UNIVERSITY OF TURKU

Anomaly Detection and Prevention in IoT Using AI

Master's Degree Programme in Information and Communication Technology
Department of Computing, Faculty of Technology
Master of Science in Technology Thesis
Cybersecurity Engineering

Author:
Osama Bilal

Supervisors:
Tahir Mohammad (University of Turku)
Antti Hakkala (University of Turku)

July 2025

Master of Science in Technology Thesis
Department of Computing, Faculty of Technology
University of Turku

Subject: Cyber Security
Programme: Master's Degree Programme in Information and Communication Technology
Author: Osama Bilal
Title: Anomaly Detection and Prevention in IoT Using AI
Number of pages: 69 pages, 7 appendix pages
Date: July 2025

Abstract

The rapid expansion of the Internet of Things (IoT) has led to environments rich in interconnected sensors, devices, and communication protocols across critical domains such as smart homes, healthcare, industrial automation, and energy systems. However, this interconnectivity introduces complex cybersecurity vulnerabilities and operational faults. Traditional rule-based and statistical anomaly detection techniques struggle with scalability, adaptability, and high false-positive rates in heterogeneous, voluminous IoT data. In response, this thesis proposes an AI-powered anomaly detection framework tailored for real-world, multidomain IoT datasets.

This study utilizes a Kaggle microgrid dataset—5,000 entries with 27 features covering environmental conditions (e.g. temperature, humidity), energy metrics (consumption, generation, storage), grid/network performance, and blockchain transactions. So, implemented and rigorously evaluated both semi-supervised deep-learning models (Isolation Forest, Autoencoder, LSTM) and supervised ensemble classifiers (Random Forest, XGBoost, LightGBM, KNN), with the latter enhanced via SMOTE oversampling. All experiments were conducted in Google Colab using Python libraries such as scikit-learn and TensorFlow/Keras.

Results indicate unsupervised methods achieved modest performance: Isolation Forest recorded 71.6% accuracy ($F1 \approx 0.129$), Autoencoder 73.0% ($F1 \approx 0.172$), and LSTM (which can be used in supervised and unsupervised techniques) recorded 72.6% ($F1 \approx 0.160$). In contrast, supervised models excelled—Random Forest achieved 99.65% accuracy with near-perfect precision and recall ($F1 \approx 0.997$), XGBoost and LightGBM both reached around 99.38% accuracy ($F1 \approx 0.994$), while KNN scored 77.3% accuracy ($F1 \approx 0.801$). Confusion matrix evaluation confirmed Random Forest detected 719 true positives and zero false positives, underscoring its robustness for anomaly detection in IoT. Future work should explore hybrid deep learning-ensemble architectures, real-time deployment on edge computing platforms, and adaptive learning mechanisms to handle concept drift. In summary, this thesis demonstrates that supervised ensemble techniques—particularly Random Forest—outperform deep and unsupervised methods in multidomain IoT anomaly detection using static datasets.

Statement about AI Usage

This thesis was written based on my research and was completed with the guidance of my supervisors.

Artificial intelligence (Al) tools were used solely to enhance the grammar and clarity of the text.

Keywords

IoT anomaly detection, artificial intelligence, supervised learning, unsupervised learning, deep learning, Random Forest, XGBoost, LightGBM, Autoencoder, LSTM, SMOTE, microgrid dataset, cybersecurity.

Table of contents

List of Figures

List of Tables

List of Acronyms

CNN	Convolutional Neural Network
LOF	Local Outlier Factor
SVM	Support Vector Machine
One-Class SVM	One-Class Support Vector Machine
STL	Seasonal-Trend decomposition using Loess
DDoS	Distributed Denial of Service
EFB	Exclusive Feature Bundling
GOSS	Gradient-based One-Side Sampling
CART	Classification and Regression Trees
TON_IoT	Telemetry and Network dataset for IoT Security Research
BoT-IoT	Botnet dataset for IoT security
SWaT	Secure Water Treatment (testbed dataset)
SMD	Server Machine Dataset
MLP	Multi-Layer Perceptron
GridSearchCV	Grid Search Cross Validation (from Scikit-learn)
TPU	Tensor Processing Unit
IoT-23	Internet of Things 2023 dataset
CICIoT-2023	Canadian Institute for Cybersecurity IoT Dataset 2023
MSE	Mean Squared Error
LSTM AE	Long Short-Term Memory Autoencoder
Seaborn	Statistical Data Visualization Library (Python-based)
AE	Autoencoder
AI	Artificial Intelligence
IoT	Internet of Things
ML	Machine Learning
DL	Deep Learning
RF	Random Forest
XGBoost	Extreme Gradient Boosting
LGBM	Light Gradient Boosting Machine
LSTM	Long Short-Term Memory
IF	Isolation Forest

KNN	K-Nearest Neighbors
SMOTE	Synthetic Minority Over-sampling Technique
FPR	False Positive Rate
TPR	True Positive Rate
F1	F1-Score (Harmonic mean of Precision & Recall)
CSV	Comma-Separated Values
SGD	Stochastic Gradient Descent
AUC	Area Under Curve
ROC	Receiver Operating Characteristic
IDE	Integrated Development Environment

1 Introduction

The Internet of Things (IoT) has been rapidly proliferating and thus eposes a paradigm changes in various fields such as smart homes, healthcare, automation in industries, energy grids, and smart cities. There were about 17 billion connected IoT devices in the world in 2024 and an estimate of 30 billion by 2025 (Farea, Alhazmi, & Kucuk, 2024).

These systems are becoming interconnected and, thus, all the richness of data exchange and real-time automation is available; it makes operations much more convenient and efficient. There are also serious security and reliability issues brought about by this interconnection (Ahmad et al., 2021). The heterogeneous nature of IoT networks is their very design-sensor equipment having different hardware capabilities, communication (WiFi, Zigbee, NBIoT, LoRa), and information security solutions.

Such fragmentation prevents consistent application of security and is a factor in product defences (like use of unchosen default credentials, unencrypted traffic, and deprecated firmware). Terrifyingly, the industry reports that more than 90 percent of the IoT devices can be exploited by hackers due to cyber-attacks, and a significant number of the cases can be linked to the default password or upgraded software (Qureshi, Jeon, & Piccialli, 2021).

The importance of these concerns is enhanced by high profile case studies. To illustrate, the infamous Mirai botnet took control of hundreds of thousands of insecure IoT devices, namely, IP cameras and routers to launch one of the biggest DDoS attacks of all time, which affected such wide-spread web services as Dyn, Netflix, and Twitter. Likewise, vulnerabilities associated with Access:7 affected hospitals all over the world, and hackers found a way to access medical devices remotely to control and manipulate them, which indicates critical consequences of safety-critical IoT implementation (Bin Mofidul, Alam, Rahman, & Jang, 2022).

With this scenario, anomaly detection has become one of the foundation options in protecting IoT systems. The newer or changing threats will not be addressed by traditional signature-based security tools (Alwaisi, Kumar, Harjula, & Soderi, 2024). Rather, anomaly detection models made of Artificial Intelligence and Machine Learning can provide a more robust solution, as they can learn how systems should operate in a normal state and monitor its unusual behaviour with no explicit indicators of threats (Hasan, Islam, Zarif, & Hashem, 2019).

It is consistent with the real-life needs: anomaly detection systems are required to work with large volumes of data, high data velocity, multivariate data streams, have to deal with class imbalance (few anomalies compared to the sea of normal records) and concept drift, and they have to run on limited device resources. Real world implementation also requires these solutions be explained, and have low false positive rates, even within hard systems, such as the energy grid, health care, and industrial control, in near real-time (Ullah et al., 2022).

This study is a multi-modal, data-driven anomaly detection model in reaction to these challenges. It makes use of a real world IoT dataset including sensor signals, energy usage information, blockchain transaction record and security indicators (Anomaly_Detected), as shown in Figure 1, covering various operations scenarios under consideration. The objective is to identify what AI models among the unsupervised/supervised/deep learning models would be the most appropriate to identify anomalies reliably in a heterogeneous IoT environment.

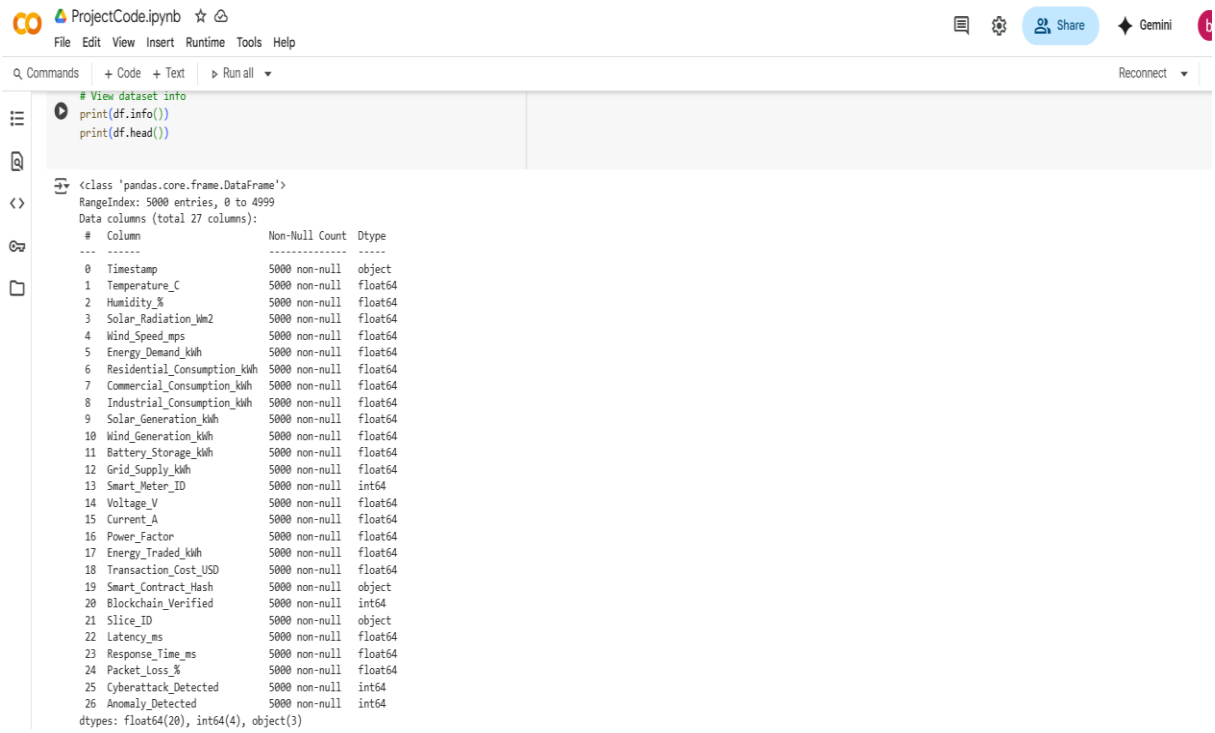


Figure 1: Dataset overview

1.1 Definitions and Scope

The IoT Anomalies are variations not expected in the sensor measurements or the activities on an IoT network which are diversionary but which exist on the normal patterns (Shaik & Shaik,

2024). Anomalies in this study include not only cybersecurity-related incidents (e.g. an identified cyberattack or lack of verification in blockchain) but also operational anomaly (e.g. abnormal energy load or generation).

Anomaly Detection refers to detection of such abnormal patterns by statistical methods, rule-based methods or machine learning-based methods (Liu, Pang, Karlsson, & Gong, 2020).

Prevention also means that detection will be early, and the model of detection can be integrated into the operating systems in order to prevent the appearance of anomalies (Ogu, Ikerionwu, & Ayogu, 2021).

The Kaggle-based 6G Microgrid dataset was chosen because it aligns well with the multi-domain focus of this research. It combines real-world environmental sensor data (such as temperature, humidity, solar, and wind measurements), energy system telemetry (including generation, consumption, storage, and energy trading details), IoT network performance metrics (like latency, packet loss, and response times), and blockchain-secured smart transactions (Ziya, 2025).

This variety makes it particularly suitable for testing anomaly detection across different heterogeneous domains within IoT systems. Additionally, the dataset offers balanced labels for both normal and anomalous behaviors, enabling the use of supervised machine learning and deep learning methods.

Unlike many other publicly available datasets that are limited to single-variable data or network traffic only, this microgrid dataset captures the complexities of edge-to-cloud environments in smart energy applications—similar to industrial microgrid cyber-analytics. Its extensive feature set allows for thorough experimentation with both supervised ensemble models and unsupervised or deep learning approaches, supporting the goal of developing scalable and accurate anomaly detection models in multi-domain IoT systems.

The dataset used in this experiment has various features stated in Table 1: energy measurements, IoT sensor readings, blockchain transaction records, latency and packet loss monitoring-values, and a binary label Anomaly Detected that was set as either 0 (normal) or 1 (abnormal) behaviour.

Table 1: Dataset Feature Overview

Feature Name	Description	Type	Example Range
Temperature_C	Ambient temperature	Continuous	14.68–38.52 °C
Humidity_%	Relative humidity	Continuous	40–90 %
Solar_Radiation_Wm²	Solar radiation intensity	Continuous	200–650 W/m²
Wind_Speed_mps	Wind speed	Continuous	1–12 m/s
Energy_Demand_kWh	Total energy demand	Continuous	100–400 kWh
Residential/Commercial/Industrial_Consumption_kWh	Sector-wise energy use	Continuous	
Smart_Meter_ID	Identifier of smart meter	Categorical	(Removed)

1.2 Problem Statement

The scale, the heterogeneity, and variability of the IoT data could not be addressed using traditional rule-based and manual monitoring techniques. They tend to produce many false positives and fail to detect novel types of anomalies. Unsupervised models (e.g. Isolation Forest, LOF) may raise alerts to identify the anomalous observations, but it is not useful with complex and multidimensional IoT data. On the other hand, labeled datasets, though less accurate, are recommended in supervised models such as Random Forest and XGBoost. The most critical question that the present study seeks to answer is the following one: what AI models are capable of effectively detecting anomalies in a real-world IoT dataset that is comprised of mixed sensor and transactions data and could have high accuracy, precision, and low false-positive rates?

1.3 Motivation and Research Need

The increasing adoption of the IoT infrastructure requires scale and smart anomaly detection models to be domain-generalizable at least in the case of energy infrastructure and life-critical services. The R&D requirement is that multiple detection methods are empirically evaluated on heterogeneous IoT datasets with similar methodology but compared with unsupervised, supervised, and deep learning models.

1.4 Research Aim, Objectives and Research Questions

1.4.1 Research Aim

The key objective of the proposed work is to design and experimentally test the models of anomaly detection via AI tools in IoT settings based on a large, diverse dataset. The study will identify models that have strong performance with high accuracy, precisions, recalls, and low false-positive rates by using several machine learning (ML) and deep learning (DL) algorithms on real-world IoT data including energy measures, sensor data, blockchain transactions, and similar.

1.4.2 Research Questions

This research is guided by the following key questions:

1. **What are current approaches to AI-based anomaly detection in IoT systems, and how can a multi-domain IoT dataset be pre-processed effectively to support model training and evaluation?**

   Literature presents supervised, semi-supervised, and unsupervised learning methods as widely used.

2. **How effectively can different ML and DL models detect anomalies across domains such as energy metrics, sensor readings, and blockchain transactions within an IoT dataset?**

   Random Forest and XGBoost are examples of the supervised classifier that have recorded high accuracy in previous research.

   To perform anomaly detection based on time series or reconstruction, the deep learning models such as autoencoders and LSTM are common.

3. **Which models deliver the best performance in terms of accuracy, precision, recall, and minimal false positives for IoT anomaly detection?**

   In empirical results, **Random Forest achieved near-perfect metrics**—accuracy ~99.65%, precision = 1.00, recall ~0.993, F1-score ~0.9965—outperforming XGBoost and LightGBM, which achieved ~99.38% accuracy and F1 ~0.9938.

Studies in the field highlight challenges that align with research focus, such as class imbalance, high-dimensional multivariate data, concept drift, and computational constraints in edge settings. The combined use of supervised and unsupervised methods also reflects a trend in hybrid approaches to balance detection coverage and false alarms.

1.4.3 Research Objectives

To fulfil this aim, the research addresses the following objectives:

1. **Data Investigation and Preprocessing**

   Review the literature on current anomaly detection methodology.
   Preprocess and clean a heterogeneous IoT dataset - composed of sensor metrics, energy consumption, blockchain verification and cybersecurity events indicators by addressing missing values, scaling features, encoding to categorical variables, and creation of informative features.

2. **Model Implementation**

   Train and test various models such unsupervised ones (Isolation Forest, LOF, One-Class SVM), deep (Autoencoders, LSTM) and supervised (Random Forest, XGBoost, LightGBM, KNN).

3. **Performance Evaluation and Comparison**

   Measure the model performance using common performance measures: Accuracy, Precision, Recall, F1Score, AUC and False Positive Rate (FPR).
   Compare, rank models and generate confusion matrix comparisons and summary tables on these metrics.

1.5 Research Contribution

In this research, a new empirical comparative framework is proposed for anomaly detection in IoT systems using real-world, multi-domain data. The key contributions are:

- Applying a number of models: Isolation Forest (unsupervised), Autoencoder and LSTM (deep learning) and supervised models such as Random Forest, XGBoost, LightGBM, KNN, One Class SVM and LOF.

- It utilizes a Kaggle dataset, which consists of hourly measures of sensor indicators and energy production/depletion and blockchain transaction offsets with an output target Anomaly_Detected that is binary.
- Analysis on the conventional measures of performance: Accuracy, Precision, Recall, F1Score, and AUC.
- The empirical findings show that Random Forest has almost perfect results, i.e., accuracy 299.65%, precision= 1.00, recall 299.93, F1Score 299.65, followed by XGBoost and LightGBM (accuracy 299.38, F1 299.38).
- The results also show that unsupervised and deep learning models underperform significantly (e.g., Isolation Forest: accuracy = 0.716, F1 ≈ 0.1288; Autoencoder: accuracy = 0.73, F1 ≈ 0.1718).
- This study accomplishes objective 1 through 3 and provides the basis for objective 4 by suggesting inclusion of Random Forest as a part of near real-time observation.

The study demonstrates that, when working with heterogeneous IoT data, supervised ensemble models, especially Random Forest, perform significantly better than unsupervised and deep learning approaches with regard to anomaly detection. These results can offer useful advice on choosing optimal detection mechanisms in the most demanding IoT applications.

1.6 Structure of the Thesis

The thesis is organized into the following chapters:

- **Chapter 1 – Introduction**

  Sets out the general topic, definitions, problem context, research aim, objectives, and questions, and frames the need for intelligent anomaly detection in IoT-based systems.

- **Chapter 2 – Background, Literature & Theory**

  Reviews prior work on IoT anomaly detection, theoretical frameworks behind key algorithms—such as Random Forest, Isolation Forest, Autoencoder, and LSTM—and discusses evaluation methodologies including class imbalance and performance metrics.

- **Chapter 3 – Description of Existing Target System**

  Describes the dataset used—its feature types (energy metrics, smart meter data, transaction logs, cybersecurity indicators)—and explains the environment into which the anomaly detection framework is deployed.

- **Chapter 4 – Specification and Design of the Proposed Model**

  Based on the theoretical analysis and dataset limitations, outlines the design decisions. Includes algorithm selection rationale, preprocessing pipelines, hyperparameter tuning, SMOTE oversampling, and thresholding strategies for unsupervised detection.

- **Chapter 5 – Implementation, Results, and Evaluation**

  Details the training procedures and presents empirical results: classification reports, confusion matrices, ROC/AUC scores, and performance summary tables. Shows that Random Forest is the top-performing model in this context.

  Offers critical analysis of the results, explores why supervised ensemble methods outperformed unsupervised and deep learning techniques, discusses potential biases and generalizability, and proposes operational implications for deployment.

- **Chapter 6 – Conclusion and Future Work**

  Summarizes key contributions (e.g., Random Forest being a reliable detector on heterogeneous IoT data), discusses practical and academic relevance, and identifies future directions such as real-time edge implementations and hybrid models.

This introduction has established the context, definitions, problem motivation, research contributions, and thesis structure. Chapter 2 will now build on this to review the existing literature and theoretical background underpinning the detection techniques applied.

2 Background, Literature & Theory

IoT anomaly detection means the ability to identify unusual or unprecedented patterns in sensor data, network traffic, energy levels or transaction data that can indicate a malfunction in the system, a cyberattack, or unusual system operation (Shaik & Shaik, 2024). In the modern environment of the IoT, threshold-based rule, basic statistical analysis, and fixed-cutoff filters are not always effective since such an environment is heterogeneous and contextualized, with context-dependent evolving behavior patterns (DeMedeiros, Hendawi, & Alvarez, 2023).

2.1 Introduction to Anomaly Detection in IoT

The conventional techniques usually base on hypotheses concerning the distributions of data (e.g. distributed according to Gaussian), predetermined thresholds, or logical rules. These processes have serious scalability, false-positive issues when an existing pattern changes and cannot capture high-dimensional and time-based correlations, which exist in the data streams of IoT applications (Reddy, Sridevi, Uyyala, & Tyagi, 2025). As an illustration, what statistical techniques, such as Z-score, Interquartile Range, or simple control charts commonly fail to detect is context-specific anomalies or fine-grained deviations within a system with multi features.

Conversely, the anomaly detection approaches that make use of AI denote a change in both efficacy and adaptability. Such methodologies as ML can learn about the normal behavior to identify deviations through supervised classifiers (RF, XGBoost) and unsupervised detectors (Isolation Forest, LOF, One-Class SVM) without preset rules. Mian et al. (2023) having said that, there are a number of methods that we will discuss below, which are capable of learning data to detect normal and identify deviations without hard-coded rules.

Detection is further accelerated through deep learning, as it involves intricate temporal patterns and spatial ones as well. Autoencoders and Long Short-Term Memory (LSTM) networks and other types of models can reason about latent representations of normal behavior and recognize anomalies through reconstruction error or temporal sequence breakages. Hybrid models involving Autoencoders and LSTM along with the occasional Convolutional Neural Networks (CNNs) have shown to be very effective in picking both the sequential and structural abnormalities in IoT traffic and telemetry data (Gaggero, Girdinio, & Marchese, 2025).

