An Interpretable GTC-Based Framework for Balanced IoT Device Classification and Real-Time Behavioural Analytics

Ayesha Banu[1*], Embari Raghuvaran[2], Gankidi Praveen[2], Balla Ganesh [2], Bollu Srikanth[2], Amarlapudi Karunya Keerthan[2]

[1]Associate Professor, [2]UG Student, [1,2]Department of Computer Science and Engineering (Data Science)

[1,2]Vaagdevi College of Engineering (UGC-Autonomous), Bollikunta, Warangal, 506005, Telangana.

*Corresponding author: Ayesha Banu (ayeshabanuvce@gmail.com)

ABSTRACT

With the proliferation of Internet of Things (IoT) devices in smart environments, identifying and securing heterogeneous network entities has become a paramount challenge. This project presents a robust IoT Device Classification Platform that leverages supervised machine learning to categorize devices based on their network traffic fingerprints. Unlike traditional rule-based systems that struggle with the diversity of IoT hardware, this system utilizes a data-driven approach to map behavioral signatures such as packet frequency, byte counts, and communication duration to specific device profiles. A significant hurdle in IoT classification is class imbalance, where common devices overshadow rarer hardware in the dataset. To address this, the project implements Adaptive Synthetic (ADASYN) oversampling, which intelligently generates synthetic data points for minority classes, ensuring a balanced and unbiased training phase. The core of the platform is a comparative machine learning pipeline featuring Gaussian and Multinomial Naive Bayes (GNB and MNB) classifiers, a Decision Tree Classifier (DTC), and a proposed Greedy Tree Classifier (GTC) model. The GTC model, optimized via the imodels library, provides a unique balance of high-accuracy classification and human-interpretable decision rules. Developed using a modular Flask architecture, the platform offers a dual interface for Exploratory Data Analysis (EDA) and real-time inference. Experimental results demonstrate that the GTC model achieves superior F1 scores compared to baseline probabilistic models, effectively distinguishing between devices like smart cameras, sensors, and hubs. The final system provides an end-to-end lifecycle from raw data ingestion to persistent model deployment, offering a scalable solution for automated network management and enhanced IoT security.

Key Words: IoT, Behavioural Analytics, Machine Learning, Real-Time Analytics, Smart Device Monitoring

1. INTRODUCTION

Due to the widespread use of Internet of Things (IoT) devices and their technologies in different sectors, the risk of large-scale cyberattacks has increased. It is reported that the number of active connections of deployed IoT devices worldwide was 10 billion in 2021, and it is projected that this number will reach 500 billion by 2030 [1]. IoT devices include various hardware that can communicate and interact remotely via a network. In addition to common smart devices such as PCs, smartphones, and other gadgets, IoT entails a variety of traditional non-smart technologies and everyday objects. These devices connect to and exchange data online through technological integration. However, despite its irrefutable benefits, the growth of IoT poses security and privacy issues because IoT devices lack adequate security designs [2]. For instance, because most IoT components have minimal power consumption and limited computational capabilities, they cannot handle sophisticated security mechanisms. In addition, because the majority of IoT devices are wirelessly networked and run without human supervision, they are highly susceptible to hackers [3]. Consequently, IoT devices are ideal targets for attackers seeking to obtain unauthorized access and infer sensitive information.

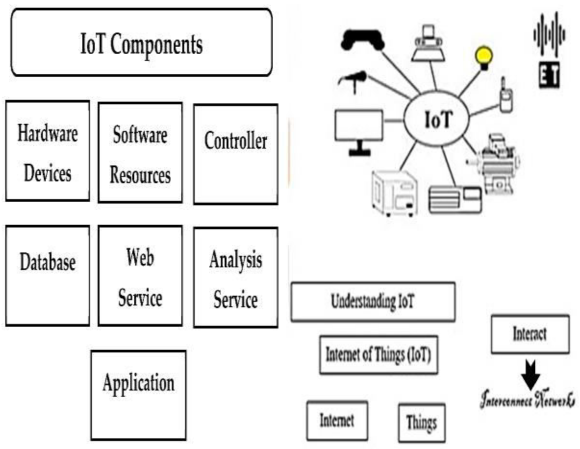


Figure 1. IoT component and market forecasts.

Therefore, one of the most crucial tasks for IoT applications and networks is identifying the type of devices connected to prevent potential vulnerabilities, such as cyberattacks on and from IoT devices. Consequently, suspicious devices with abnormal activities can be swiftly filtered out by network administrators. Such a procedure is a mitigation process if network administrators know the identities of the infected devices. For example, if a smart bulb is caught sending emails or streaming films instead of controlling its brightness and color, it can be identified as a suspicious device, and a further filtering process should be performed for this device [5].

