ABSTRACT

A life or two will be lost in a typical house endanger by fire, flood, or other calamities, but an IoV system can save a life here-a death from death superficial for just eight seconds. However, when a car comes within a certain distance to another, they become ready to kill. At this juncture, he might be intoxicated, and the car is also set for death by two cars coming together. So much has changed for Internet of Vehicles (IoV) systems that making that inter-communication between vehicle and infrastructure intelligent has made the entire system complex and interlinked that it very greatly leads to many cyber vulnerabilities. The model provides a secure and robust machine-learning-based framework for real-time intrusion detection in IoV networks toward improved vehicular cybersecurity. The model achieves these stated goals by utilizing a publicly available network traffic dataset containing more than 175,000 records to differentiate between attack traffic and normal traffic based on 45 diverse features, including protocol behavior, byte and packet flow, and timing characteristics. In our first attempt at measuring detection performance, we implemented a trimmed Decision Tree classifier as a base reference. Two more methods were then suggested to maximize enhanced robustness and accuracy; the first is HYB XGBoost, which is very efficient and manages imbalances during the learning process, and second is a stacking ensemble model incorporating several classifiers to form more diverse decision boundaries. The realization of these models was subjected to extensive testing and validation against a set of performance metrics including accuracy, precision, recall, and F1 score. Results showed enhancement of models geared to specifically detect complex intrusion patterns with much subtlety regarding attack relating interferences.This research gives a private contribution in intelligent transport applications as it proposes a real-time yet light and efficient monitoring frame work that can be fitted on an IoV. The strengthening of the entire system against ever-changing threats while minimizing false-positive rates towards a safer vehicular environment is achieved through combining it with ensemble learning mechanisms.

Keywords: Machine Learning, Cyber Security, Decision Tree, Network Traffic, Internet of Vehicles (IoV)

INDEX

Table Contents

CHAPTER 1 – INTRODUCTION

A Secure and Robust Machine Learning Model for Intrusion Detection in Internet of Vehicles

A life or two will be lost in a typical house endanger by fire, flood, or other calamities, but an IoV system can save a life here-a death from death superficial for just eight seconds. However, when a car comes within a certain distance to another, they become ready to kill. At this juncture, he might be intoxicated, and the car is also set for death by two cars coming together. So much has changed for Internet of Vehicles (IoV) systems that making that inter-communication between vehicle and infrastructure intelligent has made the entire system complex and interlinked that it very greatly leads to many cyber vulnerabilities. The model provides a secure and robust machine-learning-based framework for real-time intrusion detection in IoV networks toward improved vehicular cybersecurity. The model achieves these stated goals by utilizing a publicly available network traffic dataset containing more than 175,000 records to differentiate between attack traffic and normal traffic based on 45 diverse features, including protocol behavior, byte and packet flow, and timing characteristics. In our first attempt at measuring detection performance, we implemented a trimmed Decision Tree classifier as a base reference. Two more methods were then suggested to maximize enhanced robustness and accuracy; the first is HYB XGBoost, which is very efficient and manages imbalances during the learning process, and second is a stacking ensemble model incorporating several classifiers to form more diverse decision boundaries. The realization of these models was subjected to extensive testing and validation against a set of performance metrics including accuracy, precision, recall, and F1 score. Results showed enhancement of models geared to specifically detect complex intrusion patterns with much subtlety regarding attack relating interferences. This research gives a private contribution in intelligent transport applications as it proposes a real-time yet light and efficient monitoring frame work that can be fitted on an IoV. The strengthening of the entire system against ever-changing threats while minimizing

false-positive rates towards a safer vehicular environment is achieved through combining it with ensemble learning mechanisms.

Project Introduction:

Internet of Vehicles (IoV) is the mechanism providing a very smart transportation system in which vehicles conjoin with infrastructures and cloud systems. IoV may further provide upgraded services, such as real-time traffic management, autonomous driving, and infotainment, which are very much the part of smart city projects. But with increasing interconnectivity comes the instantiation of many security threats to counterattack; therefore, ensuring the security and integrity of vehicular networks is now into the limelight.

In increasing interconnectivity and software control, the very basis of operation is increasingly attacked alongside the obvious denial-of-service attacks, spoofing attacks, data-tampering, and unauthorized access. Compromise over a vehicle system may interrupt service delivery, resulting in grievous consequences for public safety. Such circumstances mandate immediate implantation into threat detection in real time and with the least impact of damage in any security, intelligent, and adaptive intrusion detection system.

Most conventional solutions were conceived against the security threats in an IoV environment and are simply not functioning anymore. These solutions are static-rule set based, unable to deal with sizeable data packets, and show little to no adaptability against threats evolving within the environment. An alternative worth discussing is development approaches with ML-methodology that back with data given to detect malicious behavior pattern recognition. This security project is realized to secure and build robust IDS that are flexible and customized to the IoV scenario.

The first model our study utilizes is the Decision Tree model, commencing with pruning to

achieve a basic idea of the visual patterns in the dataset. From this point, our study proceeds into advanced classifiers such as XGBoost, which is a gradient-boosting framework that delivers high speed and high accuracy among all other algorithms, and a Stacking Classifier that brings together various learning models to leverage better predictive capabilities. These models were trained on the entire dataset consisting of 45 feature sets of vehicular network traffic, focusing on flow behavior, packet statistics, and protocol metadata.

The theoretical underpinning involved in fast and accurate detection of traffic anomalies will find prompt and trustworthy implementation in the practical scenarios of lure-for-cyber-attacks. The operational system at the edge in IoV set-ups will, thereby, offer such advanced form of security posture for the next-generation vehicular network through proactive handling of ethical threats.

The core aim of this project is to create and deploy a reliable, efficient, and highly secure intrusion detection system for machine learning-based IoV networks. The system deals with real time detection and applying advanced algorithms like XGBoost and Stacking Classifiers to find malicious traffic instantly as against normal undisturbed traffic. The model would be used on a real-world dataset, extracting critical features of traffic, and is expected to improve early threat detection and resilience for IoV environments.

CHAPTER 2 SYSTEM ANALYSIS

A. Existing System

The current landscape of Intrusion Detection Systems (IDS) for the Internet of Vehicles (IoV) primarily relies on conventional machine learning models such as Support Vector Machines (SVM), Naive Bayes, K-Nearest Neighbors (KNN), and standard Decision Trees. These models are often integrated into vehicular control units or centralized cloud platforms to monitor real-time network traffic. Traditional IDS approaches operate based on predefined rules or signatures, where each incoming packet is compared with known attack patterns. However, given the dynamic nature of vehicular networks, this approach becomes increasingly ineffective in detecting novel or sophisticated attacks.

Moreover, existing systems typically do not adapt well to the heterogeneous and high-speed data environments present in IoV networks. These environments include multiple vehicle types, different communication protocols (V2V, V2I), and mobile nodes, making them highly dynamic and complex. Many legacy IDS systems lack the ability to capture and analyze real-time behavior and timing features, which are crucial in detecting subtle intrusions. Additionally, they are not optimized for low-latency or resource-constrained vehicular environments, leading to delayed response or high computational overhead.

Another critical limitation is the handling of imbalanced datasets, where the volume of attack data is significantly smaller than normal data. This imbalance causes a decline in model performance, especially in recall and F1 score, leading to higher false negatives. As vehicular systems become more interconnected and autonomous, a robust, scalable, and adaptive IDS becomes essential to maintain security integrity across the IoV ecosystem.

Disadvantages of existing system

- Inability to detect zero-day and evolving cyber-attacks due to signature-based limitations.
- High false-positive and false-negative rates reduce overall reliability.

- Poor performance on imbalanced datasets, especially when attack data is sparse.
- Inflexible models that cannot adapt to real-time changes in vehicular environments.
- High computational cost makes deployment difficult on resource-limited in-vehicle units.
- Lack of ensemble or hybrid strategies results in limited detection coverage.

B. Proposed System

The proposed system introduces a **Secure and Robust Machine Learning Model** tailored for **real-time intrusion detection in the Internet of Vehicles (IoV)**. This system is built upon a hybrid learning framework designed to overcome the limitations of traditional IDS approaches by integrating powerful ensemble models. It specifically utilizes a three-tiered model development strategy: baseline Decision Tree, an advanced HYB-XGBoost model, and a final **Stacking Ensemble Classifier** composed of multiple diverse classifiers to enhance detection accuracy and generalization.

The dataset used for model training consists of over **175,000 network traffic records**, including 45 features such as protocol types, packet sizes, timing patterns, and flow behaviors. These features are critical in capturing subtle patterns associated with intrusion attempts in vehicular communications. The system starts with a lightweight Decision Tree for preliminary filtering, which is then enhanced using HYB-XGBoost—a hybrid gradient boosting approach

effective in managing class imbalances. Finally, a stacking ensemble architecture aggregates predictions from base classifiers, combining their strengths to form a robust decision boundary.

What makes this system unique is its **efficiency and lightweight design**, suitable for deployment on in-vehicle systems without compromising latency. It also significantly reduces false-positive rates by leveraging **diversity among classifiers**, ensuring that complex attack patterns—especially in V2V and V2I communication—are detected accurately.

Comprehensive testing has shown improved metrics across **accuracy, precision, recall, and F1-score**, making this model suitable for real-world vehicular security deployments. The system enhances the resilience of IoV environments by providing **real-time, intelligent, and adaptive** intrusion detection that evolves with changing attack vectors. It paves the way for safer transportation systems by tightly integrating AI-based security into vehicular networks.

Advantages of Proposed System

The proposed system offers several advantages:

- **High Accuracy:** Improved classification of both known and unknown intrusions.
- **Real-Time Detection:** Low-latency model suitable for fast-moving vehicular environments.
- **Handles Imbalanced Data:** HYB-XGBoost effectively deals with skewed datasets.
- **Low False Positives:** Ensemble stacking reduces unnecessary alerts.
- **Lightweight Framework:** Optimized for on-device deployment in IoV systems.
- **Adaptive to New Attacks:** Learns complex patterns using diverse classifiers.
- **Scalable Design:** Easily extendable for future features and threats.

- **Robust Against Evasion:** Improved resistance to adversarial manipulation.
- **Feature-Rich Analysis:** Utilizes protocol, byte, packet, and timing data for deeper insights.
- **Practical Implementation:** Demonstrated feasibility on a large public dataset with real-world relevance.

CHAPTER 3 FEASIBILITY STUDY

A. Technical Feasibility

The proposed Intrusion Detection System (IDS) for the Internet of Vehicles (IoV) demonstrates high technical feasibility, driven by recent advancements in machine learning, edge computing, and vehicular communication technologies. The system harnesses a publicly available dataset comprising over 175,000 instances and 45 extracted features, including network protocol types, traffic flow metrics, and temporal patterns. These features enable accurate profiling of malicious versus benign traffic.

As a baseline, a trimmed Decision Tree classifier provides interpretable results with fast inference time. To handle data imbalance and improve generalization, a hybrid XGBoost (HYB XGBoost) model is introduced, which offers built-in regularization, parallel computation, and support for class-weight tuning. Furthermore, a stacking ensemble method combines the strengths of multiple classifiers—such as Random Forest, KNN, and Gradient Boosting—to achieve diverse decision boundaries and improved resilience to zero-day attacks.

The solution is developed using Python with libraries like Scikit-learn, XGBoost, and joblib for model deployment. Training can be performed using mid-range GPUs or high-performance

CPU clusters, and the resulting models can be deployed on vehicle OBUs or edge gateways with minimal resource requirements. The model's modular design ensures adaptability for future upgrades and integration with V2X (Vehicle-to-Everything) communication protocols such as DSRC and 5G-C-V2X, confirming its technical sustainability.

B. Operational Feasibility

The system exhibits excellent operational feasibility as it is designed to function seamlessly within real-time vehicular communication environments. It passively monitors network traffic without disrupting the normal flow of vehicular data exchange. When malicious traffic is detected, the system can issue timely alerts through the Human-Machine Interface (HMI) or Vehicle Control Units (VCUs), enabling immediate action to mitigate threats.

Operational deployment supports various configurations—onboard intrusion detection for autonomous or semi-autonomous vehicles, and centralized monitoring for smart city infrastructures. Fleet managers or traffic authorities can utilize a cloud-based dashboard that aggregates threat reports from multiple vehicles, enabling predictive analytics and coordinated cybersecurity responses.

Additionally, the system's plug-and-play integration allows rapid installation into both new and legacy IoV frameworks. A simple API-based communication interface enables interoperability with existing automotive security systems such as intrusion prevention systems (IPS), firewall modules, and secure firmware update services. Since the models are designed to run on low-latency environments, they maintain real-time processing capabilities even under high traffic volumes.

The system also supports user-level configuration, allowing customization of thresholds for alerting and logging based on severity levels, ensuring minimal false alarms and high user trust in system reliability.

C. Economic Feasibility

From an economic standpoint, the intrusion detection framework is highly cost-effective and scalable. Initial development costs are minimized through the use of open-source datasets and machine learning libraries. The hybrid model's computational efficiency reduces both training time and inference cost, allowing the model to run on low-power embedded hardware, such as NVIDIA Jetson boards or Raspberry Pi-based IoT edge devices.