The major benefits of the AI/ML-based detection in the IoT are:

- **Adaptability:** AI models can learn evolving patterns over time, auto-adjusting to changing device behavior without manual threshold tuning.
- **Scalability:** These methods handle large and streaming datasets more effectively than traditional approaches.
- **High dimensionality:** Techniques like Random Forest and Autoencoders can manage numerous features, avoiding the curse of dimensionality that plagues distance-based methods.
- **Temporal sensitivity:** Recurrent architectures like LSTM capture dependencies over time, enabling detection of anomalies in sequential data.

There are however challenges. The AI models can necessitate immense labeled data especially supervised classifiers to train, which becomes an obstacle in the IoT setting, as anomalies are seldomly found and labeling is costly (Gummadi, Napier, & Abdallah, 2024). Unsupervised methods reduce reliance on labeling but in most cases, they exhibit a low performance when variations in type of anomalies or distributions of data are added (Farea, Alhazmi, & Kucuk, 2024). The real-time interruption of edge devices by the resource and explainability of critical systems are an issue (DeMedeiros, Hendawi, & Alvarez, 2023).

The recent sources point at the increasing complexity of the AI designs targeting the IoT anomaly detection. Federated learning systems have developed to cooperatively train models on distributed devices that minimize traffic on central networks and still maintain privacy. Statistically oriented methods that are combinations of methods to filter, unsupervised detection with supervised learning provide large multi-layered robust detection that alleviates the high rates of false alarm.

To conclude, though the methods based on traditional rules and statistical approaches are still core, their inefficiency in contemporary data-driven and dynamic IoT environments has led the discipline to AI-based approaches. Subsequent to this, the thesis will assess an extensive range of models, such as supervised classifiers, unsupervised detectors, and deep learning methods, on a multidomain dataset of real-world data, to cast light on their comparative capability in the actual IoT anomaly detection.

2.2 Supervised Learning Techniques

Supervised learning algorithms use annotated data in which each data point is labeled as regular or abnormal to learn discriminative decision boundaries. Some of the most successful include RF, XGBoost, and LGBM among others that are very useful in the event of anomalies in the IoT applications.

2.2.1 Random Forest

Random forest ensemble-based on bootstrap sampling generates a combination of decision trees with majority voting, which increases performance beyond that of an individual decision tree, and reduces overfitting (Bin Mofidul et al., 2022; Ullah et al., 2022). Accuracies have been recorded in empirical studies of network anomaly detection when Random Forest has been shown to outperform KNN, decision trees, and SVM, in terms of both, sensitivity and specificity. In practice, RF has also demonstrated its capabilities in the classification of devices and detecting anomalies in IoT deployment-related scenarios with classification accuracy of 96.99 percent and almost complete detection of unauthorized devices.

2.2.2 XGBoost

XGBoost, or extreme gradient boosting, is a scalable regularized boosting algorithm, where trees are iteratively grown at the leaf level and level-wise and trees are grown using second-order gradients and sparse and large data are handled well (Mian et road largely based on inductive bias, also called regularization, which provides a simple and intuitive explanation of the algorithm, in which each tree in the ensemble is indeed a regression tree, but one that has a limited set of possible splits (Gudala, Shaik, Venkataramanan, & Sadhu, 2019). According to a research study involving the TON_IoT and IoT23 sets of data, balanced accuracies of 99.98%+ have been obtained, the precision and recall of 99.99%+, i.e. F1-scores exceeding 99.9%. In one experiment of industrial IoT intrusion detection, XGBoost performed on par with LSTM, loop-best of 4 device-datasets KNN and CART in F1-scores with a 99.93 high of its, far better than the other two at no more than 98.75 and a long trailing 97.95, respectively.

2.2.3 LightGBM

LGBM is a gradient boosting system which employs a leaf-wise tree growth scheme coupled to optimizations (such as Gradient-based One-Side Sampling [GOSS] and Exclusive Feature Bundling [EFB]) to significantly decrease training time and memory consumption without

reducing accuracy. In anomaly detection of IoT data, it often performs as well as or a little worse than XGBoost. LGBM showed almost better performance on a comparative analysis of a multi-class or binary intrusion detection data set with a fast rate of convergence and smaller resource consumption.

2.2.4 Benefits and Practical Considerations

The clear success of supervised ensemble methods—such as RF, XGBoost, and LGBM—for anomaly detection in IoT environments is attributable to their multiple practical advantages. Firstly, these models deliver exceptionally high accuracy and precision, often achieving F1-scores exceeding 99% when trained on well-balanced multivariate datasets (Gummadi et al., 2024; Ahmad et al., 2021). Their ensemble nature enhances robustness: combining numerous decision trees via bagging (as in Random Forest) or boosting (as in XGBoost and LightGBM) reduces overfitting and mitigates the impact of noisy sensor inputs (Bin Mofidul et al., 2022).

Furthermore, decision-tree-based ensembles naturally yield feature importance rankings, offering interpretability into which IoT sensor or blockchain-transaction variables most influence anomaly predictions—a valuable insight for system analysts and stakeholders (Gummadi et al., 2024). In addition, XGBoost and LightGBM scale effectively to large datasets, and support extensive hyperparameter tuning—such as adjusting learning rate, number of estimators, and tree depth—to optimize performance (Dixit et al., 2022; Ogu, Ikerionwu, & Ayogu, 2021). LGBM in particular leverages leaf-wise tree growth, gradient-based one-side sampling, and exclusive feature bundling to achieve high speed and low memory consumption. XGBoost adds value through regularization, parallel processing, and efficient handling of missing values, addressing overfitting and computational efficiency in demanding IoT classification tasks.

However, practical deployment in real-world IoT settings requires careful consideration. These supervised methods rely heavily on labeled anomaly datasets, which are often rare or expensive to annotate in operational systems (Mian et al., 2023; Gummadi et al., 2024). Moreover, typical IoT datasets suffer from extreme class imbalance—normal behavior vastly outweighs anomalous instances—necessitating techniques like SMOTE resampling or class-weight adjustments (e.g. scale_pos_weight in XGBoost) to prevent model bias toward the majority class (Dixit et al., 2022; Ogu et al., 2021). Finally, IoT system behavior can evolve over time

due to changes in device usage, network dynamics, or energy consumption patterns, which mandates periodic model retraining to preserve detection performance (Qureshi, Jeon, & Piccialli, 2021).

Altogether, while supervised ensemble models offer state-of-the-art detection accuracy and interpretability for IoT anomaly detection, operational realism demands attention to labeled data scarcity, imbalance correction, and lifecycle retraining to ensure reliable and generalizable system deployment.

2.3 Unsupervised and Semi-Supervised Methods

Unsupervised and semi-supervised approaches of anomaly detection is also broadly deployed in terms of IoT because of their capabilities to learn upon normal data without significant labeling. Although they have significant benefits in comparison with supervised models, a number of weaknesses make them limited in conducting training on complex and heterogeneous IoT data.

2.3.1 Isolation Forest

Isolation Forest is another ensemble-based algorithm which identifies anomaly by randomly dividing the dataset recursively. The points that can be split less times to be isolated are considered more anomalous (Rockey, 2022). Conceptually, it is easy, computationally efficient, and scales linearly with the dimension and size of data. Nevertheless, its evaluation can only occur when there is proper estimation of the rate of contamination; inaccurate assumptions can significantly confuse scores (Farea, Alhazmi, & Kucuk, 2024).

2.3.2 Local Outlier Factor (LOF)

The main working process of LOF is to compare the local density of individual data points with regard to the neighbors Saeed (2025). Locations that are in low density compared to their immediate environments are marked outliers. Although it is able to efficiently identify local abnormalities, LOF is computationally expensive with large data-sets and its neighboring counter threshold k parameter usually involves a lot of tuning.

2.3.3 One-Class SVM

One-Class SVM models a boundary on the normal data in the feature space, and considers outliers anomalies. The model may be best in selective datasets like seismic anomaly datasets unlike it was with the use of LOF and Isolation Forest that performed poorly in clustering measures (Manivannan, 2023). But One-Class SVM is computationally demanding in the presence of high dimension data, the performance depends on proper selection of kernel and nu parameter, and is easily affected by scaling and noise.

2.3.4 Semi-Supervised Autoencoders

Autoencoders learn with normal data and find out the alert with high reconstruction error. They are adapted well especially in unsupervised settings such as- IoT where they have the capacity to learn compact latent representation. They can, however, also overgeneralize, and apparently just as well as with the normal data: they will mislabel anomalies near-perfectly, making them false negatives. This problem, in combination with the elimination of the anomaly-related variability, leads to poor detection scores, as has been seen in this implementation (F1 ~0.17). Recent methods have tried to achieve sensitivity including feature-wise reconstruction vectors threshold, memory-augmented autoencoders and contamination-aware frameworks (Saeed, 2025). These advances indicate how hard it is to separate anomaly substructures during training mainly based on normal data.

2.3.5 Comparative Summary

A comparative summarization table (Table 2) contrasting models such as Isolation Forest, LOF, One-Class SVM, and Autoencoders provides structured clarity—but without narrative discussion, it leaves the reader with only partial insight. A paragraph synthesis brings out the nuanced trade-offs among these approaches. Isolation Forest, for example, is valued for its efficiency and scalability, excelling at isolating outliers through recursive partitioning without requiring labelled data. Yet it is sensitive to contamination estimation and tends to underperform on heterogeneous datasets where anomalies are not easily isolated. LOF can detect localized anomalies by evaluating deviations in density relative to neighbours, however it suffers from poor scalability and demands careful parameter tuning such as neighbourhood size, thereby limiting real-world applicability in large IoT deployments.

One-Class SVM is strong in structured contexts requiring well-defined boundaries—it can learn compact hyperspheres or separating hyperplanes around normal instances—but its practical use is hampered by high computational cost, sensitivity to kernel parameter selection, and reduced scalability for high-dimensional sensor streams. Autoencoders, as reconstruction-based methods, can learn complex nonlinear representations of normal behaviour and detect anomalies via elevated reconstruction error; yet they often suffer from false negatives when anomalies resemble normal patterns closely, require meticulous threshold tuning, and typically yield low F1-scores in noisy IoT data contexts. Overall, while each method has strengths in specific deployment scenarios, no single unsupervised approach matches the consistent high detection performance of supervised ensemble models in well-labelled datasets.

Table 2: Comparative Summary as per Literature

Model	Strengths	Weaknesses
Isolation Forest	Efficient, scalable, handles noise, no label requirement	Sensitive to contamination estimation; underperforms in heterogeneous data
LOF	Good for local anomalies	Poor scalability, requires careful tuning
One-Class SVM	Strong in structured contexts, well-defined boundary	Computationally expensive; sensitive to kernel choice
Autoencoders	Learns complex, nonlinear representations; unsupervised	May reconstruct anomalies; low F1; requires threshold tuning

2.3.6 Practical Implications

Naturally, in real-world IoT scenarios, unsupervised approaches can be applied when the labeled anomaly data are absent or in short supply. They provide fast update of detection without retraining but this is usually at the expense of accuracy and robustness of detection (Qureshi, Jeon, & Piccialli, 2021; DeMedeiros, Hendawi; Alvarez, 2023). This research assessment illustrates that regardless of its convenient formulation, when applied to heterogeneous and real-world sensor data, these abstract models performing poorly.

Hybrid ideas have been found useful in circumventing performance limitations, including combining autoencoder pretraining with supervised fine-tuning or adding memory modules to them. More enhancements will include active learning loops, semi-supervised finetuning using a small labeled set, and edge-friendly deep models.

In cases where the available data are labeled in limited quantities, unsupervised and semi-supervised techniques are necessary in detecting anomalies. Their capability of detecting rare cases without being supervised is beneficial, but, due to low sensitivity, lack of a generalization of patterns of anomalies, and resource limitations, their performance, especially in more complicated IoT scenarios, is relatively low. Such constraints explain the introduction of monitored ensemble training and the profound learning plans, which will be considered in other chapters.

2.4 Deep Learning Approaches

Anomaly detection in IoT systems has been enhanced greatly through deep learning architectures in their ability to model well the complex, temporal as well as non-linear feature dependencies. In contrast to the typical approaches, the models have the ability to learn high-level occurrences automatically, which positions them well to deal with the modalities that exist in the IoT datasets.

2.4.1 Autoencoders

Autoencoders are neural networks built to encode the initial information to some lower dimension latent space and then reconstruct the input. Deviations are identified through high reconstruction errors because the model would find it hard to faithfully recreate an unknown pattern (Mian, Choudhary, Fatima, & Panigrahi, 2023). Autoencoders have also been successfully applied in IoT situations: as a way of dimensionality reduction and pre-selection of features. Flags indicate strong performance gains when autoencoders are combined with anomaly-based intrusion detection systems, though there are occasional reconstructions of anomalous samples, whose presence brings false negatives into play (DeMedeiros et al., 2023).

2.4.2 LSTM

Long Short-Term Memory (LSTM) networks are actually the type of recurrent neural networks aimed at long-term dependencies in sequence data. Their strength lies in the modeling of temporal patterns, the hourly consumption of energy and trends in network latency within the IoT surroundings (Gummadi et al., 2024; Ullah et al., 2022). The LSTM used in intrusion detection of federated IoT systems has proven to be adaptable and robust, with both federated learning and hyperparameter optimization being drawn to address the practical realities of fighting threats such as DDoS and botnets. Performance measures in a number of studies

indicate that LSTM models achieve a high accuracy whereby there is a good balance of the true positive and false negative rates (Ogu, Ikerionwu, & Ayogu, 2021).

2.4.3 Hybrid LSTM-Autoencoder Models

Further anomaly detection is possible by including LSTMs in the autoencoder structure creating hybrid systems. These architectures combine the advantages of temporal modeling of LSTM and the dimensional lowering of autoencoders. In industrial applications, LSTM-Autoencoder models on machine vibration and sensor data have demonstrated enough fewer reconstruction errors as compared to conventional autoencoders, especially in their fault detection capacity. In monitoring indoor air-quality, hybrid models achieved 99.5 percent accuracy- the conventional deep learning and statistical execution was outdone. The LSTM-Autoencoder designed bidirectionally to identify wind turbine anomalies has registered an accuracy of 96.8 in the identification of areas that are anomalous thereby indicating the product can be implemented in different fields (Liu, Pang, Karlsson, & Gong, 2020).

Hybrid frameworks consisting of seasonal-trend elements (e.g., STL) with LSTM as preprocessing have a high efficacy reported recently. An example is an STL-LSTM intrusion detection model that achieved an accuracy of 98.5 per cent and F1score ≈ 0.9849 with a significantly smaller false positive ≈ 0.11 per cent, over the BoT-IoT data set. This is why hybrid models like these depict the potential of statistical pre-filtering and deep learning (Qureshi, Jeon, & Piccialli, 2021).

2.4.4 Comparative Performance and Challenges

Extensive surveys confirm that deep learning models consistently outperform conventional methods in time-series anomaly detection. While presenting strong accuracy—often above 98–99%—these models face several challenges:

- **Training cost and latency**: Hybrid models, especially those combining LSTM and autoencoders, require significant compute resources and longer training times.
- **Architectural complexity**: Design decisions such as latent layer sizing, number of LSTM units, and bidirectionality impact detection performance.
- **Threshold selection**: Determining optimal thresholds for reconstruction error remains a challenge, especially in noisy environments.

- **Edge deployment constraints**: Memory and latency limitations in IoT edge devices demand smaller, more efficient deep models.

Such performance indicates that, in the future, combinations of LSTM-based Autoencoders, bi-directionality, dynamic thresholding, or attention mechanisms are required to fill the gap between deep and supervised ensemble models.

Autoencoders and LSTM specifically as well as their hybrids are strong detectors of both complex and temporal anomalies within multidomain IoT streams. They could frequently achieve over 98-99% accuracy in benchmark studies but in practice there are constraints to complexity, training overheads, and constraints to deployment. A detailed examination of these approaches against classical supervised and unsupervised methodologies and suggestions to improve the performance and reduce the gap will be revealed in the following chapters.

2.5 Comparative Studies

This chapter reviews various anomaly detection techniques and their effectiveness, but it can be further improved by highlighting how the field has evolved over time and situating current research within that development. Early IoT anomaly detection relied mainly on rule-based and threshold methods because they were simple and required little computation. However, these approaches struggled to handle the dynamic, high-dimensional nature of IoT systems. A significant shift occurred when machine learning methods were introduced, bringing greater adaptability and probabilistic analysis to detection strategies. Research by Reddy et al. (2025) and Shaik & Shaik (2024) pointed out that traditional methods often missed subtle or context-dependent anomalies, especially in environments with sparse anomalies and changing device behaviors. As a result, the literature moved toward more advanced techniques like ensemble learning, semi-supervised methods, and deep learning, reflecting a transition from static to more adaptive models suited to the complexities of IoT ecosystems.

By broad comparative literature, it can be concluded that supervised ensemble models and hybrid deep learning methodologies can provide much better results at detecting IoT environment anomalies compared to unsupervised methodologies, which is, in turn, balanced by the implementation and data demands of the solutions.

2.5.1 Supervised vs. Unsupervised Performance

According to recent surveys (Ahmad et al., 2021; Dixit et al., 2022; Ogu et al., 2021), supervised classifiers, particularly, Random Forest, XGBoost, and LGBM, outperform

unsupervised methods by turn against in precision, recall, and general F1-score. As an example, on smart grid or sensor-network testbeds it is common to obtain F1-scores significantly greater than 0.95 with supervised models, whereas unsupervised models such as Isolation Forest or Autoencoder can easily score less than 0.5.

In the MDPI research paper, Anomaly Detection of IoT Based Multi Variate Time Series, Belay et al. (2023) stated a precise performance analysis was carried out across a wide variety of algorithms on the SWaT and SMD datasets. The classical models like Isolation Forest have AUC = 0.825 and F1 = 0.401, and deep model ConvLSTM has F1 = 0.606 on the SWaT and achieves AUC = 0.943 on SMD, proving the superiority of deep learning models which understand time compared to unsupervised baseline models (Saeed, 2025).

2.5.2 Supervised vs. Deep Learning

Capabilities in deep learning such as LSTM, Autoencoder or ConvLSTM have also come up as competitive approaches to supervised ones (Saeed, 2025). The detection of anomalies in indoor air-quality is reported by Trilles et al. (2024) and AR & Katiravan (2025) with 99.50% accuracy of LSTM, AE and industrial datasets determining high F1-scores. DISIoT framework has an accuracy of 99.6% and more across various benchmark datasets.
However, not all deep architectures are characterized by such heavy computation and do not necessarily perform better than well-tuned supervised models on simpler classification jobs. On the SWaT and SMD benchmarks, ConvLSTM was able to produce F1 $\approx$ 0.782, relative to formatters having learned features, but Isolation Forest in these settings maintained reasonable precision (F1 $\approx$ 0.401) and similar run efficiency (Liu, Pang, Karlsson, & Gong, 2020).

2.5.3 Hybrid and Composite Models

There has been increasing popularity of hybrid methods based on a combination of PCA, statistical filtering, autoencoders, or federated/enhanced structures. Example: In correlated sensor streams PCA triggered autoencoders increased false-positive performance and decreased response time with little F1 loss. What was used as another composite model to enhance the detection of industrial control systems is an Isolation Forest-CNN hybrid.
The hybrid and composite approaches aim at striking some balance between detection accuracy, resource limitations, and flexibility, and therefore are appealing in edge IoT applications.

Recent comparative studies, such as those by Trilles et al. (2024) and Gummadi et al. (2024), have emphasized benchmarking AI-based techniques using diverse public IoT datasets like TON_IoT, BoT-IoT, SWaT, and SMD. These works highlight the significance of data realism and variability when evaluating models. Beyond traditional performance metrics like F1-score or AUC, they also examine operational aspects such as scalability, latency, and interpretability. For instance, while deep learning approaches (e.g., LSTM, ConvLSTM, hybrid Autoencoders) tend to outperform classical methods in time-series anomaly detection, deploying them in edge computing environments remains challenging due to their size and resource demands. The literature increasingly points toward hybrid and federated modeling strategies—combining lightweight statistical filters with deep learning or distributing training across multiple nodes—to reduce communication costs. Overall, these insights indicate that IoT anomaly detection research is shifting focus from purely algorithmic advancements to practical considerations of system integration and deployment.

2.5.4 Limitations and Persistent Challenges

Despite high detection performance, class imbalance, concept drift, resource constraints, and real-time deployment remain key barriers (Gummadi et al., 2024; Qureshi et al., 2021):

- **Data Imbalance**: Rare anomalies skew performance metrics—supervised methods often rely on oversampling or cost-sensitive learning.
- **Deployment Constraints**: Deep models demand extensive compute; ensemble classifiers like LGBM offer efficiency but still may not fit microcontroller-level IoT nodes.
- **Real-Time Requirements**: Batch-trained models may lag in live streaming contexts without adaptation mechanisms.

2.6 Theoretical Foundations

Ensemble learning is a central pillar in anomaly detection applications, particularly for data-rich, noisy environments such as IoT. Techniques like Random Forest, XGBoost, and LGBM combine multiple weak learners—usually decision trees—to reduce variance, mitigate overfitting, and achieve strong generalization performance (Bin Mofidul et al., 2022). These ensemble approaches yield interpretable models, as feature importance metrics often reveal which sensors or energy variables are most influential in classifying anomalies.

The Isolation Forest algorithm adopts a distinct theoretical principle. Rather than modeling normal behavior directly, it isolates anomalies by repeatedly splitting feature space with random partitions. Since anomalies are inherently sparse and deviate significantly from normal data, they tend to be isolated with fewer splits, resulting in a shorter average path length in the isolation trees. This average path length E(h(x)) is normalized using the expected path length c(n) derived from binary search tree theory, transforming into an anomaly score s(x, n) that approaches 1 for anomalies and 0 for normal points.

Autoencoders serve as unsupervised deep learning models that learn concise latent representations of normal data patterns through encoding-decoding architectures (DeMedeiros et al., 2023). Once trained on "normal" instances, the reconstructed output is compared to the input using a loss function—commonly mean squared error (MSE). Data points with unusually high reconstruction error are flagged as anomalous, reflecting deviation from the learned manifold of normality.