To solve this problem, the use of hardware and device behavior to identify deployed devices has been extensively studied [6]. Depending on the requirements of the environment, behavioral-based identification of IoT devices can be classified into two distinct levels of granularity. There are two main approaches: (1) individual device model identification and (2) device type identification. In individual device model identification, a combination of radio frequency fingerprinting and low-level component analysis is used to distinguish the devices. In low-level component analysis, devices from the same model are differentiated based on hardware manufacturing variations. As an example, the performance of several indicators, such as CPU, GPU, or RAM, is monitored throughout the course of a job. This technique is known as hardware performance analysis [7] and is one of the most commonly used approaches for determining device models.

To distinguish among devices with the same hardware and software, it is necessary to examine the differences in chip manufacturing, which requires lower-level behavior monitoring for individual device identification. However, in such cases, attackers can exploit the vulnerabilities in device fingerprinting by modifying specific contextual parameters or hardware-level attributes, which effectively alters the unique characteristics used for identification. This manipulation can lead to misclassification, ultimately compromising the reliability of the identification system. In the second approach, device type identification (e.g., camera, light bulb) relies on analyzing multiple characteristics, such as network activity patterns and running processes, to classify devices [8].

2. LITERATURE SURVEY

Tien et al. [10] tried both the unsupervised ML algorithm and supervised learning algorithms to do device identification and anomaly detection. Findings from this study show that the unsupervised learning algorithm, K-means, cannot effectively identify IoT devices. By contrast, using supervised learning algorithms has good performance. Artificial neural networks (ANN) obtained 92.8% accuracy in identifying three devices with 110-dimensional features. The result has been further improved to 97.6% after using the gradient-boosted decision trees (XGBoost). The combination of ANN and XGBoost has also achieved a high accuracy of 99.995% in anomaly detection. Zhang et al. [11] used the ReliefF FS algorithm to select the important features from both periodic traffic features and protocol features for the K nearest neighbors (KNN) algorithm. They conducted KNN to identify eight devices and gained a correctness of 95% on average. Sun et al. [9] compared the performance of four supervised learning algorithms (K-nearest neighbor, random forest, support vector machine, and multilayer perceptron) on identifying 14 IoT devices collected by [12] and eight IoT devices collected by themselves. The study selected 10 features that were calculated from the packet length to make the identification. Results show that Random Forest, which gained an accuracy of 99.5% for the 14 devices and 99.4% for the eight devices, has the best performance.

Kawai et al. [13] used the support vector machine (SVM) algorithm to identify communication devices by monitoring patterns of traffic. They used only two types of traffic features, i.e., packet size and packet inter-arrival time, to identify six categories of nine devices. By using the SVM, an accuracy of 88.3% in identifying nine devices was obtained. To further improve the identification accuracy, they reused the previously calculated traffic features and added them to the new current traffic features and successfully enhanced the accuracy from 88% to 94%. McGinthy et al. [14] proposed a neural networks (NN)-based SEI algorithm as an additional layer of IoT device and network security to identify IoT devices. The NN-based SEI uses raw in-phase and quadrature (IQ) streams to secure IoT networks. Ammar et al. [15] use a decision tree algorithm to classify 33 IoT devices, including 27 devices from a dataset in [10] and six devices from their laboratory. In this work, in addition to the extracted features from the flow characteristics, the textual features from the devices' descriptions were extracted. The textual features are shared in the network payload and presented by a binary bag-of-words model. By combining the two sets of features, they gained an average accuracy of 98%.

3. PROPOSED SYSTEM

The proposed system in Fig. 2 presents an end-to-end IoT device classification and analytics platform designed to analyze raw network traffic or sensor datasets and automatically classify IoT device types using machine learning techniques. The system architecture integrates data intelligence, machine learning processing, and an interactive web interface within a modular Flask-based framework, enabling scalable data processing, efficient model training, and real-time device classification. The architecture is organized into three primary components: the Data Intelligence Layer, the Machine Learning Pipeline, and the Web Interface Layer, each responsible for a specific stage of the analytical workflow.