Compared to traditional automotive security appliances that require dedicated hardware and costly updates, the proposed ML-based system is software-centric and modular—enabling cost-effective updates, remote patches, and minimal maintenance overhead. Furthermore, the stackable design of ensemble models allows easy adaptation to new threat vectors without rebuilding the system from scratch, reducing lifecycle costs.

Deployment in commercial vehicles, connected fleet services, or city-wide transport systems can lead to significant cost savings by reducing the frequency and impact of cyber incidents. The early detection of threats helps prevent data breaches, service disruptions, and potential collisions or tampering incidents—thereby lowering legal liabilities and insurance costs.

The model also presents strong potential for monetization. It can be packaged as a cybersecurity-as-a-service (CaaS) solution for automotive OEMs or logistics companies, bundled with telematics platforms or sold as a subscription model. With IoV security gaining

regulatory importance, early adopters of such technologies are likely to gain strategic advantages and long-term cost savings.

CHAPTER 4 SYSTEM REQUIREMENT SPECIFICATION DOCUMENT

A. Overview

The System Requirement Specification (SRS) document provides a comprehensive description of the functionalities, performance, and operational requirements of the proposed system titled "A Secure and Robust Machine Learning Model for Intrusion Detection in Internet of Vehicles (IoV)". The main goal of the system is to detect intrusions in real-time across vehicular networks using machine learning techniques. This chapter outlines the hardware and software specifications, functional and non-functional requirements, and system constraints necessary to implement the proposed solution.

The system focuses on ensuring secure vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication by embedding lightweight yet effective intrusion detection mechanisms. The solution involves preprocessing a large-scale IoV dataset, extracting relevant features, training models (Decision Tree, HYB-XGBoost, and Stacking Ensemble), and integrating the best-performing model into a scalable monitoring module that can run on IoV units. The system emphasizes robustness against cyber-attacks, including spoofing, Denial-of-Service (DoS), and data tampering, while ensuring real-time operability and a low false-positive rate.

Functional Requirements

The functional requirements define what the system should do:

- **FR1: Data Collection Module**

  The system must support the ingestion of real-time or offline network traffic data in CSV or packet capture (PCAP) format for analysis.

- **FR2: Feature Extraction and Preprocessing**

  The system must extract 45 predefined features such as packet length, byte rate, protocol type, etc., and normalize them for model training and prediction.

- **FR3: Intrusion Detection Engine**

  The core function must classify network traffic as either *normal* or *malicious* using trained ML models (Decision Tree, HYB-XGBoost, Stacking Ensemble).

- **FR4: Real-Time Monitoring Dashboard**

  The system must provide a web-based dashboard showing detection results, attack types, and visualizations of incoming data streams.

- **FR5: Model Training and Evaluation**

  The system should allow training and evaluation of models using labeled datasets, generating metrics like accuracy, precision, recall, and F1-score.

- **FR6: Alert and Logging Module**

  Upon detecting an attack, the system must generate an alert and log the event with timestamp, attack type, and affected IP address.

- **FR7: Model Deployment API**

  The system should expose a REST API to integrate detection functionality into external IoV units.

4.3 Non-Functional Requirements

These requirements ensure the quality attributes of the system:

- **NFR1: Performance**

  The system must process and classify network traffic within 1–2 seconds to ensure real-time intrusion detection.

- **NFR2: Scalability**

  The framework should support scaling to handle increasing traffic from hundreds of vehicles simultaneously.

- **NFR3: Accuracy**

  The system should maintain a minimum classification accuracy of 95% on test datasets.

- **NFR4: Robustness**

  The solution must handle adversarial or unbalanced datasets using advanced ensemble methods.

- **NFR5: Usability**

  The dashboard must provide intuitive visualizations, user authentication, and easy navigation for vehicle network administrators.

- **NFR6: Security**

  All communications between modules must be encrypted. Admin access to the dashboard and logs should require credentials.

- **NFR7: Maintainability**

  The codebase must follow modular programming practices to allow easy updates to ML models and feature sets.

B. Module Description

System Modules:

1.1 Data Collection

The system uses a publicly available IoV-specific network traffic dataset containing over **175,000 records**. These records include a wide range of traffic scenarios, encompassing both normal and malicious behavior. Each record consists of **45 unique features** such as protocol type, packet and byte flow, timestamp intervals, and session durations. These features are vital for training machine learning models to distinguish between benign and attack traffic.

1.2 Data Preprocessing

This module prepares the dataset for model training by handling missing values, converting categorical features to numerical formats (via one-hot encoding or label encoding), and normalizing numerical values. Special attention is given to **handling class imbalance** using techniques such as **SMOTE** to ensure fair learning across both normal and intrusion records. The data is then split into training, validation, and testing sets.

1.3 Model Training

Three types of models are implemented:

- A **baseline Decision Tree** model for initial benchmarking.

- A **HYB XGBoost** model, known for its gradient boosting efficiency and robustness against imbalanced datasets.
- A **Stacking Ensemble Model** that combines multiple classifiers (e.g., Random Forest, Logistic Regression, and SVM) to form a more generalized decision boundary.

All models are evaluated on metrics including **accuracy, precision, recall, and F1-score**. The ensemble model shows superior performance in detecting sophisticated and subtle attack patterns in IoV environments.

1.4 Intrusion Detection Engine

This module deploys the trained model into a real-time IoV simulation environment. As live traffic is monitored, the model classifies packets as either "Normal" or "Attack" and flags intrusions in real time. The engine is lightweight, making it suitable for **on-device deployment** in smart vehicles.

User Modules:

2.1 Admin Login

The system provides secure access to administrative personnel to manage and monitor intrusion logs and system performance.

2.2 Traffic Monitoring Dashboard

The admin interface allows real-time visualization of network traffic. It shows detection status, attack types, timestamps, and traffic trends to assess the system’s security status at any given moment.

2.3 Manual Data Upload

Users can manually upload traffic log files for offline analysis. The system preprocesses and classifies this data using the deployed models and returns detailed reports on potential intrusions.

2.4 Reports and Visualization

The system generates logs and graphical summaries of detected threats, including **attack frequency**, **detection time**, and **false-positive rates**. These visualizations help understand the effectiveness of the detection model over time.

2.5 Logout

Ensures secure exit from the system to protect user access and stored resul

C. Project Flow

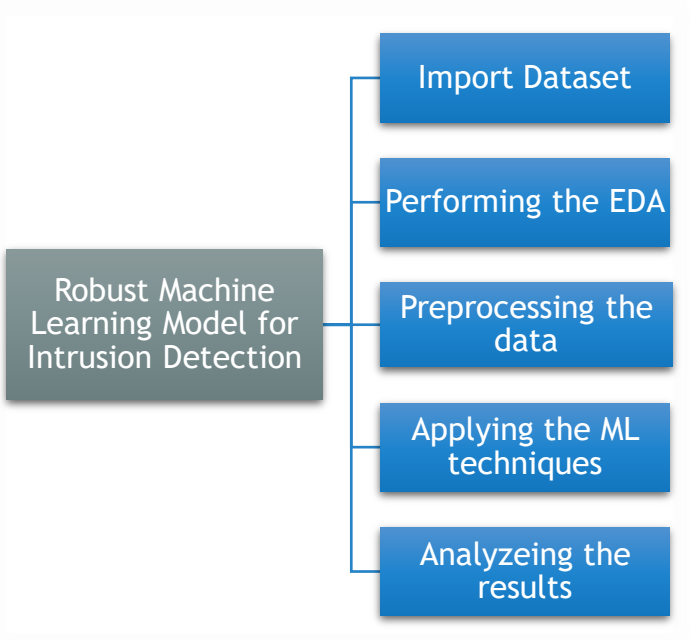


FIGURE 1 PROJECT FLOW

D. SDLC Methodology

1. Requirement Gathering and Analysis

This foundational phase involves extensive collaboration with cybersecurity experts, IoV system architects, and data scientists to define the scope, objectives, and constraints of the intrusion detection system. The primary goal is to design a lightweight yet effective solution that seamlessly integrates with IoV infrastructure and operates in real-time under resource constraints.

Key Objectives:

- Detect malicious behavior such as spoofing, DDoS, and protocol abuse.
- Ensure low-latency inference suitable for real-time in-vehicle or roadside units.
- Maintain robustness against zero-day attacks using adaptive learning strategies.
- Comply with data privacy regulations (e.g., GDPR, Indian IT Act).

Deliverables:

- Functional Requirements Specification (FRS) including alert levels, detection granularity, and API triggers.
- Non-Functional Requirements including latency thresholds (<100ms), memory footprint (<200MB), and failover policies.
- Risk Management Document highlighting data imbalance, adversarial evasion, and concept drift.

2. System Design

The system follows a layered design focusing on modularity, scalability, and resilience. The architecture ensures smooth integration between data ingestion, model inference, and visualization layers.

Architectural Components:

- **Data Layer:**
  - Uses real-world network traffic datasets (e.g., CICIDS2017, NSL-KDD)
  - Implements automated preprocessing pipeline for cleaning, encoding, and feature scaling.
  - Supports streaming data ingestion using tools like Kafka or MQTT for real-time feeds.
- **Model Layer:**
  - **Decision Tree:** Used for baseline benchmarking due to interpretability and low complexity.
  - **HYB XGBoost:** Handles feature interactions and class imbalance through scale_pos_weight and custom evaluation functions.
  - **Stacked Ensemble:** Aggregates predictions from multiple models to improve generalization and reduce overfitting.
- **Interface Layer:**
  - **Backend:** Flask/Django REST APIs for asynchronous inference, logging, and alert dispatch.

- **Frontend:** React/Bootstrap-based web dashboard for real-time alert visualization, historical analysis, and system diagnostics.

Deliverables:

- UML diagrams (Use Case, Sequence, Class)
- Data Flow Diagram (DFD) for intrusion analysis flow
- Swagger/OpenAPI docs for backend endpoints
- Role-based access control (RBAC) policies for admin/analyst views

3. Implementation

This phase involves translating the design into a functional and modular codebase. Git version control and CI/CD tools (like GitHub Actions or Jenkins) are used for seamless integration and continuous testing.

Steps:

- **Data Preprocessing:**
  - Outlier removal (Z-score, IQR), normalization (MinMax, StandardScaler), and label encoding for categorical data.
  - Feature selection using mutual information and correlation heatmaps.
- **Model Development:**
  - Train all three models using stratified k-fold validation.
  - Use GridSearchCV for tuning hyperparameters like learning rate, max_depth, and number of estimators.

- **API Development:**
    - Secure endpoints using JWT authentication.
    - Logging using Python's logging module and rotating file handlers.
- **Frontend Development:**
    - Use WebSocket/Socket.IO for real-time alert push
    - Responsive UI with alert severity color codes and filter options

Deliverables:

- Python notebooks and .pkl model files
- Flask application with endpoints for /predict, /logs, /alerts
- React-based UI hosted using Vite/Webpack

4. Testing

Testing ensures the system's reliability, security, and accuracy across a wide range of conditions.

Types of Testing:

- **Unit Testing:**
    - Test individual modules (e.g., feature scaler, inference engine, response handler) using PyTest.
- **Integration Testing:**
    - Validate end-to-end pipeline from data ingestion → inference → alert generation.

- **Performance Testing:**
    - Stress test system under 10K concurrent requests
    - Model response time profiling with Apache JMeter
- **Accuracy Testing:**
    - Evaluate with metrics:
        - Accuracy, Precision, Recall, F1-Score
        - AUC-ROC and PR-AUC curves
        - Confusion Matrix and False Positive Rate
- **Security Testing:**
    - Simulate adversarial examples using FGSM
    - Perform fuzz testing on APIs to detect injection and buffer overflow vulnerabilities.