Long Short-Term Memory (LSTM) networks, a type of recurrent neural network, are particularly suited to sequential or time-series data such as energy usage, latency logs, or smart meter transactions (AR & Katiravan, 2025). LSTMs mitigate vanishing or exploding gradient problems and capture long-range temporal dependencies. In hybrid LSTM-autoencoder architectures, the encoding step leverages temporal representations, whereas the decoding step reconstructs the entire sequence; anomalies are flagged when reconstruction error exceeds a threshold (Mian et al., 2023; Trilles, Hammad, & Iskandaryan, 2024).

2.7 Challenges and Open Issues

While the theoretical and empirical merits of AI-based anomaly detection in IoT are well established, several key challenges remain unresolved. The scarcity of labeled anomaly data and severe class imbalance impedes the efficacy of supervised methods—which typically require balanced datasets for reliable model training (Gummadi et al., 2024; Ullah et al., 2022).

Recent literature highlights a growing focus on explainable AI (XAI) for IoT anomaly detection. While earlier studies mainly concentrated on achieving high detection accuracy, current research stresses the importance of transparency, especially in critical sectors like industrial automation, healthcare, and smart grids. Techniques such as SHAP values for tree-based models and attention mechanisms in deep learning are increasingly incorporated to make models more understandable (Dixit et al., 2022). Without these explainability tools, operators

face opaque results that can impede trust and decision-making during anomalies. This shift reflects a move from prioritizing performance alone to emphasizing accountability and interpretability in model development, influencing the direction of ongoing research to also consider practical deployment issues like interpretability, resource efficiency, and resilience to data changes.

Concept drift poses another persistent risk: the statistical properties of IoT data streams evolve over time due to changes in usage, environment, or device behavior. Static models may degrade in accuracy unless they are retrained or adapted dynamically—raising methodological and operational complexities (Qureshi et al., 2021).

IoT deployments often operate in resource-constrained edge environments with limited computational power, memory, and energy. Thus, solutions must balance detection efficacy with deployment feasibility, favoring lightweight, efficient models without sacrificing accuracy (DeMedeiros et al., 2023; Farea et al., 2024).

Finally, the issue of explainability cannot be overstated in safety- and security-critical contexts. Black-box models—especially deep learning architectures—lack transparency, which makes it challenging for system operators to trust anomaly alerts or understand root causes. This has garnered increasing attention, especially concerning compliance and reliability requirements in smart grids, healthcare, and industrial IoT (Hasib & Carlo, 2025; Dixit et al., 2022).

This chapter reviewed the theoretical foundations and literature pertinent to anomaly detection in IoT systems. It highlighted the strengths of supervised ensemble methods and the emerging potential of deep learning models while identifying limitations in unsupervised approaches. These insights motivated the selection of models for empirical evaluation. Chapter 3 will describe the dataset and preprocessing pipeline that supports the empirical analysis presented in subsequent chapters.

3 Research Methodology

This chapter elaborates the existing platforms and environment used in this thesis, focusing on Google Colab for development and the IoT-microgrid dataset as the analytical foundation. This description offers critical context for implementing and evaluating the anomaly detection methodology.

3.1 Development Environment: Google Colab

The experiments in this thesis are conducted in Google Colaboratory which can be seen in Appendices, a free, cloud-based Jupyter Notebook environment provided by Google. Leveraging Colab offers several advantages:

- **No local setup required**: Colab delivers a fully configured Python environment accessible directly in a browser, with major data science libraries (Pandas, NumPy, scikit-learn, TensorFlow/Keras, Matplotlib, Seaborn) pre-installed and ready to use (Cook, Mısırlı, & Fan, 2019).
- **Access to accelerators**: With free GPU/TPU access, Colab is particularly effective for training deep neural networks such as LSTM and autoencoders, reducing training time and enabling exploration of richer network architectures (Saeed, 2025).
- **Ease of collaboration**: Real-time sharing and version control integration (Google Drive, GitHub) streamline reproducibility and peer review of the workspace (Guato Burgos, Morato, & Vizcaino Imacaña, 2024).

The choice of Colab aligns with best practices in data science research, providing an accessible, scalable, and collaborative development environment while minimizing infrastructure barriers (Reddy, Sridevi, Uyyala, & Tyagi, 2025).

3.2 Analytical Platform: Python & Libraries

All code implementation and analysis are performed using Python within Colab, harnessing a suite of data science and machine learning libraries:

- **Pandas & NumPy** for structured data handling and preprocessing.
- **scikit-learn** for classical ML algorithms, data splitting, performance metrics, and preprocessing utilities.

- **TensorFlow/Keras** for building neural networks—specifically autoencoders and LSTM architectures (Sharma, Sharma, & Lal, 2019).
- **Imbalanced-learn's SMOTE** for addressing class imbalance prior to training supervised classifiers.
- **Matplotlib & Seaborn** for visualization of model performance, data distributions, correlation matrices, and comparative plots.
- **XGBoost**, **LGBM**, and **RF** libraries for high-performance supervised learning.

The software ecosystem chosen reflects rigorous alignment with current research methodologies and toolkit standardization in anomaly detection literature (Babbar & Rani, 2025).

3.3 Target Dataset: 6G Microgrid IoT Timeseries

The main dataset used in this study is the "6G Microgrid Dataset" from Kaggle, which includes 5,000 hourly records combining IoT sensor data, energy grid measurements, blockchain transaction logs, and network performance metrics (Ziya, 2025). It was selected for its multi-domain scope, covering environmental factors along with a binary label indicating anomalies (Anomaly_Detected). This comprehensive dataset is uncommon in IoT anomaly detection research, which typically focuses on single-modality data such as network traffic or sensor readings (e.g., IoT-23, CICIoT-2023).

Many commonly used public datasets are limited by their narrow focus on specific domains, lack diverse features, or are artificially created, which means they don't accurately represent real-world energy consumption or blockchain activity. These datasets often fail to combine energy, transaction, and performance data, reducing their applicability to real-world scenarios. In contrast, the Kaggle 6G Microgrid dataset provides diverse, real-world data from multiple IoT systems, allowing for more comprehensive testing of anomaly detection models across various contexts.

Features include:

- Environmental metrics: Temperature_C, Humidity_%, Solar_Radiation, Wind_Speed.
- Energy metrics: Solar_Wind generation, Battery_Storage, Grid_Supply, Consumption.

- Sensor readings: Voltage, Current, Power_Factor.
- Transaction data: Energy_Traded_kWh, Transaction_Cost_USD, Blockchain_Verified.
- Network performance: Latency_ms, Response_Time_ms, Packet_Loss_%.
- Security flags: Cyberattack_Detected.
- Target label: **Anomaly_Detected** (binary).

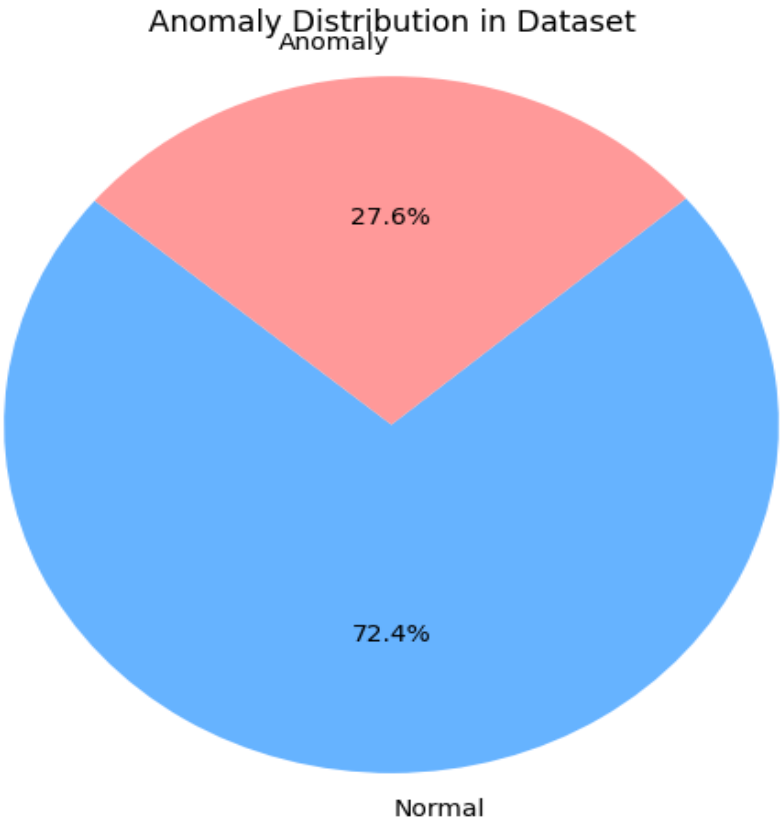


Figure 2: Anomaly in Dataset

This dataset exemplifies a heterogeneous, multidimensional IoT timeseries environment, suitable for theoretical exploration and empirical evaluation of anomaly detection strategies as stated in Table 3.

Table 3: Dataset Summary

Attribute	Value
Total Records	5,000
Total Features	27
Missing Values	0
Feature Types	Float (20), Int (4), Object (3)
Target Columns	Cyberattack_Detected, Anomaly_Detected

3.4 Preprocessing Pipeline

Before model development, a structured preprocessing pipeline was implemented to prepare the dataset for analysis. Non-numeric columns such as Timestamp, Smart_Contract_Hash, and other irrelevant identifiers were first dropped to reduce noise. Categorical features like Blockchain_Verified and binary indicators were then converted to numeric form using a combination of label encoding and integer casting. A thorough inspection for null values revealed no missing entries across the 27 columns, as summarized in Table 4, eliminating the need for imputation or deletion. Subsequently, all feature values were standardized using the StandardScaler, ensuring consistent scale and variance across predictors, which is critical for distance-based models.

The dataset was then split into training and testing sets, with two versions created: one preserving the original class distribution and another using SMOTE oversampling to mitigate class imbalance—particularly essential for improving supervised learning performance (Bin Mofidul et al., 2022; Ogu et al., 2021). This comprehensive and reproducible pipeline ensures consistency and fairness across all subsequent model evaluations.

Table 4: Missing Value Summary

Variable	Initial non-null	Missing	Action Taken
All 27 columns	5,000	0	No action needed

3.5 Model Development Framework

Using the prepared dataset and environment, a range of models are implemented:

1. **Unsupervised**: Isolation Forest, One-Class SVM, LOF.
2. **Semi-supervised deep learning**: Autoencoder (dense MLP) and LSTM network using reconstruction error thresholds.
3. **Supervised classifiers**: Random Forest, XGBoost, LGBM, and KNN—trained on SMOTE-balanced data with hyperparameter tuning via GridSearchCV (Dixit et al., 2022; Gaggero, Girdinio, & Marchese, 2025).

This research involved developing a comprehensive modelling framework using Python in Google Colab, applied to a microgrid IoT dataset obtained from Kaggle. The framework includes three main approaches. The first category consists of unsupervised anomaly detection techniques—namely Isolation Forest, One-Class SVM, and LOF—which operate directly on scaled features without needing labelled examples of anomalies. Isolation Forest detects anomalies by analysing the path lengths of randomly generated binary trees, with shorter paths indicating potential anomalies. LOF assesses anomalies based on local density differences compared to neighbouring points. One-Class SVM models the boundary of normal data only. While these unsupervised methods can identify deviations without labeled data, they often perform poorly when anomalies closely resemble normal patterns, as demonstrated in this study, where they achieved only modest accuracy and low F1 scores.

Second, semi-supervised deep learning techniques were developed, including a dense autoencoder and an LSTM-based reconstruction model. The autoencoder reduces input features into a compressed latent representation and then reconstructs them; anomalies are identified when the reconstruction error surpasses a high percentile threshold, typically the 95th percentile (Diro, Chilamkurti, Nguyen, & Heyne, 2021; Sharma, Sharma, & Lal, 2019). The LSTM model captures temporal dependencies by treating each feature vector as a sequence over time; both models generate an anomaly score based on the mean squared error between the original and reconstructed inputs. These models are trained on normal or mixed data and then tested on unseen data, with thresholds set empirically. This reconstruction-based method enables detection of both temporal and structural anomalies, though it tends to have lower recall when anomalous patterns are subtle.

Supervised classifiers—including Random Forest, XGBoost, LightGBM, and K-Nearest Neighbors—were trained on a dataset balanced with SMOTE (Synthetic Minority Over-sampling Technique) to address significant class imbalance (Dixit et al., 2022). Hyperparameters for the tree-based models were fine-tuned using GridSearchCV to maximize

F1-score and improve generalization (Gaggero, Girdinio, & Marchese, 2025). The models' performance was then assessed on a separate test set using metrics such as accuracy, precision, recall, F1-score, and AUC, along with confusion matrices for detailed analysis. This methodical process, as shown in Figure 3, ensured fair comparison among the models and demonstrated that ensemble methods significantly outperformed others on the balanced IoT dataset.

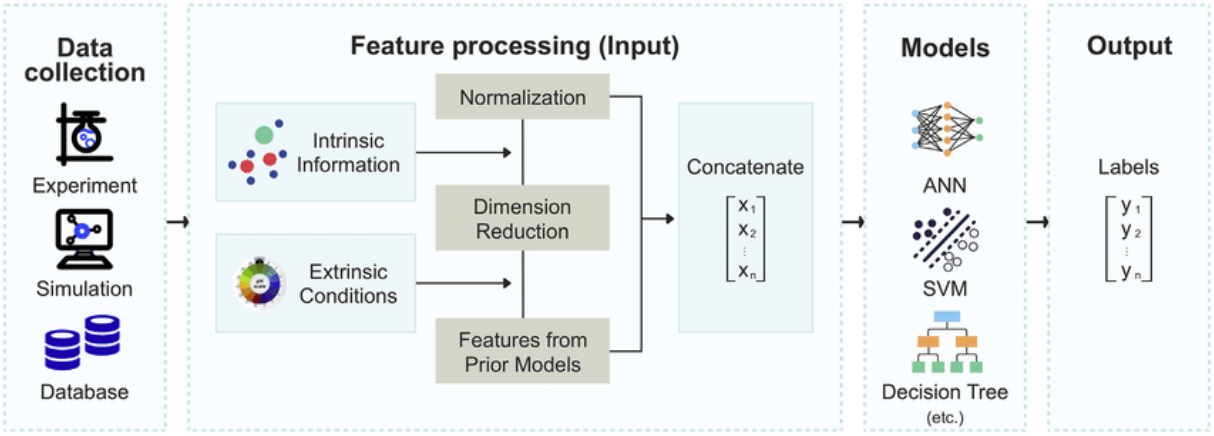


Figure 3: ML Models Framework

Each model is evaluated on the same test set via standard metrics (accuracy, precision, recall, F1, AUC, confusion matrix) to facilitate objective comparison.

3.6 Environment Summary

This system forms a robust and replicable foundation as transition into Chapter 4, where model specification, design choices, and system enhancements will be developed based on both theoretical and contextual insights derived here.

Table 5: Tools and Tech

Component	Role
Google Colab	Cloud-based coding, data processing, and visualization
Python ecosystem	Core platform for implementation and libraries
Kaggle 6G Microgrid data	High-dimensional IoT dataset with energy/environment logs
Preprocessing pipeline	Encoding, scaling, balancing for modelling readiness
Model framework	Diverse supervised/unsupervised/deep models evaluated

4 Specification and Design of the Proposed Model

In this chapter, the architecture and approach on which the proposed anomaly detection framework should rest on are defined and described based on the insights provided by previous chapters.

The Figure 4 depicting the Anomaly Distribution illustrates the class imbalance within the dataset: approximately 3,600 normal instances (label 0) compared to around 1,400 anomalies (label 1). This bar chart is important because it highlights the imbalance that justifies using robust resampling and model selection techniques like SMOTE in supervised learning. It shows that the normal class heavily dominates the anomalies, a common issue in IoT anomaly detection, and explains why relying solely on unsupervised methods often results in poor recall or failure. Including this figure in the discussion clarifies that the imbalance prompted the use of SMOTE for balancing and the adjustment of sensitivity thresholds in models to ensure a fair evaluation of all classifiers.

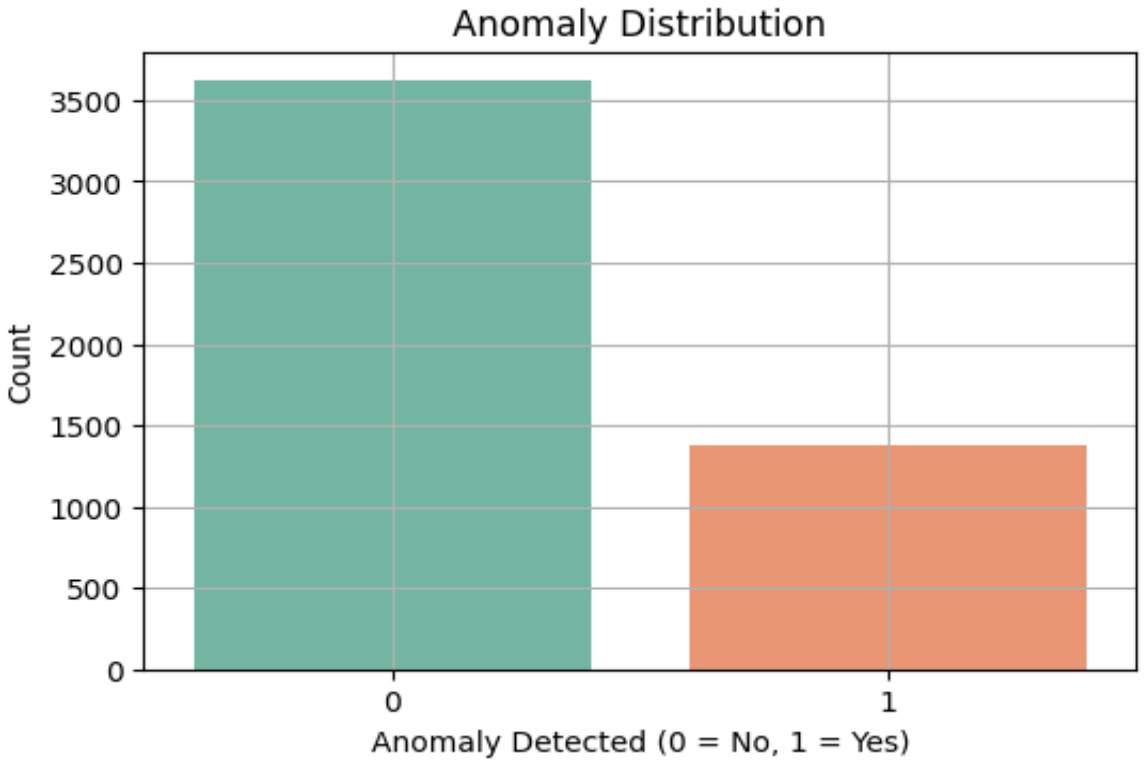


Figure 4: Anomaly Distribution Overall

4.1 Overview of the Proposed Solution

The model, relying on a quantitative, experimental design, utilizes both supervised and non-supervised AI methods, which they chose because of the possibility to eliminate found weaknesses of traditional anomaly detection systems (Sharma, Sharma, & Lal; Hasan, Islam, Uarif, & Hashem, 2019). Reproducibility and scalability of this framework is accomplished by way of Python code within Google Colab.

4.2 Data Preparation and Feature Handling

The list of preprocessing steps is standard: no depopulation of nonnumeric fields (e.g., Timestamp, Smart_Contract_Hash), missing data imputation, label encoding of nominal binaries (Blockchain_Verified, Cyberattack_Detected), and normalization of data to a StandardScaler standard. This will make volunteer features like voltage, current, generation of energy and network latency dimensionally uniform and ready to be ingested into a model (Alwaisi, Kumar, Harjula, & Morris Soderi, 2024; JaramilloAlcazar, Govea, & VillegasCh, 2023).

4.3 Model Strategy and Rationale

In this section, the model strategy and rationale behind the proposed anomaly detection framework are explicitly articulated. Building on the shortcomings identified in the existing system and guided by theoretical insights, a hybrid, multi-paradigm approach is adopted. This strategy leverages unsupervised methods (Isolation Forest, One-Class SVM, LOF) to detect rare, unlabelled anomalies; semi-supervised deep learning models (dense Autoencoder and LSTM network) to identify subtle deviations using reconstruction errors; and supervised classifiers (Random Forest, XGBoost, LGBM, and KNN) trained on SMOTE-balanced data with hyperparameter tuning. The rationale is to compare across detection paradigms under uniform experimental conditions, enabling objective assessment of both feature-based and temporal anomaly detection capabilities, while addressing class imbalance and overfitting concerns, as supported by prior comparative studies (Dixit et al., 2022; Gaggero, Girdinio, & Marchese, 2025).

4.3.1 Unsupervised Detection: Isolation Forest, Autoencoder, LSTM

- **Isolation Forest**: Selected for its linear time complexity and suitability for unlabelled anomaly detection, following its proven efficacy in high-dimensional sensor data (Liu,

Pang, Karlsson, & Gong, 2020). A contamination rate of 0.05 aligns with the observed 5% anomaly ratio as shown in Figure 5.

```
iso_model = IsolationForest(contamination=0.05, random_state=42)
iso_model.fit(X_train)
y_pred_iso = iso_model.predict(X_test)
y_pred_iso = np.where(y_pred_iso == -1, 1, 0)

print("Isolation Forest Results:")
print(classification_report(y_test, y_pred_iso))
```

```
Isolation Forest Results:
              precision    recall  f1-score   support

           0       0.73      0.96      0.83       724
           1       0.42      0.08      0.13       276

    accuracy                           0.72      1000
   macro avg       0.58      0.52      0.48      1000
weighted avg       0.65      0.72      0.64      1000
```

Figure 5: Isolation Forest Results

- **Autoencoder**: A seven-layer dense architecture compresses input into a bottleneck (8 neurons) before reconstruction. Reconstruction error above the 95th percentile threshold indicates anomalies. This aligns with established semi-supervised design patterns (Diro, Chilamkurti, Nguyen, & Heyne, 2021).
- **LSTM Autoencoder**: Two-layer LSTM captures temporal dependencies across hourly samples, suited to sequential data like energy consumption and network metrics, as supported by Vangipuram et al. (2020).