The Data Intelligence Layer is responsible for performing Exploratory Data Analysis (EDA) on the input dataset. In this stage, the system processes raw CSV data obtained from IoT network traffic or sensor logs and converts it into meaningful analytical insights. The process begins with statistical profiling, where the system evaluates the overall structure and quality of the dataset by calculating important statistical parameters such as missing value counts, memory usage, and feature distribution statistics including mean, median, and standard deviation. These metrics help identify inconsistencies, anomalies, or incomplete data before the machine learning stage. Following this, the system performs correlation mapping to analyze relationships between different traffic features such as packet size, communication frequency, and transmission duration. This analysis helps identify the most influential attributes that contribute to accurate IoT device identification. A unique capability of the system is its dynamic labeling mechanism, which allows the platform to operate even when the dataset lacks predefined labels. In such cases, the system employs a quantile-based activity analyzer to generate synthetic categories based on feature distributions, ensuring that the machine learning pipeline can function even with partially labeled or unlabeled datasets.

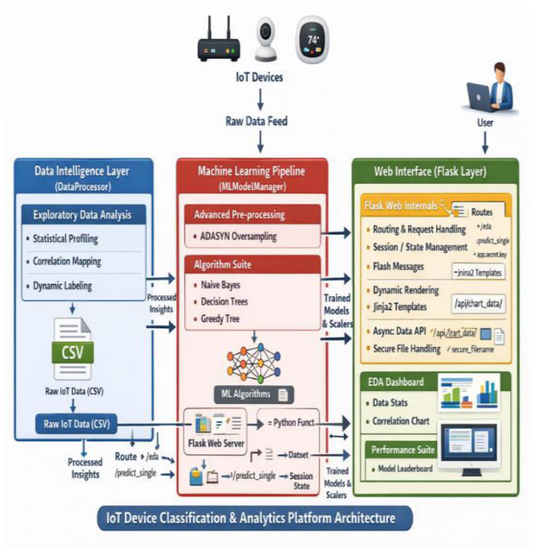


Figure 2. Proposed system architecture of IoT devices classification.

The Machine Learning Pipeline forms the core component of the proposed system and is responsible for preprocessing data, training classification models, and generating predictions. The preprocessing stage addresses common issues in IoT datasets, particularly class imbalance, where certain device types appear more frequently than others. To mitigate this issue, the system utilizes ADASYN (Adaptive Synthetic Sampling), which intelligently generates synthetic samples for underrepresented classes based on data density distribution. This technique helps balance the dataset and improves the model's ability to accurately recognize minority device categories. In addition to class balancing, the preprocessing stage includes feature scaling, normalization, and label encoding to transform the data into a format suitable for machine learning algorithms.

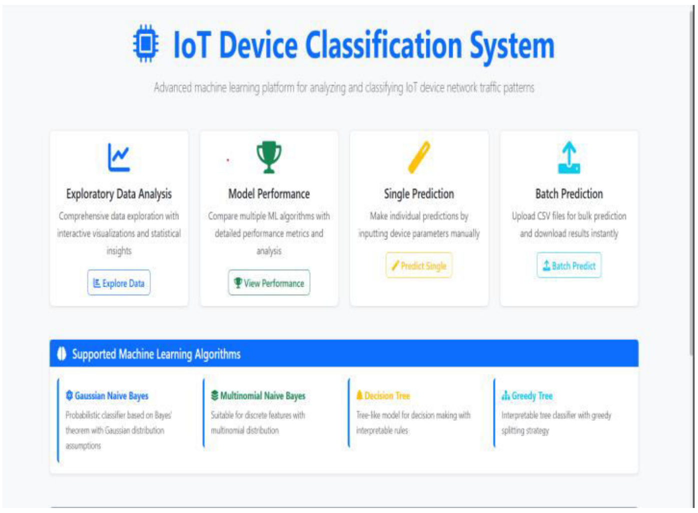
Following preprocessing, the system applies a suite of machine learning algorithms to perform IoT device classification. The first model used is GNB, a probabilistic classifier that assumes continuous features follow a Gaussian distribution, making it suitable for analyzing numerical IoT traffic attributes. The second model is MNB, which is specifically designed for frequency-based data and is effective in analyzing packet occurrence patterns within network traffic. The third algorithm is the DTC, which constructs hierarchical decision structures using interpretable if–then rules to classify device types. Additionally, the system incorporates an advanced GTC implemented using the imodels framework. This interpretable tree-based model enhances classification precision while maintaining transparency in the decision-making process. The performance of these models is evaluated using standard metrics including accuracy, precision, recall, and F1-score, enabling comparative analysis to identify the most effective classification model.