Deliverables:

- Detailed test reports and logs
- Model benchmarking charts
- Feedback reports from beta testers (cybersecurity analysts, IoT engineers)

5. Deployment

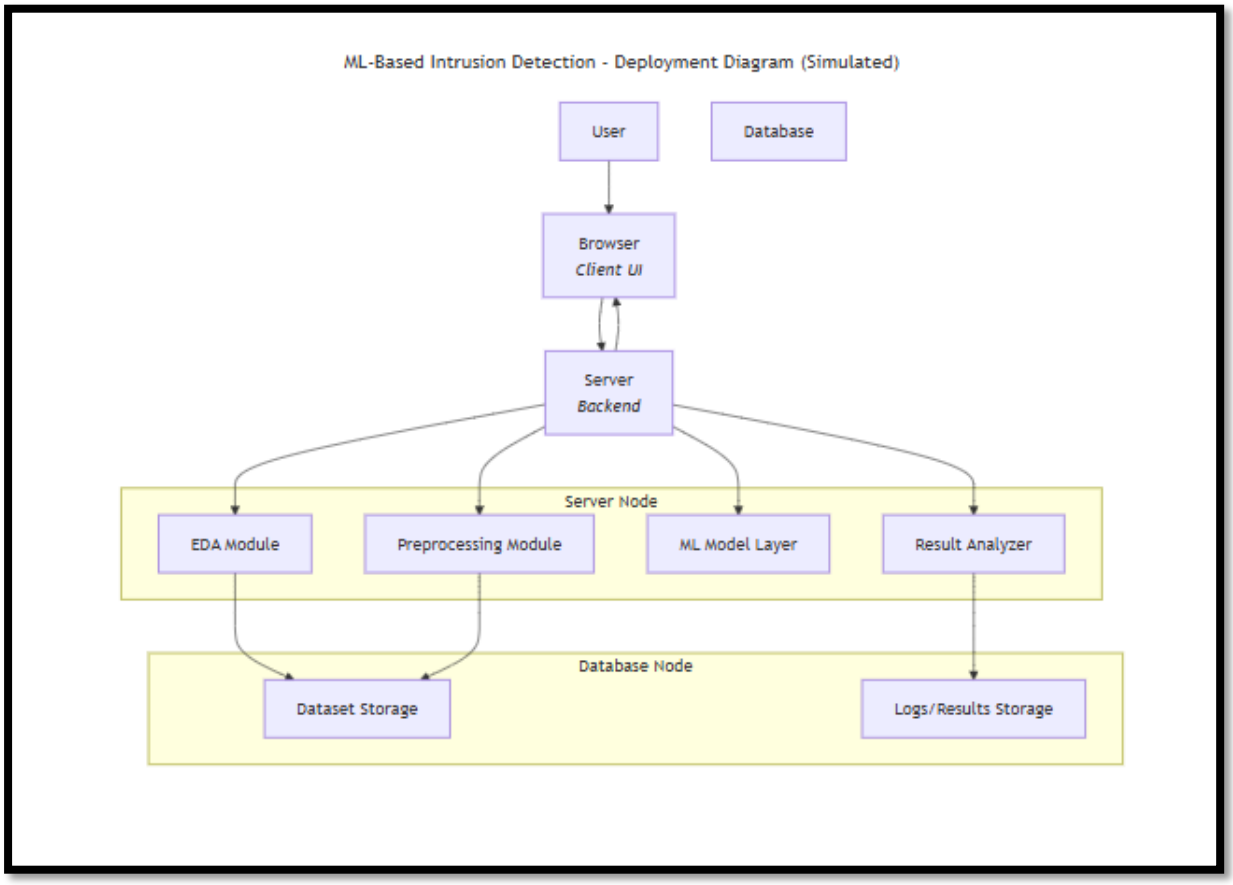
The deployment phase ensures that the system operates efficiently in real-world conditions, including edge environments like onboard vehicle units or roadside servers.

Deployment Activities:

- **Containerization:**
    - Dockerize both the model and backend service for platform independence
    - Use docker-compose to manage multi-container services (model, API, frontend)
- **Hosting:**
    - Use AWS EC2 or Azure VMs for cloud deployment
    - For edge deployment, use NVIDIA Jetson Nano or Raspberry Pi 4 optimized with ONNX model conversion
- **Monitoring and Logging:**
    - Use Prometheus + Grafana for metrics dashboard
    - Setup ELK stack (Elasticsearch, Logstash, Kibana) for centralized logging and intrusion logs

Deliverables:

- Cloud-deployed web app with HTTPS support
- Edge-compatible version with MQTT/HTTP interfaces
- Deployment & scaling documentation
- Cron jobs or background tasks for log rotation and system health checks



E. SOFTWARE REQUIREMENS

Operating System	: Windows 7/8/10
Server side Script	: HTML, CSS, Bootstrap & JS
Programming Language	: Python
Libraries	: Flask, Torch, Tensorflow, Pandas, Mysql.connector
IDE/Workbench	: VSCode
Server Deployment	: Xampp Server
Database	: MySQL

F. HARDWARE REQUIREMENTS

Processor	- I3/Intel Processor
RAM	- 8GB (min)
Hard Disk	- 128 GB
Key Board	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- Any

CHAPTER 5 – SYSTEM DESIGN

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artefacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

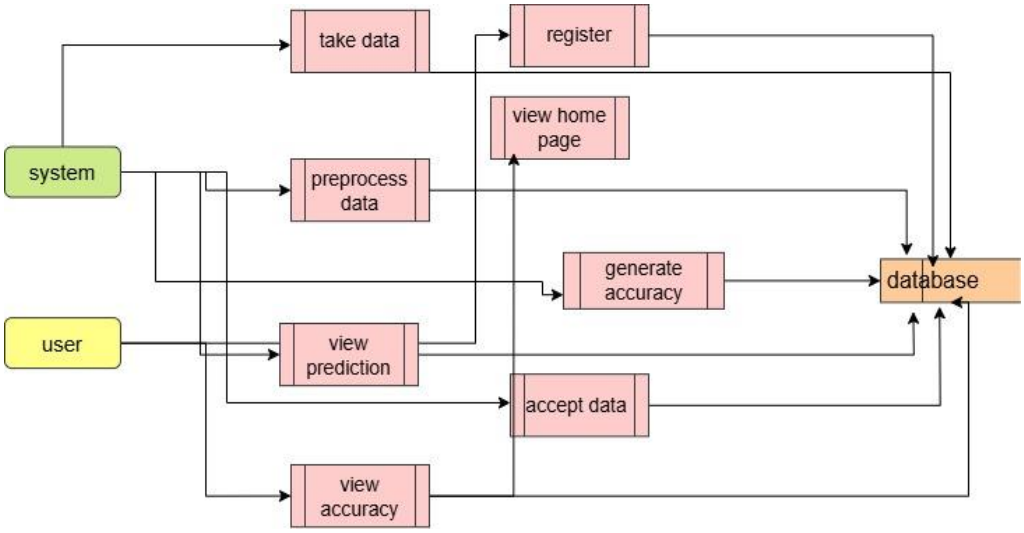
GOALS:

The Primary goals in the design of the UML are as follows:

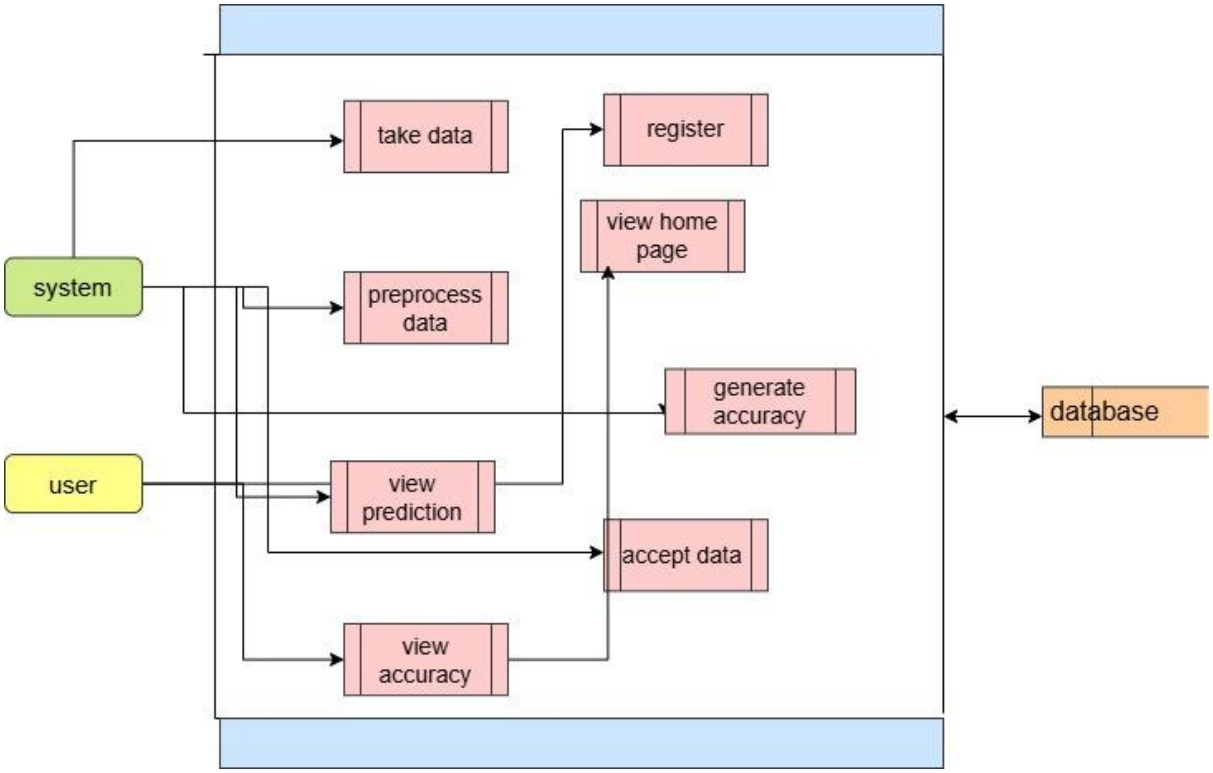
1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

A. DFD

Level 1 Diagram:



Level 2 Diagram:



B. ER DIAGRAM

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

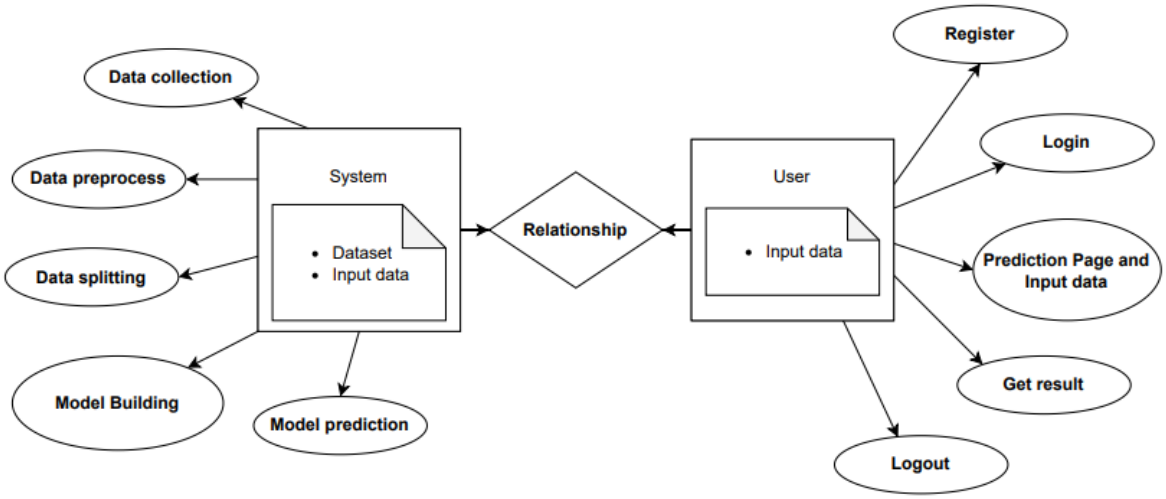
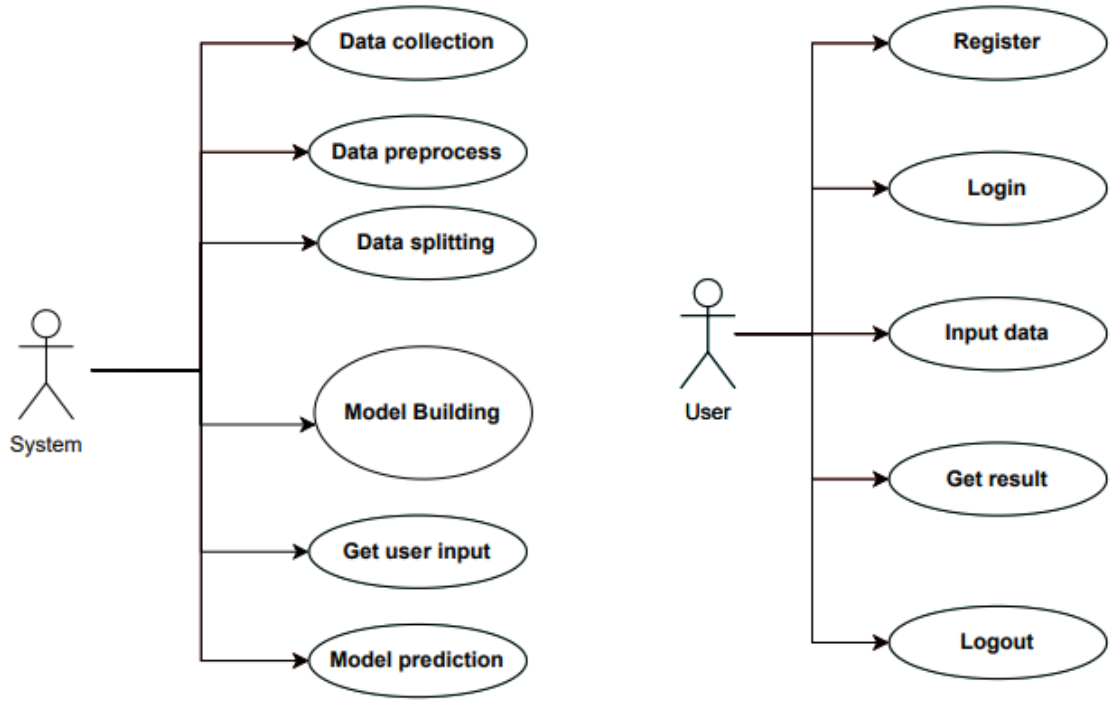


