ENHANCED POWER GRID PREDICTION USING VARIOUS MACHINE-LEARNING TECHNIQUES

A PROJECT

Submitted by

(42) Anusree Mannathuparambu Satheeshkumar	CB.EN.U4EEE23104
(45) Nivetha R.	CB.EN.U4EEE23122
(46) Ponabirami A.	CB.EN.U4EEE23126
(47) Samyugtha J.	CB.EN.U4EEE23154
(48) Harshita V. P.	CB.EN.U4EEE23155

in partial fulfilment for the award of the degree of

MINOR IN
ARTIFICIAL INTELLIGENCE AND
MACHINE LEARNING
(Python For AI - 23AIE232M)

AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

AMRITA SCHOOL OF ENGINEERING
AMRITA VISHWA VIDYAPEETHAM
COIMBATORE - 641 112 (INDIA)
April - 2025

AMRITA SCHOOL OF ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641 112

BONAFIDE CERTIFICATE

This is to certify that the project entitled **“Enhanced Power Grid Prediction using various Machine Learning Techniques”** submitted by:

(42) Anusree Mannathuparambu Satheeshkumar	CB.EN.U4EEE23104
(45) Nivetha R.	CB.EN.U4EEE23122
(46) Ponabirami A.	CB.EN.U4EEE23126
(47) Samyugtha J.	CB.EN.U4EEE23154
(48) Harshita V. P.	CB.EN.U4EEE23155

for the award of the **Degree of Minor** in **“ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING (23AIE232M)”** is a bonafide record of the work carried out by them under my guidance and supervision at Amrita School of Artificial Intelligence, Coimbatore.

Chandni M
Project Guide
Assistant Professor

Submitted for the university examination held on 25/04/2025

INTERNAL EXAMINER **EXTERNAL EXAMINER**

AMRITA SCHOOL OF ENGINEERING
AMRITA VISHWA VIDYAPEETHAM
COIMBATORE - 641 112

DECLARATION

We, Anusree Mannathuparambu Satheeshkumar (CB.EN.U4EEE23104), Nivetha R. (CB.EN.U4EEE23122), Ponabirami A. (CB.EN.U4EEE23126), Samyugtha J. (CB.EN.U4EEE23154) and Harshita V. P. (CB.EN.U4EEE23155) hereby declare that this thesis entitled **"Enhanced Power Grid Prediction using various Machine Learning Techniques"**, is the record of the original work done by me under the guidance of **Chandni M**, Assistant professor, Amrita School of AI, Coimbatore. To the best of my knowledge this work has not formed the basis for the award of any degree/diploma/ associateship/fellowship/or a similar award to any candidate in any University.