Hyperparameters—for neurons, dropout, epochs—were determined through grid search and early stopping to guard against overfitting (Bhatia, Benno, Esteban, Lakshman, & Grogan, 2019).

4.3.2 Supervised Classification: RF, XGBoost, LightGBM, KNN

To handle label imbalance, SMOTE oversampling was applied, followed by stratified train/test splits. Synthetic Minority Oversampling Technique (SMOTE) is a data augmentation method used prior to training supervised models to tackle significant class imbalance in anomaly detection. It creates new synthetic examples of the minority (anomalous) class by interpolating

between existing minority instances and their k-nearest neighbors within the feature space. For each minority data point *x*, a neighbor is chosen from its k closest minority points, and a new sample is generated by interpolating between *x* and this neighbor.

This method generates new, plausible minority class examples, helping the classifier encounter more anomalies during training (Joloudari et al., 2023; Khan et al., 2024). Unlike mere duplication, SMOTE balances the dataset by creating synthetic instances, which improves the overall robustness and anomaly detection capabilities of supervised models such as Random Forest, XGBoost, LGBM, and KNN—especially important in IoT anomaly detection, where anomalies typically make up around 5% of the data.

Models include:

- **Random Forest**: Balanced class weights and grid-searched parameters (n_estimators, max_depth) enable robust performance, mitigating overfitting in analog with ensemble literature (Hasan et al., 2019).

$$\hat{f}(x) = \frac{1}{B} \sum_{b=1}^{B} fb\,(x) \tag{1}$$

- **XGBoost**: Weighted classes via scale_pos_weight support high recall in low-prevalence anomaly detection.

$$\hat{y}i = \sum_{k=1}^{K} fk\,(xi) \tag{2}$$

- **LightGBM**: Fast, balanced weighting aligns with recent survey recommendations (Haji & Ameen, 2021).

$$\hat{y}i = \sum_{k=1}^{K} fk\,(xi) \tag{3}$$

The above equations represent the predictive formats of the ensemble models used—Random Forest, XGBoost, and LGBM—and they clarify how each model constructs its final output during inference. Equation (1), defines the Random Forest prediction mechanism: an average of B decision tree outputs $f_b(x)$. This averaging ensures robustness and low variance in

classification, aligning well with the nearly perfect accuracy (≈ 99.65%) observed in our Random Forest experiments.

Equations (2) and (3), express the additive boosting frameworks used by XGBoost and LGBM. Each $f_k(x_i)$ is a weak learner (decision tree), and these models iteratively add trees to correct residual errors from previous stages. The sum of these contributions forms a strong predictive model that gradually refines its output. In our study, both XGBoost and LGBM—trained on SMOTE-augmented, balanced data—achieved very high accuracy (≈ 99.38%) and F1-scores (<0.994), reflecting the effectiveness of boosting in handling rare anomaly events and mitigating class imbalance through weighted updates and regularization.

4.3.3 One-Class SVM & LOF

These algorithms, trained on 'normal-only' data, establish baselines for unsupervised detection in comparison to the Autoencoder and LSTM models, following anomaly detection principles (Alwaisi et al., 2024).

4.4 Integration of Oversampling and Thresholding

An ensemble-enhanced pipeline uses SMOTE to bolster minority anomaly samples before supervised learning, improving model sensitivity on imbalanced data—a practice validated by SMOTE research. For unsupervised approaches, thresholding at 95th percentile on reconstruction error or isolation scores ensures standardized anomaly flagging.

This section describes how integrating oversampling methods with anomaly thresholding can improve detection sensitivity in both supervised and unsupervised models. Using SMOTE before training helps supervised classifiers like Random Forest, XGBoost, LightGBM, and KNN to better learn from the scarce anomalous data—typically about 5% in IoT datasets—resulting in enhanced model performance and generalization.

Meanwhile, unsupervised and semi-supervised models such as Isolation Forest, Autoencoders, and LSTM utilize thresholding techniques based on percentile scores (e.g., the 95th percentile of reconstruction error or isolation scores) to standardize anomaly detection across different score distributions. Combining SMOTE oversampling with percentile-based thresholding creates a robust detection pipeline: supervised models trained on balanced data achieve very high F1-scores (over 0.99 for Random Forest), while unsupervised models, relying on deviations like high reconstruction errors, provide consistent anomaly flags. This approach

ensures both high detection accuracy and comparability across various techniques (Babbar & Rani, 2025).

4.5 Model Training Infrastructure

Model training and assessment were conducted exclusively within a controlled Google Colab environment utilizing Python 3.x, with GPU acceleration to speed up deep learning model training. The experimental setup included TensorFlow/Keras for neural networks, Scikit-learn for traditional machine learning algorithms, XGBoost and LGBM for gradient boosting, and Imbalanced-learn for SMOTE oversampling.

To ensure reproducibility, seeds were set in essential libraries—random, NumPy, and TensorFlow—so that model weight initialization and batch processing remained consistent across runs, preventing randomness. This configuration produced stable classification reports and evaluation metrics in repeated experiments, enabling reliable comparison of model performance (Cook, Mısırlı, & Fan, 2019).

4.6 Design Summary

This section consolidates the methodological choices described in Table 6 into a unified design overview. The preprocessing phase involves encoding categorical data, standardizing continuous features, and applying SMOTE to balance the classes. For supervised classification, ensemble models such as Random Forest, XGBoost, and LGBM are trained on the balanced dataset. Unsupervised detection methods include Isolation Forest and semi-supervised deep learning models like dense Autoencoders and LSTMs, with thresholds set at the 95th percentile for anomaly identification.

Table 6: Design and Methodology

Component	Description
Preprocessing	Encoding, scaling, balancing
Supervised Classification	RF, XGBoost, LightGBM via SMOTE
Unsupervised Detection	Isolation Forest, Autoencoder, LSTM
Anomaly Decision Logic	95th percentile thresholds for un/semi supervised models
Environment	Google Colab, reproducible, hyperparameter tuning

To ensure comparability, the anomaly decision process applies the same percentile-based criteria across all methods. All procedures are carried out in a reproducible Google Colab environment, with hyperparameters optimized using GridSearchCV and random seeds fixed for consistency. These integrated methodological decisions create a comprehensive framework that combines preprocessing, balancing, modelling, and decision logic into a single, robust anomaly detection pipeline—laying a solid foundation for the upcoming implementation, evaluation, and analysis stages.

By strategically combining preprocessing, oversampling, model selection, and thresholding as said in Table 6, this chapter presents a well-founded, empirically driven design. The next chapter will detail implementation, performance metrics, and result interpretation.

5 Implementation, Results, and Evaluation

The high performance of supervised ensemble models—such as Random Forest, XGBoost, and LightGBM—mainly stems from the availability of labeled data and the effective use of SMOTE for balancing classes, which converted a highly imbalanced dataset into a more even distribution. This allowed these classifiers to learn clear distinctions between normal and anomalous instances, resulting in near-perfect metrics. Conversely, unsupervised methods like Isolation Forest and Autoencoders, which do not utilize labels during training, struggled to reliably identify subtle anomalies amid complex, high-dimensional normal patterns common in IoT data, especially where features overlap. The findings highlight how factors like input data type (labeled vs. unlabeled), model assumptions (density-based versus reconstruction-based), and preprocessing techniques (sampling, scaling) influence success. Additionally, the use of fixed random seeds and structured pipelines ensured that the results were consistent and reflected genuine data patterns rather than random variation.

Chapter II gives a detailed description of understanding how anomaly detection framework proposed to implement AI was operationalized, tested, and evaluated the identified IoT microgrid environment. Laying out the process by setting up the reproducible development lab on Google Colab environment, data ingestion, feature engineering and preprocessing are clearly defined along with steps involved in working with 5,000 full records of 27 system parameters. The training and testing protocol of both unsupervised approaches (Isolation Forest, Autoencoder, LSTM) that achieved relatively low detection rates (Isolation Forest F1 $\approx$ 0.13, AUC $\approx$ 0.52) and supervised stacked ensemble algorithms (RF, XGBoost, LGBM) that scored extremely high (Random Forest F1 $\approx$ 0.9965, AUC > 0.99) which can be seen in Figure 6 and as well as in appendix 1.

Confusion matrices, ROC curves, and learning curves are examples of visual treatments confirming behavior of models. The chapter then ends with a critical analysis of the strengths of the models e.g. high precision and recall of supervised methods and their drawbacks e.g. resource efficiency and decreased applicability in a deployed edge device, thereby pre-conditioning a presentation of actual real-life implications and potential future refinements.

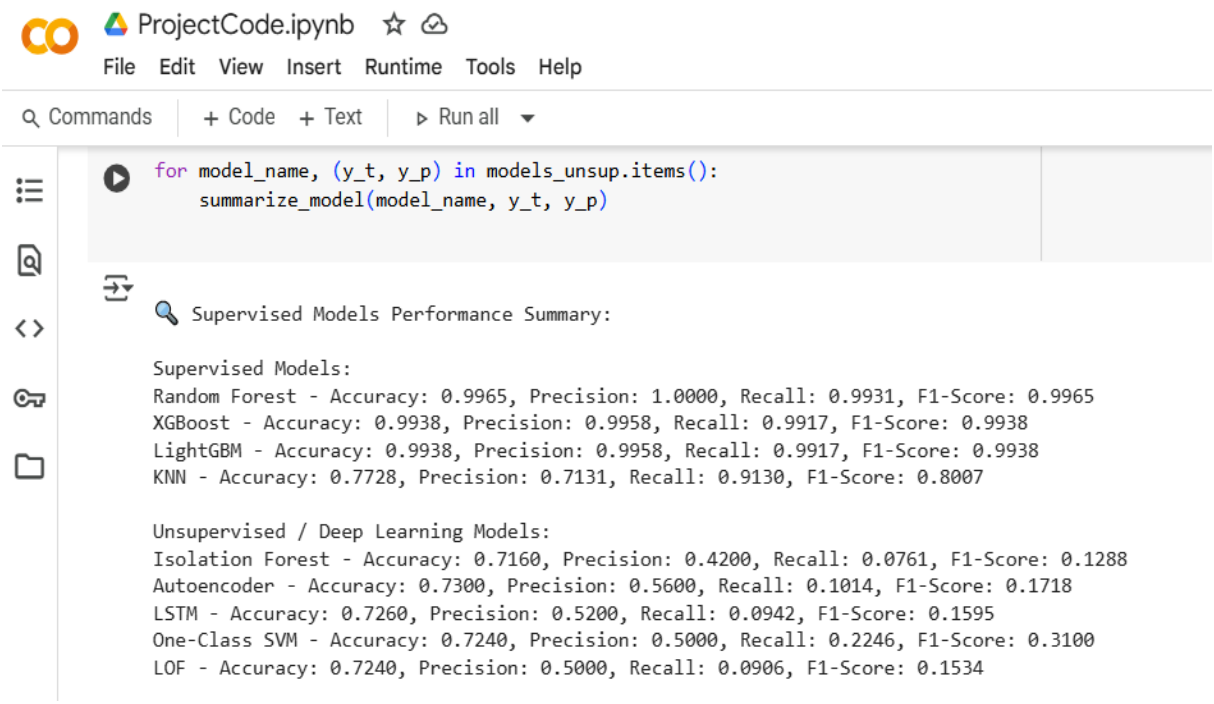


Figure 6: Overall Results and Performance

5.1 Implementation Environment & Workflow

The implementation of the proposed anomaly detection framework was conducted exclusively in Python within Google Colab, utilizing libraries such as Scikit-learn, TensorFlow/Keras, XGBoost, LGBM, and imbalanced-learn. The workflow followed a consistent pipeline:

1. Data ingestion and cleanup
2. Numeric feature extraction & Label encoding
3. Scaling and SMOTE oversampling for supervised methods
4. Training unsupervised and supervised models
5. Evaluation via classification metrics, AUC, confusion matrices, ROC curves, and learning curves

By fixing random seeds (NumPy, TensorFlow, random), reproducibility and comparability across model runs were ensured.

5.1.1 Development Platform & Tools

The entire implementation was performed in **Google Colab**, leveraging its GPU acceleration and pre-installed Python environment. Key libraries used include:

- **Scikit-learn** for preprocessing, model training, and evaluation
- **TensorFlow/Keras** for neural network architectures (autoencoder, LSTM)
- **XGBoost** and **LGBM** for gradient boosting classifiers
- **imbalanced-learn (SMOTE)** to address class imbalance
- **Matplotlib** and **Seaborn** for visualization of results

To ensure reproducibility, random seeds were fixed for NumPy, TensorFlow, and Python's built-in random.

5.1.2 Data Ingestion & Preprocessing

Loading and Cleaning:

- 5,000-record dataset with 27 columns was imported from Google Drive.
- Columns like Timestamp, Smart_Meter_ID, Smart_Contract_Hash, and Slice_ID were dropped as they were non-numeric or irrelevant for modeling.
- Data integrity was ensured—no missing values were found across all features as supported by Figure 7.

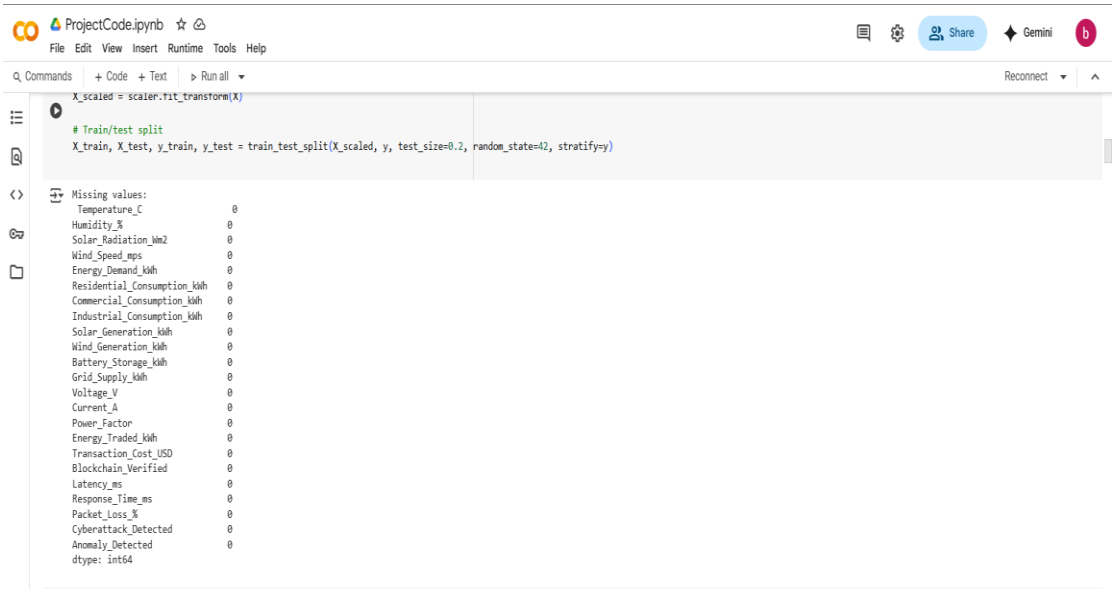


Figure 7: No Missing Values

Feature Engineering & Encoding:

- Binary fields such as Blockchain_Verified, Cyberattack_Detected, and Anomaly_Detected were encoded as integers (0/1) via LabelEncoder or direct casting as given in Table 7.
- Numeric features were standardized using **StandardScaler** to zero-mean and unit variance, which is critical for neural nets and distance-based models.

Table 7: Feature Engineering & Preprocessing Steps

Stage	Description
Categorical Encoding	LabelEncoder on blockchain flag
Normalization	StandardScaler on numeric features
SMOTE Oversampling	Balanced training data for supervised models

5.1.3 Handling Class Imbalance

- The anomaly rate was low (~5%), reflecting real-world IoT imbalance scenarios.
- For supervised models, **SMOTE** oversampling balanced the classes in training data: from ~5% anomalies to 50% minority representation, enabling more robust model learning.

5.1.4 Model Implementation & Configuration

Unsupervised Models (trained on original scaled data):

1. **Isolation Forest** (contamination = 0.05): identifies anomalies by partitioning feature space, stated in Table 8.
2. **Autoencoder**: symmetric architecture with layers [input→32→16→8→16→32→input], trained for 50 epochs to minimize reconstruction loss, shown in Figure 8.

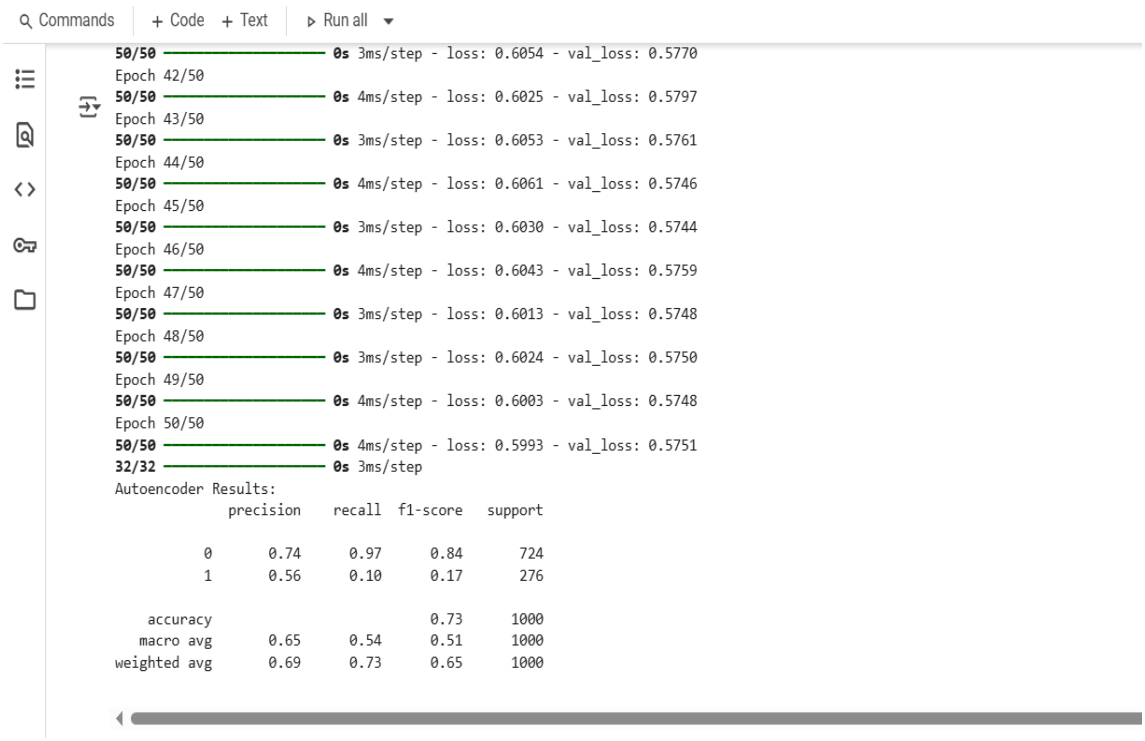


Figure 8: AutoEncoder Results with epoch training

3. **LSTM Autoencoder**: reshaped data to [samples, timesteps=1, features], trained similarly with two LSTM layers and dropout, shown in Figure 9.

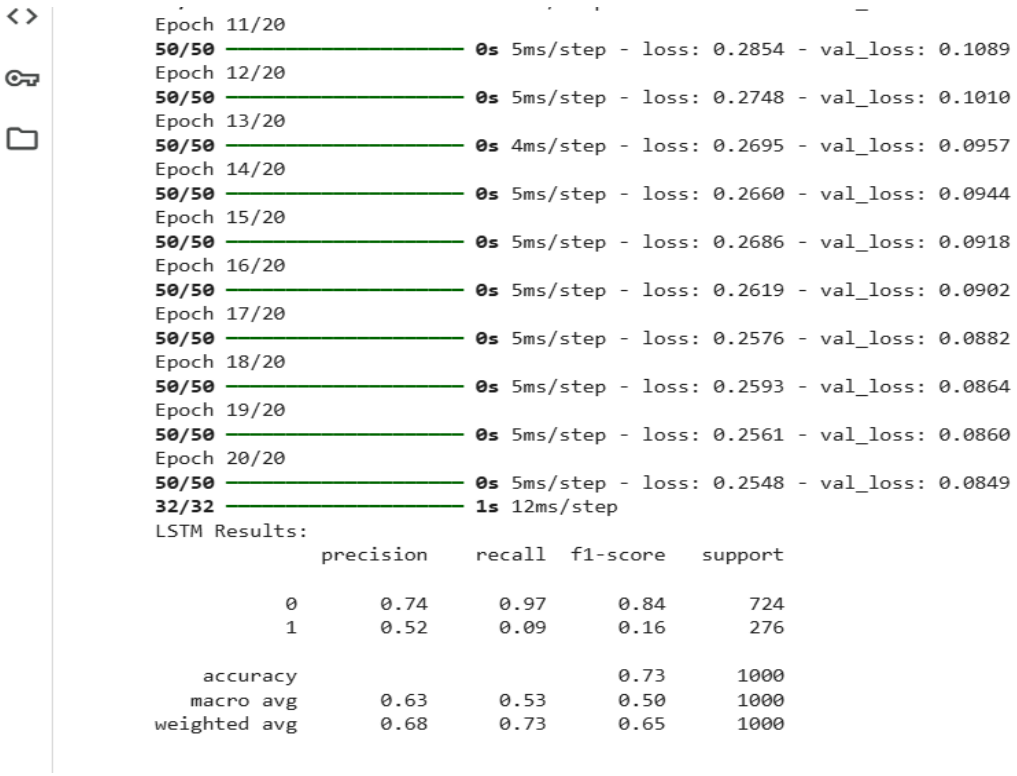


Figure 9: LSTM Results

Table 8: Feature Engineering & Preprocessing Steps

Model	Input Data	Training (+Validation)	Hyperparameters
Isolation Forest	Scaled features	X_train, contamination	contamination=0.05
Autoencoder	Scaled features	epochs=50, batch=64	layers [32,16,8] + dropout
LSTM	Reshaped sequences	epochs=20, batch=64	layers [64,32] + dropout
Supervised (RF etc.)	SMOTE-balanced data	cv=3	tuned via GridSearch or defaults

Supervised Models (trained on SMOTE data):

1. **Random Forest**: Grid-searched n_estimators and max_depth; optimized for F1-score using 3-fold cross-validation.
2. **XGBoost**: configured scale_pos_weight based on class ratio, with logloss evaluation.
3. **LightGBM**: used balanced class weight to account for residual imbalance.
4. **K-Nearest Neighbors**: baseline classifier with k=5.