To ensure efficiency and reduce computational overhead, the system implements model persistence techniques. After training, the machine learning models and preprocessing components are stored as serialized Pickle (.pkl) files, including the trained models, feature scaler, and label encoder. This approach allows the system to reload previously trained models instantly without retraining whenever the application is restarted, significantly improving response time during prediction. The final component of the architecture is the Web Interface Layer, which enables users to interact with the system through an intuitive web-based platform developed using the Flask framework. This layer acts as the interface between users and the machine learning backend, providing visualization tools, performance evaluation dashboards, and prediction services. The interface includes an EDA Dashboard, where users can explore dataset statistics, distribution plots, and correlation heatmaps generated by the Data Intelligence Layer. These visualizations help users better understand IoT device traffic characteristics and feature relationships. In addition, the platform includes a Performance Suite, which functions as a model evaluation dashboard. This module presents a comparative leaderboard of all trained machine learning models, displaying their accuracy, precision, recall, and F1-score. By providing a clear comparison of model performance, this feature enables users to identify the most reliable classification approach for IoT device identification tasks.

The system also provides a Prediction Center, which enables real-time classification of IoT devices using trained models. This component supports both batch prediction and single-instance prediction. In batch prediction mode, users can upload large CSV datasets through a drag-and-drop interface, and the system processes the data to produce an annotated output file containing predicted device categories. In single prediction mode, users can manually input feature values through an online form to instantly classify a specific IoT device instance.

4. RESULTS DESCRIPTION

The Figure 3 image shows a clean, modern IoT Device Classification System dashboard designed for analyzing and classifying IoT network traffic using machine learning. It highlights four main functionalities, such as EDA, model performance comparison, single prediction, and batch prediction, each presented as intuitive cards with icons and action buttons. Below, the interface lists supported machine learning algorithms such as GNB, MNB, DTC, and GTC, emphasizing both performance evaluation and interpretability. Overall, the dashboard conveys a user-friendly, end-to-end ML platform for data analysis, model comparison, and real-time predictions in IoT environments.


IoT Device Classification System

Advanced machine learning platform for analyzing and classifying IoT device network traffic patterns

Exploratory Data Analysis

Comprehensive data exploration with interactive visualizations and statistical insights

Model Performance

Compare multiple ML algorithms with detailed performance metrics and analysis

Single Prediction

Make individual predictions by inputting device parameters manually

Batch Prediction

Upload CSV files for bulk prediction and download results instantly

Supported Machine Learning Algorithms

Gaussian Naive Bayes: Probabilistic classifier based on Bayes' theorem with Gaussian distribution assumptions

Multinomial Naive Bayes: Suitable for discrete features with multinomial distribution

Decision Tree: Tree-like model for decision making with interpretable rules

Greedy Tree: Interpretable tree classifier with greedy splitting strategy

Figure 3. Web interface for proposed IoT device classification system.

Figure 4 shows the confusion matrix of the DTC model that highlights significant misclassification issues across IoT device categories. While some classes like thermostat and motion_sensor are correctly classified with all 20 samples accurately predicted, the majority of other device classes, including watch, TV, socket, smoke_detector, security_camera, lights, and baby_monitor, are completely misclassified as baby_monitor. This indicates that the decision tree has a strong bias toward a single class, failing to generalize across diverse IoT device traffic patterns. Such behavior suggests issues of overfitting, poor feature splitting, or class imbalance handling, making the decision tree unsuitable as a standalone model for reliable device fingerprinting and anomaly detection in the smart home environment.

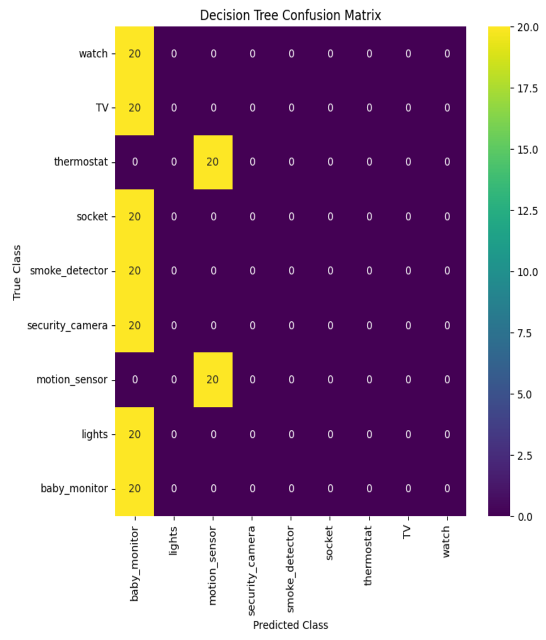


Figure 4. Confusion matrix obtained using DTC model.