Date: 25-04-2025

Dr. K.P.Soman
Professor and Head
Amrita School of AI

Contents

Acknowledgements……………………………………………………………………...I

List of Figures……………………………………………………………………….II

List of Tables………………………………………………………………….……..III

Abstract……………………………………………………………………..……...IV

1 Introduction………………………………………………………………………..1

1.1 Background………………………………………………………………………..1

1.2 Problem Statement………………………………………………………………..1

1.3 Objectives…………………………………………………………………………2

1.4 Paper Organization………………………………………………………………2

2 Literature Review…………………………………………………………………3

2.1 Related Works……………………………………………...……..……………3

2.2 Research Gaps……………………………………………………………………4

2.3 Improvements…………………………………………………………………..5

3 Technologies and Libraries Used…………………………………………………6

3.1 Variables and Control Structures………………………………………………..6

3.2 Database Integration……………………………………………………..………..7

3.3 Libraries Used…………………………………………………………………..7

4 Methodology…………………………………………………..…………………9

4.1 Data Collection……………….…………………………………………..……..9

4.1.1 Dataset Source……………………………………………………………………9

4.1.2 Data Cleaning & preprocessing………………………………………………..10

4.2 Exploratory Data Analysis…………………………………….……………...11

4.2.1 Target Variable Distribution Analysis……………………………………………11

4.2.2 Correlation Analysis……………………………………………………………….12

4.3 Feature Engineering………………………………………………………………..13

4.3.1 Feature Transformation……………………………………………………..13

4.3.1 Feature Selection……………………………………………………………….14

4.3.1 Dimensionality Reduction………………………………..………………..14

4.4 Model Selection……………………………………………………………………..15

4.5 Proposed Model……………………………………………………………………..16

5 Results and Future Works……………………………………………..…………...17

5.1 Results and Cross-Validation……………………………………………..……..17

5.1.1 Hyperparameter tuning results………………….……………………..…………17

5.1.2 Performance Metrics Analysis……………………..…………………………18

5.1.3 Confusion Matrix………………………………..…………………………..19

5.1.4 AUC-ROC Curve………………...……………………………………………20

5.2 Limitations…………………………………………………………..……………22

5.3 Future Works………………………………………………………………………22

6 Conclusion……………………………………………………………………………23

6.1 Challenges Faced…………………………………………………………………23

6.2 Conclusion…………………………………………………………………………...23

7 References…………………………………………………………………………24

Acknowledgement

We would like to express our wholehearted thank you to everyone who helped us finish this project successfully. Firstly, I would like to thank our faculty, Chandini M, for her advices and support, which was very useful for us to proceed with the project. And next, we would like to express our gratitude to our college, Amrita Vishwa Vidyapeetham, for providing us the knowledge and resources to complete this project. We would also like to thank our team members Anusree Mannathuparambu Satheeshkumar, Nivetha R, Samyugtha J, Harshita V P, and Ponabirami A for their exceptional collaboration, effort, and dedication to the project's success. We have had excellent conversations and a great working relationship. Additionally, I want to thank other faculty members.This project has been a valuable learning experience, and we are truly thankful to everyone who made it possible to complete the project successfully.

List of Figures

Figure 4.1 Data Source……………………………………………………………..…...9

Figure 4.2 Target Variable Distribution……….……………………………………..….11

Figure 4.3 Correlation Heatmap of Top 10 Most Correlated Features………………..…12

Figure 5.1 Comparison of metrics for each model…………………………….……..19

Figure 5.2 Confusion matrix…………………………………………………………..…20

Figure 5.3 ROC-AUC Curves……………………………………………………………21

List of Tables

TABLE 5.1 Hyperparameter Tuning results…………….…………………………………17

TABLE 5.2 Performance Metrics of each model…………………………………………..18

TABLE 5.3 Confusion Matrix Analysis………….…………………………………………21

Abstract

This study aims to develop a machine learning based model for predictive analysis and anomaly detection in power grids . With increasing complexity of power grids due to renewable energy sources and cyber threats, there is a need for smarter solutions. The project uses a real-world dataset that includes both natural disturbances and cyber attack status. The dataset was fully processed using cleaning, transformation, feature selection and feature reduction. Missing data is handled using median imputation and the data is normalized using Yeo-Johnson Power Transformation. The most relevant features were obtained using ANOVA F-test and Principal Component Analysis was used for dimensionality reduction while preserving 95% variance. To address class imbalance between 'Natural' and 'Attack' events, an oversampling technique, ADASYN was used. The study compares four advanced machine learning models - Decision Tree, Random Forest, XGBoost and LightGBM, optimized using Hyperparameter tuning. Among the models, LightGBM performs best achieving the highest accuracy of 94.47%, F1-score of 0.9663, ROC-AUC score of 0.9759 proving its prominent efficiency in generalization and precision. Cross-validation technique ensure the reliability of the model. Additionally, the project integrated SQL for storing and retrieving data making it suitable for real-time application. Overall, this study supports the the development of proactive anomaly detection systems, ensuring more secure and efficient power grids.

Keywords: Machine learning, Power grids, Boosting techniques, Anomaly detection, Cyber security, Feature engineering.

Chapter 1
Introduction

1.1 Background

In the modern era, power grids have become complex because of growing energy requirements, integration of renewable sources, and changing cyber threats. Classical electrical grid infrastructures were built for steady and predictable supply-demand conditions and hence they find it hard to cope with this evolution. Such infrastructures tend to be slow in reacting to abnormalities or disruptions and end up being inefficient, resulting in massive blackouts [2], [3]. The advent of smart grids and the use of Machine Learning (ML) have provided new opportunities for better ways of monitoring, controlling, and managing energy systems. Machine learning allows analyzing huge volumes of sensor and control unit data across the grid and detecting patterns indicating equipment faults, unusual operation, or even cyberattacks. As opposed to the conventional systems operating on pre-defined rules, ML algorithms learn from examples and change over time and, therefore, are better appropriate in dynamic and uncertain situations [1], [4].

Researchers have already shown the applicability of ML in some fields of power systems like load forecasting, voltage stability, and detection of energy theft [4], [5], [6]. Fewer studies have been conducted on the detection of important events like attacks or grid disturbances in real time. Further, most existing systems are still reactive—they detect issues after they have happened. This necessitates the creation of predictive models that will detect threats early enough and facilitate proactive grid management [7], [8].

The present project is found on these developing requirements. It uses real-world datasets and contrasts different machine learning models—Decision Trees, Random Forests, XGBoost, and LightGBM to find the most effective method for forecasting abnormal events in the power grid. Through this we could construct a smarter and more secure energy system that can not only detect but also forecast disruptions prior to their impact on operations.

1.2 Problem statement

Power grid are collapsed or disturbed and due to natural occurrences, technical issues, or cyberattacks. They can cause service disruptions, economic losses, and even public safety concerns. The majority of current detection systems are rule-based or they rely on human intervention, due to which they restrict their capacity to identify subtle or unforeseen anomalies in real time [9], [10].

