Anomaly Detection in Network Traffic Logs Using Machine Learning

First we have taken network traffic log from this site **: https://www.cecresearch.com/**  
Now we .

1. take the sample.csv file as unsupervised log data .
2. make this data classified using **Isolation Forest Algorithm** .
3. it level tahe data with ‘**Normal** ’ or ‘**anamoly** ’ based on [

'Flow\_Duration', 'Total\_Fwd\_Packets', 'Total\_Backward\_Packets',

'Fwd\_Packet\_Length\_Max', 'Bwd\_Packet\_Length\_Max',

'Flow\_Bytes\_s', 'Flow\_Packets\_s',

'Fwd\_IAT\_Mean', 'Bwd\_IAT\_Mean', 'Packet\_Length\_Variance',

'Average\_Packet\_Size'] values . and create ‘classified\_traffic\_full.csv’ which is classified .

1. now we train the model mechine with **Random Forest Classifier .**
2. It create **‘rf\_anomaly\_model.pkl’** as trained model .
3. Now we tell the model to check the traffics for anomaly detection on ‘sample.csv’ which is non-leveled data.
4. Mechine check the data for any anomaly and give a output name ‘**test\_output.csv’** , it contain the leveled data by mechine.
5. Now we match the leveled output data with ‘**classified\_traffic\_full.csv’** data which was leveled by **‘Isolation Forest Algorithm’ .**
6. We calculate the error percentse and Fasle positive and false negative ratio of anaomoly data .

Program sample :

Input 1 :

import os

import pandas as pd

import numpy as np

import joblib

from sklearn.ensemble import RandomForestClassifier, IsolationForest

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report, f1\_score

from colorama import Fore, Style, init

import seaborn as sns

import matplotlib.pyplot as plt

# Initialize colorama

init(autoreset=True)

# Feature list

features = [

    'Flow\_Duration', 'Total\_Fwd\_Packets', 'Total\_Backward\_Packets',

    'Fwd\_Packet\_Length\_Max', 'Bwd\_Packet\_Length\_Max',

    'Flow\_Bytes\_s', 'Flow\_Packets\_s',

    'Fwd\_IAT\_Mean', 'Bwd\_IAT\_Mean', 'Packet\_Length\_Variance',

    'Average\_Packet\_Size'

]

Model train :

 df\_labeled = pd.read\_csv(labeled\_file)

    df\_labeled.columns = df\_labeled.columns.str.strip().str.replace(' ', '\_').str.replace('/', '\_')

    # Filter rows where class is not null

    df\_labeled = df\_labeled.dropna(subset=['class'])

    df\_labeled['class'] = df\_labeled['class'].map({'normal': 0, 'anomaly': 1})

    X\_train = df\_labeled[features].replace([np.inf, -np.inf], np.nan).dropna()

    y\_train = df\_labeled.loc[X\_train.index, 'class']

    clf = RandomForestClassifier(n\_estimators=100, random\_state=42, class\_weight='balanced')

    clf.fit(X\_train, y\_train)

    joblib.dump(clf, model\_file)

    print(Fore.GREEN + f"✅ Model trained and saved as '{model\_file}'")

confution matrix generate :

user\_input = input(Fore.YELLOW + "\n📈 Do you want to view the confusion matrix plot? (yes/no): ").strip().lower()

if user\_input in ['yes', 'y']:

    plt.figure(figsize=(10, 7))

    sns.heatmap(conf\_matrix, annot=True, fmt='d', cmap='Blues',

                xticklabels=['Predicted Normal', 'Predicted Anomaly'],

                yticklabels=['Actual Normal', 'Actual Anomaly'])

    stats\_text = (

        f"Total: {total}    Correct: {right\_total} ({right\_percent:.2f}%)\n"

        f"Wrong: {wrong\_total} ({wrong\_percent:.2f}%)\n"

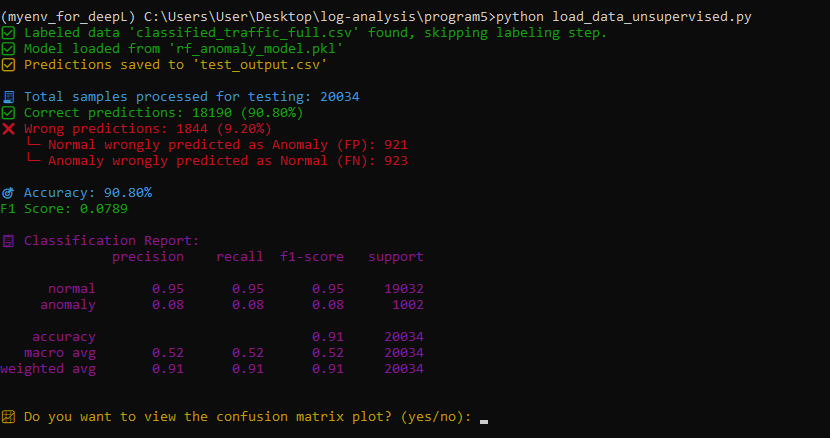
        f"Normal → Anomaly (FP): {fp}   Anomaly → Normal (FN): {fn}\n"