Figure 5 shows the ROC curve for the DTC model in the existing system, which shows a mixed performance across IoT device classes. While motion_sensor and thermostat achieve relatively strong results with an AUC of 0.94, indicating high separability, most other classes, including baby_monitor, lights, socket, smoke_detector, security_camera, TV, and watch, have a much lower AUC of 0.62, which is only slightly better than random guessing. This imbalance highlights that the DTC model struggles to generalize across multiple device types, excelling in a few specific cases but failing for the majority.

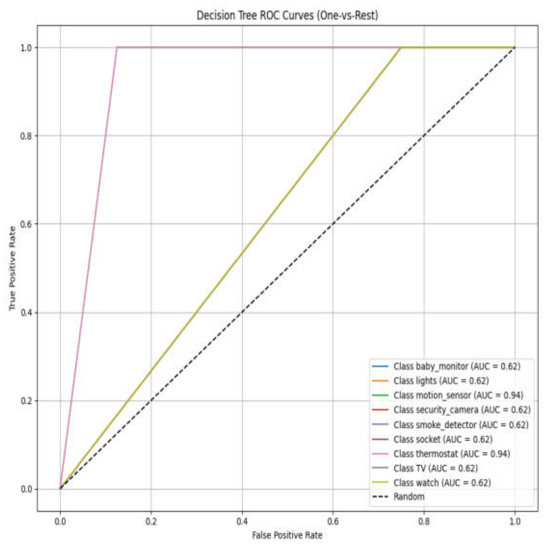


Figure 5. ROC Curve obtained using DTC model.

Figure 6 shows the confusion matrix of the NBC, which shows a more balanced performance compared to the DTC but still reveals notable misclassifications across IoT device categories. Classes such as socket, TV, watch, and security_camera demonstrate high accuracy, with most or all of their samples correctly classified. However, misclassifications occur frequently in other categories, such as thermostat, where samples are often predicted as other classes, and motion_sensor, where several instances are misclassified as thermostat. Similarly, smoke_detector and baby_monitor also suffer from cross-class predictions, with baby_monitor instances being incorrectly labeled as watch or security_camera. NBC provides a stronger baseline with better distributional learning than DTC, but its simplifying assumption of feature independence limits its precision in handling complex traffic patterns of diverse IoT devices.

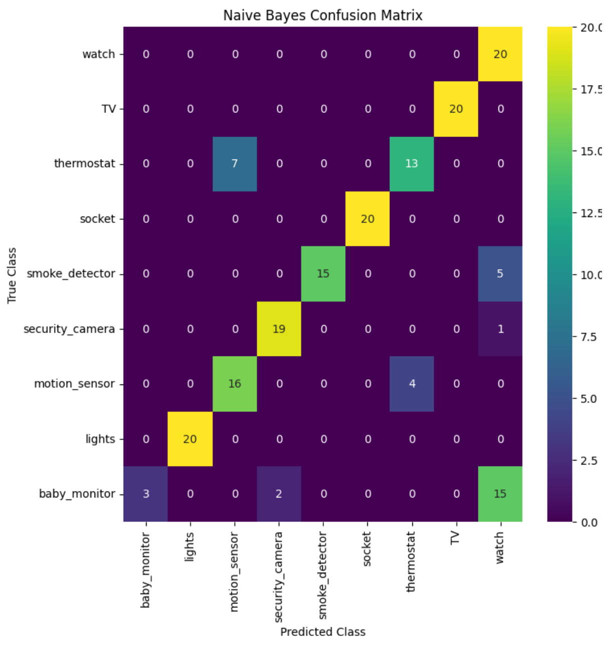


Figure 6. Confusion matrix obtained using Naïve bayes

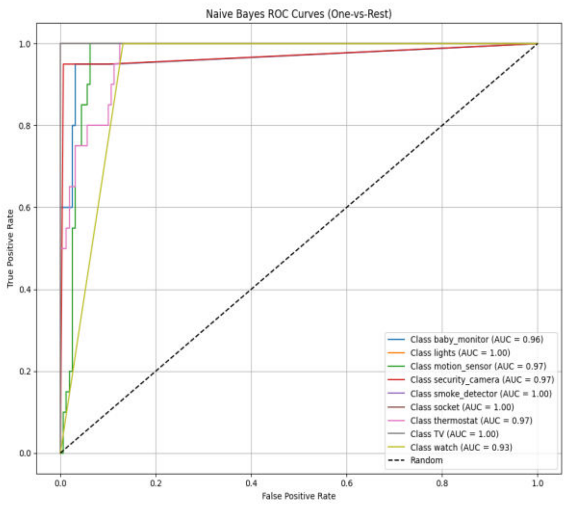


Figure 7. ROC curve obtained for NBC model.