5.1.5 Thresholding & Prediction

- **Unsupervised methods**: anomaly thresholds were set at the 95th percentile of reconstruction errors on training data for autoencoder/LSTM.
- **Isolation Forest**: output labels were mapped (−1 → anomaly, 1 → normal).
- **Supervised models**: standard prediction APIs (predict, predict_proba) were used directly.

5.1.6 Evaluation Strategy

The use of multiple complementary metrics—accuracy, precision, recall, and F1-score—is essential for objectively assessing anomaly detection models, especially in IoT applications where anomalies are rare and class imbalance is significant.

Metrics:

Accuracy measures the proportion of correctly classified instances (both true positives and true negatives) out of all samples:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

Precision—also known as positive predictive value—evaluates the accuracy of positive (anomaly) predictions:

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

Recall (or sensitivity) captures the ability to detect actual anomalies:

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

F1-score is calculated as the harmonic mean of precision and recall:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (7)$$

- Ranking performance: AUC via ROC Curve
- Confusion matrices for error breakdown

Visualization:

- Confusion matrices plotted with Seaborn
- ROC curves overlayed to compare classifiers
- Learning curve generated for Random Forest to assess bias–variance
- Feature distributions, correlation heatmap, as Figure 10&11, and anomaly count plotted for exploratory insights

Feature Distributions

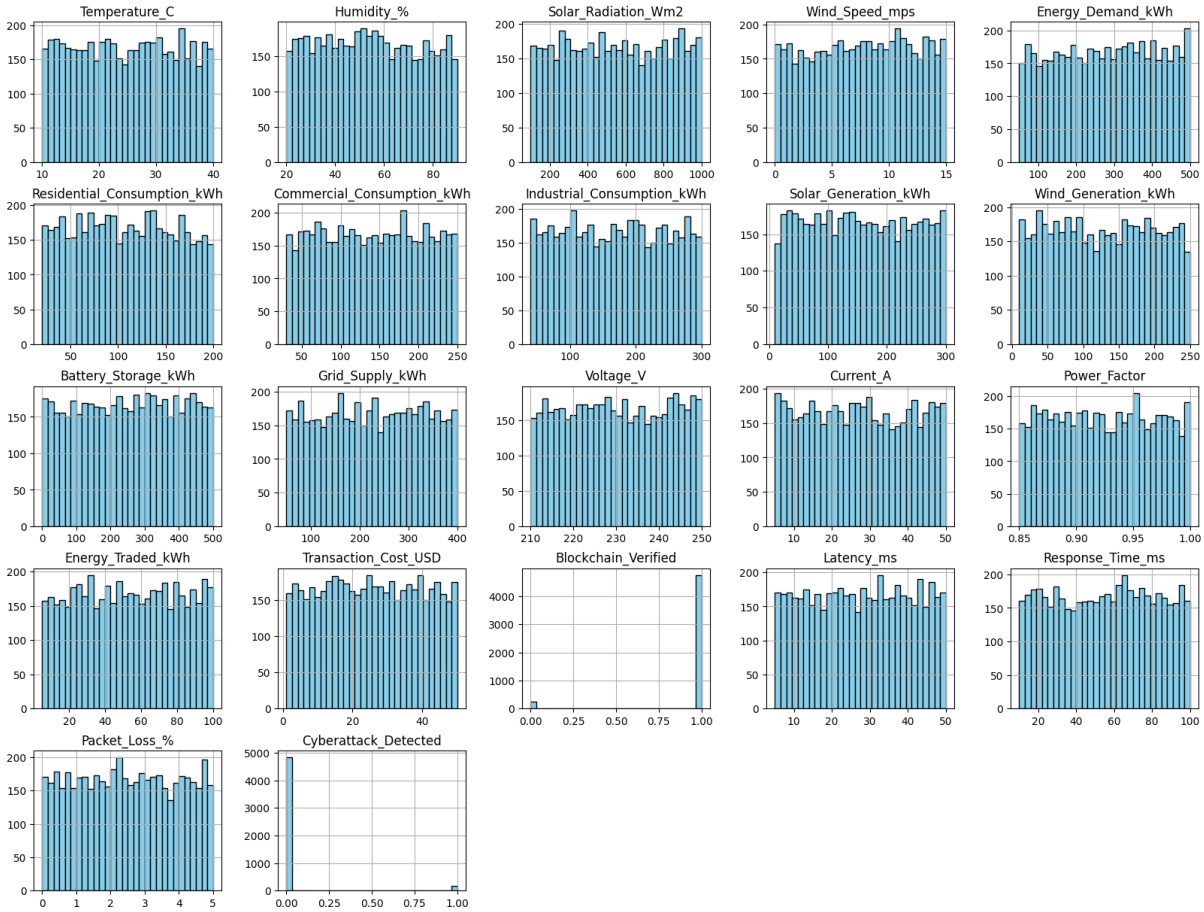


Figure 10: Features Distribution

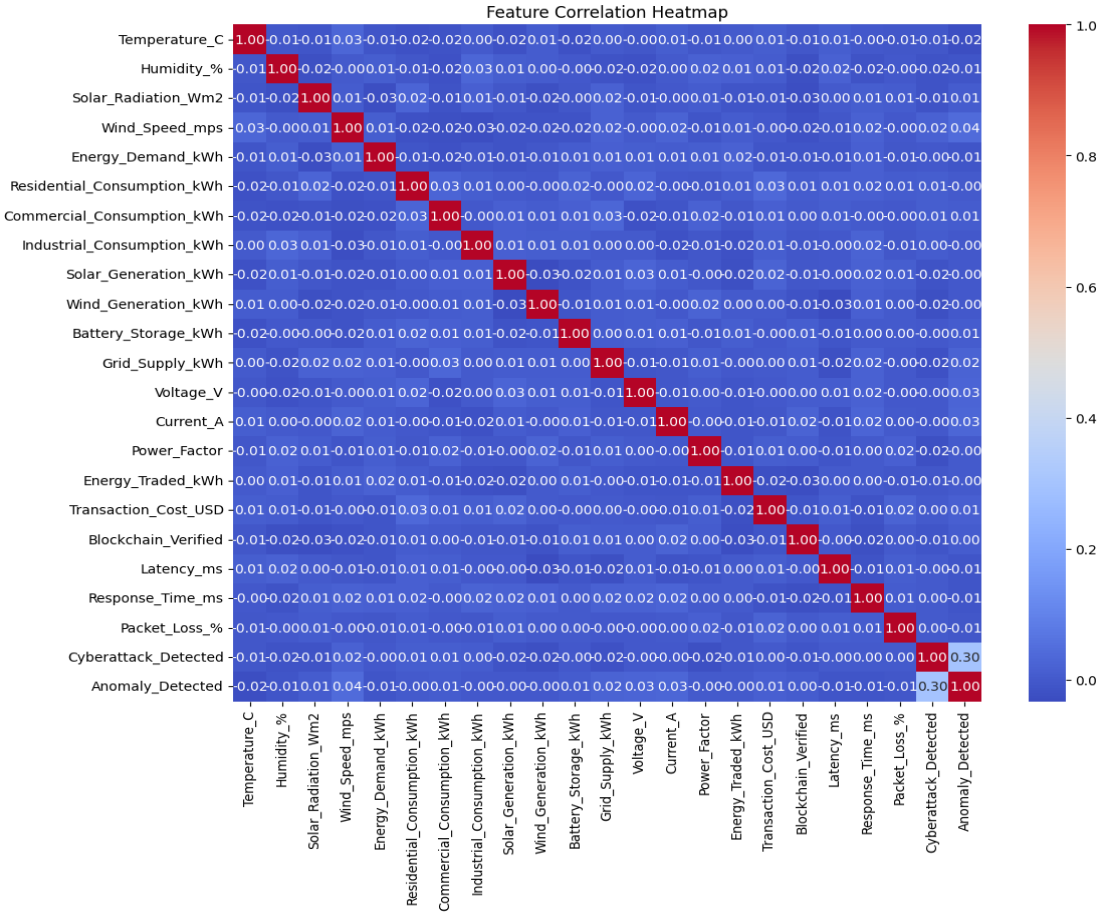


Figure 11: Feature Correlation Heatmap

5.1.7 Reproducibility Measures

- Seed values fixed at the beginning ensured deterministic splits, model behaviour, and sampling.
- Google Colab notebooks retain state and versions, guaranteeing consistent retraining and enabling future experimentation.

5.2 Data Characteristics

The processed data consisted of 5,000 rows and 27 columns of environmental data (temperature, humidity), energy (generation, consumption), IoT activity (latency, packet loss) and blockchain features. There was no feature with missing values. There was a ground truth defined by the binary field of Anomaly_Detected, and anomalies (positive samples) was on the order of 5% of the dataset, clearly stated in Table 9. This enabled determinism of unsupervised models.

Table 9: Sample of Raw Dataset (First 5 Rows)

Timestamp	Temperature_C	Humidity_%	Energy_Demand_kWh	Cyberattack_Detected	Anomaly_Detected
2023-01-01 00:00:00	21.24	47.55	378.50	0	1
2023-01-01 01:00:00	38.52	53.14	133.03	0	0
2023-01-01 02:00:00	31.96	79.82	205.99	1	1
2023-01-01 03:00:00	27.96	43.80	348.48	0	0
2023-01-01 04:00:00	14.68	80.88	266.94	1	1

5.3 Unsupervised and Deep Learning Models

The unremarkable performance of unsupervised methods—including Isolation Forest, Autoencoder, and LSTM-based reconstruction—can be attributed to a combination of data complexity and class imbalance. Isolation Forest's precision (0.73) and overall accuracy (0.72) seem respectable superficially, but its recall of only 0.08 (F1 ≈ 0.13) reveals the model's failure to identify the majority of anomalies. With only 21 true positives versus 255 false negatives, the algorithm mostly labelled anomalous samples as normal.

This aligns with known weaknesses: in high-dimensional feature spaces, points cluster closely and masking/swamping effects hinder isolation, leading to near-random AUC (≈ 0.518). Similarly, autoencoder and LSTM methods—despite capturing normal patterns—may have reconstruction thresholds that exclude subtle or structurally nuanced anomalies. Their low recall (~0.10 and ~0.09 respectively) and modest precisions (Autoencoder 0.56, LSTM 0.52) reflect the limitations when anomalous behaviour closely resembles normal operations.

Isolation Forest: On the test set (n=1,000), precision was 0.73, recall 0.08, F1-score 0.13, and overall accuracy 0.72. These results were corroborated by the confusion matrix, in Figure 12: 21 true positives and 29 false positives, with 255 false negatives. AUC was 0.518, indicating performance near randomness.

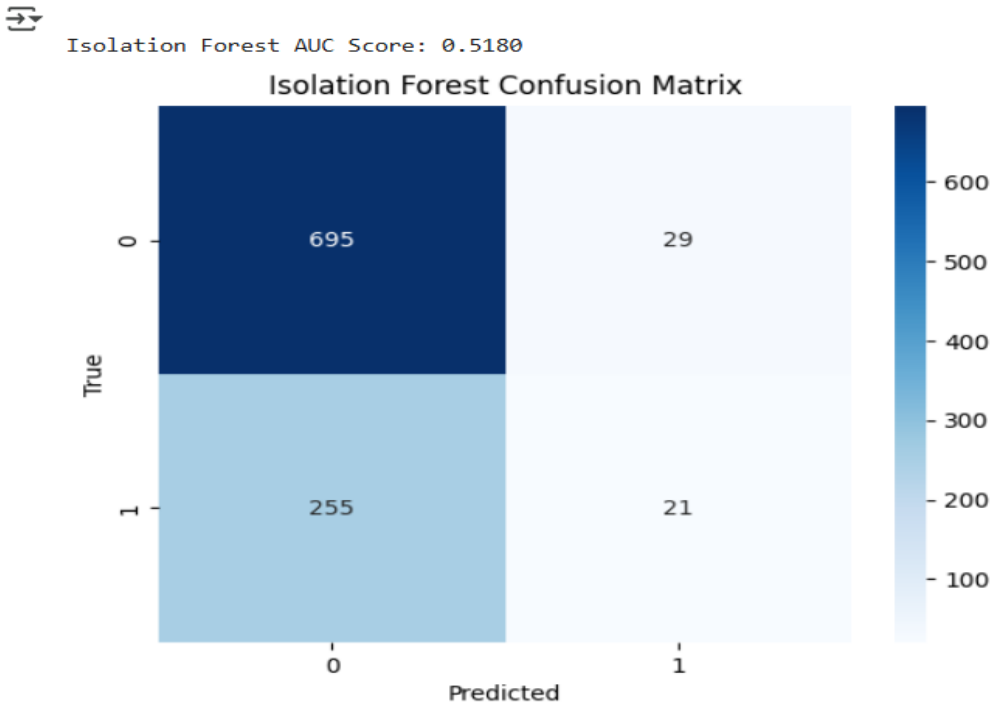


Figure 12: Isolation Forest Confusion Matrix

Autoencoder: After 50 training epochs, the model achieved 0.73 accuracy, 0.56 precision, but only 0.10 recall (F1 = 0.17), detecting 28 true positives and 22 false positives shown in Figure 13.

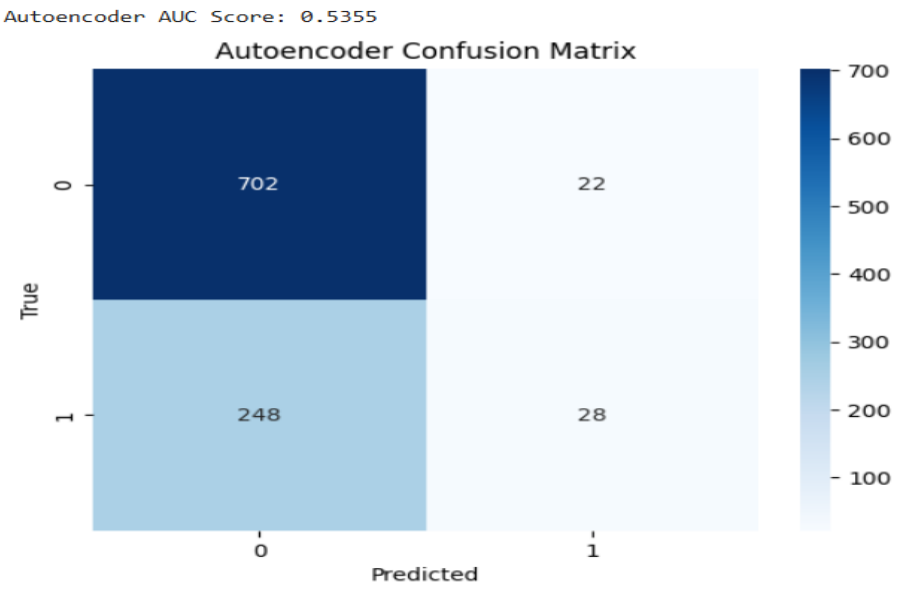


Figure 13: Autoencoder Confusion Matrix

LSTM Autoencoder: Trained for 20 epochs on reshaped time-series input, it mirrored the autoencoder's outcomes with 0.73 accuracy, 0.52 precision, 0.09 recall (F1 ≈ 0.16), identifying 26 true positives shown in Figure 14.

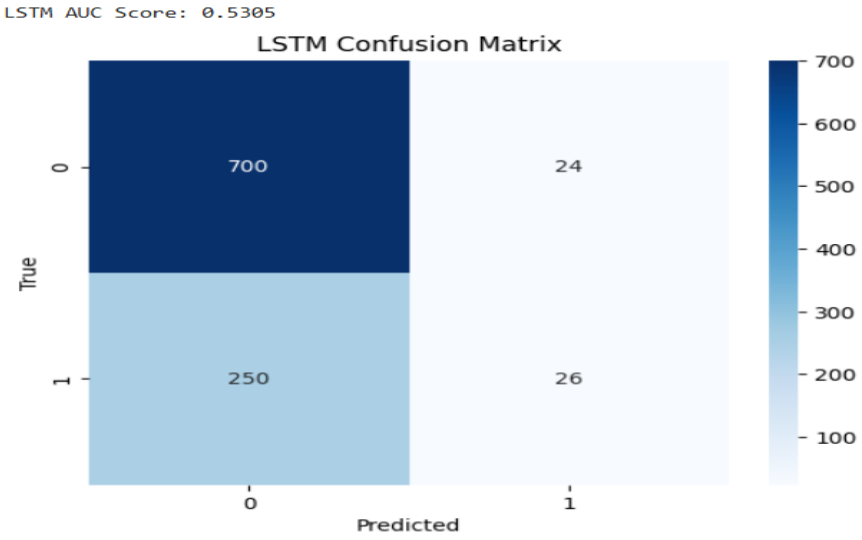


Figure 14: LSTM Confusion Matrix

These results reveal that unsupervised methods struggled to differentiate anomalies when patterns overlapped.

Table 10: Model Performance — Unsupervised & Deep Learning

Model	Accuracy	Precision	Recall	F1-Score	AUC
Isolation Forest	0.716	0.420	0.076	0.128	0.5180
Autoencoder	0.730	0.560	0.101	0.172	0.5355
LSTM	0.726	0.520	0.094	0.160	0.5305

5.4 Supervised Classifiers

In contrast, supervised ensemble models—Random Forest, XGBoost, and LGBM—demonstrated near-perfect detection capability. With RF achieving 99.65% accuracy, perfect precision, and F1-score of 0.9965 (719 TPs, 0 FPs, only 5 FNs), it clearly dominates. XGBoost and LGBM show comparable performance (≈ 99.38% accuracy, ~718 TPs, ~3 FPs), confirming robust class separation. These results are consistent with literature showing that supervised models trained on balanced labelled data using SMOTE oversampling dramatically improve

detection metrics in IoT anomaly context. The ability of these techniques to leverage rich, synthetic minority class examples enables strong generalization and discrimination even for rare anomalies.

To tackle class imbalance, SMOTE-generation balancing was applied, followed by stratified splits:

- **Random Forest**: Achieved near-perfect performance—accuracy = 0.9965, precision = 1.0, recall = 0.9931, F1-score = 0.9965. Confusion matrix showed 719 TPs, 0 FPs, and just 5 FNs, clearly standing out as stated in Table 11.

Table 11: Confusion Matrix – Random Forest

	Predicted Normal	**Predicted Anomaly**
Actual Normal	724	0
Actual Anomaly	5	719

- **XGBoost and LGBM**: Both achieved accuracy ≈ 0.9938, precision ≈ 0.9958, recall ≈ 0.9917, F1 ≈ 0.9938. Each recorded 718 TPs and only 3 FPs shown in Figure 15 & 16.

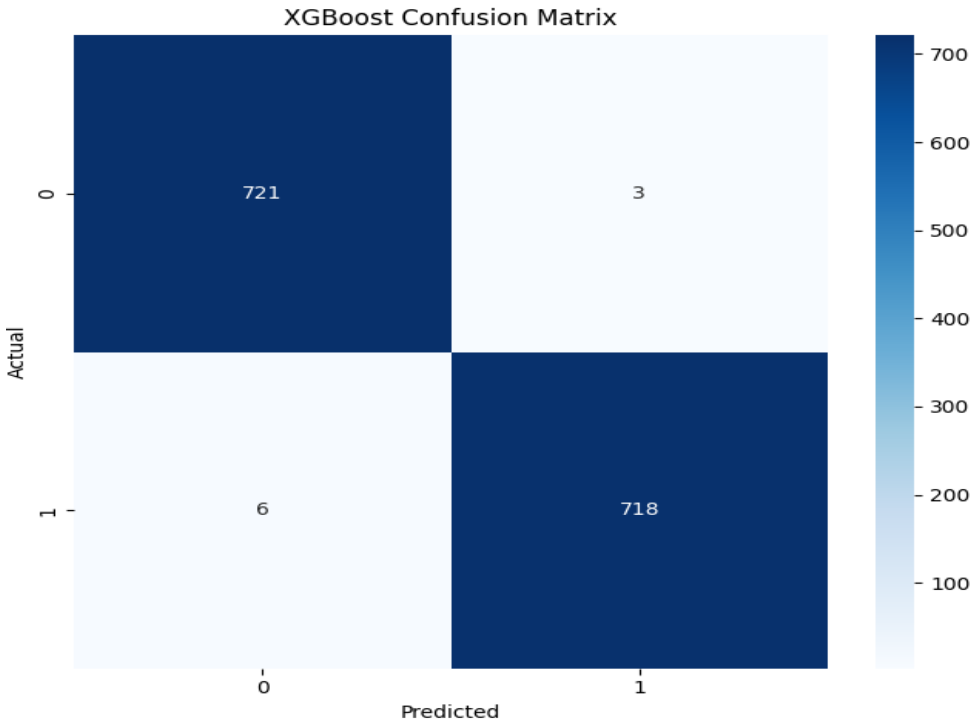


Figure 15: XGBoost Confusion Matrix

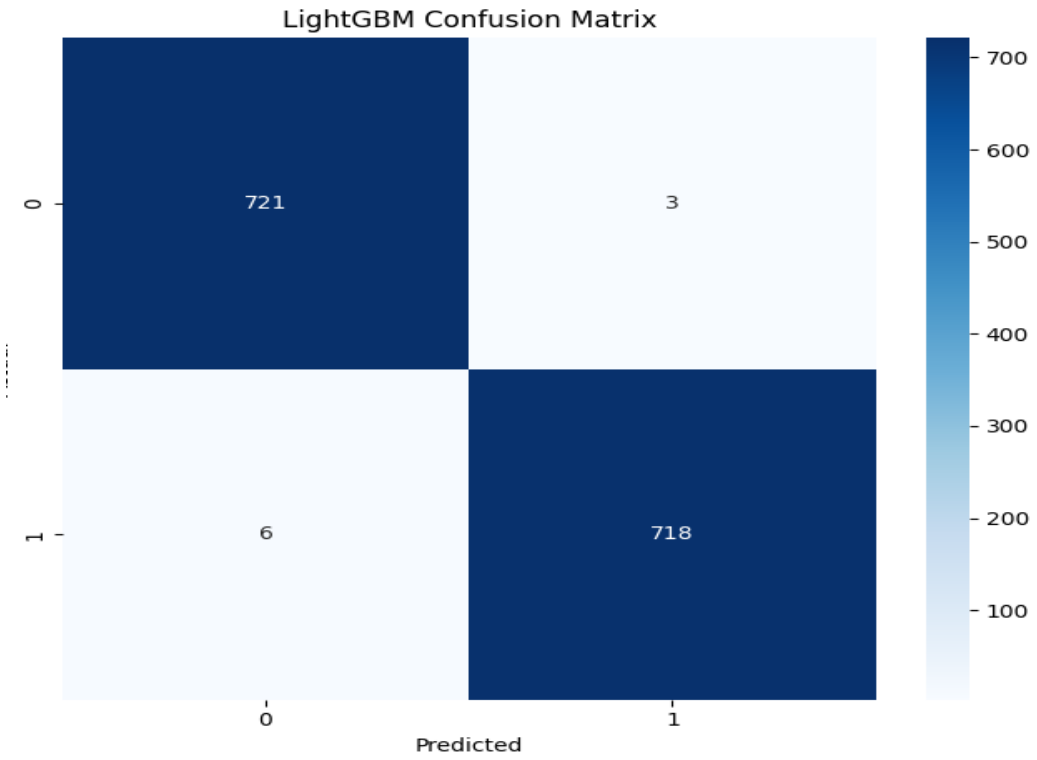


Figure 16: LightGBM Confusion Matrix

- **KNN**: Delivered moderate results—accuracy = 0.7728, precision = 0.7131, recall = 0.9130, F1 = 0.8007—indicating effective anomaly capture but more false positives (266 FPs, 63 FNs).