In addition, most systems have poor utilization of historical data and do not incorporate optimized machine learning methods that have the potential to enhance performance. Moreover, most systems lack an ability to sense various kinds of events, physical as well as cyber-related, by employing one robust predictive model. Hence, a data-driven intelligent system for identifying power grid anomalies and attacks is an essential need.

1.3 Objectives

The main aim of this project is to develop a machine learning-driven framework to enhance anomaly detection in power grid systems.

- ❖ Developing predictive models from real power grid datasets to flag events as normal or abnormal.
- ❖ Comparing some supervised Machine learning algorithm that include Decision Tree, Random Forest, XGBoost, and LightGBM to determine the best model.
- ❖ Utilizing methods like ADASYN for balancing classes, Yeo-Johnson transformation for normalization, and PCA and ANOVA F-test for dimensionality reduction and feature selection.
- ❖ Using cross-validation and hyperparameter tuning to make the models well-tuned and generalize across datasets.
- ❖ Measuring model performance using multiple metrics like accuracy, precision, recall, F1-score, and ROC-AUC to make sure the model generalizes well across all prediction aspects.

This project helps in the development of a smarter, more robust, and proactive power grid monitoring system that can trigger timely warnings and minimize the effects of possible threats.

1.4 Paper Organization

This report follows the following organization:

•Chapter 2 offers an extensive literature review of the current research on machine learning applications in power grid systems and an overview of the gaps that this project seeks to address.

•Chapter 3 outlines the technologies, libraries, and tools used to develop the model.

•Chapter 4 outlines the procedure, that is, data preprocessing, model training, and the evaluation.

•Chapter 5 presents the outcomes of various models, comparison and findings using graphs and parameters.

•Chapter 6 concludes the report with an overview of the important findings, the problems encountered, and the future improvement recommendations.

Chapter 2
Literature Review

2.1 Related Works

Over the last few years, numerous researchers have explored how machine learning can enhance power grid efficiency, particularly fault detection, outages prediction, and system reliability.One of the early research works [8] was aimed at applying machine learning models for fault detection in the New York City power grid. Their work showed that early detection of potential failures using machine learning could reduce downtime and improve grid responsiveness. However, this study mainly used basic models and lacked ensemble methods that could further boost performance.

[4] made a general survey of machine learning applications for power system analysis. Their study quoted how the implementation of Machine learning algorithms would prove beneficial in load forecasting, voltage regulation, and power quality observation. While they gave a brief overview of applications, the study did not examine anomaly detection thoroughly or model comparison.

[6] compared the evolving trends of smart grids with machine learning and big data. Their comparison showed that machine learning helps in managing vast amounts of energy data, which can be utilized for predictive maintenance and load optimization.They did not consider analyzing how these algorithms behave in the event of attacks or faults, which are very important for grid security.

[5] investigated the application of ML with renewable energy sources and how it can be used to enhance grid stability. Although this research was aimed at grid sustainability, it did not consider system security or fault detection in real-time.

[11] and [12] proposed deep reinforcement learning and optimal power flow models for power grids. These models are able to learn control strategies over time, and their long-term performance is improved. Their strategies tend to be computationally complex and may not be suitable for real-time decision-making in real grid settings.

[1], in their recent work, experimented and compared different machine learning algorithms like SVM, ANN, RNN, and Decision Trees. They applied these models over a simulated dataset of natural events and cyberattacks on power grids. Their results showed that Support Vector Machines gave the maximum accuracy and recall. Although this piece of work did contribute significantly, the study didn't explore more recent ensemble methods like XGBoost or LightGBM, which have better performed on classification problems.

There also exist research works on fault classification with time-series data, in which recurrent neural networks (RNNs) are employed. RNNs are capable of detecting patterns in time, but are generally slower to train and more difficult to understand, which could hinder their use in real time [13]. There has been an attempt to use IoT for real-time monitoring of the grid parameters

[7], but such systems are reactive rather than predictive. They trigger alarms when a fault is found, not avoiding it.

In short, while there have been many attempts in research to apply machine learning to power grid analysis, most just try to do prediction without consideration for anomalies, or are restricted to certain uses like load forecasting or voltage stability.

2.2 Research Gaps

Despite the widespread acceptance of machine learning into power grid systems, there remain a number of important knowledge gaps in the literature:

- Limited application of boosting and ensemble techniques

The earlier work largely utilized simple models such as SVMs, ANNs, and Decision Trees for classification.It is only that few studies explored ensemble methods or boosting methods such as XGBoost and LightGBM, which have been lauded for their excellent performance, particularly for handling complex, nonlinear patterns in grid data.

- Underexploited Anomaly and Attack Detection

Despite the widespread deployment of machine learning to load forecasting, power quality analysis, and fault detection tasks, relatively little research has dealt with predictive anomaly detection or the classification of cyberattacks. Most systems, however, only identify faults once they have been executed, and even then they fail to possess a proactive strategy anticipating disturbances before it is too late.