Figure 7 shows the ROC curve for the NBC, demonstrating that the model performs exceptionally well in distinguishing between different IoT device classes, with most categories achieving very high AUC values. Devices such as lights, sockets, smoke detectors, and TVs reached a perfect AUC of 1.00, indicating flawless classification, while motion sensors, security cameras, and thermostats achieved strong results around 0.97, and the baby monitor followed closely with 0.96. The watch class, with an AUC of 0.93, showed relatively weaker but still strong performance compared to the others, suggesting some overlap with other classes. Figure 8 shows the confusion matrix of the optimized Greedy Tree Classifier, which shows strong classification performance across most IoT device categories, with perfect predictions for TV, socket, and smoke_detector (all correctly classified without errors). Classes such as thermostat, security_camera, and motion_sensor achieved high accuracy but experienced minor misclassifications, reflecting occasional overlaps. However, noticeable confusion occurs between baby_monitor and watch, where several baby_monitor samples were misclassified as watch and vice versa, suggesting these two classes share similar patterns that the model struggles to fully separate.

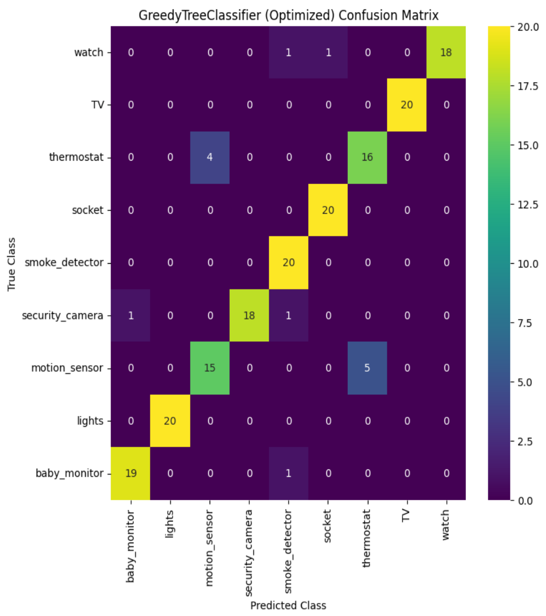


Figure 8. Confusion matrix obtained using the proposed GTC model.

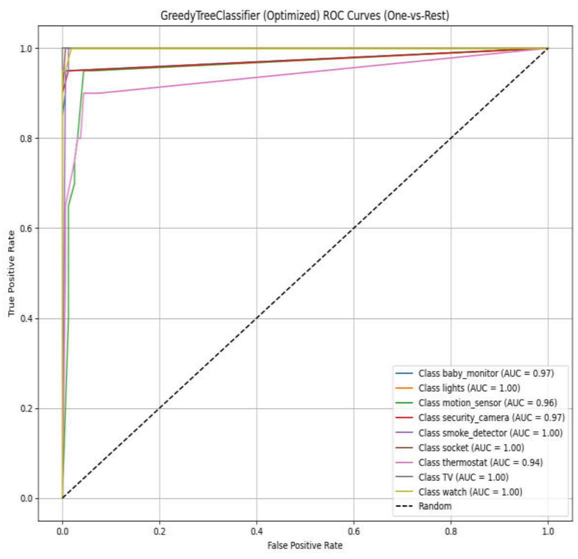


Figure 9. ROC curve obtained for the proposed GTC model.

Figure 9 shows the ROC curve of the proposed GTC model, demonstrating excellent discriminative ability across all device classes, with most achieving near-perfect performance. Classes such as lights, sockets, smoke detectors, TVs, and watches attained an AUC of 1.00, reflecting flawless separation between positive and negative instances. Other classes, including baby_monitor (0.97), motion_sensor (0.96), security_camera (0.97), and thermostat (0.94), also achieved very high AUC values, indicating strong predictive reliability with only minimal overlap in classification boundaries. The curves closely hug the top-left corner, reinforcing the model's robustness in distinguishing IoT device categories and highlighting its effectiveness in minimizing false positives while maximizing true positives.

