Github Link: <https://github.com/Gopisankar19/data-analytic1.git>

# Project Title: DETECTING CYBER THREATS THROUGH ANOMALY DETECTION IN NETWORK TRAFFIC DATA.

# PHASE-2

# Student Name: GOPISANKAR M

# Register Number: 611823104019

# Institution: P.S.V. College of Engineering and Technologies

# Department: B.E–COMPUTER SCIENCE AND ENGINEERING

# Date of Submission: 8 MAY 2026

## Problem Statement

The rapid expansion of digital communication and cloud-based services has significantly increased the volume, velocity, and variety of network traffic, making it more challenging to safeguard systems from cyber threats. Traditional rule-based security systems are increasingly inadequate against sophisticated, evolving, and previously unseen cyberattacks. These systems often suffer from limitations such as delayed response times and high false positive rates.

Intelligent Data Analytics (IDA), leveraging techniques such as machine learning, deep learning, and statistical modeling, presents a viable solution for enhancing threat detection capabilities. By analyzing large-scale network traffic data in real time, IDA can identify patterns and anomalies that indicate potential intrusions or malicious activity.

This research aims to apply intelligent data analytics techniques for anomaly detection in network traffic data to detect cyber threats efficiently and accurately. The focus is on building models that not only detect known attack patterns but can also generalize to identify novel threats with minimal human intervention, low latency, and high precision.

## Project Objectives

 **To collect and preprocess network traffic data** from publicly available datasets or simulated environments to ensure quality and relevance for anomaly detection.

 **To identify and extract key features** from raw network traffic that contribute to distinguishing normal behavior from malicious activity.

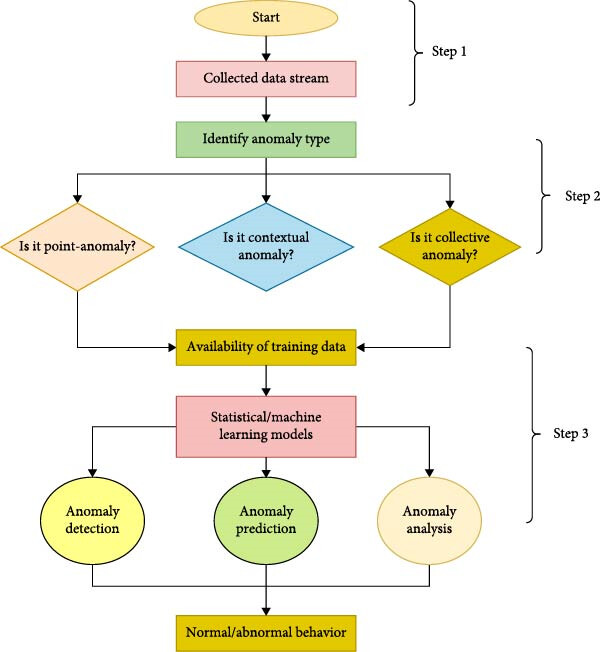
 **To develop and implement intelligent data analytics models** (e.g., machine learning, deep learning, or statistical approaches) for detecting anomalies in network traffic.

 **To evaluate the performance** of the developed models using appropriate metrics such as accuracy, precision, recall, F1-score, and false positive rate.

 **To compare multiple anomaly detection techniques** (e.g., supervised vs. unsupervised learning, statistical vs. neural models) to determine the most effective approach for cyber threat detection.

 **To ensure real-time or near real-time detection capabilities** for practical deployment in modern cybersecurity infrastructures.

## Flowchart of the Project Workflow

****

1. **Data Description**

**4.1. Dataset Source**

* **Example Datasets:**
  + **CICIDS2017: Provided by the Canadian Institute for Cybersecurity; includes realistic benign and attack traffic (e.g., DDoS, brute force, port scanning).**
  + **NSL-KDD: A refined version of the KDD’99 dataset; commonly used for benchmarking intrusion detection systems.**
  + **UNSW-NB15: Generated using IXIA PerfectStorm, includes nine attack types with modern traffic patterns.**

**4.2. Data Type**

* **Structured Tabular Data: Each row represents a network connection or packet flow.**
* **Format: CSV (Comma-Separated Values), PCAP (for raw packet capture), or ARFF (for machine learning tools).**

**4.3. Key Features**

* **Basic Features:**
  + **Duration**
  + **Protocol type (TCP, UDP, ICMP)**
  + **Source and Destination IP/Port**
* **Traffic Features:**
  + **Number of bytes sent/received**
  + **Packet counts in forward/backward directions**
  + **Flow duration**
* **Time-based Features:**
  + **Time between packets**
  + **Active/idle times**
* **Label:**
  + **Class: Normal or Anomaly (sometimes with specific attack labels)**

**4.4. Preprocessing Needs**

* **Handle missing or inconsistent values.**
* **Normalize numerical features.**
* **Encode categorical variables (e.g., protocol type).**

## Data Preprocessing

 **Handling Missing Values**:

Remove rows with excessive missing data.