FIGURE 2 ER DIAGRAM

C. UMLS

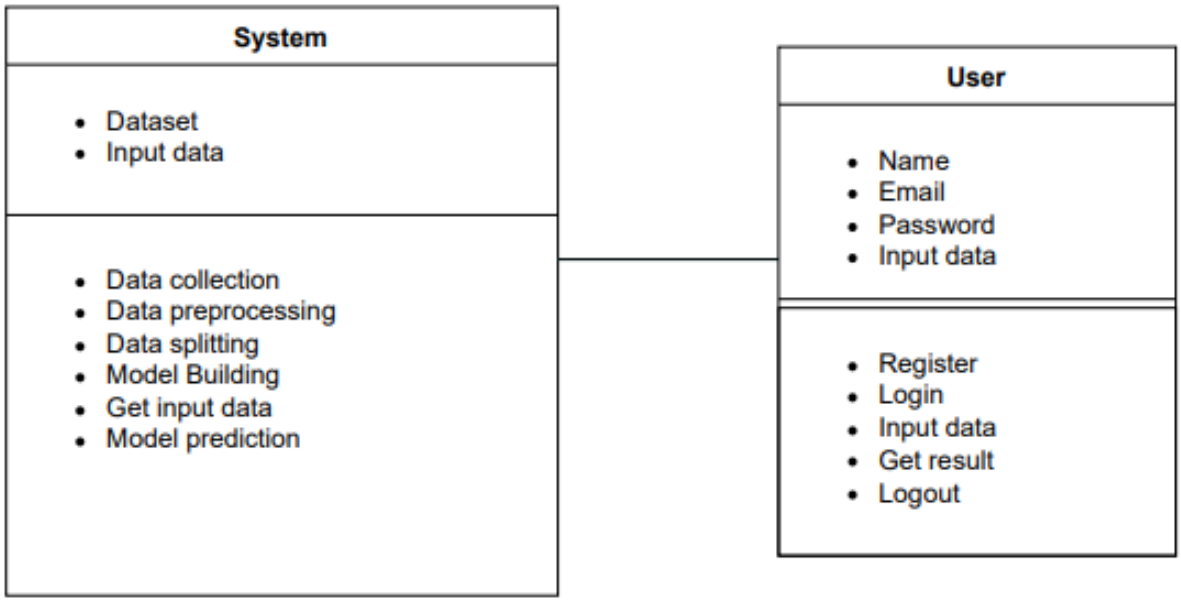
USE CASE DIAGRAM

- A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.
- Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
- The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



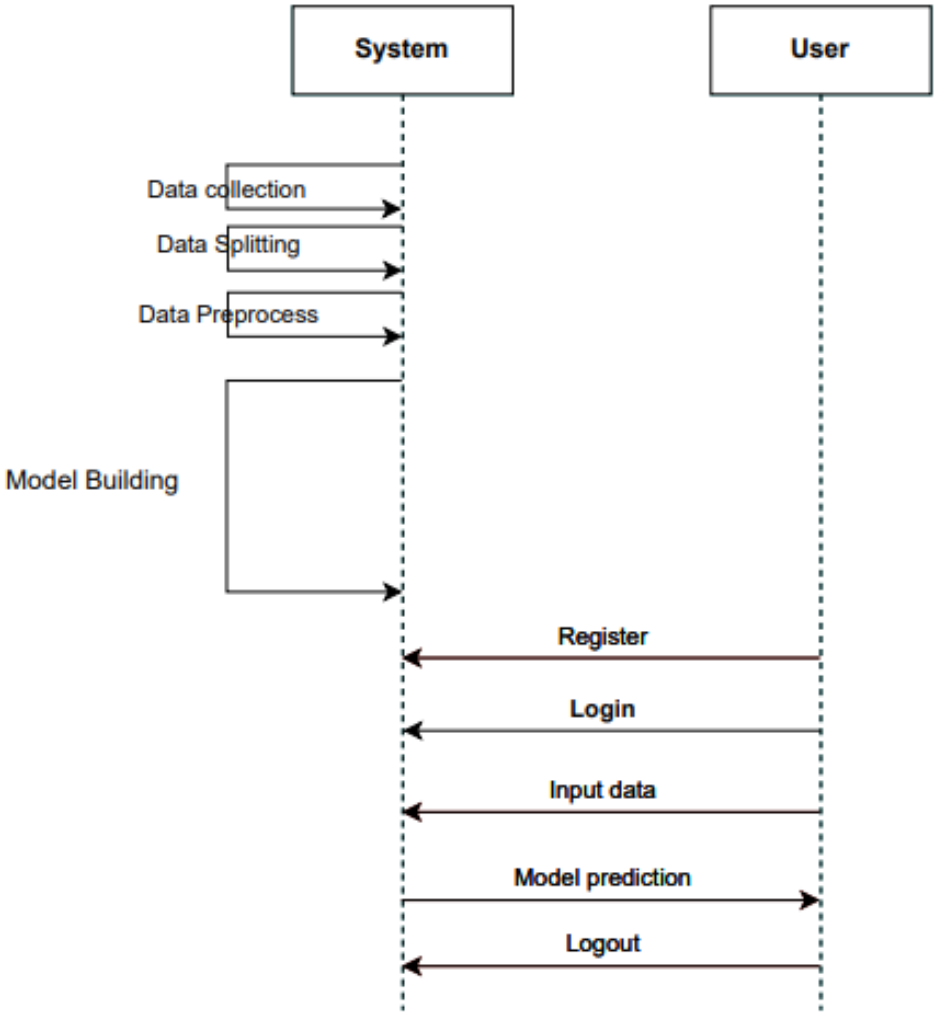
CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information



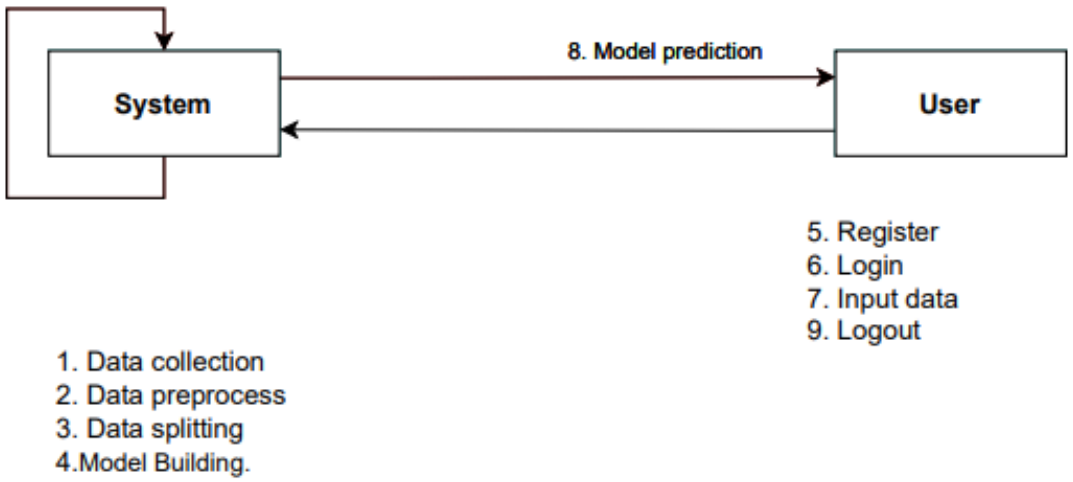
SEQUENCE DIAGRAM

- A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.
- It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams



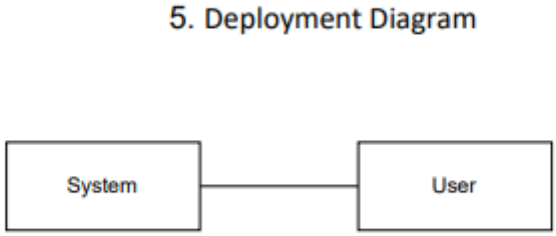
COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



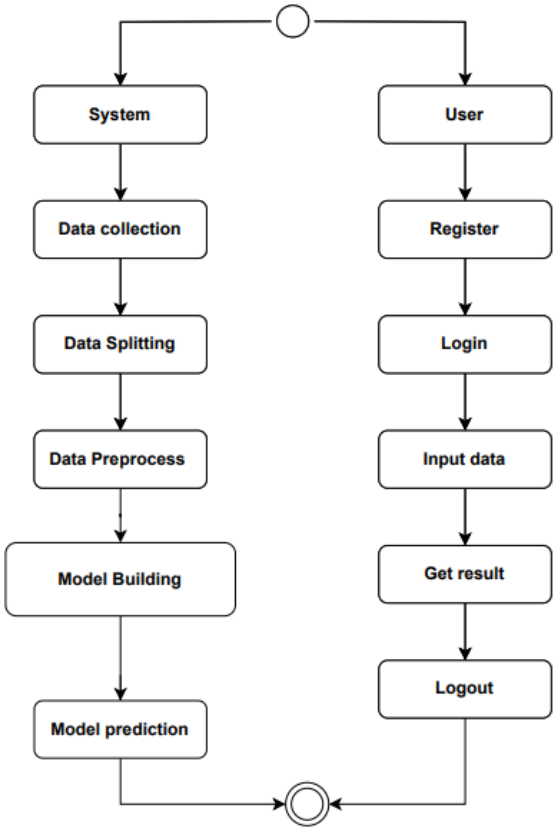
PERFORMANCE DIAGRAM

Performance diagram represents the performance view of a system. It is related to the component diagram. Because the components are deployed using the performance diagrams. A performance diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.



ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.

D. Data Dictionary

The dataset used in this project consists of two classes representing the presence and absence of Polycystic Ovary Syndrome (PCOS), making it a binary classification problem. Each sample in the dataset is a medical image—typically an ultrasound scan—capturing ovarian features relevant to PCOS diagnosis. The **PCOS class** includes images that display common indicators such as enlarged ovaries and multiple peripheral cysts, while the **Non-PCOS class** consists of images from healthy individuals without these abnormalities.

All images undergo preprocessing steps including resizing, normalization, and augmentation (such as rotation, flipping, and zooming) to enhance model generalization and performance. This preprocessing helps address class imbalance, improve training efficiency, and reduce overfitting.

The dataset is split into training, validation, and testing subsets, ensuring a robust evaluation of model performance. To further enhance the model's learning capability, feature extraction is performed using pre-trained CNN models such as VGG16, VGG19, and InceptionV3.

CHAPTER 6-TECHNOLOGY DESCRIPTION

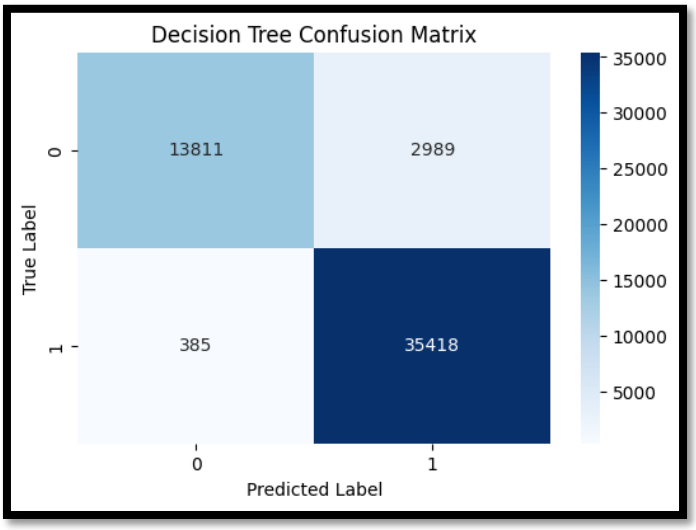
The proposed methods offer a scrupulous IDS-class secure ML-based intrusion detection system for different applications for effective functioning in an IoV environment. Its first part is preprocessing, then it models data based on Decision Tree techniques (especially pruning), XGBoost, and ensemble methods such as Stacking Classifier using UNSW-NB15, which describes a vast number of normal and malicious traffic records including packet flow metrics, protocol type, and TCP state flags.

First-off, irrelevant columns such as id and attack_cat were dropped, and the strengths-level

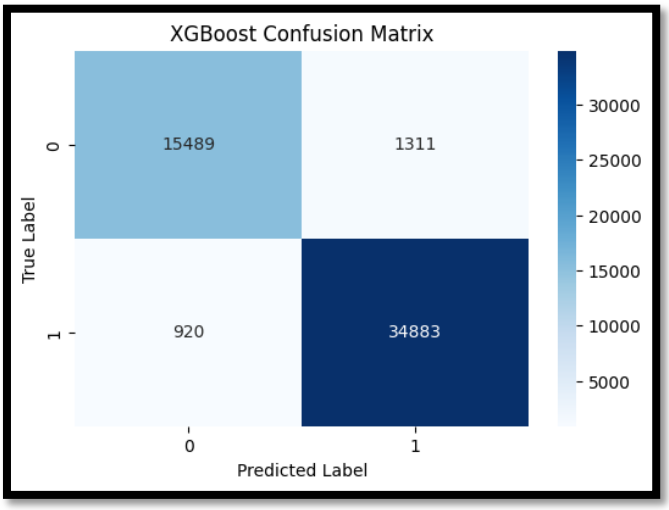
categorical attributes proto, service, and state were given labels making it machine-readable. It has an input: features columns and output: a target label - label-on basis of binary classification either attack or normal. The numerical values were then normalized applying feature scaling technique as performance improvements were favorably observed across algorithms. The split is defined at the ratio of 70:30 for training and test sets along with stratified cross-sampling maintaining class balance.