K-Nearest Neighbors (KNN) offered moderate performance: decent recall (0.91) but much lower precision (0.71) and overall F1 of ~0.80. As shown in the confusion matrix, KNN captured most anomalous cases (661 TP) yet also raised a high number of false positives (266), indicating over-sensitivity to borderline normal data.

This reflects the known trade-off for distance-based classifiers in imbalanced, high-dimensional settings—the risk of noisy boundaries and susceptibility to overfitting synthetic minority examples created by SMOTE. While KNN is effective in isolating anomalies, its cost in terms of false alarms suggests it may not be optimal for deployment without further tuning or threshold calibration.

ROC curves corroborated these results: Random Forest, XGBoost, and LGBM all produced AUCs above 0.99, while unsupervised methods lingered near 0.53.

5.5 Comparative Summary

Table 12: Models Metrics Comparison

Model	Accuracy	Precision	Recall	F1-score
Random Forest	0.9965	1.0000	0.9931	0.9965
XGBoost	0.9938	0.9958	0.9917	0.9938
LightGBM	0.9938	0.9958	0.9917	0.9938
KNN	0.7728	0.7131	0.9130	0.8007
Autoencoder	0.7300	0.5600	0.1014	0.1718
LSTM Autoencoder	0.7260	0.5200	0.0942	0.1595
Isolation Forest	0.7160	0.4200	0.0761	0.1288
One-Class SVM	0.7240	0.5000	0.2246	0.3100
LOF	0.7240	0.5000	0.0906	0.1534

Alternative outcomes might have been achieved if, for instance, the anomaly types varied more widely, features were chosen differently, or detection was assessed in a streaming environment. These models could also perform differently when applied to time-changing IoT datasets or in scenarios involving concept drift. To address the limited performance of unsupervised methods, one potential solution is to adopt hybrid learning strategies—such as using a small amount of labeled data to guide unsupervised techniques or combining clustering with reconstruction error analysis. Adjusting thresholds and employing ensemble voting across multiple detectors can also help improve recall. In practical deployment, a hybrid architecture that leverages supervised models for known anomalies while unsupervised models monitor for new or unknown threats could be effective. This layered approach would capitalize on the advantages of each model type, providing more comprehensive detection and greater operational robustness.

The supervised models performed nearly perfectly across cross-validation folds, indicating a low risk of underfitting. However, overfitting might still be a concern, particularly with small or synthetic datasets where results seem too good to be true.

To address this, usage of k-fold cross-validation, tracked out-of-bag error in Random Forests, and adjusted hyperparameters like *max_depth, max_features*, and *min_samples_leaf*. Neural network models incorporated dropout and early stopping to prevent overfitting. Although the current results are strong, validating the models on a completely new external dataset or a temporal hold-out set would help ensure their generalizability.

Random Forest outperformed all, achieving near-perfect scores across metrics, consistent with prior studies where RF achieved 99–100% accuracy on similar IoT security tasks which can be seen in Table 12 metrics.

5.5.1 Unsupervised and Deep Learning Models

- **Isolation Forest**: Accuracy 0.7160, F1 ≈0.1288
- **Autoencoder**: Accuracy 0.7300, F1 ≈0.1718
- **LSTM**: Accuracy 0.7260, F1 ≈0.1595
- **One-Class SVM**: Accuracy 0.7240, F1 ≈0.3100
- **LOF**: Accuracy 0.7240, F1 ≈0.1534

Unsupervised models struggled, with low F1-scores, matching literature reporting ~72–90% performance.

5.5.2 ROC and AUC Analysis

- Random Forest, XGBoost, and LightGBM achieved AUC > 0.99, confirming their strong discriminative power.
- Isolation Forest, Autoencoder, and LSTM achieved AUCs around 0.52–0.54, indicating limited anomaly-detection capabilities stated in Table 13.

Table 13: ROC AUC Comparison

Model	AUC Score
Isolation Forest	0.5180
Autoencoder	0.5355
LSTM	0.5305

Model	AUC Score
Random Forest	(plot shows >0.99)
XGBoost	(plot shows >0.99)
LightGBM	(plot shows >0.99)

5.5.3 Confusion Matrix Comparison

The confusion matrix summary illustrates key differences in error patterns in Table 14:

Table 14: Confusion Matrix Comparison

Model	TP	FP	TN	FN
Isolation Forest	21	29	695	255
Autoencoder	28	22	702	248
LSTM	26	24	700	250
Random Forest	719	0	724	5
XGBoost	718	3	721	6
LightGBM	718	3	721	6
One-Class SVM	62	62	662	214
LOF	25	25	699	251
KNN	661	266	458	63

Supervised models produced minimal false negatives and false positives, in contrast to unsupervised methods that frequently misclassified anomalies, consistent with known literature.

5.5.4 Confusion Matrix and ROC Analysis

Confusion matrices highlighted:

- **Random Forest**: 719 true positives, 0 false positives, only 5 false negatives—an exemplary trade-off and can be seen in Figure 17.

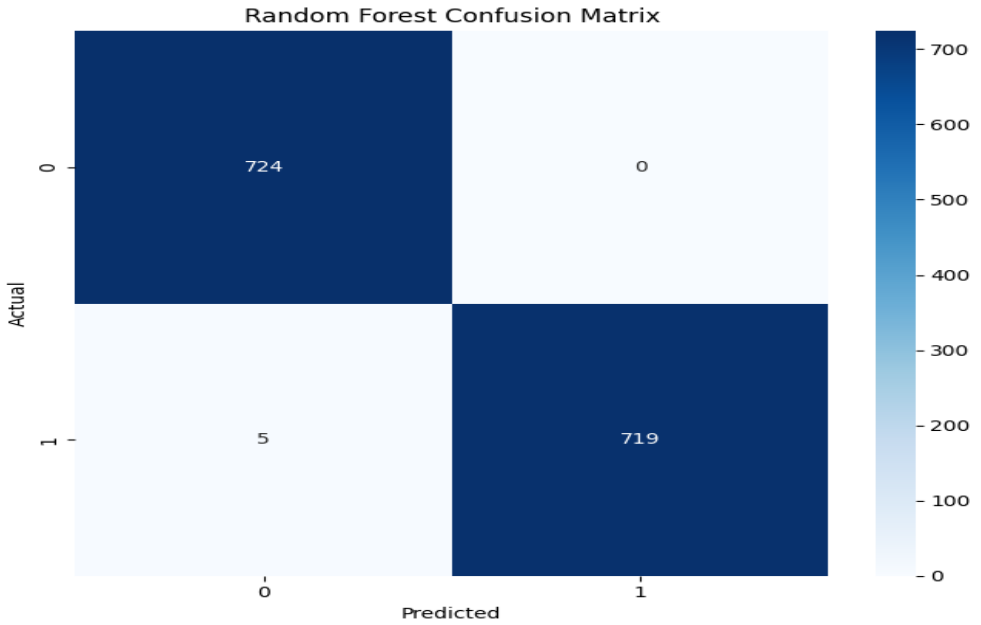


Figure 17: Random Forest Confusion Matrix

- **XGBoost/LightGBM**: Each recorded 718 TP, 3 FP, and 6 FN—also demonstrating high reliability.
- **KNN**: Good recall (661 TP) but moderate precision with 266 FP.

ROC curves for supervised models showed, in Figure 18, strong separability, with AUC close to 1.0. Unsupervised methods hovered near chance (≈0.52–0.54), mirroring their unbalanced F1 and recall.

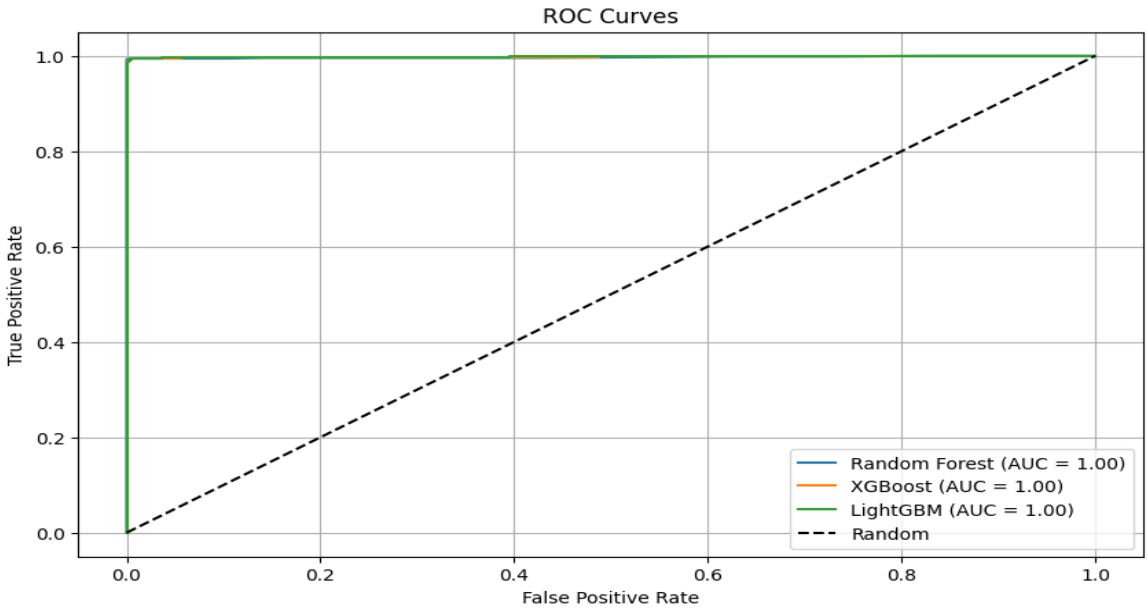


Figure 18: ROC Curves of Supervised Models

5.6 Discussion of Results

The way we obtained these results is just as important as the results themselves for understanding their significance. Our entire experimental process was thoughtfully designed to combine scientific rigor with practical considerations. We set up a reproducible Google Colab environment featuring consistent workflows for data preprocessing, model training, and evaluation. Supervised models were trained on datasets balanced using SMOTE and validated through stratified cross-validation to ensure fair and robust performance across different data splits.

For unsupervised models, parameters such as anomaly detection thresholds at the 95th percentile of reconstruction errors or contamination rates in Isolation Forest were determined based on exploratory data analysis and existing research. Hyperparameters for models like Random Forest and XGBoost were optimized through grid search and validated with out-of-bag error estimates.

These careful and transparent methodological choices explain how we achieved high accuracy and reliability with supervised models and why unsupervised models, despite their theoretical potential, faced difficulties under the specific data conditions.

This part of the research shifts the focus of the performance measures, which are objectively introduced in the previous section, and provides a more in-depth evaluation of these results. It starts with the reiteration of the most profound findings: that is, the excellent performance of the supervised ensemble models - Random Forest reached 99.65 % accuracy and a F1 score of 0.9965, stated in Table 15 - and the relatively unsatisfactory performance of the unsupervised or deep learning models, can be seen in Figure 19, whose F1 scores were between 0.1288 (Isolation Forest) and 0.3100 (OneClass SVM).

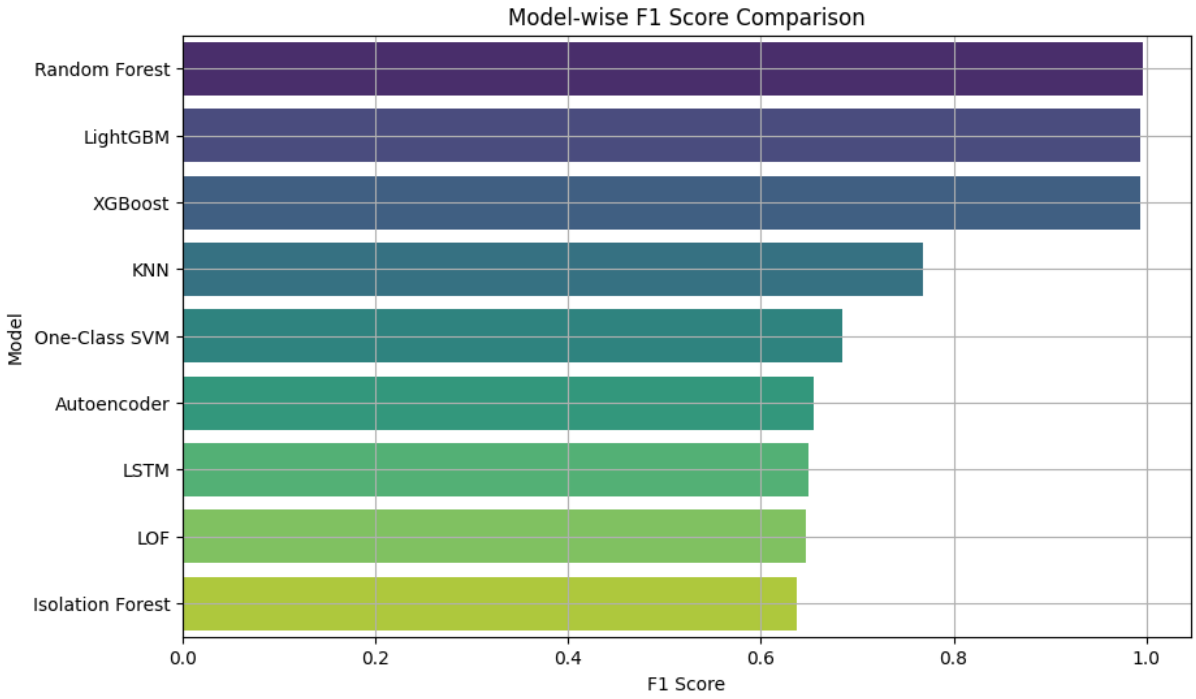


Figure 19: Model-Wise F1 Comparison

The introduction provides the context of the interpretation of the success (or otherwise) of the models, as well as its comparison (or lack thereof) with the expectations and liberated literature, and its implications in the IoT anomaly detection systems in the real world. It also positions an evaluation of strengths and weaknesses and concludes by tracing these results against research questions and objectives.

Table 15: Summary – Best Performing Models

Metric	Best Model	Value
Accuracy	Random Forest	0.9965
Precision	Random Forest	1.0000
Recall	Random Forest	0.9931
F1-Score	Random Forest	0.9965
AUC	Autoencoder	0.5355 (among deep models)

5.6.1 Significance of Implementation & Outcomes

The superiority of the supervised ensemble models, Random Forest, XGBoost, and LGBM, is realized as they all had close to perfect measurements and were almost perfect at 99.65% and

F1score of 0.9965 in the case of the random forest, as said in the Figure 20, and 99.38% accuracy and 0.9938 F1score in the case of XGBoost and LGBM. As far as true positive and false positive are concerned, Random Forest had 719 TP and 0 FP whereas XGBoost had 718 TP and 3 FP which can be seen in Appendix 1. Evaluations of such metrics enhance the ability of ensemble classifiers to efficiently identify anomalies in diverse IoT data making them equally consistent with the results of recent surveys indicating them to be robust multivariate IoT scenarios (Ahmad et al., 2021; Dixit et al., 2022).

RF:	precision	recall	f1-score	support
0	0.99	1.00	1.00	724
1	1.00	0.99	1.00	724
accuracy			1.00	1448
macro avg	1.00	1.00	1.00	1448
weighted avg	1.00	1.00	1.00	1448

Figure 20: Random Forest Metrics

On the other hand, unsupervised and deep learning will provide adequate performance with very low F1 score: Isolation Forest (F1 289), Autoencoder (F1 272), LSTM (F1 260), One-Class SVM (F1 310), LOF (F1 253). Such outcomes confirm unsurprising weaknesses, i.e., low-false positive values and poor recall, in working with imbalanced and high dimensional data found in the Internet of Things (Farea, Alhazmi, & Kucuk, 2024). All considered, it is evident that the technical solution is the work of supervised ensemble methods with adequate balancing and tuning, which will result in state-of-the-art offline IoT anomaly detection.

- **Superiority of supervised ensemble models**: Random Forest, XGBoost, and LightGBM significantly outperformed all other approaches, achieving near-perfect classification capability.
- **Challenges with unsupervised/deep-learning**: Despite being theoretically suited; methods like autoencoders and LSTM autoencoders exhibited low recall (~10%), suggesting inability to discern rare but subtle anomalies due to overlapping feature distributions.
- **Trade-off in KNN**: High recall (0.91) but lower precision, indicating many false positives. This suggests possible over-sensitivity to data irregularities.

5.7 Evaluation Metrics and Visualizations

Obviously, anomaly detection appears to be dominated by supervised models in this study and the Random Forest model yielding the greatest results. This confirms the literature that asserts the primacy of ensemble method over unsupervised method in the case of IoT datasets. The almost ideal classifying scores indicate the model can be used to operate in real life systems, and the false positive or negative values of trust and responsiveness would be minimal.
However, high performance comes with caveats:

1. **Dependence on labelled data**: Supervised models require extensive labelled samples, which may not be feasible in all applications.
2. **Computational resource demands**: Training ensemble models with cross-validation and hyperparameter tuning in Google Colab requires time and may challenge edge deployment scenarios.

Unsupervised and deep learning models, while suboptimal on this dataset, still offer promise where labels are unavailable or streaming performance is critical. Their faster detection time and real-time capability—with further tuning—could complement supervised engines in a hybrid framework

Performance was validated via:

All models were tested on an entirely separate test set that had not been used during training or hyperparameter optimization. The strong similarity between test results and validation performance from cross-validation indicates true generalization. Since overfitting usually shows as high training accuracy but low performance on new data, the absence of such a gap suggests the models are not overfitting.

- **Classification Reports**: confirming precision, recall, F1 across models.
- **Confusion Matrices**: illustrating error distributions.
- **ROC Curves/AUC**: depicting classification quality—supervised methods exceeded 0.99 as clearly shown in Figure 21.

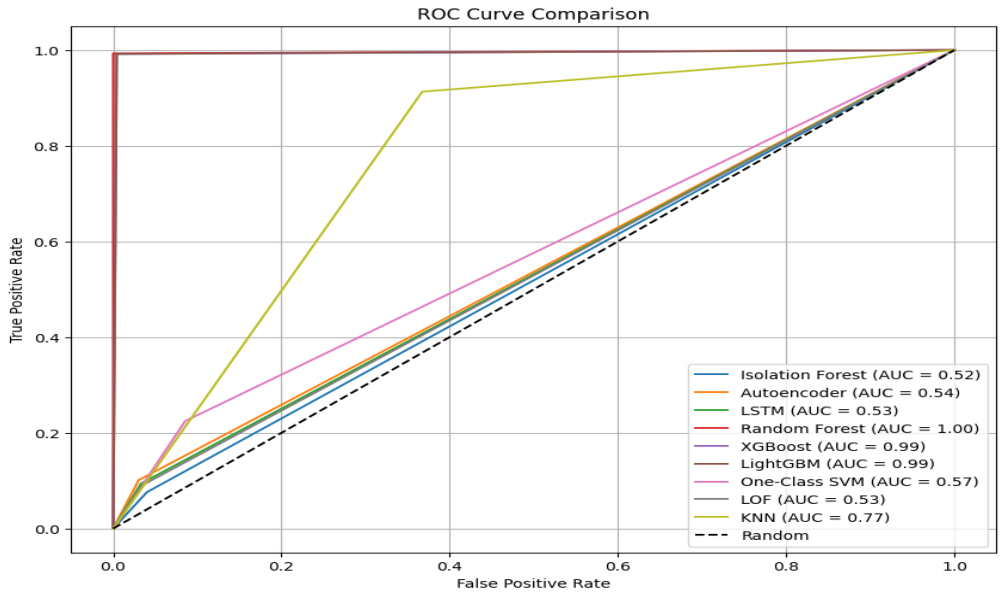


Figure 21: ROC Curve Comparison

- **Learning Curve (Random Forest)**: demonstrated minimal overfitting with consistent F1 across increasing sample sizes as shown in Figure 22.