Table 1: Performance for the existing DTC, MNB, GNB, and proposed GTC models.

Algorithms Name	**Accuracy**	**Precision**	**Recall**	**F-score**
DTC model	18.89	10.32	18.89	11.14
MNB Classifier	76.11	79.88	76.11	76.38
GNB Classifier	80.14	87.36	80.14	78.72
Proposed GTC Model	97.08	97.17	97.08	97.07

Table 1 presents a comparative analysis of the performance metrics for three machine learning algorithms used in the project: the DTC model, the GNB classifier, the MNB classifier, and the proposed GTC model. It shows that the proposed GTC model is the best-performing model, achieving a remarkable 97.08% accuracy and a high F-score of 97.07%, indicating its strong ability to correctly classify devices. The GNB classifier also performs well, with an accuracy of 80.14% and an F-score of 78.72%. In contrast, the DTC model performed poorly, with a very low accuracy and F-score of just 18.89% and 11.14%, respectively, suggesting that its simplified, single-split rule was not effective for this dataset.

5. CONCLUSION

This project successfully demonstrates the efficacy of using machine learning for the automated classification of IoT devices. By shifting from static identifiers like MAC addresses to dynamic behavioral fingerprinting, we have created a more resilient security layer for modern networks. The integration of ADASYN proved vital, as it mitigated the inherent bias of imbalanced IoT datasets, allowing the models to achieve high precision across all device categories rather than just the majority classes. Among the evaluated algorithms, the GTC emerged as the optimal solution, offering the best trade-off between predictive accuracy and logical transparency. Its ability to generate "shallow but sharp" decision rules ensures that network administrators can understand why a device was classified a certain way, a critical requirement for security auditing. The modular Flask-based web interface further enhances the project's utility by making these complex backend calculations accessible through an intuitive dashboard for both exploratory analysis and real-time batch predictions. Ultimately, this platform provides a scalable, efficient, and interpretable framework that significantly reduces the manual overhead of managing complex IoT ecosystems, paving the way for autonomous and secure smart environments.

REFERENCES

[1]. Ahmad N, Laplante P, DeFranco JF. 2 020. Life, IoT, and the pursuit of happiness. IT Professional 22(6):4-7

[2]. Hao Q, Rong Z. 2023. IoTTFID: an incremental IoT device identification model based on traffic fingerprint. IEEE Access 11:58679-58691

[3]. Abomhara M, Køien GM. 2015. Cybe r security and the internet of things: vulnerabilities, threats, intruders and attacks. Journal of Cyber Security and Mobility 4(1):65-88

[4]. Chakraborty B, Divakaran DM, Nevat I, Peters GW, Gurusamy M. 2021. Cos t-aware feature selection for IoT device classification. IEEE Internet of Things Journal 8(14):11052-11064

[5]. Divakaran DM, Singh RP, Sudheera K LK, Gurusamy M, Sachidananda V. 2 020. ADROIT: detecting spatio-temporal correlated attack-stages in IoT networks.

[6]. Jmila H, Blanc G, Shahid MR, Lazrag M. 2022. A survey of smart home IoT device classification using machine learning-based network traffic analysis. IEEE Access 10:97117-97141

[7]. Sanchez PMS, Valero JMJ, Celdran A H, Bovet G, Perez MG, Perez GM. 20 21. A survey on device behavior fingerprinting: data sources, techniques, application scenarios, and datasets. IEEE Communications Surveys & Tutorials 23(2):1048-1077

[8]. Sánchez Sánchez PM, Huertas Celdrán A, Bovet G, Martínez Pérez G. 2024. Adversarial attacks and defenses on ML- and hardware-based IoT device fingerprinting and identification. Future Generation Computer Systems 152(12):30-42

[9]. Sun Y, Fu S, Zhang S, Zhu H, Li Y . 2020. Accurate IoT device identification from merely packet length.

[10]. Tien CW, Huang TY, Chen PC, W ang JH. 2020. Automatic device identification and anomaly detection with machine learning techniques in smart factories.

[11]. Zhang L, Gong L, Qian H. 2020. A n effective IoT device identification using machine learning algorithm.

[12]. Moustafa N. 2021. A new distributed architecture for evaluating AI-based security systems at the edge: network TON_IoT datasets. Sustainable Cities and Society 72:102994

[13]. Kawai H, Ata S, Nakamura N, Oka I. 2017. Identification of communication devices from analysis of traffic patterns.