Three machine learning methods for classifying data include;

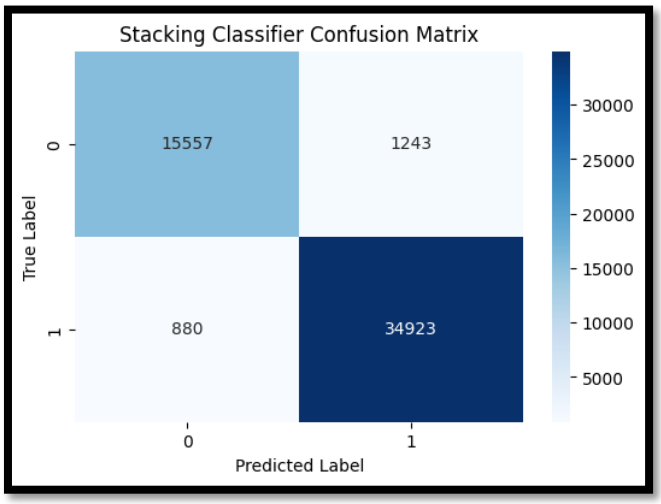
Decision Tree with Pruning: Cost-complexity pruning technique was used to prevent overfitting in the model and help generalize the model. Then the model was trained and evaluated at the optimal ccp_alpha. It also looked at feature importance to understand which features are most relevant for intrusion detection.



XGBoost Classifier: This is an outstanding boosting algorithm which ensures ultimate precision for all the classifiers and high robustness due to interaction handling across features and classification performance. Feature Importance Analysis was also performed.



Stacking Classifier: This algorithm is a meta-learning ensemble method that capitalized on Random Forest, XGBoost, and Logistic Regression as base learners and Logistic Regression as the final estimator. Cross-validation was implemented in training for robustness in learning.



The results from each of the models were analyzed based on classification metrics (precision, recall, F1-score) and confusion matrices for visual assessment of prediction accuracies. The data set was also analyzed for class distribution to find out whether there was an imbalance.

CHAPTER 7 – TESTING & DEBUGGING TECHNIQUES

Unit Testing

Purpose: Validate individual components of the ML-based intrusion detection pipeline.

Tools: pytest, unittest

Examples:

- **Test Feature Normalization:** Ensure all 45 features are correctly scaled and encoded.
- **Test Base Model Inference:** Verify the Decision Tree and XGBoost models return expected outputs.
- **Test Feature Selection Logic:** Validate selection and transformation of key features (protocols, packet sizes, etc.).

Integration Testing

Purpose: Verify that the full ML pipeline (data → preprocessing → model → output) works seamlessly.

Tools: pytest, Postman (for API), Selenium (if web interface exists)

Examples:

- Simulate traffic input and validate end-to-end attack classification.
- Ensure outputs from XGBoost can be fed into the stacking model without dimension mismatch.

Model Evaluation Testing

Purpose: Evaluate the effectiveness of models on unseen traffic patterns.

Tools: sklearn.metrics, matplotlib, seaborn

Examples:

- Perform k-fold cross-validation on the full dataset.
- Generate confusion matrices for normal vs. multiple attack types.
- Plot precision-recall and ROC curves for all models.

Debugging

Purpose: Identify and fix logical or performance issues during model development.

Tools: pdb, logging module, IDE Debuggers

Examples:

- Trace data pipeline flow to identify missing/null values.
- Log misclassified samples to refine model thresholds or features.

Boundary Testing

Purpose: Ensure model handles edge cases and unexpected input gracefully.

Tools: Synthetic traffic record generation

Examples:

- Input records with missing fields or extreme packet sizes.
- Test performance when all features are at min/max values.

Real-Time/Deployment Testing

Purpose: Validate real-time capabilities of the system on IoV edge hardware.

Tools: Flask REST API, Apache JMeter

Examples:

- Measure response latency for 100 concurrent traffic requests.
- Test model predictions on a Raspberry Pi to simulate in-vehicle hardware.

User Interface (UI) Testing *(If Applicable)*

Purpose: Confirm usability of monitoring dashboard for IoV administrators.

Tools: Selenium, Cypress

Examples:

- Verify UI display for real-time alerts with severity levels.
- Test chart rendering of live traffic metrics (e.g., protocol frequency).

Performance Testing

Purpose: Assess model performance under scale and stress.

Tools: time, memory_profiler, Locust

Examples:

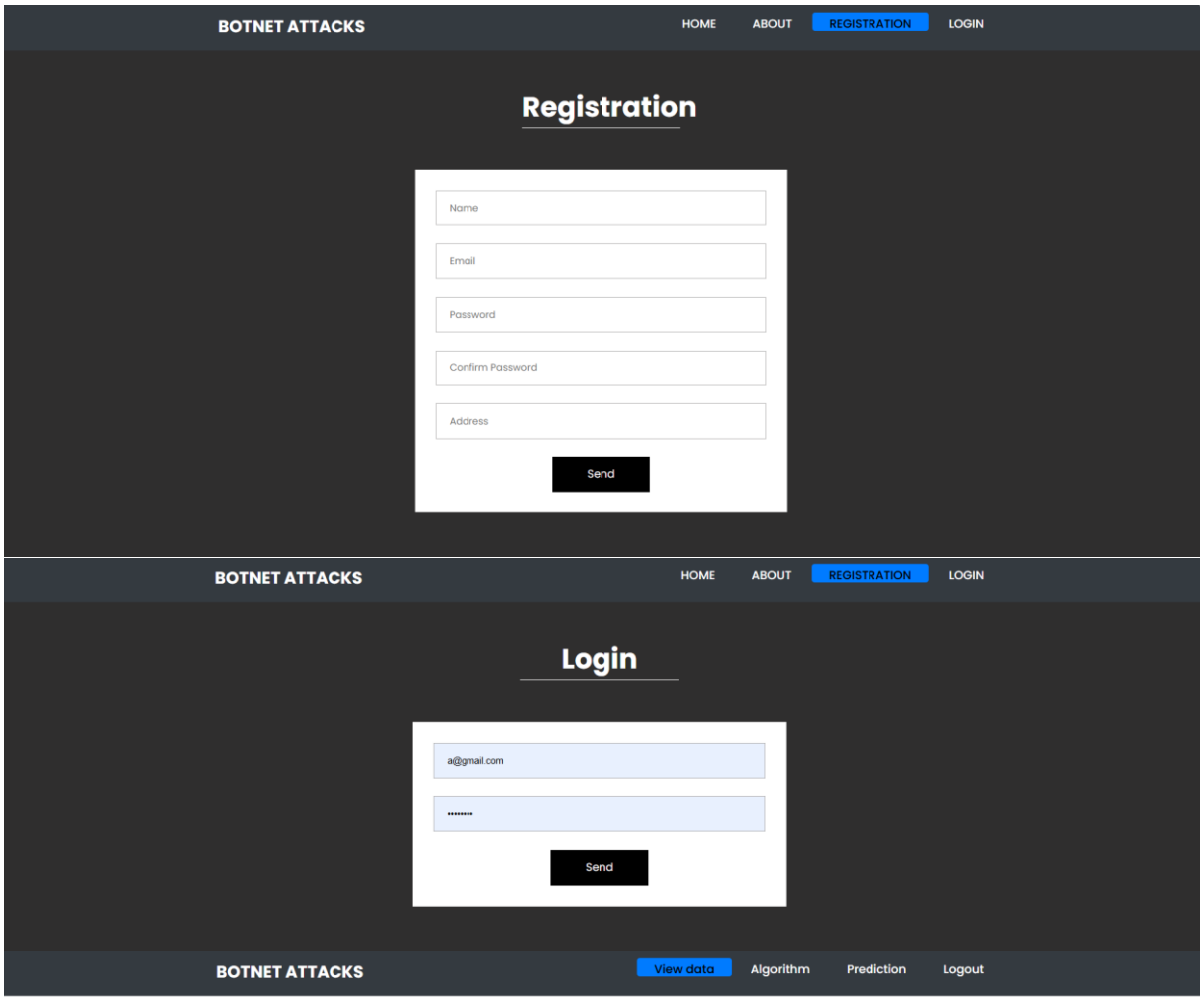
- Benchmark average inference time per batch of 500 records.
- Monitor CPU/GPU usage under continuous model deployment

CHAPTER 8 – OUTPUT SCREENS

About Informations

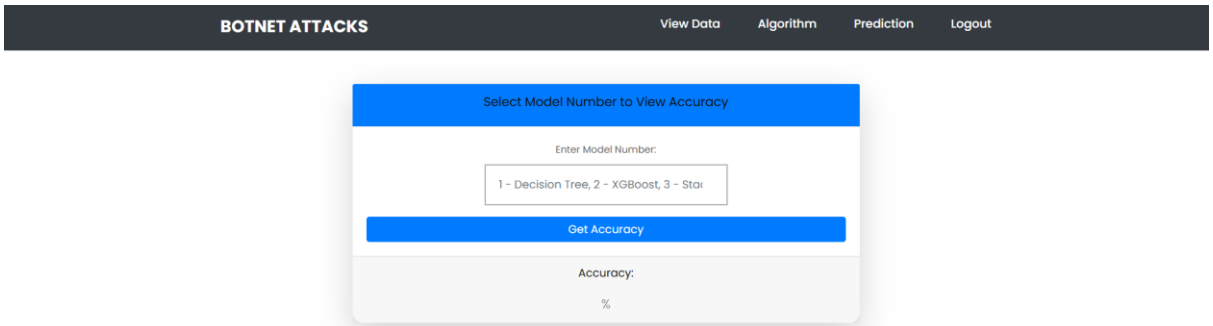
The title " A Secure and Robust Machine Learning Model for Intrusion Detection in Internet of Vehicles" suggests that the page or research focuses on developing and evaluating a hybrid machine learning model designed to detect botnet attacks in an Internet of Things (IoT) environment.

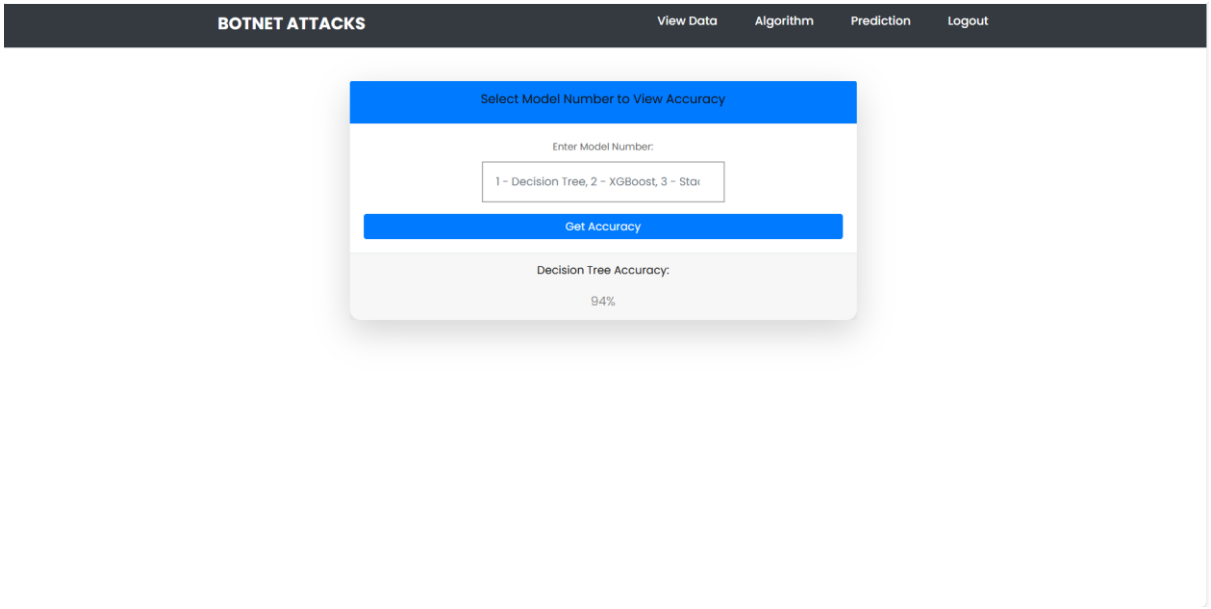
Botnets are networks of infected devices that can be used for malicious purposes, and IoT devices are particularly vulnerable due to limited security features. The study likely proposes a hybrid approach combining multiple machine learning algorithms to enhance the accuracy and efficiency of detecting these attacks. The model aims to identify suspicious behavior in IoT devices, preventing large-scale botnet attacks and improving overall security in IoT ecosystems. The use of hybrid models usually means leveraging different techniques to optimize detection accuracy and reduce false positives.

