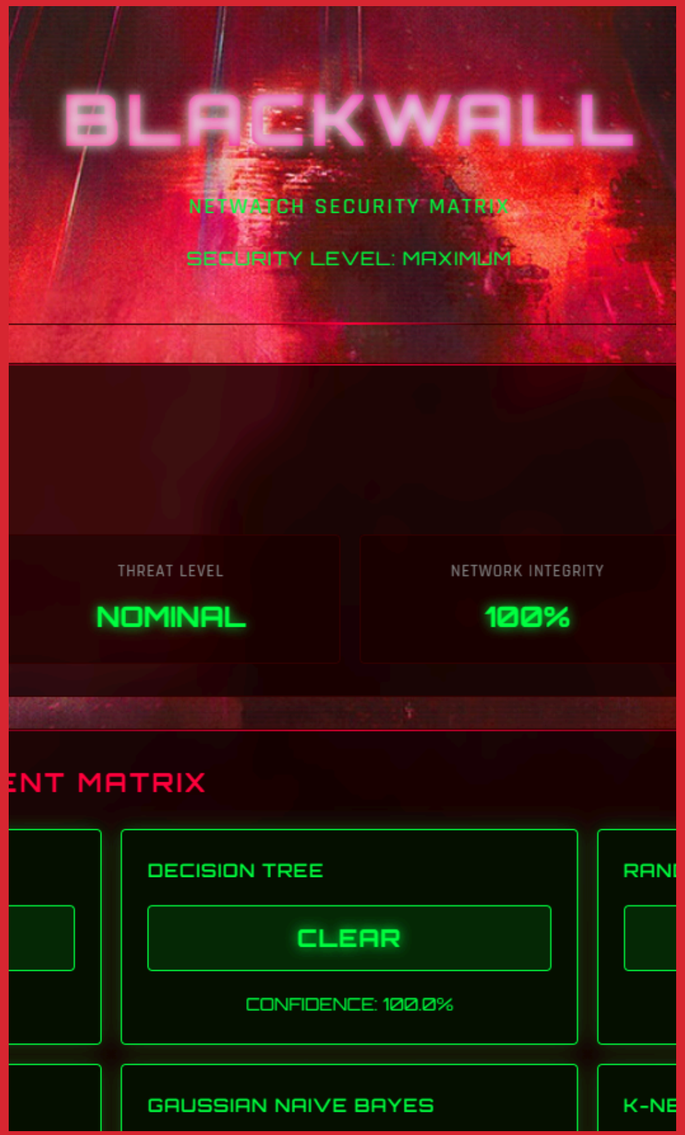


Model Overviews

LOGISTIC REGRESSION - 1ST BEST MODEL*





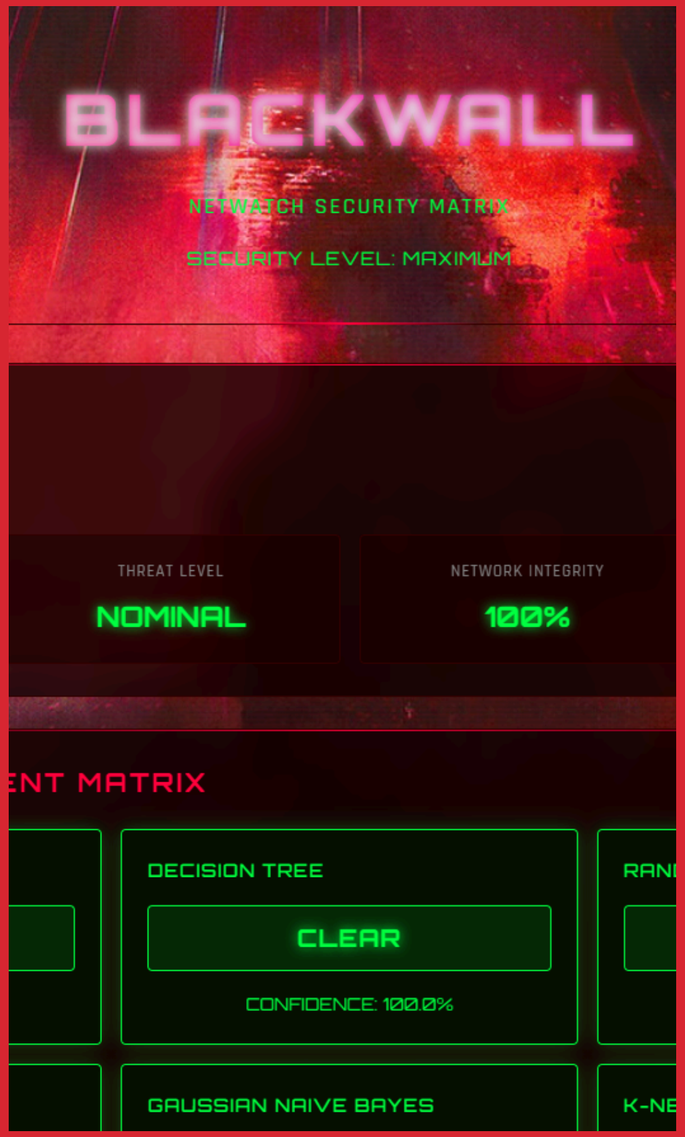


Model Overviews

LOGISTIC REGRESSION

	precision	recall	f1-score	support
Normal	0.985	0.955	0.970	66835
Attack	0.819	0.934	0.873	14553
accuracy			0.951	81388
macro avg	0.902	0.945	0.921	81388
weighted avg	0.955	0.951	0.953	81388

ROC-AUC: 0.981



Model Overviews

ALL MODEL COMPARISON

Methods We Tried

Hyperparameter Tuning

The extreme class imbalance and dataset size limited its effectiveness.

Then what about using SMOTE?

Methods We Tried

SMOTE

SMOTE is a bad idea for BlackWall because it creates fake network attacks that never existed.

UnderSampling

Random undersampling was avoided because removing large volumes of benign traffic reduces the diversity of normal network behavior.

Methods We Tried

PyShark → Scapy

We planned to use PyShark first, but we switched to Scapy because it gave us faster packet access and more control over flow construction in real time.

Methods We Tried

Deep Learning Testing - MLP

Without temporal or raw packet representations, deep learning does not gain a structural advantage over classical models.

When applied to raw packet streams or temporal sequences, deep learning typically outperforms classical models.
However, with aggregated flow features, tree-based methods are more suitable.

Methods We Tried

Deep Learning Testing - MLP

```python
mlp = MLPClassifier(
    hidden_layer_sizes=(64, 32),
    activation='relu',
    solver='adam',
    alpha=1e-4,
    learning_rate='adaptive',
    max_iter=40,
    early_stopping=True,
    validation_fraction=0.15,
    random_state=42
)
```

```
=== MLP Classification Report ===
              precision    recall  f1-score   support

           0       0.99      0.99      0.99    127763
           1       1.00      0.99      0.99    188684

    accuracy                           0.99    316447
   macro avg       0.99      0.99      0.99    316447
weighted avg       0.99      0.99      0.99    316447

=== Confusion Matrix ===
[[126946    817]
 [  1126 187558]]
{'ATTACK': 0, 'BENIGN': 1}
```

Methods We Tried

Isolation Forest Testing

Attacks are not anomalies here.

The attacks actually creates a cluster.

High-dimesionality was such a huge problem.

Thank you!

Demo Time