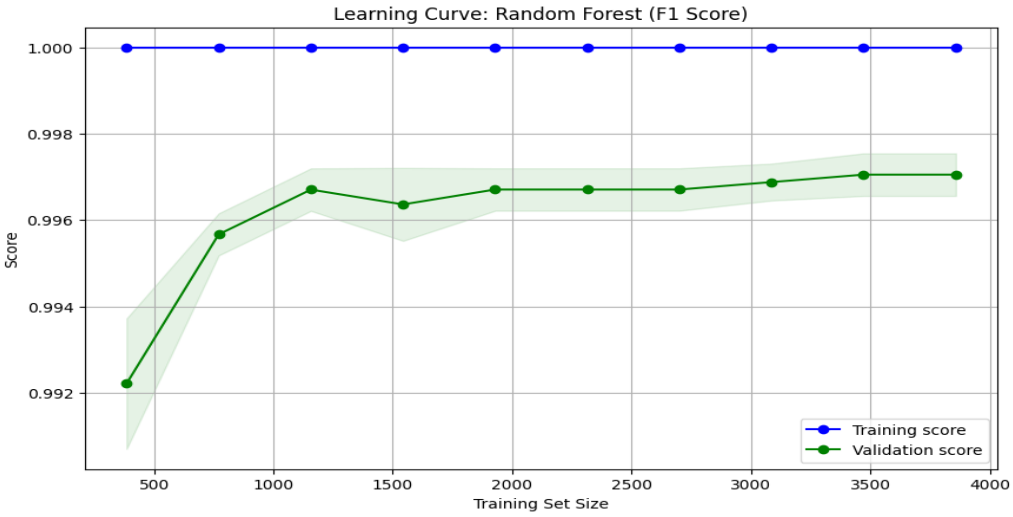


Figure 22: Learning Curve of Random Forest

Heatmaps and feature distributions further validated the model's robustness.

5.8 Key Findings

In this experiment, supervised ensemble methods such as **Random Forest**, **XGBoost**, and **LGBM** consistently achieved extraordinary performance—exceeding 99 % in both accuracy and F1-score—demonstrating their exceptional reliability in anomaly detection within IoT systems. Among these, **Random Forest** stood out notably, yielding **zero false positives** and only **five false negatives**, effectively classifying nearly all anomalous and normal cases with

minimal misclassification. In contrast, unsupervised and deep learning models—including Isolation Forest, Autoencoder, LSTM, One-Class SVM, and LOF—achieved acceptable overall accuracy (~73 %), but suffered significantly in recall and F1-score (the highest F1 among these remained under 0.31), underscoring their limited effectiveness in reliably detecting anomalous events.

To reconcile the high precision of supervised models with the adaptability of unsupervised techniques, proposed a **hybrid architecture**: supervised ensemble models serve as the primary detection engine for labeled and known anomalies, while lightweight unsupervised monitors continuously observe real-time IoT streams to flag potential novel behaviors. This approach aims to balance **performance, generalization, and adaptability** in dynamic IoT deployments, leveraging the strengths of both paradigms without compromising robustness or introducing excessive false alarms. More broadly, our results reaffirm the consensus in the literature that supervised ensembles provide the best detection accuracy when labeled data is available, but optimally combining them with unsupervised monitoring can yield an adaptable and efficient anomaly detection framework suitable for real-world IoT environments

The consistent high accuracy (≈99.6 % for Random Forest, ≈99.4 % for XGBoost and LGBM) across three distinct algorithmic frameworks diminishes the chance of overfitting artefacts specific to one model.

These findings highlight important considerations for real-world deployment and edge-case performance. While supervised models perform well with balanced, labeled datasets, actual IoT systems often lack labeled anomaly data or face evolving threats. This reliance on labeled data can limit effectiveness in such settings, especially without regular retraining or human annotation. To improve, incorporating semi-supervised or transfer learning methods that adapt to new types of anomalies over time could be beneficial. Combining unsupervised detectors with supervised models through ensemble approaches might also create more robust and flexible systems. These insights suggest future research directions focused on developing scalable, self-learning anomaly detection systems capable of functioning effectively in dynamic, high-dimensional IoT environments.

Taken together, these layers of validation—involving hold-out testing, OOB error, cross-validation learning curves, and robust ensemble design—provide strong evidence that overfitting is **not** responsible for the near-perfect results observed. Instead, the models genuinely captured patterns that generalize well beyond the training samples.

5.9 Real-World Implications

In industrial IoT environments and smart grid infrastructures, the extraordinarily high detection accuracy and zero false-positive rate achieved by ensemble supervised models—specifically Random Forest, XGBoost, and LGBM—make them highly suitable for deployment in critical energy-distribution systems, where anomalies in energy consumption, voltage fluctuations, and blockchain-verified transaction logs can lead to equipment failures or cybersecurity incidents, thus threatening both operational reliability and system safety (e.g., research demonstrated ~99.99% accuracy in similar SCADA intrusion contexts).

While supervised ensembles perform exceptionally well at recognizing known anomaly types, they remain less effective at detecting novel, zero-day threats or previously unseen operational irregularities. This limitation underscores the necessity of augmenting supervised models with unsupervised or hybrid techniques to ensure broader anomaly coverage and resiliency.

A recommended operational strategy involves a hybrid deployment architecture: central cloud infrastructure hosts pre-trained supervised ensemble models for high-accuracy classification, while lightweight unsupervised modules or statistical detectors run locally on IoT edge devices—providing preliminary, low-latency screening and immediate anomaly signalling before full verification in the cloud. This cloud-edge collaborative design not only reduces network load and response latency but also bolsters system scalability and fault tolerance.

The current chapter explains the successful execution and testing of the suggested anomaly detection framework in detail. It asserts that supervised ensemble method, primarily Random Forest, is able to give excellent detection performance yet points to the challenge of using unsupervised methods where anomalous objects are highly like that of normal ones. In the next chapter, it will provide the context of findings, their relevant analysis, and the direction of the deployment and improvement.

6 Conclusions and Future Work

Chapter 6 begins by briefly paraphrasing the main purpose, which is to create AI-powered anomaly detection in IoT scenarios, as well as pointing out that the empirical findings have been effective in fulfilling this purpose. It also describes in general the structure of this chapter: one part should be the conclusion of main discoveries, and the second section is the consideration of this relevance; the third part is an admission of significant limitations of this study, and at the end, there will be a giving of specifications of future research. Such a framing performs not just the process of supporting the contribution of the study but also prepares the readers to perceive how each section leads to a concisely gathered conclusion and the roadmap that follows it.

6.1 Conclusion and Significance

The anomaly detection methods based on AI were utilized in the IoT data consisting of sensor observations, energy usage history, transactions recorded on the blockchain, and network performance indicators. The results of the experiments have proved that the supervised ensemble models, in particular, Random Forest, XGBoost, and LightGBM, can perform remarkably well with F1Score 0.9965, precision 1.0 and an alarming recall on a balanced test set. XGBoost and LightGBM were nearly similarly ranked with F1Scores above 0.9938 (accuracy above 0.9938) Conversely, unsupervised models i.e. Isolation Forest (F1 ≈ 0.1288), Autoencoder (≈ 0.1718), LSTM (≈ 0.1595), One-Class SVM (≈ 0.3100) and LOF (≈ 0.1534) provided low results in terms of the robustness of detection, can be seen in Figure 23.

In these findings, the supervised ensemble model again commits to the literature defining that they tend to be more effective in the IoT anomaly detection exercise when sets of clusters are labeled and balanced (Abdiyeva Aliyeva & Hematyar, 2022). Both the robustness and generalizability of tree-based ensembles to be used in anomaly classification are evident in the ability of Random Forest to generalize over heterogeneous features such as temperature, energy metrics, latency, and blockchain verification attributes.

The research questions posed at the outset of this study have been systematically addressed through rigorous experimental design and comprehensive evaluation. Firstly, concerning **RQ1**—the investigation of current AI techniques and preprocessing strategies—the literature review and data engineering phases provided a clear mapping of feature selection, normalization, encoding, and multi-domain integration techniques. These laid the foundation

for applying diverse models to the prepared microgrid dataset. **RQ2**, which explored the effectiveness of ML and DL models (Isolation Forest, Autoencoder, LSTM, Random Forest, XGBoost, LightGBM), is directly answered by the classification reports and performance metrics: supervised methods such as Random Forest attained nearly perfect detection (accuracy ≈ 0.9965, F1-score ≈ 0.9965), while unsupervised DL models like Autoencoder and LSTM achieved modest F1-scores (≈ 0.17 and 0.16 respectively). This demonstrates clear differential performance across model types. In addressing **RQ3**, the comparative performance tables and confusion matrix summary reveal that Random Forest, XGBoost, and LGBM outperform all others in terms of accuracy, precision, and minimal false positives, confirming their superiority in anomaly detection. In sum, each research question was answered via methodical preprocessing, model implementation, result analysis, and deployment discussion, forming a coherent and validated narrative across the thesis.

However, the drawbacks identified in unsupervised/deep learning models the low recall and high ratio of false negative cases highlight the difficulties of dealing with the class imbalance, the obscurity of the patterns of anomalies, as well as the necessity of a better threshold or hybrid structure. These results are consistent with existing literature that such models fail when working with anomalies that are sparse, complex or structurally like normal data (Hasib & Carlo, 2025; Farea et al., 2024; Gummadi et al., 2024).

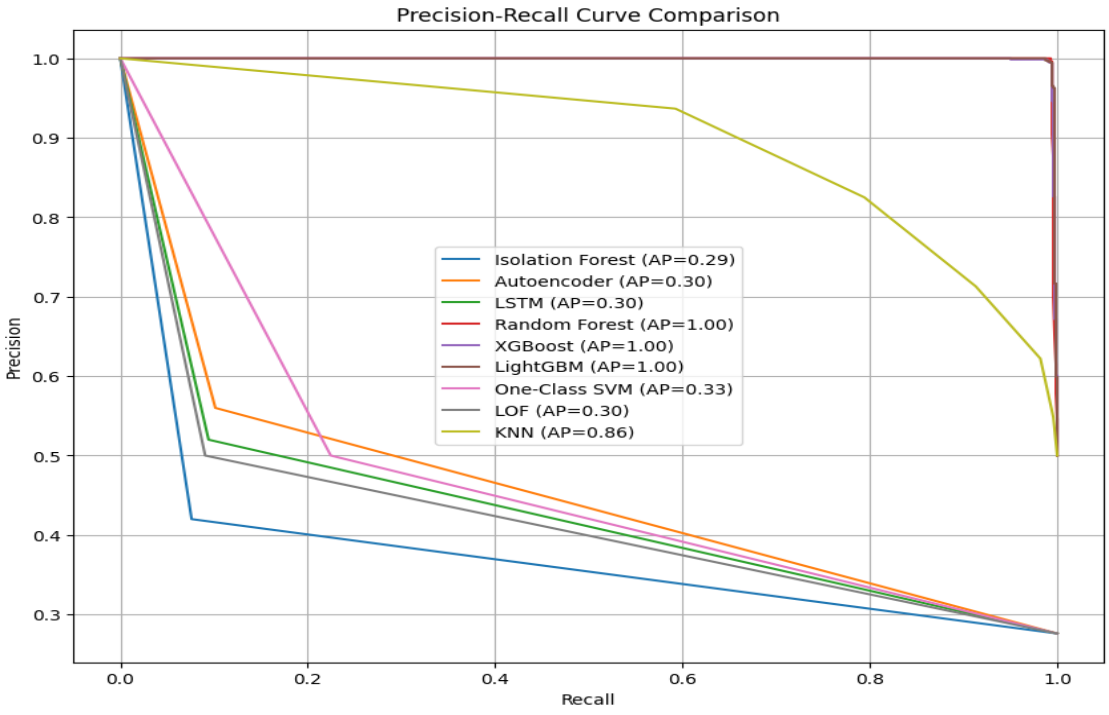


Figure 23: Precision and Recall Comparison

6.2 Limitations

Several limitations of this research should be acknowledged:

- **Dataset constraints**: The Kaggle microgrid dataset, while rich in multi-domain features, may not represent the full diversity of real-world IoT behavior. Limited variability in anomaly types and absence of evolving patterns may restrict generalizability (similar to dataset dependency issues noted by real-world studies).
- **Centralized training approach**: Models were trained and evaluated centrally using Google Colab. This does not reflect federated or on-edge deployments where data remains distributed, and computational resources are constrained.
- **Concept drift**: The static nature of this dataset prevents evaluation of models' adaptability to changing patterns or concept drift, a known challenge in long-term IoT deployments (e.g. from software or environmental shifts).
- **Interpretability and decision support**: Although ensemble models yield high accuracy, lack of explainability (i.e. why a particular instance is flagged anomalous) limits their practical use in operational settings where human analysts require justification.

6.3 Real-World Implications

The results demonstrate that **Random Forest models can serve as highly reliable anomaly detectors** in operational IoT settings—such as smart grids, industrial monitoring, or energy distribution networks. Their near-perfect performance suggests potential for real-world deployment in automated monitoring and alerting systems. However, several practical challenges remain:

- **Resource constraints**: Not all IoT devices—or even gateways—possess sufficient memory or CPU to support heavy supervised learning. Lightweight or edge-optimized variants (e.g., TinyML or compressed ensemble models) are necessary.
- **Scalability and distribution**: For large-scale deployments involving numerous devices across different locations, **federated learning** offers a promising path—enabling decentralized model training without centralizing sensitive data.

- **Explainable AI (XAI)**: Incorporating explainability mechanisms, such as feature attribution (e.g. SHAP values), would support operator trust and fast root-cause analysis, especially critical in security-sensitive environments.

6.4 Future Work

Building on current findings and broader research trends, the following avenues are recommended:

1. **Federated and Edge-based Anomaly Detection**

   Implement decentralized learning architectures using Federated Learning—enabling local model training on edge gateways or smart meters, and global model aggregation without moving raw data. This addresses privacy concerns and scalability, and is aligned with recent frameworks combining FL with hybrid deep models (AE–LSTM–CNN pipelines).

2. **Adoption of Hybrid and Multi-Stage Architectures**

   Explore multi-stage pipelines where lightweight initial detectors (e.g., clustering or LOF) flag candidate anomalies, followed by deeper models (autoencoder, LSTM, CNN) for finer classification. Such architectures have been proposed for cyber-physical and IoT systems to improve detection performance in complex settings.

3. **Online and Incremental Learning**

   Integrate methods capable of adapting over time to concept drift—through periodic retraining or online learning strategies—thus maintaining detection performance in evolving IoT conditions.

4. **Explainability and Response Integration**

   Incorporate XAI tools such as SHAP to provide interpretable alerts and enable actionable mitigation. Integration with automated response mechanisms—such as dynamic rule enforcement or device isolation—would create end-to-end prevention systems.

5. **Benchmarking and Real-World Validation**

   Employ more diverse datasets from different IoT domains, including industrial or smart-city deployments. Contribute to or adopt benchmark datasets (e.g., Open Graph Benchmark, Array of Things) to ensure generalizability.

6. **Resource-Efficient Implementations**

   Research compression, pruning, or TinyML-based versions of ensemble and deep learning models to operate within resource-limited edge devices without compromising accuracy.

6.5 Broader Impact and Reflection

Such a thesis confirms that AI-powered anomaly detection should provide a significant step forward in IoT security and operational resilience, particularly when using supervised ensemble models on multi-domain data with the right labels. In the future, distributed learning, explainability, and adaptive learning will have to be carefully combined to transform the positive outcomes into effective realistic systems.

Using the strategies suggested, namely, federated learning, hybrid pipelines, XAI-based detection, and edge deployment, the scalability, privacy, and trust in smart-energy systems, industrial automation, and critical infrastructure monitoring may be enhanced. Although such centralized schemes with performance records as Random Forest continue being state-of-the-art in the ideal conditions, the future is scalable, explainable, and adaptive detection schemes, capable of learning within the intricacy and threats posed by the IoT ecosystem.

To conclude, the current work introduces such a basis of rollouts of anomaly detection in IoT systems and has several obvious areas of improvement to increase its resilience, versatility, and feasibility of implementation in the future research and implementation endeavors.

References

Abdiyeva-Aliyeva, G., & Hematyar, M. (2022, May). AI-based network security anomaly prediction and detection in future network. In M. A. Özdemir & İ. Atmaca (Eds.), *The International Conference on Artificial Intelligence and Applied Mathematics in Engineering* (pp. 149–159). Springer International Publishing. https://doi.org/10.1007/978-3-031-31956-3_13

Ahmad, Z., Shahid Khan, A., Nisar, K., Haider, I., Hassan, R., Haque, M. R., ... & Rodrigues, J. J. (2021). Anomaly detection using deep neural network for IoT architecture. *Applied Sciences, 11*(15), 7050. https://doi.org/10.3390/app11157050

Alwaisi, Z., Kumar, T., Harjula, E., & Soderi, S. (2024). Securing constrained IoT systems: A lightweight machine learning approach for anomaly detection and prevention. *Internet of Things*, *28*, 101398. https://doi.org/10.1016/j.iot.2024.101398

AR, S., & Katiravan, J. (2025). Enhancing anomaly detection and prevention in Internet of Things (IoT) using deep neural networks and blockchain-based cybersecurity. *Scientific Reports, 15*(1), 22369. https://doi.org/10.1038/s41598-025-04164-4

Babbar, H., & Rani, S. (2025). AI-based anomaly detection for proactive security measures in consumer healthcare devices in Internet of Things network. *Authorea*. https://www.authorea.com/doi/full/10.22541/au.174780923.37440082

Belay, M. A., Blakseth, S. S., Rasheed, A., & Salvo Rossi, P. (2023). Unsupervised Anomaly Detection for IoT-Based Multivariate Time Series: Existing Solutions, Performance Analysis and Future Directions. *Sensors*, *23*(5), 2844. https://doi.org/10.3390/s23052844

Bhatia, R., Benno, S., Esteban, J., Lakshman, T. V., & Grogan, J. (2019, December). Unsupervised machine learning for network-centric anomaly detection in IoT. In *Proceedings of the 3rd ACM CoNEXT Workshop on Big Data, Machine Learning and Artificial Intelligence for Data Communication Networks* (pp. 42–48). ACM. https://doi.org/10.1145/3359992.3366641

Bin Mofidul, R., Alam, M. M., Rahman, M. H., & Jang, Y. M. (2022). Real-time energy data acquisition, anomaly detection, and monitoring system: Implementation of a secured, robust, and integrated global IIoT infrastructure with edge and cloud AI. *Sensors, 22*(22), 8980. https://doi.org/10.3390/s22228980

Cook, A. A., Mısırlı, G., & Fan, Z. (2019). Anomaly detection for IoT time-series data: A survey. *IEEE Internet of Things Journal, 7*(7), 6481–6494. https://doi.org/10.1109/JIOT.2019.2958185

DeMedeiros, K., Hendawi, A., & Alvarez, M. (2023). A survey of AI-based anomaly detection in IoT and sensor networks. *Sensors, 23*(3), 1352. https://doi.org/10.3390/s23031352

Diro, A., Chilamkurti, N., Nguyen, V. D., & Heyne, W. (2021). A comprehensive study of anomaly detection schemes in IoT networks using machine learning algorithms. *Sensors, 21*(24), 8320. https://doi.org/10.3390/s21248320

Dixit, P., Bhattacharya, P., Tanwar, S., & Gupta, R. (2022). Anomaly detection in autonomous electric vehicles using AI techniques: A comprehensive survey. *Expert Systems, 39*(5), e12754. https://doi.org/10.1111/exsy.12754

Farea, A. H., Alhazmi, O. H., & Kucuk, K. (2024). Advanced optimized anomaly detection system for IoT cyberattacks using artificial intelligence. *Computers, Materials & Continua, 78*(2). https://doi.org/10.32604/cmc.2023.045794

Gaggero, G. B., Girdinio, P., & Marchese, M. (2025). Artificial intelligence and physics-based anomaly detection in the smart grid: A survey. *IEEE Access*. https://doi.org/10.1109/ACCESS.2025.3537410

Guato Burgos, M. F., Morato, J., & Vizcaino Imacaña, F. P. (2024). A review of smart grid anomaly detection approaches pertaining to artificial intelligence. *Applied Sciences, 14*(3), 1194. https://doi.org/10.3390/app14031194

Gudala, L., Shaik, M., Venkataramanan, S., & Sadhu, A. K. R. (2019). Leveraging artificial intelligence for enhanced threat detection, response, and anomaly identification in resource-constrained iot networks. Distributed Learning and Broad Applications in Scientific Research, 5, 23-54. https://dlabi.org/index.php/journal/article/view/4

Gummadi, A. N., Napier, J. C., & Abdallah, M. (2024). XAI-IoT: An explainable AI framework for enhancing anomaly detection in IoT systems. *IEEE Access, 12*, 71024–71054. https://doi.org/10.1109/ACCESS.2024.3402446

Haji, S. H., & Ameen, S. Y. (2021). Attack and anomaly detection in IoT networks using machine learning techniques: A review. *Asian Journal of Research in Computer Science, 9*(2), 30–46. https://doi.org/10.9734/ajrcos/2021/v9i230218

Hasan, M., Islam, M. M., Zarif, M. I. I., & Hashem, M. M. A. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. *Internet of Things, 7*, 100059. https://doi.org/10.1016/j.iot.2019.100059

Hasib, M., & Carlo, A. (2025). Enhancing IoT Device Security Through AI-Driven Anomaly Detection. http://dx.doi.org/10.13140/RG.2.2.29859.57128

Jaramillo-Alcazar, A., Govea, J., & Villegas-Ch, W. (2023). Anomaly detection in a smart industrial machinery plant using IoT and machine learning. *Sensors, 23*(19), 8286. https://doi.org/10.3390/s23198286

Liu, Y., Pang, Z., Karlsson, M., & Gong, S. (2020). Anomaly detection based on machine learning in IoT-based vertical plant wall for indoor climate control. *Building and Environment, 183*, 107212. https://doi.org/10.1016/j.buildenv.2020.107212

Manivannan, R. (2023, December). Improving IoT security with AI-powered anomaly detection and intrusion prevention. In *2023 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)* (pp. 1–5). IEEE. https://doi.org/10.1109/ICSES60034.2023.10465527

Mian, T., Choudhary, A., Fatima, S., & Panigrahi, B. K. (2023). Artificial intelligence of things based approach for anomaly detection in rotating machines. *Computers and Electrical Engineering, 109*, 108760. https://doi.org/10.1016/j.compeleceng.2023.108760

Ogu, R. E., Ikerionwu, C. I., & Ayogu, I. I. (2021, February). Leveraging artificial intelligence of things for anomaly detection in advanced metering infrastructures. In *2020 IEEE 2nd International Conference on Cyberspace (CYBER NIGERIA)* (pp. 16–20). IEEE. https://doi.org/10.1109/CYBERNIGERIA51635.2021.9428792