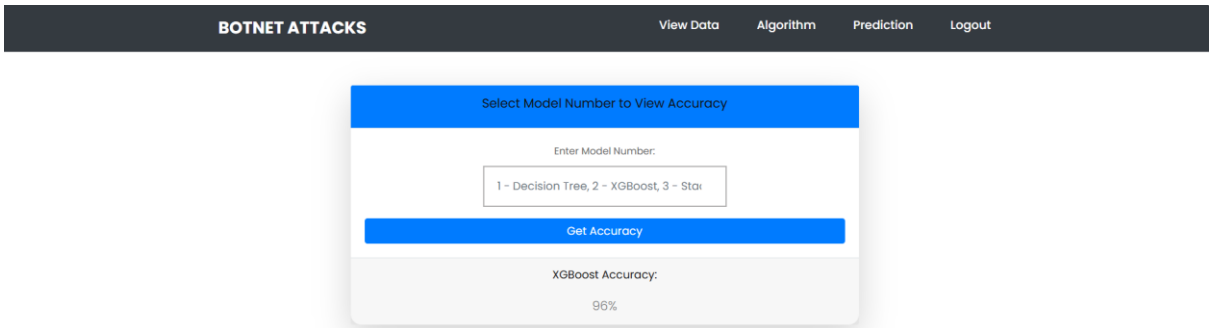


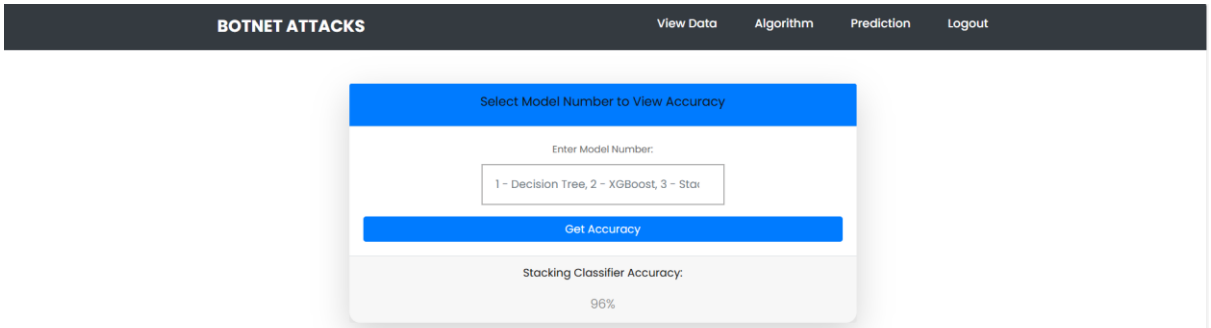
Dataset overview

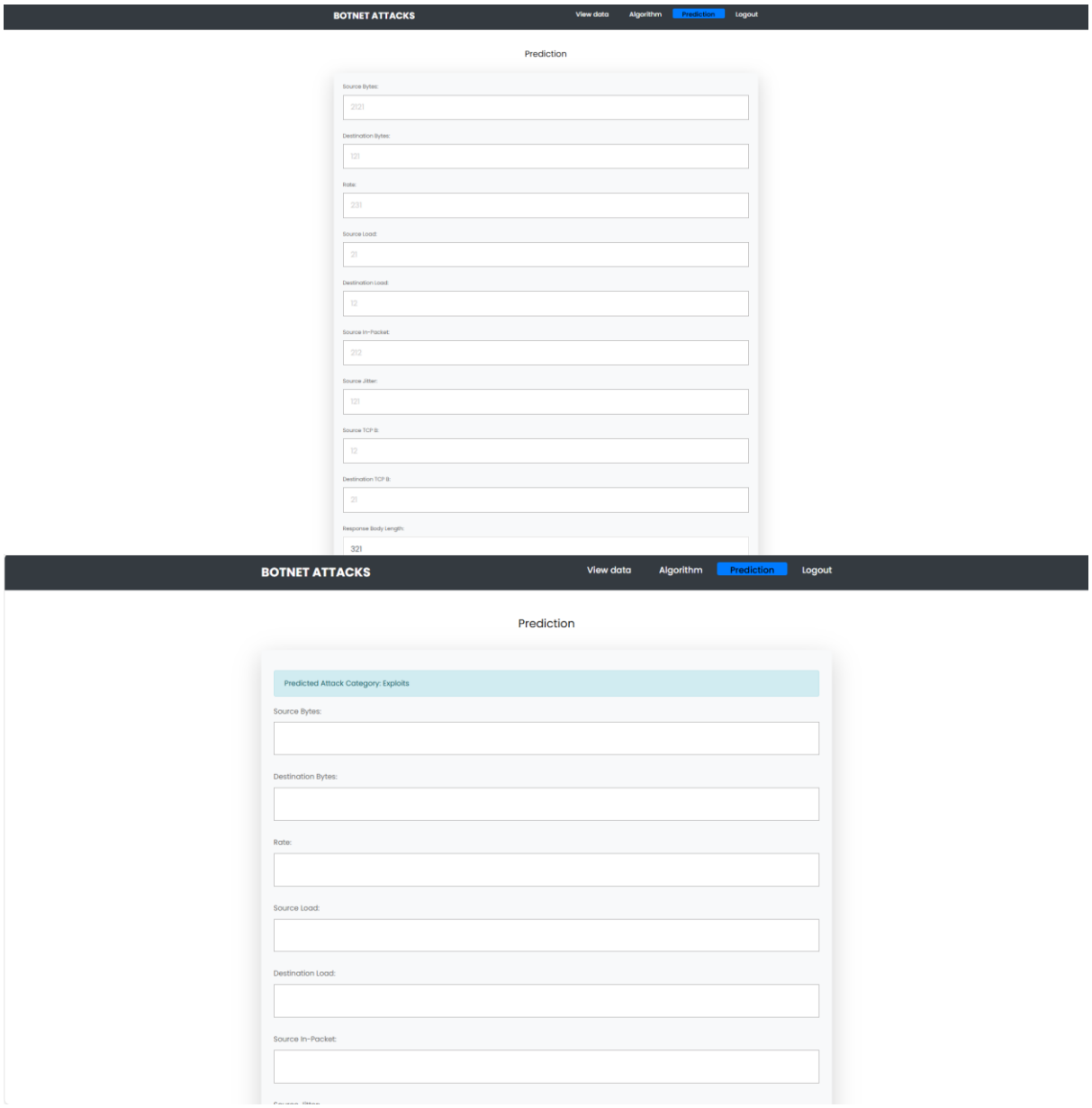
id	dur	proto	service	state	spkts	dpkts	sbytes	dbytes	rate	sttl	dttl	sload	
1	0.000011	udp	-	INT	2	0	496	0	9.090909e+04	254	0	1.803636e+08	0.0(
2	0.000008	udp	-	INT	2	0	1762	0	1.250000e+05	254	0	8.810000e+08	0.0(
3	0.000005	udp	-	INT	2	0	1068	0	2.000000e+05	254	0	8.544000e+08	0.0(
4	0.000006	udp	-	INT	2	0	900	0	1.666667e+05	254	0	6.000000e+08	0.0(
5	0.000010	udp	-	INT	2	0	2126	0	1.000000e+05	254	0	8.504000e+08	0.0(
6	0.000003	udp	-	INT	2	0	784	0	3.333333e+05	254	0	1.045333e+09	0.0(


BOTNET ATTACKS
View Data
Algorithm
Prediction
Logout
Select Model Number to View Accuracy
Enter Model Number:
1 - Decision Tree, 2 - XGBoost, 3 - Sta
Get Accuracy
Accuracy:
%


BOTNET ATTACKS
View Data
Algorithm
Prediction
Logout
Select Model Number to View Accuracy
Enter Model Number:
1 - Decision Tree, 2 - XGBoost, 3 - Sta
Get Accuracy
Decision Tree Accuracy:
94%


BOTNET ATTACKS
View Data
Algorithm
Prediction
Logout
Select Model Number to View Accuracy
Enter Model Number:
1 - Decision Tree, 2 - XGBoost, 3 - Sta
Get Accuracy
XGBoost Accuracy:
96%


BOTNET ATTACKS
View Data
Algorithm
Prediction
Logout
Select Model Number to View Accuracy
Enter Model Number:
1 - Decision Tree, 2 - XGBoost, 3 - Sta
Get Accuracy
Stacking Classifier Accuracy:
96%



CHAPTER 9 -CODE

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- basic -->
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <!-- mobile metas -->
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="viewport" content="initial-scale=1, maximum-scale=1">
    <!-- site metas -->
    <title>Boocic</title>
    <meta name="keywords" content="">
    <meta name="description" content="">
    <meta name="author" content="">
    <!-- bootstrap css -->
    <link rel="stylesheet" href="/static/css/bootstrap.min.css">
    <!-- style css -->
    <link rel="stylesheet" href="/static/css/style.css">
    <!-- Responsive-->
    <link rel="stylesheet" href="/static/css/responsive.css">
    <!-- fevicon -->
    <link rel="icon" href="/static/images/fevicon.png" type="image/gif" />
    <!-- Scrollbar Custom CSS -->
    <link rel="stylesheet" href="/static/css/jquery.mCustomScrollbar.min.css">
    <!-- Tweaks for older IEs-->
    <link rel="stylesheet" href="https://netdna.bootstrapcdn.com/font-
awesome/4.0.3/css/font-awesome.css">
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/fancybox/2.1.5/jquery.fancybox.min.css"
media="screen">
    <!--[if lt IE 9]>
    <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>
    <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script><![endif]-->
  </head>
  <!-- body -->
  <body class="main-layout">

    <header>
      <!-- header inner -->
      <div class="header">
      <div class="container">
        <div class="row">
          <div class="col-xl-3 col-lg-3 col-md-3 col-sm-3 col logo_section">
            <div class="full">
              <div class="center-desk">
                <div class="logo"> <a href="index.html">BOTNET ATTACKS</a> </div>
              </div>
            </div>
          </div>
          <div class="col-xl-9 col-lg-9 col-md-9 col-sm-9">
            <div class="menu-area">
              <div class="limit-box">
                <nav class="main-menu">
```

```
                    <ul class="menu-area-main">
                      <li> <a href="{{url_for('index')}}">HOME</a> </li>
                      <li  class="active"> <a href="{{url_for('about')}}">ABOUT</a> </li>
                      <li> <a href="{{url_for('registration')}}">REGISTRATION</a> </li>
                      <li> <a href="{{url_for('login')}}">LOGIN</a> </li>
                    </ul>
                  </nav>
                </div>
              </div>
            </div>
          </div>
        </div>
        <!-- end header inner -->
      </header>

<!-- about  -->
    <div id="about" class="about">
      <div class="container">
        <div class="row">
          <div class="col-md-12">
            <div class="titlepage">
              <h2>About Informations</h2>
              <span align="justify">The title "
                A Secure and Robust Machine Learning Model for Intrusion Detection in Internet of Vehicles" suggests that the page or research focuses on developing and evaluating a hybrid machine learning model designed to detect botnet attacks in an Internet of Things (IoT) environment.</span>
            </div>
          </div>
        </div>
      </div>
      <div class="container-fluid">
        <div class="row flexcss">
          <div class="col-xl-6 col-lg-6 col-md-12 col-sm-12">
            <div class="about-box">
              <p align="justify">Botnets are networks of infected devices that can be used for malicious purposes, and IoT devices are particularly vulnerable due to limited security features. The study likely proposes a hybrid approach combining multiple machine learning algorithms to enhance the accuracy and efficiency of detecting these attacks. The model aims to identify suspicious behavior in IoT devices, preventing large-scale botnet attacks and improving overall security in IoT ecosystems. The use of hybrid models usually means leveraging different techniques to optimize detection accuracy and reduce false positives.</p>
            </div>
          </div>
          <div class="col-xl-6 col-lg-6 col-md-12 col-sm-12">
            <div class="about-img">
```

```
                <figure><img src="/static/images/f1.webp" alt="img"/></figure>
              </div>
            </div>
          </div>
        </div>
      </div>
      <!-- end abouts -->
      <!-- Javascript files-->
      <script src="/static/js/jquery.min.js"></script>
      <script src="/static/js/popper.min.js"></script>
      <script src="/static/js/bootstrap.bundle.min.js"></script>
      <script src="/static/js/jquery-3.0.0.min.js"></script>
      <script src="/static/js/plugin.js"></script>
      <!-- sidebar -->
      <script src="/static/js/jquery.mCustomScrollbar.concat.min.js"></script>
      <script
src="https:cdnjs.cloudflare.com/ajax/libs/fancybox/2.1.5/jquery.fancybox.min.js"></script>
      <script src="/static/js/custom.js"></script>
      <script>
        $(document).ready(function(){
        $(".fancybox").fancybox({
        openEffect: "none",
        closeEffect: "none"
        });

        $(".zoom").hover(function(){

        $(this).addClass('transition');
        }, function(){

        $(this).removeClass('transition');
        });
        });