- Insufficiency in Highlighting Real-Time Adaptability and Security

Most of the available models are not intended for real-time use or do not address the dynamic nature of threats in smart grids. There is scant research on how machine learning models react under cyberattack conditions or severe grid disruptions—factors that are central to system resilience.

- Inefficient Handling of Class Imbalance and High-Dimensional Data

Power grid datasets are usually dominated by extreme class imbalance, such as between attack and normal events. Most recent work has avoided this problem or used naive strategies. Likewise, the high-dimensionality of relay and PMU data hasn't been tackled appropriately, and it affects the interpretability and efficiency of models.

- Reactive use of IoT without predictive integration

There have been some studies that have used IoT for real-time monitoring, but these systems remain mostly reactive, only sending alarms after failure. The capability to leverage IoT data with predictive machine learning models is still not being utilized to its full potential.

Briefly stated, although numerous research attempts have been made to implement machine learning for analysis in the power grid, most only attempt prediction without regard to anomalies, or are limited to specific applications such as load forecasting or voltage stability.

2.3 Improvements

This work overcomes most of the drawbacks present in earlier studies. For starters, rather than employing a single machine learning model, we employ multiple models such as Decision Tree, Random Forest, XGBoost, and LightGBM. These models have been renowned for their excellent performance in classification issues and are more appropriate to deal with intricate, nonlinear relationships present in power grid data.

In order to address the issue of class imbalance, typically overlooked in older research, the ADASYN technique is employed, which generates synthetic data points for the minority class. It prevents the model from becoming biased towards the majority class, usually the case with attack data that is scarce or available in short numbers.

We also utilize Power Transformation (Yeo-Johnson) to handle non-normal and skewed feature distributions that impact model training. In addition, we use ANOVA F-test for feature selection and PCA for reducing dimensionality, which minimizes noise and maximizes training speed.

Unlike previous studies that employed default model parameter settings, our method employs hyperparameter tuning through GridSearchCV, making every model optimal. We also employ various metrics for assessment such as accuracy, precision, recall, F1-score, and ROC-AUC to analyze the performance of the model as a whole and pick the best one.

Another major improvement is that our project involves predictive anomaly detection rather than real-time monitoring. This means that the system can alert operators of potential dangers in advance before they happen, so the system is proactive, not reactive.

Chapter 3
Technologies and Libraries Used

3.1 Variables and Control Structures

In this project, variables are used extensively to store and process data at every stage of the machine-learning pipeline. The variables are named descriptively to represent the type of data or operation being performed, which improves code readability and maintainability.
Some key examples include:

- **Data Storage Variables**:
    - df: Stores the raw dataset loaded from a CSV file using Pandas.
    - X, y: Represent the input features and target labels respectively. X is obtained by slicing the dataset to exclude the target column, while y is computed using NumPy to convert categorical labels into binary values.
    - X_train, X_test, y_train, y_test: These hold the split versions of the data used for training and testing
    - X_res, y_res: Contain the oversampled training data using ADASYN.
- **Variables under Preprocessing and Transformation**:
    - transformer: A PowerTransformer object is used to normalize the feature values.
    - X_transformed, X_selected, X_pca: Sequentially store the output of transformed, selected, and PCA-reduced features.
- **Variables under Model and Evaluation Variables**:
    - param_grid: A dictionary holding the hyperparameters to be tested during grid search.
    - grid_search: Stores the GridSearchCV object responsible for hyperparameter tuning.
    - best_model variables like best_dt, best_rf, best_xgb, and best_lgbm: These store the best-performing model obtained from grid search.
    - y_pred, y_proba: Hold the model's predictions and predicted probabilities for the test data.
    - Evaluation metrics like accuracy, precision, recall, f1, and roc_value are computed and stored for reporting model performance.

These variables facilitate smooth data flow and modular execution of the ML workflow, from data loading and transformation to model training, evaluation, and visualization.

Control structures are used in the project primarily for flow control, configuration, and parameter testing. The incorporates the following Python control structures.

This code does not make use of explicit control structures such as if statements, for or while loops. Instead, it relies heavily on high-level functions provided by Python libraries like Scikit-learn, Pandas, and Imbalanced-learn. These libraries abstract away the need for manual iteration or conditional branching by offering optimized built-in methods. For example, operations like grid search (GridSearchCV), PCA transformation (pca.fit_transform()), oversampling (adasyn.fit_resample()), and evaluation metrics (accuracy_score, roc_auc_score, etc.) internally handle control flow and computations. As a result, the code remains compact, modular, and efficient without requiring traditional Python control structures.