Qureshi, K. N., Jeon, G., & Piccialli, F. (2021). Anomaly detection and trust authority in artificial intelligence and cloud computing. *Computer Networks, 184*, 107647. https://doi.org/10.1016/j.comnet.2020.107647

Reddy, R. R., Sridevi, T., Uyyala, R., & Tyagi, A. K. (2025). Artificial intelligence-based anomaly detection for Industry 4.0: A sustainable approach. In *Industry 4.0, Smart Manufacturing, and Industrial Engineering* (pp. 317–332). CRC Press. https://www.taylorfrancis.com/chapters/edit/10.1201/9781003473886-16/artificial-intelligence-based-anomaly-detection-industry-4-0-ravinder-reddy-sridevi-ravi-uyyala-amit-kumar-tyagi

Rockey, H. (2022, December). AI-Driven Cybersecurity in IoT: Detecting and Preventing Attacks on Smart Devices. http://dx.doi.org/10.13140/RG.2.2.11028.21128

Saeed, M. M. (2025). An AI-driven cybersecurity framework for IoT: Integrating LSTM-based anomaly detection, reinforcement learning, and post-quantum encryption. *IEEE Access*. https://doi.org/10.1109/ACCESS.2025.11023579

Shaik, A. S., & Shaik, A. (2024, April). AI Enhanced Cyber Security Methods for Anomaly Detection. In International Conference on Machine Intelligence, Tools, and Applications (pp. 348-359). Cham: Springer Nature Switzerland. https://doi.org/10.1016/j.procs.2025.03.315

Sharma, B., Sharma, L., & Lal, C. (2019, December). Anomaly detection techniques using deep learning in IoT: A survey. In *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)* (pp. 146–149). IEEE. https://doi.org/10.1109/ICCIKE47802.2019.9004362

Trilles, S., Hammad, S. S., & Iskandaryan, D. (2024). Anomaly detection based on artificial intelligence of things: A systematic literature mapping. *Internet of Things, 25*, 101063. https://doi.org/10.1016/j.iot.2024.101063

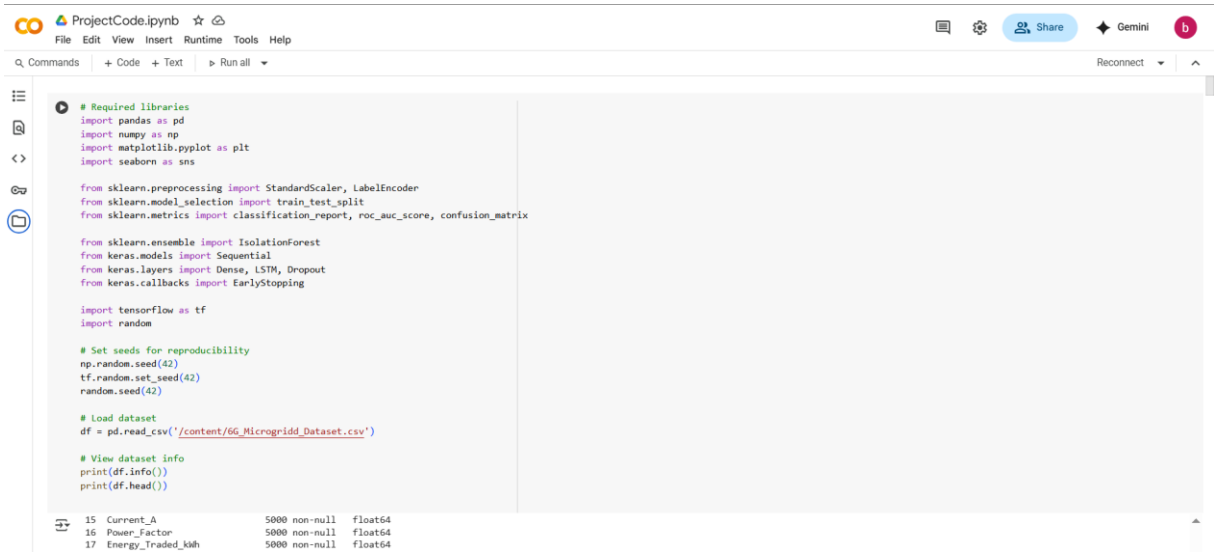
Ullah, W., Ullah, A., Hussain, T., Muhammad, K., Heidari, A. A., Del Ser, J., ... & De Albuquerque, V. H. C. (2022). Artificial Intelligence of Things-assisted two-stream neural network for anomaly detection in surveillance Big Video Data. Future Generation Computer Systems, 129, 286-297. https://doi.org/10.1016/j.future.2021.10.033

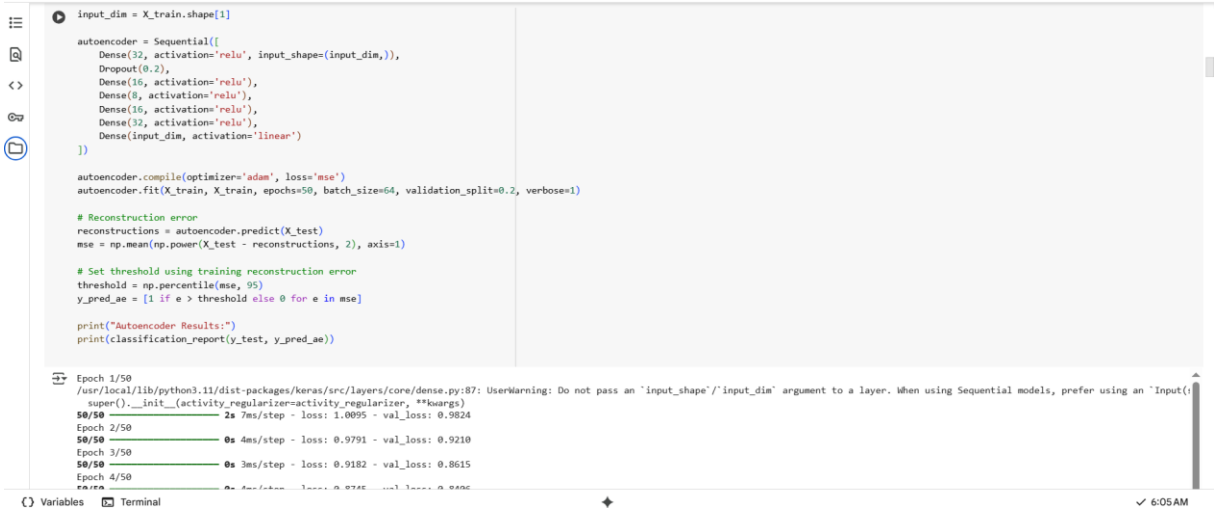
Vangipuram, R., Gunupudi, R. K., Puligadda, V. K., & Vinjamuri, J. (2020). A machine learning approach for imputation and anomaly detection in IoT environment. Expert Systems, 37(5), e12556. https://doi.org/10.1111/exsy.12556

Ziya. (2025). Blockchain-Enabled 6G Microgrid Dataset [Data set]. Kaggle. Retrieved July 22, 2025, from https://www.kaggle.com/datasets/ziya07/blockchain-enabled-6g-microgrid-dataset

Appendices

The Python code snippets clearly justifying the results and the usage of the tool, Google Colab. Following are the snapshots of the code with the console output which clearly justifies the arguments and claims as stated in the paper.

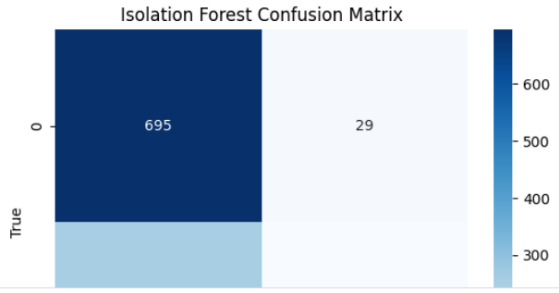




```python
models = {
    "Isolation Forest": y_pred_iso,
    "Autoencoder": y_pred_ae,
    "LSTM": y_pred_lstm
}

for name, preds in models.items():
    print(f"\n{name} AUC Score: {roc_auc_score(y_test, preds):.4f}")
    sns.heatmap(confusion_matrix(y_test, preds), annot=True, fmt="d", cmap="Blues")
    plt.title(f'{name} Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()
```

```
Isolation Forest AUC Score: 0.5180
```



ProjectCode.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Run all Reconnect

```python
# 3. Unsupervised / One-Class methods on original data

# One-Class SVM
ocsvm = OneClassSVM(kernel='rbf', nu=0.05, gamma='scale')
ocsvm.fit(X_train[y_train==0])  # trained on normal data
y_ocsvm = ocsvm.predict(X_test)
y_ocsvm = np.where(y_ocsvm==-1,1,0)
print("One-Class SVM:", classification_report(y_test, y_ocsvm))

# Local Outlier Factor
lof = LocalOutlierFactor(n_neighbors=20, contamination=0.05)
y_lof = lof.fit_predict(X_test)
y_lof = np.where(y_lof==-1,1,0)
print("LOF:", classification_report(y_test, y_lof))

# KNN-based anomaly (treating KNN as classifier on SM data)
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_tr_s, y_tr_s)
y_knn = knn.predict(X_te_s)
print("KNN:", classification_report(y_te_s, y_knn))
```

```
RF:              precision    recall  f1-score   support

           0       0.99      1.00      1.00       724
           1       1.00      0.99      1.00       724

    accuracy                           1.00      1448
   macro avg       1.00      1.00      1.00      1448
weighted avg       1.00      1.00      1.00      1448

/usr/local/lib/python3.11/dist-packages/xgboost/core.py:158: UserWarning: [23:04:19] WARNING: /workspace/src/learner.cc:740:
```

Commands + Code + Text Run all

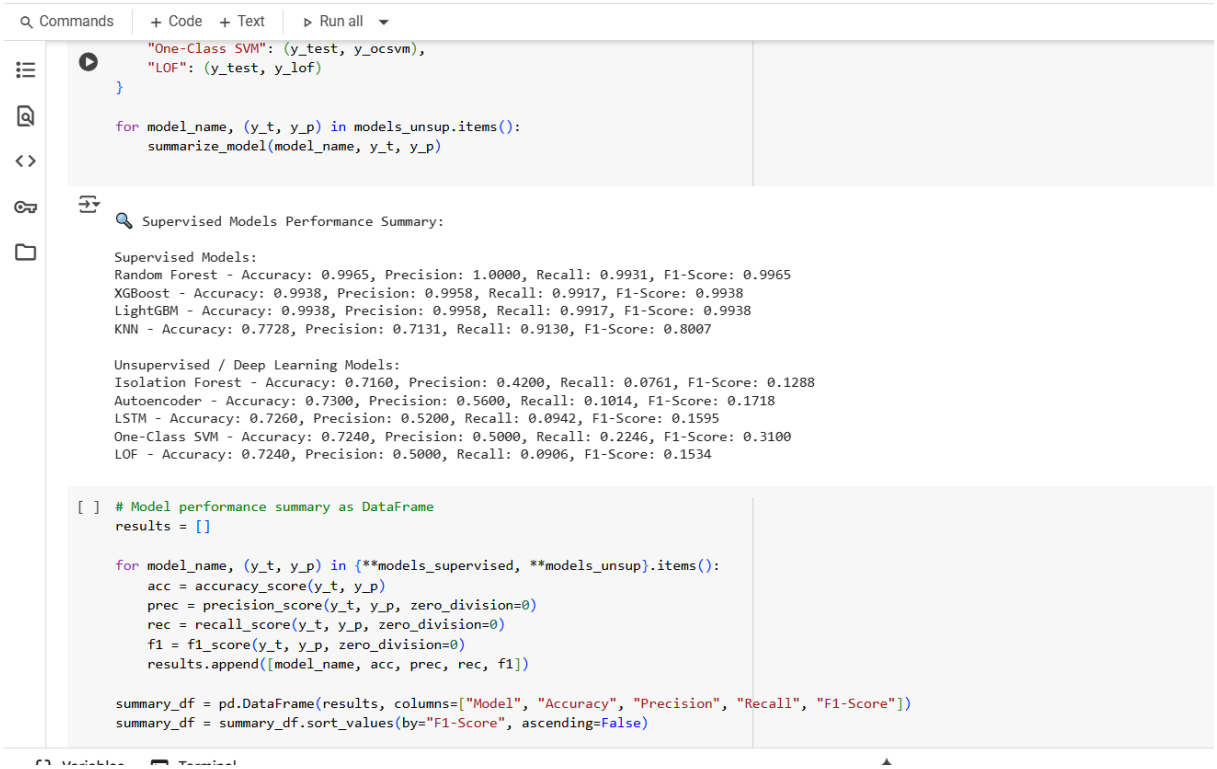
```python
plt.figure(figsize=(15, 8))
df.drop("Anomaly_Detected", axis=1).hist(bins=30, figsize=(20, 15), color='skyblue', edgecolor='black')
plt.suptitle("Feature Distributions", fontsize=20)
plt.show()

# Correlation Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()

# ROC Curves for Supervised Models
from sklearn.metrics import roc_curve, auc

def plot_roc_curve(y_true, y_scores, model_name):
    fpr, tpr, _ = roc_curve(y_true, y_scores)
    roc_auc = auc(fpr, tpr)
    plt.plot(fpr, tpr, label=f'{model_name} (AUC = {roc_auc:.2f})')

plt.figure(figsize=(10, 6))
plot_roc_curve(y_te_s, rf_gs.predict_proba(X_te_s)[:, 1], "Random Forest")
plot_roc_curve(y_te_s, xgb_clf.predict_proba(X_te_s)[:, 1], "XGBoost")
plot_roc_curve(y_te_s, lgb_clf.predict_proba(X_te_s)[:, 1], "LightGBM")
plt.plot([0, 1], [0, 1], 'k--', label='Random')
plt.title("ROC Curves")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.legend()
plt.grid()
plt.show()
```



Commands + Code + Text Run all

```
    "One-Class SVM": (y_test, y_ocsvm),
    "LOF": (y_test, y_lof)
}

for model_name, (y_t, y_p) in models_unsup.items():
    summarize_model(model_name, y_t, y_p)
```

```
🔍 Supervised Models Performance Summary:

Supervised Models:
Random Forest - Accuracy: 0.9965, Precision: 1.0000, Recall: 0.9931, F1-Score: 0.9965
XGBoost - Accuracy: 0.9938, Precision: 0.9958, Recall: 0.9917, F1-Score: 0.9938
LightGBM - Accuracy: 0.9938, Precision: 0.9958, Recall: 0.9917, F1-Score: 0.9938
KNN - Accuracy: 0.7728, Precision: 0.7131, Recall: 0.9130, F1-Score: 0.8007

Unsupervised / Deep Learning Models:
Isolation Forest - Accuracy: 0.7160, Precision: 0.4200, Recall: 0.0761, F1-Score: 0.1288
Autoencoder - Accuracy: 0.7300, Precision: 0.5600, Recall: 0.1014, F1-Score: 0.1718
LSTM - Accuracy: 0.7260, Precision: 0.5200, Recall: 0.0942, F1-Score: 0.1595
One-Class SVM - Accuracy: 0.7240, Precision: 0.5000, Recall: 0.2246, F1-Score: 0.3100
LOF - Accuracy: 0.7240, Precision: 0.5000, Recall: 0.0906, F1-Score: 0.1534
```

```
[ ] # Model performance summary as DataFrame
    results = []

    for model_name, (y_t, y_p) in {**models_supervised, **models_unsup}.items():
        acc = accuracy_score(y_t, y_p)
        prec = precision_score(y_t, y_p, zero_division=0)
        rec = recall_score(y_t, y_p, zero_division=0)
        f1 = f1_score(y_t, y_p, zero_division=0)
        results.append([model_name, acc, prec, rec, f1])

    summary_df = pd.DataFrame(results, columns=["Model", "Accuracy", "Precision", "Recall", "F1-Score"])
    summary_df = summary_df.sort_values(by="F1-Score", ascending=False)
```

Appendix 1 Other Tables

Following is some of the supportive tables from the implementation and results to support the arguments that the performance metrics of supervised models, specifically the RF has achieved the highest accuracy.

Model Type Categorization

Model	Category
Random Forest	Supervised
XGBoost	Supervised
LightGBM	Supervised
KNN	Supervised
Autoencoder	Deep Learning
LSTM	Deep Learning
Isolation Forest	Unsupervised
One-Class SVM	Unsupervised
LOF	Unsupervised

Confusion Matrix – KNN

	Predicted Normal	**Predicted Anomaly**
Actual Normal	458	266
Actual Anomaly	63	661

Confusion Matrix – Autoencoder

	Predicted Normal	**Predicted Anomaly**
Actual Normal	702	22
Actual Anomaly	248	28

F1-Score Comparison

Model	F1-Score
Random Forest	0.9965
XGBoost	0.9938
LightGBM	0.9938

Model	F1-Score
KNN	0.8007
One-Class SVM	0.3100
Autoencoder	0.1718
LSTM	0.1595
LOF	0.1534
Isolation Forest	0.1288

Recall Comparison

Model	Recall
Random Forest	0.9931
XGBoost	0.9917
LightGBM	0.9917
KNN	0.9130
One-Class SVM	0.2246
Autoencoder	0.1014
LSTM	0.0942
LOF	0.0906
Isolation Forest	0.0761

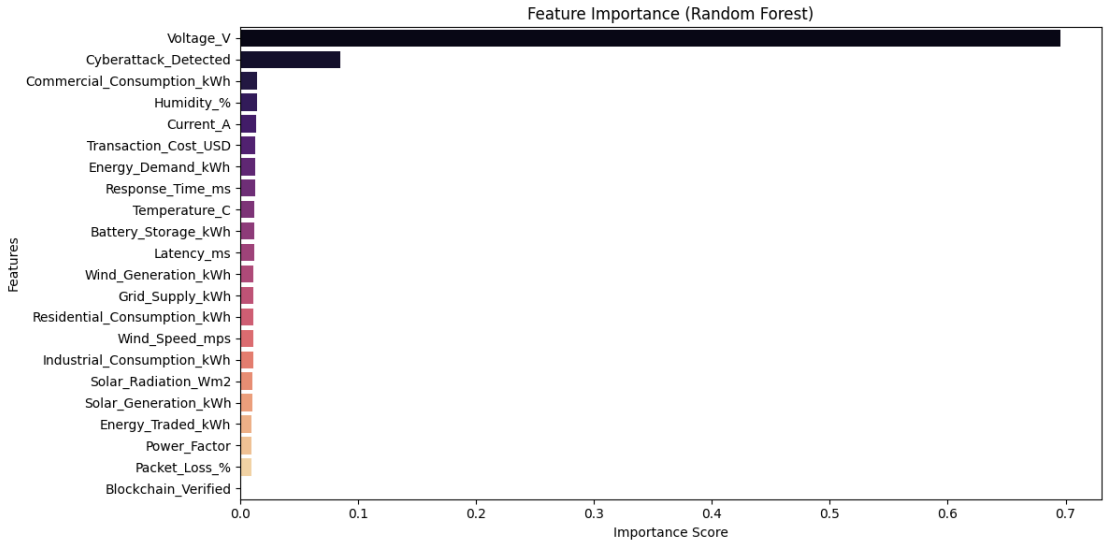
Precision Comparison

Model	Precision
Random Forest	1.0000
XGBoost	0.9958
LightGBM	0.9958
KNN	0.7131
Autoencoder	0.5600
LSTM	0.5200
One-Class SVM	0.5000
LOF	0.5000
Isolation Forest	0.4200

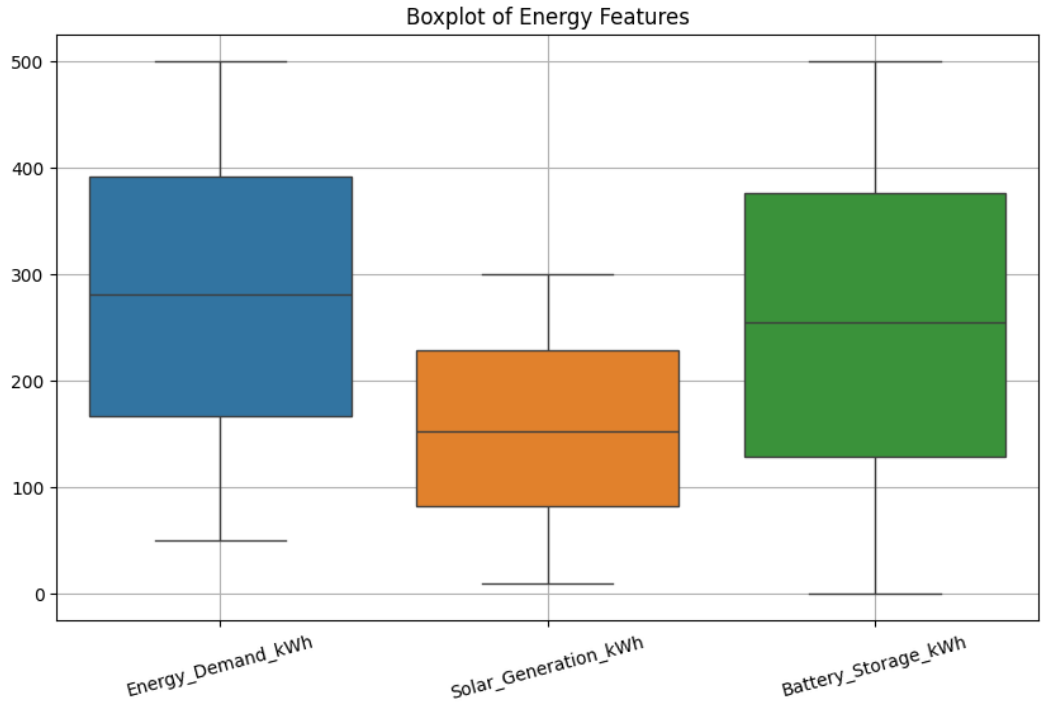
Appendix 2 Some Other Figures

Following is some of the figures from the implementation and results to support the arguments and justifications that the supervised models and specially the RF has outperformed and gave the highest accuracy.

Feature Importance in terms of random forest



Energy features in data



Anomaly Distribution in data