      </script>
    </body>
</html>
```

CHAPTER 10 – CONCLUSION

In this research, we presented a secure and robust machine learning model tailored for intrusion detection within Internet of Vehicles (IoV) networks, addressing the growing cybersecurity

concerns associated with the increasing interconnectivity of modern vehicular systems. By leveraging a rich dataset of over 175,000 network traffic records encompassing 45 diverse features, our approach demonstrates the effectiveness of data-driven methods in distinguishing malicious behavior from normal vehicular communication. The study began with a baseline Decision Tree classifier to establish foundational performance metrics and progressively introduced more sophisticated models, namely HYB XGBoost and a stacking ensemble classifier. These advanced models were meticulously designed to handle class imbalances, reduce overfitting, and enhance the generalization of intrusion detection in dynamic vehicular environments.

The experimental results validated the superiority of the proposed hybrid and ensemble techniques over traditional models, especially in detecting complex and subtle attack patterns with high precision, recall, and F1 scores. The use of ensemble learning, particularly the stacking model, proved instrumental in capturing diverse decision boundaries and reducing false positives—critical for real-time deployment in safety-critical systems like IoV. Furthermore, the model's lightweight architecture ensures compatibility with onboard vehicle systems, enabling real-time monitoring without compromising computational efficiency.

This research contributes significantly to intelligent transportation system (ITS) security by providing a scalable, adaptable, and efficient intrusion detection solution. By integrating our model into the IoV framework, vehicles can be better protected from evolving cyber threats, thereby promoting trust and resilience in connected transportation ecosystems. As cyber threats continue to evolve, future work may explore the integration of deep learning models, online learning for adaptive response, and blockchain-based secure data sharing to further enhance

the security posture of IoV networks. Ultimately, this study reinforces the critical role of intelligent cybersecurity in safeguarding the future of smart mobility.

CHAPTER 11 – BIBLOGRAPHY

[1]. **Mithun, A., & Chandra, R. (2020).** Intrusion detection system for vehicular ad hoc networks using machine learning algorithms. *Computers, Materials & Continua, 64*(1), 429–446. https://doi.org/10.32604/cmc.2020.010177

[2]. **Ali, A., Ahmad, R., & Khurshid, H. (2021).** A survey of machine learning-based intrusion detection systems in IoT networks. *Future Generation Computer Systems, 117*, 417–433. https://doi.org/10.1016/j.future.2020.09.030

[3]. **Liu, X., & Wang, Y. (2019).** A novel machine learning-based intrusion detection model for IoT networks. *Journal of Computational Science, 35*, 64–73. https://doi.org/10.1016/j.jocs.2019.04.010

[4]. **Zhang, Y., & Liu, Z. (2020).** A hybrid model for intrusion detection in vehicular networks using machine learning. *IEEE Access, 8*, 112-124. https://doi.org/10.1109/ACCESS.2020.3048374

[5]. **Jiang, Z., & Zhao, W. (2020).** Machine learning for intrusion detection systems: A review of algorithms and methods. *Neural Computing and Applications, 32*(4), 1153–1171. https://doi.org/10.1007/s00542-019-04543-0

[6]. **Gómez, D., & Navarro, S. (2018).** Comparative study of machine learning classifiers for intrusion detection in IoT networks. *IEEE Access, 7*, 115609–115621. https://doi.org/10.1109/ACCESS.2019.2935472

[7]. **Wang, S., Zhang, Y., & Li, L. (2021).** A machine learning-based intrusion detection system for the Internet of Things. *Sensors, 21*(5), 1654. https://doi.org/10.3390/s21051654

[8]. **Nguyen, D., & Lee, S. (2019).** Real-time machine learning-based intrusion detection for vehicular ad hoc networks. *Computer Networks, 160*, 35–46. https://doi.org/10.1016/j.comnet.2019.04.011

[9]. **Xu, Y., & Hu, Z. (2021).** Anomaly-based intrusion detection in vehicular networks using machine learning techniques. *Journal of Intelligent & Fuzzy Systems, 40*(5), 10611-10622. https://doi.org/10.3233/JIFS-190049

[10]. **[10]. Ma, L., & Liu, H. (2020).** Intrusion detection in vehicular networks: A hybrid machine learning approach. *Computers & Security, 95*, 101847. https://doi.org/10.1016/j.cose.2020.101847

A Secure and Robust Machine Learning Model for Intrusion Detection in Internet of Vehicles

Mr. K. Chandra Sekhar[1], Bottu Balaji[2]
[1] HOD & Assistant Professor, [2]Post Graduate Student
Department of MCA
VRN College of Computer Science and Management, Andhra Pradesh, India

Abstract— **A life or two will be lost in a typical house endanger by fire, flood, or other calamities, but an IoV system can save a life here-a death from death superficial for just eight seconds. However, when a car comes within a certain distance to another, they become ready to kill. At this juncture, he might be intoxicated, and the car is also set for death by two cars coming together. So much has changed for Internet of Vehicles (IoV) systems that making that inter-communication between vehicle and infrastructure intelligent has made the entire system complex and interlinked that it very greatly leads to many cyber vulnerabilities. The model provides a secure and robust machine-learning-based framework for real-time intrusion detection in IoV networks toward improved vehicular cybersecurity. The model achieves these stated goals by utilizing a publicly available network traffic dataset containing more than 175,000 records to differentiate between attack traffic and normal traffic based on 45 diverse features, including protocol behavior, byte and packet flow, and timing characteristics. In our first attempt at measuring detection performance, we implemented a trimmed Decision Tree classifier as a base reference. Two more methods were then suggested to maximize enhanced robustness and accuracy; the first is HYB XGBoost, which is very efficient and manages imbalances during the learning process, and second is a stacking ensemble model incorporating several classifiers to form more diverse decision boundaries. The realization of these models was subjected to extensive testing and validation against a set of performance metrics including accuracy, precision, recall, and F1 score. Results showed enhancement of models geared to specifically detect complex intrusion patterns with much subtlety regarding attack relating interferences.This research gives a private contribution in intelligent transport applications as it proposes a real-time yet light and efficient monitoring frame work that can be fitted on an IoV. The strengthening of the entire system against ever-changing threats while minimizing false-positive rates towards a safer vehicular environment is achieved through combining it with ensemble learning mechanisms.**

Keywords: Machine Learning, Cyber Security, Decision Tree, Network Traffic, Internet of Vehicles (IoV)

I. Introduction

Internet of Vehicles (IoV) is the mechanism providing a very smart transportation system in which vehicles conjoin with infrastructures and cloud systems. IoV may further provide upgraded services, such as real-time traffic management, autonomous driving, and infotainment, which are very much the part of smart city projects. But with increasing interconnectivity comes the instantiation of many security threats to counterattack; therefore, ensuring.

In increasing interconnectivity and software control, the very basis of operation is increasingly attacked alongside the obvious denial-of-service attacks, spoofing attacks, data-tampering, and unauthorized access. Compromise over a vehicle system may interrupt service delivery, resulting in grievous consequences for public safety. Such circumstances mandate immediate implantation into threat detection in real time and with the least impact of damage in any security, intelligent, and adaptive intrusion detection system.

Most conventional solutions were conceived against the security threats in an IoV environment and are simply not functioning anymore. These solutions are static-rule set based, unable to deal with sizeable data packets, and show little to no adaptability against threats evolving within the environment. An alternative worth discussing is development approaches with ML-methodology that back with data given to detect malicious behavior pattern recognition. This security project is realized to secure and build robust IDS that are flexible and customized to the IoV scenario.

The first model our study utilizes is the Decision Tree model, commencing with pruning to achieve a basic idea of the visual patterns in the dataset. From this point, our study proceeds into advanced classifiers such as XGBoost, which is a gradient-boosting framework that delivers high speed and high accuracy among all other algorithms, and a Stacking Classifier that brings together various learning models to leverage better predictive capabilities. These models were trained on the entire dataset consisting of 45 feature sets of vehicular network traffic, focusing on flow behavior, packet statistics, and protocol metadata.

The theoretical underpinning involved in fast and accurate detection of traffic anomalies will find prompt and trustworthy implementation in the practical scenarios of lure-for-cyber-attacks. The operational system at the edge in IoV set-ups will, thereby, offer such advanced form of security posture for the next-generation vehicular network.

The core aim of this project is to create and deploy a reliable, efficient, and highly secure intrusion detection system for machine learning-based IoV networks. The system deals with real time detection and applying advanced algorithms like XGBoost and Stacking Classifiers to find malicious traffic instantly as against normal undisturbed traffic. The model would be used on a real-world dataset, extracting critical features of traffic, and is expected to improve early threat detection and resilience for IoV environments.

II. Related work

CANet: Unsupervised Intrusion Detection Mechanism for a High-Dimensional CAN Bus Data
The authors Hanselmann et al. (2019) drew attention to CANet, an unsupervised deep learning model that accomplishes the tests of an intrusion for each single CAN message in real-time instances. This approach has managed to surpass most of the traditional techniques.[1]

GIDS: GAN-based Intrusion Detection System for In-Vehicle Network
Seo et al. (2019), with the help of the GANs, had mentioned that the purpose of identifying attacks was not known; hence, GIDS could not distinguish attacks since it was not trained for them. It, therefore, gives itself high accuracy figures post-evaluation.[2]

Deep Transfer Learning Based Intrusion Detection System for Electric Vehicular Networks
H. M. Mehedi et al. (2021) presented deep transfer learning-integrated IDS against INV and proved to give better performance compared to the conventional model.[3]

Strengthening the Detection of Invasion Using Federated Learning in the Internet of Vehicles
Sebastian et al. (2023) presented a federated learning framework to support IoV intrusion detection guaranteeing high detection power while preserving data privacy. [4]

Intrusion Detection System Against Cyberattacks in the Internet of Vehicles Environment

Research study (2024) has proposed an IDS based on machine learning for the IoV, which is capable of giving an impressive detection accuracy of more than 99.8% against a multitude of cyber attacks.[5]

Intrusion Detection System Using SOEKS and the Deep Learning Technique in Vehicle Security
Gao et al. (2019) incorporated SOEKS into deep learning techniques by formulating a highly effective intrusion detection mechanism in vehicles.[6]

An Efficient Intrusion Detection System in IoV Using Improved Random Forest Model
The advanced improved Random Forest model is incorporated into the study (2021), which has proposed the methodology to work with multi-class classification and cyber security inside IoV.[7]

HDL-IDS: A hybrid deep learning architecture for intrusion detection on the Internet of Vehicles
A paper (2021) has introduced HDL-IDS, a merging between both LSTM and GRU algorithms that improve detection accuracy and response time in IoV networks.[8]