3.2 Database Integration

Key aspects The integration primarily uses SQLite, a lightweight and self-contained relational database engine. Python's SQLAlchemy library is utilized to create and manage the database connection, offering an easy interface to interact with SQL databases using Python code. After preprocessing the dataset — including the replacement of infinite values, handling of missing data, and conversion of the target variable into binary format — the cleaned dataset is stored into the SQL database under the table name original_data. This step ensures that the raw processed form of the data is persistently saved and can be accessed at any point for validation or further analysis.

Following the transformation of features using the Yeo-Johnson method, selection of top features with ANOVA F-value, and dimensionality reduction using Principal Component Analysis (PCA), the resulting dataset is stored in another SQL table named pca_data. This table contains the final set of principal components along with the binary target column.

The stored PCA dataset is later retrieved from the pca_data table using a SQL SELECT query. This simulates a common scenario in production systems where model training or evaluation is performed on data pulled from a database.

Incorporating SQL provides multiple benefits. First, it introduces data persistence, allowing the processed data to be saved beyond the runtime of the program. Second, it lays the foundation for scalable deployment, where datasets and models can be integrated into larger data architectures. Third, it promotes modular design, enabling components of the pipeline (such as feature engineering or model evaluation) to work independently with SQL-stored data. This integration also allows easy extension to include full CRUD operations (Create, Read, Update, Delete), offering flexibility in dynamic data workflows.

3.3 Libraries used

- **NumPy** was used for high-performance mathematical and array computations
- **Pandas** enabled efficient loading, transformation, and cleaning of tabular data.
- **Matplotlib and Seaborn** helped in visualizing key aspects of the model, such as performance metrics (confusion matrix, ROC curve) and PCA-explained variance.

- **Scikit-learn** formed the core of the machine learning pipeline, supporting feature selection, dimensionality reduction, model building, hyperparameter tuning, and evaluation.
- **Imbalanced learning (ADASYN)** was used to balance the dataset, mitigating bias toward the majority class.

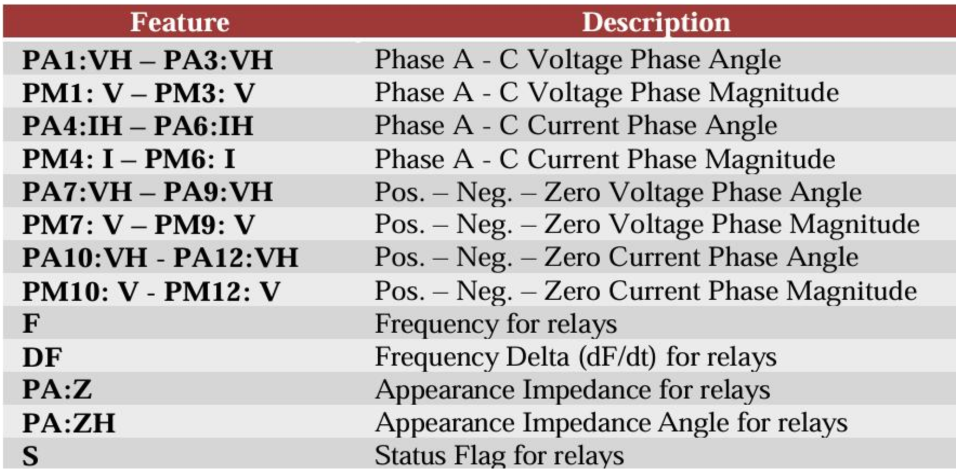
Chapter 4
Methodology

4.1. Data Collection

4.1.1 Dataset Source

The dataset used in this project, titled data1.csv, is a binary classification subset derived from a comprehensive power system simulation dataset developed by Mississippi State University and Oak Ridge National Laboratory, dated April 15, 2014. This dataset was specifically designed to simulate various real-world scenarios in a smart power grid, including natural faults, maintenance activities, normal operations, and a range of cyber-attacks on intelligent grid infrastructure. Originally, the full dataset contained fifteen sets, each comprising 37 distinct event scenarios categorized as Natural Events (8 scenarios), No Events (1 scenario), and Attack Events (28 scenarios). These were further grouped into binary, three-class, and multiclass datasets and made available in both ARFF and CSV formats for compatibility with tools Python. For this project, a binary classification version was used, where each row in the dataset represents one power system event, labelled as either a Natural Event (0) or an Attack Event (1). The dataset consists of 128 columns:

- The first 116 columns contain real-time measurements collected from Phasor Measurement Units (PMUs). These include values such as voltage magnitude and angle, current magnitude and angle, frequency, and impedance, captured from four PMU devices (R1–R4) deployed across different parts of the grid.
- The next 12 columns contain control and relay logs, including breaker status, Snort alerts, and relay commands, which help contextualize operational behaviour during each event.
- The final column is the target label, indicating the nature of the event — either a legitimate (natural) scenario or a malicious (attack) event.