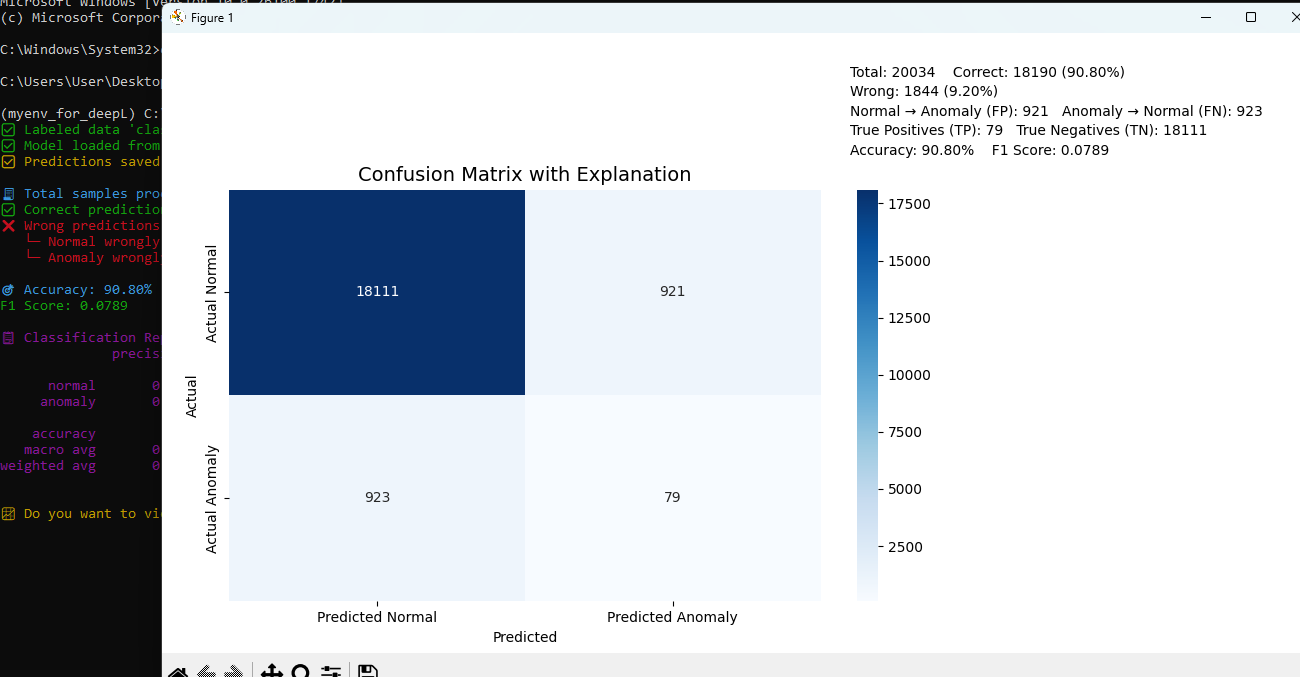
        f"True Positives (TP): {tp}   True Negatives (TN): {tn}\n"

        f"Accuracy: {accuracy \* 100:.2f}%    F1 Score: {f1:.4f}"

    )

Run the program :



Matrix outpur with false positive and false negative :   


**Summary: How Isolation Forest Detects Normal vs. Anomalous Traffic**

Isolation Forest is an **unsupervised anomaly detection algorithm** that identifies unusual network traffic by isolating data points based on their feature values.

* The algorithm builds multiple **random decision trees** by recursively splitting data on randomly selected features and values.
* **Normal traffic** data points tend to cluster together with similar feature values (e.g., moderate flow duration, balanced packet counts, consistent packet sizes). These points require **more splits to isolate** because they lie in dense regions.
* **Anomalous traffic** data points have **rare or extreme values** in one or more features (e.g., very short flow duration, unusually high packet rate, abnormal packet sizes). Because of their rarity and distinctiveness, they get **isolated quickly with fewer splits**.
* The **path length** (number of splits needed to isolate a point) averaged over all trees determines an **anomaly score**.
* Points with **short average path lengths** are flagged as anomalies, while those with longer paths are considered normal.

By analyzing key features like **flow duration, packet counts, packet sizes, and packet timing**, Isolation Forest effectively separates normal network behavior from suspicious traffic without needing pre-labeled data.

summary for **Random Forest Classifier** :

### Summary: How Random Forest Classifier Works for Anomaly Detection

Random Forest is a **supervised ensemble learning algorithm** that builds multiple decision trees to classify data points, such as distinguishing normal from anomalous network traffic.