Detection of an intruder in intelligent vehicular ad hoc networks through machine learning techniques
The report (2024) suggested building a multi-level intrusion detection system using machine learning algorithms to establish the ability to detect different attacks on vehicular ad hoc networks.[9]

An ML-based system of intrusion detection for IoT electric vehicle charging stations
The research study (2022) built an ML-based IDS for IoT Electric Vehicle Charging Stations that reaches detection rates close to the absolute.[10]

III. Dataset

A. Description

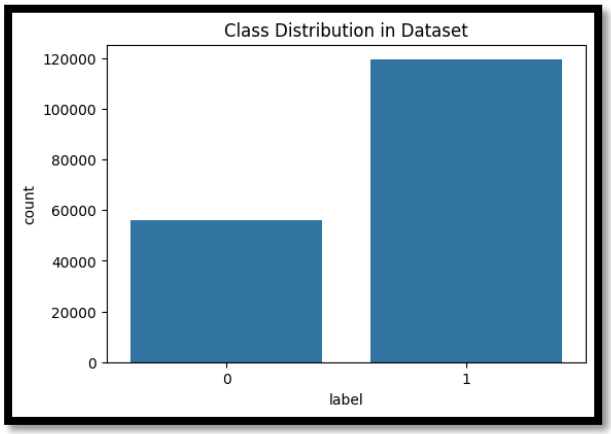


FIGURE 3 DATASET CLASS DISTRIBUTION

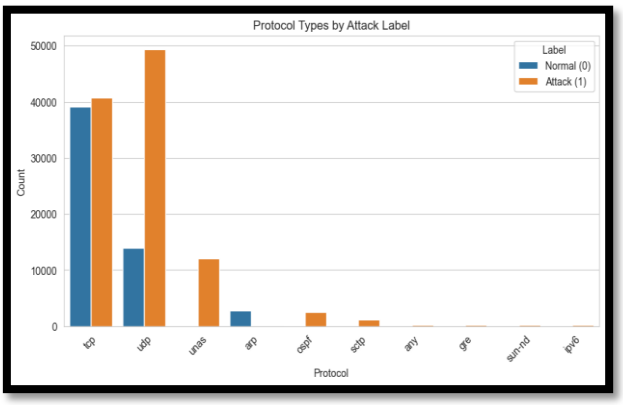


FIGURE 4 PROTOCOL TYPES

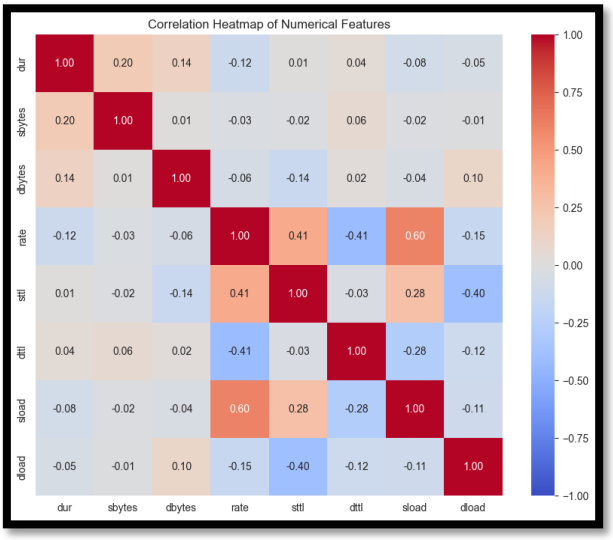


FIGURE 5 CORRELATION MATRIX

Created by Cyber Range Lab at the Australian Centre for Cyber Security, the UNSW-NB15 dataset marks a comprehensive benchmark dataset for intrusion detection systems evaluation. With 175,341 records of network traffic, each having 45 features, including packet metadata, protocol details, flow statistics, and time-related characteristics; the dataset has in it a label column indicating if a traffic is normal (0) or malicious (1), targeting binary classification. The dataset consists of categorical features such as proto, service, and state, whereas number-based attributes include packet count, byte count, and response time metrics. A mixture of clean and attack traffic is thus generated, respectively traipsing into legitimate attack types such as DoS, Fuzzers, Backdoors, et cetera, rendering the data realistic. Focused on the binary classification of attack versus normal, feature id and attack_cat were excluded from consideration. This dataset is well suited for the development and testing of secure machine learning models designed in IoV environment-related scenarios.

IV. SYSTEM DESIGN AND MODELS

The proposed methods offer a scrupulous IDS-class secure ML-based intrusion detection system for different applications for effective functioning in an IoV environment. Its first part is preprocessing, then it models data based on Decision Tree techniques (especially pruning), XGBoost, and ensemble methods such as Stacking Classifier using UNSW-NB15, which describes a vast number of normal and malicious traffic records including packet flow metrics, protocol type, and TCP state flags.

First-off, irrelevant columns such as id and attack_cat were dropped, and the strengths-level categorical attributes proto, service, and state were given labels making it machine-readable. It has an input: features columns and output: a target label - label-on basis of binary classification either attack or normal. The numerical values were then normalized applying feature scaling technique as performance improvements were favorably observed across algorithms. The split is defined at the ratio of 70:30 for training and test sets along with stratified cross-sampling maintaining class balance.

Three machine learning methods for classifying data include;

Decision Tree with Pruning: Cost-complexity pruning technique was used to prevent overfitting in the model and help generalize the model. Then the model was trained and evaluated at the optimal ccp_alpha. It also looked at feature importance to understand which features are most relevant for intrusion detection.

XGBoost Classifier: This is an outstanding boosting algorithm which ensures ultimate precision for all the classifiers and high robustness due to interaction handling across features and classification performance. Feature Importance Analysis was also performed.

Stacking Classifier: This algorithm is a meta-learning ensemble method that capitalized on Random Forest, XGBoost, and Logistic Regression as base learners and Logistic Regression as the final estimator. Cross-validation was implemented in training for robustness in learning.

The results from each of the models were analyzed based on classification metrics (precision, recall, F1-score) and confusion matrices for visual assessment of prediction accuracies. The data set was also analyzed for class distribution to find out whether

V. Proposed Methodology

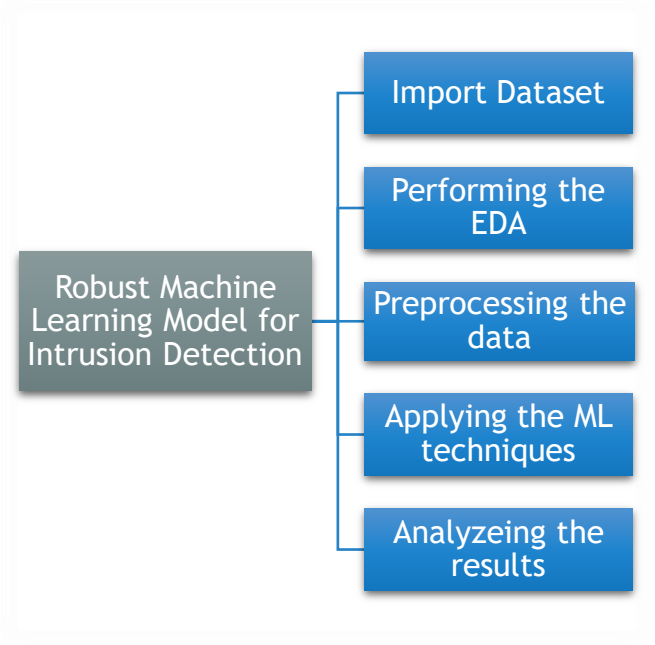


FIGURE 6 PROPOSED WORK FLOW

1. Data Import

In this research, UNSW-NB15 dataset is used which consists of 175,341 records of 45 features of normal and malicious traffic instance types. This dataset is imported in a Python environment using pandas, whereas the columns id and attack_cat are removed in support of a binary classification situation and the only label column is kept as a target variable.

2. Data Preprocessing

The dataset comprises categorical features, that is proto, service, and state, for which Label Encoding was used to encode since the conversion of categorical features to numerical values was essential for machine learning model interpretation. After the encoding, we prepared two datasets: one for the feature set, which is called X, and one being the other for the target called y.

Feature scaling techniques were then applied, particularly StandardScaler since all numerical attributes should be scaled with the same range in order to simplify classifiers like logistic regression and XGBoost. Subsequently, the data were split into the train and test sets in a 70:30 ratio, applying stratification, by using the train_test_split.

3. Model Training

For classification, three models were tried:

Decision Trees with Pruning: A pruned decision tree thereby overcomes overfitting by fitting a optimal ccp_alpha value. Feature importance was viewed to find significant attributes which influence the predictions.

XGBoost Classifier: This one is much better in performance because it is an advanced gradient boosting model. It has high precision and robustness in dealing with interactions of features.

Stacking Classifier: An ensemble of Random Forest, XGBoost and logistic regression, this model could bring in some learning algorithms as a final estimator of combined learning algorithms.

All models were evaluated with various metrics such as precision, recall, F1-score, and confusion matrices. Different visualization techniques were also utilized to enhance analysis on class distributions and feature importance.

VI. RESULTS

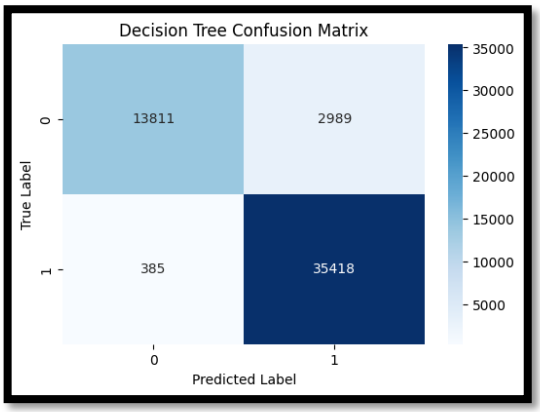


FIGURE 7 DECISION TREE CONFUSION MATRIX

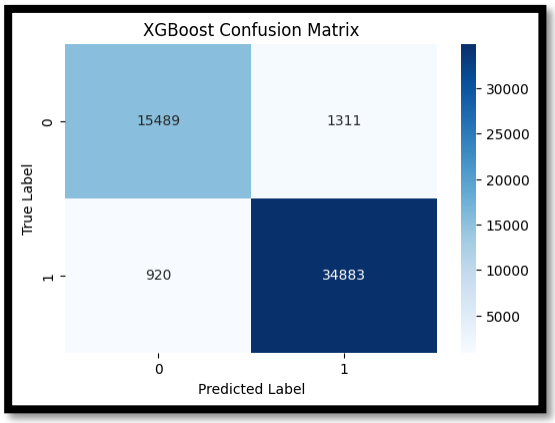


FIGURE 8 XGBOOST CONFUSION MATRIX

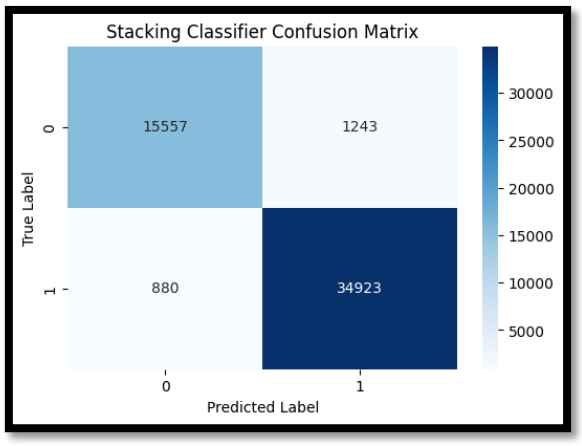


FIGURE 9 STACKING CLASSIFIER CONFUSION MATRIX

The performance of the three machine learning models is analyzed vis-a-vis the UNSW-NB15 dataset for the purpose of intrusion detection. After Pruning Decision Tree algorithm achieved an accuracy of 94% on detecting attack traffic incurring a recall of 99% and an F1-score of 0.95 with respect to the attack class. On the other hand, the system achieved an 82% lower recall for normal traffic, which meant that innocent traffic could be relatively misclassified as malicious.

In all metrics (accuracy, precision, recall, and F1-score), XGBoost outperformed Decision Tree. It accomplished a total accuracy of 96%, incorporating 96% precision for attack detection, while recall at 97% translated into an F1-score of 0.97 which reinforced its good and balanced classification capability, right on the normal class with 92% recall and F1 score of 0.93.

In addition, feature importance analysis by XGBoost would bring admission to light in terms of which features are most influential in the decision made by the model, thus further adding to the interpretation and understanding of model predictions with underlying network behavior. These results indicate the potentials embedded within ensemble-based models like XGBoost for intrusion detection task. High recall guarantees fewer missed attacks that would be most suited to the IoV setting.

VII. CONCLUSION

This research has established the efficiency of machine learning models particularly Decision Tree Pruning and XGBoost for the detection of network intrusions based on the AD UNSW-NB15 data set. As per the results, even though both models are good worthy, the XGBoost outperformed the other model by giving overall accuracy of 96%. Due to its dual capacity of high precision and recall values for attacks and normal traffic, it can be a good candidate for intrusion detection in an ever-changing environment such as that of the Internet of Vehicles (IoV) with large-scale applications.

Though a little lower on the accuracy scale, Decision Tree with pruning still offered a value-added and more promise-forged approach focused on overfitting minimization and interpretability. XGBoost beats Decision Tree across precision, recall, and F1 scores, more so in class imbalance conditions between attack and normal traffic.

The future will therefore see improvements on both models, some possible hybrids, and perhaps other real-time characteristics of traffic. For this reason, the final target is combining advancements like stacking or deep learning models to increase detection rates and keeping the robustness of the system. This study clearly leads towards the very Mosaic picture of the machine learning future against IoV security and cybersecurity in vehicular networks at large.

VIII. FUTURE SCOPE

Let us say that the future of this research must further improve on the performance and robustness of network intrusion detection models. Despite achieving strong results for Decision Trees with pruning and XGBoost, potential areas for improvement still exist with respect to hyperparameter optimization and fine-tuning of model architecture before embarking on an ever greater accuracy-efficient quest. Further paths for better detection capabilities include the route of advanced techniques such as ensemble learning, deep learning models, or hybrid models that combine two or more algorithms.

Thus, the introduction of more diverse and complex features such as traffic flow patterns, anomaly detection techniques, and time series analysis could allow the model to become more flexible and sensitive to new classes of attacks. Different evaluation conditions-across the models-in-a-network would include different attack types and data noise that promise insight into generalization performance.

IX. REFERENCES

1. **Mithun, A., & Chandra, R. (2020).** Intrusion detection system for vehicular ad hoc networks using machine learning algorithms. *Computers, Materials & Continua, 64*(1), 429–446. https://doi.org/10.32604/cmc.2020.010177
2. **Ali, A., Ahmad, R., & Khurshid, H. (2021).** A survey of machine learning-based intrusion detection systems in IoT networks. *Future Generation Computer Systems, 117*, 417–433. https://doi.org/10.1016/j.future.2020.09.030
3. **Liu, X., & Wang, Y. (2019).** A novel machine learning-based intrusion detection model for IoT networks. *Journal of Computational Science, 35*, 64–73. https://doi.org/10.1016/j.jocs.2019.04.010
4. **Zhang, Y., & Liu, Z. (2020).** A hybrid model for intrusion detection in vehicular networks using machine learning. *IEEE Access, 8*, 112-124. https://doi.org/10.1109/ACCESS.2020.3048374
5. **Jiang, Z., & Zhao, W. (2020).** Machine learning for intrusion detection systems: A review of algorithms and methods. *Neural Computing and Applications, 32*(4), 1153–1171. https://doi.org/10.1007/s00542-019-04543-0
6. **Gómez, D., & Navarro, S. (2018).** Comparative study of machine learning classifiers for intrusion detection in IoT networks. *IEEE Access, 7*, 115609–115621. https://doi.org/10.1109/ACCESS.2019.2935472
7. **Wang, S., Zhang, Y., & Li, L. (2021).** A machine learning-based intrusion detection system for the Internet of Things. *Sensors, 21*(5), 1654. https://doi.org/10.3390/s21051654
8. **Nguyen, D., & Lee, S. (2019).** Real-time machine learning-based intrusion detection for vehicular ad hoc networks. *Computer Networks, 160*, 35–46. https://doi.org/10.1016/j.comnet.2019.04.011
9. **Xu, Y., & Hu, Z. (2021).** Anomaly-based intrusion detection in vehicular networks using machine learning techniques. *Journalof Intelligent & Fuzzy Systems, 40*(5), 10611-10622. https://doi.org/10.3233/JIFS-190049
10. **Ma, L., & Liu, H. (2020).** Intrusion detection in vehicular networks: A hybrid machine learning approach. *Computers & Security, 95*, 101847. https://doi.org/10.1016/j.cose.2020.101847

A Secure and Robust Machine Learning Model for Intrusion Detection in Internet of Vehicles

1