Feature	Description
PA1:VH – PA3:VH	Phase A - C Voltage Phase Angle
PM1: V – PM3: V	Phase A - C Voltage Phase Magnitude
PA4:IH – PA6:IH	Phase A - C Current Phase Angle
PM4: I – PM6: I	Phase A - C Current Phase Magnitude
PA7:VH – PA9:VH	Pos. – Neg. – Zero Voltage Phase Angle
PM7: V – PM9: V	Pos. – Neg. – Zero Voltage Phase Magnitude
PA10:VH - PA12:VH	Pos. – Neg. – Zero Current Phase Angle
PM10: V - PM12: V	Pos. – Neg. – Zero Current Phase Magnitude
F	Frequency for relays
DF	Frequency Delta (dF/dt) for relays
PA:Z	Appearance Impedance for relays
PA:ZH	Appearance Impedance Angle for relays
S	Status Flag for relays

Figure 4.1 Data Source

This data captures a wide spectrum of power system behaviour under both normal and compromised conditions, simulating a real smart grid environment. The combination of electrical measurements and control logs provides a rich feature set for training machine learning models to accurately classify and predict system anomalies or cyber-attacks.

4.1.2 Data Cleaning & preprocessing

To prepare the dataset for effective machine learning model training, a comprehensive data cleaning and preprocessing pipeline was implemented. The purpose of this step was to handle any irregularities in the data, standardize the feature distributions, reduce dimensionality, and retain only the most informative features to ensure efficient and accurate model performance.

Handling Missing and Infinite Values -The raw dataset included several instances of infinite values, which typically arise from division errors or invalid sensor readings in power systems. These were first converted to missing values NaN to flag them for cleaning. After this, all missing values were imputed using the median of each respective feature column. The median was selected over the mean as an imputation strategy due to its robustness against outliers, which are common in electrical grid measurements such as voltage or current spikes. This step ensured numerical consistency across the dataset and prevented computational errors during model training.

Normalization using Power Transformation- After addressing missing and infinite values, the dataset was normalized using the PowerTransformer from Scikit-learn, configured with the Yeo-Johnson method. This transformation method was chosen because it can handle both positive and negative values while making the feature distributions more Gaussian-like. The Yeo-Johnson transformation stabilizes the variance and reduces skewness in the data, which is beneficial for many machine learning models that assume normally distributed inputs. This normalization step enhanced the reliability and performance of subsequent models, particularly tree-based classifiers and ensemble methods.

Target Label Conversion- As part of the preprocessing pipeline, the original dataset contained categorical labels in the last column, identifying each scenario as either a “Natural” event or an “Attack.” Since most machine learning algorithms require numerical inputs, the categorical labels were converted into binary numeric values. This was accomplished using NumPy's where() function, where the label "Natural" was mapped to 0 and all other event types (representing attacks) were mapped to. This transformation enabled the use of classification algorithms that expect numeric target variables.

4.2 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a vital first step in any machine learning project. It is used to understand the dataset's patterns, relationships and anomalies before proceeding to the feature engineering and model selection. This project is based on power grid cyber attack prediction, where EDA helps in the following ways:

1. Data understanding: EDA aids in comprehending the structure of feature columns and how the values are distributed in them. We can identify the missing values, anomalies and outliers that could affect the model prediction and performance.
2. Class Imbalance Handling: The dataset used in the project is based on the cyber-attacks on power grids. The chances of an 'Attack' are generally very rare and thus the dataset may not have a balance in the 'Natural' and 'Attack' data points. EDA determines if there are any imbalances in the target and gives insights about the requirements of oversampling or undersampling techniques.
3. Feature Behaviour Analysis: A correlation matrix can be generated to identify discriminatory features and understand the relationship between different pairs of features. Distribution plots can also be used to determine the statistical distribution of the measured values.

The power grid data is high-dimensional and complex. Employing EDA simplifies this complexity by highlighting the most relevant features for attack detection. The following are some of the analyses done on the dataset:

4.2.1. Target Variable Distribution Analysis

The bar graph (Figure) illustrates the distribution of the target variable in the dataset. It shows how the power grid observation data classifies the 'Natural' and 'Attack' events. It is very clear from the graph that the data points for the 'Attack' event are much higher than the 'Natural' data. There is a severe class imbalance present in the target feature. About 85% of the data points refer to the 'Attack' condition, while the 'Natural' conditions are very minimal. This implies that the dataset was collected such as for the designed model to understand the trend in the 'Attack' conditions, which are very rare in real life. However, the imbalance may affect the model prediction by detecting False Positives (FP), i.e. there may be unnecessary indications of attack on the power system. This emphasises that there is a requirement to implement a Class Balancing technique to equally prioritize both classes.