Impute missing values using mean/median for numerical features or mode for categorical ones.

 **Removing Duplicates**:

 **Dropping Irrelevant Features**:

* Exclude fields such as timestamps or IDs that do not contribute to anomaly detection.

 **Selecting Informative Features**:

* Use correlation analysis or feature importance (e.g., mutual information or tree-based methods) to identify the most predictive features.

Eliminate duplicate entries that can bias the learning process.

## Exploratory Data Analysis (EDA)

##  Dataset Overview:

## Number of rows and columns

## Data types (categorical, numerical)

## Summary statistics (mean, median, standard deviation)

##  Class Distribution:

## Count of normal vs. anomalous records

## Visualization: Bar plot or pie chart

## 7.Feature Engineering

**1. Feature Selection**

* **Remove Irrelevant/Redundant Features**:
  + Drop identifiers (e.g., Flow ID, Source IP) and timestamps unless time-based features are explicitly needed.
* **Correlation Analysis**:
  + Use a correlation matrix or mutual information to eliminate highly correlated or low-impact features.

**2. Feature Transformation**

* **Encoding Categorical Features**:
  + Use **one-hot encoding** for variables like Protocol, Flag, or Service.
* **Log Transformations**:
  + Apply log transformation to skewed features (e.g., Packet Size, Duration) to normalize their distribution.

**3. Feature Creation**

* **Traffic Flow Features**:
  + Flow Bytes/s, Flow Packets/s
  + Average Packet Size, Packet Length Variance
* **Time-based Features**:
  + Idle Time, Active Time, Time Between Packets
* **Directional Features**:
  + Forward Packet Count, Backward Packet Count
  + Bytes Sent vs. Received Ratio

## 8. Model Building

**A. Unsupervised Models *(No labeled data required)*:**

* **Isolation Forest: Efficiently isolates anomalies based on random partitioning of feature space.**
* **One-Class SVM: Learns the boundary of the normal class and flags outliers.**
* **Autoencoders (Neural Networks): Learns to reconstruct normal traffic; high reconstruction error indicates an anomaly.**
* **K-Means Clustering: Identifies anomalies based on distance from cluster centroids.**

**B. Supervised Models *(Requires labeled data: normal vs. attack)*:**

* **Random Forest: Robust and interpretable; works well with mixed feature types.**
* **XGBoost/LightGBM: Gradient boosting models that are fast and accurate.**
* **Logistic Regression: Simple and efficient for binary classification.**
* **Deep Neural Networks: Suitable for capturing complex, nonlinear patterns.**

**8.2. Model Training Workflow**

1. **Split Data: Train/test split (e.g., 70/30 or 80/20)**
2. **Train the Model: Fit the model using the training dataset**
3. **Hyperparameter Tuning: Use Grid Search or Random Search with Cross-Validation**
4. **Model Saving: Save the best model using joblib or pickle for later use**

## 9. Visualization of Results & Model Insights

##  Confusion Matrix

## Shows True Positives, False Positives, True Negatives, and False Negatives.

## Helps evaluate how well the model distinguishes between normal and malicious traffic. *Tool: seaborn.heatmap()*

##  ROC Curve and AUC Score

## Illustrates the trade-off between True Positive Rate and False Positive Rate.

## Higher AUC indicates better discrimination between classes. *Tool: sklearn.metrics.roc\_curve()*

##  Precision-Recall Curve

## Especially useful for imbalanced datasets to evaluate detection quality. *Tool: sklearn.metrics.precision\_recall\_curve()*

## 10. Tools and Technologies Used

**Python:**

* **The primary language for data processing, model building, and analysis due to its rich ecosystem of libraries and ease of use in machine learning and data science.**

 **Pandas**:

* Used for data manipulation, cleaning, and transformation. It provides data structures such as DataFrames for easy handling of tabular data.

 **NumPy**:

* Used for numerical computations, especially for handling arrays and matrices that are fundamental for machine learning tasks.

 **scikit-learn**:

* A versatile library for classical machine learning models (e.g., Isolation Forest, One-Class SVM, Random Forest) and evaluation metrics (e.g., ROC-AUC, Precision, Recall).

 **TensorFlow / Keras**:

* Used for building deep learning models like Autoencoders for unsupervised anomaly detection. Provides a flexible framework for neural networks and other advanced models.

## 11. Team Members .

* + - PRASANTH V  **-** *Data cleaning*
    - PERARASU S **-** *EDA*
    - SAMUEL Y **-** *Feature engineering*
    - GOPISANKAR M **-** *Model development*
  + *Clearly mention who worked on:*
    - *Data cleaning*
    - *EDA*
    - *Feature engineering*
    - *Model development*
    - *Documentation and reporting]*