* It trains many **decision trees** on random subsets of the labeled training data and features.
* Each tree independently predicts the class (normal or anomaly) for a given data point.
* The **final prediction** is made by aggregating votes from all trees (majority voting).
* This ensemble approach **reduces overfitting** and improves generalization compared to a single decision tree.
* The model learns from **labeled data**, identifying complex patterns across features like flow duration, packet counts, and packet sizes.
* It is effective in handling high-dimensional data and capturing nonlinear relationships, making it suitable for detecting subtle anomalies in network traffic.

By using a labeled dataset (created via Isolation Forest or manual labeling), Random Forest can **accurately classify traffic as normal or anomalous**, providing interpretable results and feature importance insights.

# Workflow Diagram Description: Isolation Forest & Random Forest with Math

Conceptual Diagram Layout (suggested)

┌─────────────────────────────┐

│ Input Dataset │

│ Unlabeled (Isolation Forest)│

│ Labeled (Random Forest) │

└──────────────┬──────────────┘

│

┌─────────────────────────┴───────────────────────────┐

│ │

┌───────────────┐ ┌───────────────┐

│ Isolation │ │ Random Forest │

│ Forest │ │ Classifier │

│ (Unsupervised)│ │ (Supervised) │

└───────┬───────┘ └───────┬───────┘

│ │

Random splits (random feature & split) Bootstrap sampling

│ │

Build many trees recursively Build many decision trees

│ │

Compute path length \(h(x)\) for each sample Use impurity measures (Gini)

│ │

Calculate anomaly score \(s(x)\) Each tree votes class label

│ │

Anomaly if \(s(x)\approx 1\), else normal Majority vote aggregates

│ │

└───────────────┐ ┌───────┘

│ │

Generate anomaly labels Predict traffic labels

│ │

└──────────────┬─────────────┘

│

Train Random Forest model (supervised)

│

Predict & classify test samples

│

Evaluate & report

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Feature | Value | Interpretation | Predicted Label | | Flow\_Duration | 15000 | Normal duration | Normal | | Total\_Fwd\_Packets | 20 | Balanced number of packets | Normal | | Flow\_Bytes\_s | 500 | Moderate bytes per second | Normal | | Fwd\_IAT\_Mean | 100 | Reasonable average packet interval | Normal | | Flow\_Duration | 100 | Very short flow duration | Anomaly | | Total\_Fwd\_Packets | 1 | Very low packets | Anomaly | | Flow\_Bytes\_s | 10000 | Extremely high bytes per second | Anomaly | | Fwd\_IAT\_Mean | 1 | Very high frequency packets | Anomaly |  |  |  |  |  | | --- | --- | --- | --- | | **Feature** | **Typical Normal Value Range/Behavior** | **Typical Anomalous Value Behavior** | **Why It Matters for Anomaly Detection** | | **Flow\_Duration** | Moderate to high values (e.g., 1000 ms to minutes) | Extremely low or very high values | Anomalies may have unusually short or very long flow durations | | **Total\_Fwd\_Packets** | Balanced packet counts in forward direction | Very low or very high packet counts | Too few or too many packets can indicate suspicious behavior | | **Total\_Backward\_Packets** | Balanced packet counts in backward direction | Very low or none, or extremely high | Absence or excess of response packets can indicate anomalies | | **Fwd\_Packet\_Length\_Max** | Normal max packet sizes (depends on protocol, e.g., 40-1500 bytes) | Very small or abnormally large packet sizes | Abnormal packet sizes can reflect malformed or malicious traffic | | **Bwd\_Packet\_Length\_Max** | Normal max packet sizes in backward packets | Very small or very large | Same as forward direction | | **Flow\_Bytes\_s** | Moderate bytes per second typical for flow | Extremely high or low byte rates | Sudden spikes or drops in traffic rates signal anomalies | | **Flow\_Packets\_s** | Typical packet rates consistent with normal flow | Very high packet rate or very low | Burst traffic or idle flows can be anomalous | | **Fwd\_IAT\_Mean** | Moderate average time between packets (milliseconds) | Very low (high frequency) or very high (delays) | Unusual timing between packets can indicate scanning or flooding | | **Bwd\_IAT\_Mean** | Similar to forward IAT mean | Very low or very high | Same reason as above | | **Packet\_Length\_Variance** | Normal variance indicating diversity in packet sizes | Very low (all packets same size) or very high variance | Too uniform or erratic packet sizes can be suspicious | | **Average\_Packet\_Size** | Average sizes consistent with normal communication | Very small or very large average packet sizes | Indicates abnormal payloads or malformed packets | |