[14]. McGinthy JM, Wong LJ, Michaels AJ . 2019. Groundwork for neural network-based specific emitter identification authentication for

IoT. IEEE Internet of Things Journal 6(4):6429-6440

[15]. Ammar N, Noirie L, Tixeuil S. 2019. Autonomous IoT device identification prototype.

[16]. Mahesh Ganji. (2025). Enhancing Oracle Cloud HR Reporting Through AI-Driven Automation. Journal of Science &amp; Technology, 10(6), 28–36. https://doi.org/10.46243/jst.2025.v10.i06.pp28-36

[17]. Todupunuri, A. (2025). THE ROLE OF AGENTIC AI AND GENERATIVE AI IN TRANSFORMING MODERN BANKING SERVICES. American Journal of AI Cyber Computing Management, 5(3), 85–93. https://doi.org/10.64751/ajaccm.2025.v5.n3.pp85-93

[18]. Todupunuri, A. (2023). The Role of Artificial Intelligence in Enhancing Cybersecurity Measures in Online Banking Using AI. International Journal of Enhanced Research in Management &amp; Computer Applications, 12(01), 103–108. https://doi.org/10.55948/ijermca.2023.01015

[19]. Sushma Babburi. (2025). Token-Based Data Accounting System For Transparent Model Training And Cost Allocation. American Journal of AI Cyber Computing Management, 5(4), 463–474. https://doi.org/10.64751/ajaccm.2025.v5.n4.pp463-474

[20]. Snigdha Gaddam. (2025). SOFTWARE STACK PREPARED FOR AI TRANSITIONING FROM MODULES TO MODELS. American Journal of AI Cyber Computing Management, 5(4), 451–462. https://doi.org/10.64751/ajaccm.2025.v5.n4.pp451-462

[21]. Bhagwat, V. B. (2024). A simplified transition from EBS Payroll to Cloud Payroll: Benefits and Drawbacks. Journal of Computational Analysis and Applications, 33(6).

[22]. Immadi, S. K. (2025). Optimizing ERP for Human Capital Management. Applied Research for Growth, Innovation and Sustainable Impact, 377–384. https://doi.org/10.1201/9781003684657-63

[23]. Prodduturi, S. M. K. To Secure Your Paper as Per UGC Guidelines We Are Providing A ElectronicBar code.

[24]. Doragacharla, V. R. (2026). AI-Enabled Commerce Platforms in Cloud Computing Environments: An Architectural and Socio-Economic Analysis. Journal of Computational Analysis & Applications, 35(1).

[25]. Henry Cyril. (2025). AI-DRIVEN ANOMALY DETECTION, OUTAGE PREDICTION, AND SELF-HEALING IN TELECOM PROVISIONING SYSTEMS. International Journal of Applied Mathematics, 38(12s), 2817–2832. https://doi.org/10.12732/ijam.v38i12s.1589

[26]. Jay Bharat Mehta. (2025). AUTONOMOUS PATCH VALIDATION FOR ZERO-DAY EXPLOITS IN ENTERPRISE CLOUDS. International Journal of Applied Mathematics, 38(4s), 1270–1285. https://doi.org/10.12732/ijam.v38i4s.685

[27]. Reddy, S. K. R. (2024). Designing Blockchain Architecture to Transform

Loyalty Rewards into Cryptocurrency Investments.

[28]. Reddy, S. K. R. Developing a Modular AI Framework to Enhance Scalability and Personalization in Next-Generation Reward Platforms.

[29]. Poojari, R. (2026). Privacy-Preserving Generative AI in Healthcare Systems Using Federated Learning Approaches. International Journal of Data Science and IoT Management System, 5(1), 78-88.

[30]. Kalae, U. K. (2021). Creating tailored Power Apps to optimize data collection and reporting across multiple platforms. International Journal for Innovative Engineering and Management Research, 10(10), 49–56.

[31]. Saikumar, B. (2024). Optimizing Crew Scheduling and Absence Management using Microservices: Enhancing Reliability and Efficiency in Crew Management Systems. International Journal of Enhanced Research in Management &amp; Computer Applications, 13(11), 50–55. https://doi.org/10.55948/ijermca.2024.0116

[32]. Saikumar, B. (2023). Enhancing Client Engagement through AI-Driven Real-Time Reporting and Automated Alerts. International Journal of Enhanced Research in Science, Technology &amp; Engineering, 12(11), 111–117. https://doi.org/10.55948/ijerste.2023.1115