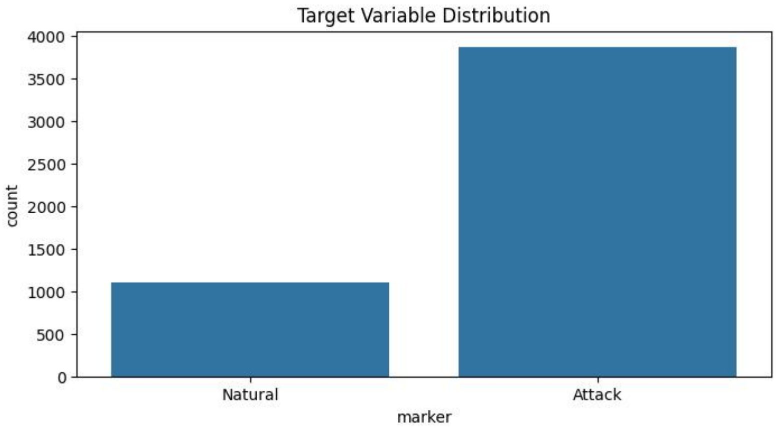


Figure 4.2: Target Variable Distribution

4.2.2. Correlation Analysis

In data science, we are majorly interested in understanding the relationships between various features in the dataset. This can be done with the help of a correlation matrix. A correlation matrix shows how strongly or weakly pairs of features are related to each other. The Figure illustrates such a correlation heatmap where we could observe the positive and negative association between the top 10 highly correlated features in the power grid dataset. These features consist of voltage/current harmonics and phasor measurements from different PMUs (R1-R4). There are strong positive correlations between R3-PM10:I and R3-PM6:I, which exhibits a near-perfect relation between current magnitudes. On the other hand, R1-PA6:IH and R2-PA6:IH are anti-phase current harmonics that show a very weak relationship. We could also observe moderate correlations between voltage harmonics (R3-PA3:VH, R2-PA3:VH) and current harmonics like R2-PA10:IH. Certain decisions can be made based on the correlation heatmap, which includes the following:

1. Drop unnecessary feature columns: Some feature pairs have a correlation coefficient of 1.0, either of which could be dropped to reduce dimensionality (eg. R3-PA3:VH and R2-PA3:VH, R3-PM10:I and R3-PM6:I). Thus, to eliminate redundancy, we can apply feature selection methods like Principal Component Analysis (PCA) or even manual removal can be done.
2. Attack-Detection Signals: Anti-correlated features like R1-PA6:IH and R2-PA6:IH may indicate cyber-attacks or could serve as noise filters.

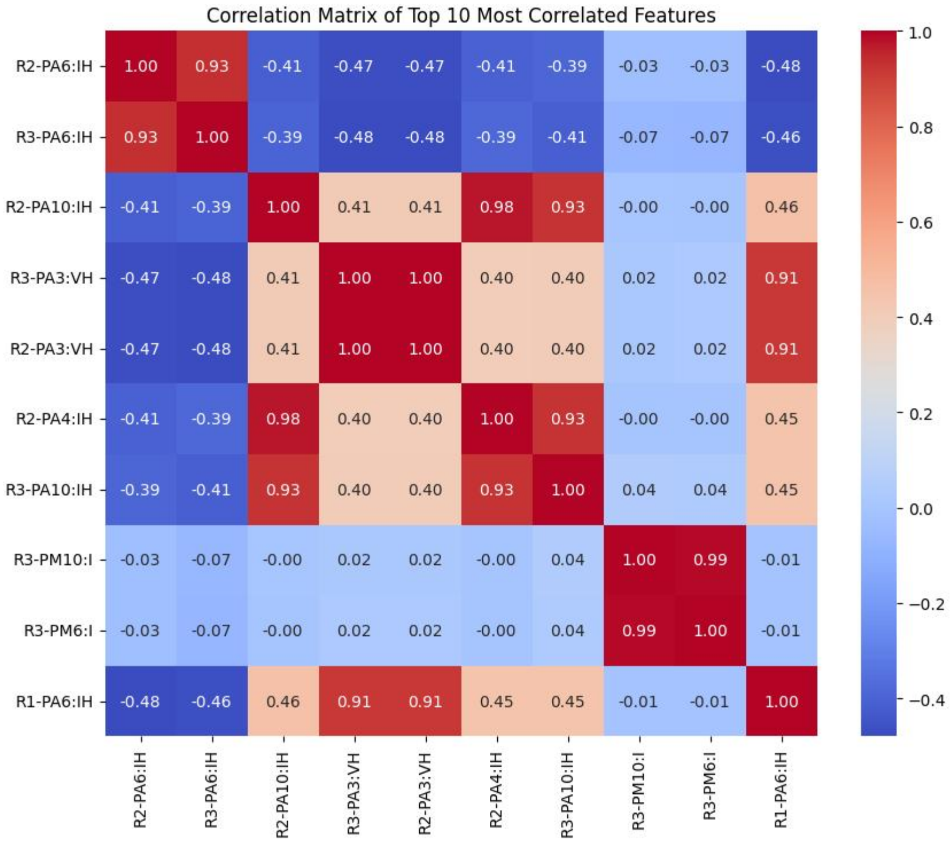


Figure 4.3: Correlation Heatmap of Top 10 Most Correlated Features

4.3 Feature Engineering

Feature Engineering refers to the process of creating new features or transforming existing features to improve the performance of a machine-learning model. This includes selecting, extracting and transforming the most relevant features from the available data to build more accurate and efficient machine learning models. Power grids are increasingly targeted by cyber attacks that manipulate sensor data (eg. PMU measurements) or relay commands to cause large-scale blackouts. Thus, the following feature engineering techniques are used to improve the model performance:

4.3.1 Feature Transformation

Phasor Measurement Unit (PMU) data in power systems often exhibits non-Gaussian distributions, particularly for measurements such as current magnitude, voltage phase angles and frequency deltas. Many machine learning algorithms, including LighGBM, perform optimally when input features are normally disturbed or at least symmetrically disturbed. Skewed feature distributions can lead to suboptimal model convergence, reduced sensitivity to subtle attack signatures and biased feature importance calculations in tree-based models. This project uses Yeo-Johnson power transformation which is defined by:

$$f(x;\lambda) = \begin{cases} ((1+x)^{\lambda} - 1)/\lambda & for\ \lambda \neq 0,\ x \geq 0 \\ \log(x+1) & for\ \lambda = 0,\ x \geq 0 \\ -\dfrac{(1-x)^{2-\lambda} - 1}{2-\lambda} & for\ \lambda \neq 2,\ x < 0 \\ -\log(-x+1) & for\ \lambda = 2,\ x < 0 \end{cases}$$

The Yeo-Johnson Power transformation acts as a major step in the power grid cyber attack problem due to the following reasons:

1. Handling zero and negative values: Current phase angles (PA4:IH to PA6:IH), frequency deltas (DF) during fault conditions and Impedance measurements (PA:Z) are commonly found to be zero or negative values, which can be effectively managed by the power transformation method.
2. Managing skewness patterns: Some features, like the voltage magnitudes (PM1:V to PM3:V), are generally right-skewed during overloads, while PM7:V to PM12:V are often left-skewed during balanced conditions. This can lead to uneven weighting and suboptimal model convergence. The Yeo-Johnson method of transformation handles these issues.

The key benefits of this process include sensitivity enhancement, improved training stability and better feature discrimination. The method's ability to handle the diverse value ranges in the PMU data made it particularly suitable for this security application.

4.3.2 Feature Selection

The dataset chosen is high dimensional data and thus, it may take huge computational time to train and test the predictive model. There are 120+ features in the dataset from the PMUs and relays which makes it difficult for real-time processing. This project uses the ANOVA (Analysis of Variance) F-test to improve the statistical robustness and computational efficiency by choosing the 40 best-correlated features. ANOVA F-test is a statistical method used to determine whether there are significant differences between the means of two or more groups. The Null Hypothesis is used along with the F1-score calculation as mentioned below:

$$Fscore = \frac{Variation\ between\ groups}{Variation\ within\ groups} = \frac{MS(between)}{MS(within)}$$

The major reason why this feature selection helps in the model is that it helps in filtering irrelevant features and works effectively with continuous data. This process requires normally distributed data, which is accomplished in the previous feature transformation step. This method has helped in the enhancement of attack detection and better interpretability. ANOVA F-test also quantifies how well a feature separates classes and increases the F1-score along with decreased precision value. The method is thus highly used in power grid problems to select PMU/relay features that best detect cyber attacks.

4.3.3 Dimensionality Reduction

The dimensionality reduction technique used in this project is Principal Component Analysis (PCA). It is a statistical technique that transforms high-dimensional data into lower-dimensional space while preserving most of the original variability. It identifies principal components, which are new uncorrelated variables that are linear combinations of the original features. Below is the mathematical expression for the covariance matrix in PCA, from which the eigenvalues and vectors are determined::

$$cov_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N - 1}$$

PMUs generated 120+ features that caused the issue of high dimensionality. Out of all these features, a minimal number contributes to the target variable. Most features are redundant information carriers. There are various ways in which PCA helps in this project, which include:

1. Noise reduction: There exists a large number of features that have noise and irrelevant signal data. PCA helps in dropping low-variance features like the sensor noise. In the case of the dataset used, high-frequency noise in DF (frequency data) is an example, that is filtered using PCA.
2. Focus on Attack Signatures: There are several principal components created after the process of PCA. Each of these helps in capturing common features that affect the target similarly. For example, PC1 may often capture voltage/current anomalies while PC2 may represent frequency deviations.

3. Faster detection: The features are reduced from 128 to 12 principal components after all these feature engineering techniques. Training the model with 12 PCs is about 5 times faster than that with 128 features.

Thus, PCA is a feature engineering process that helps reduce the dimensions of the given dataset to improve the efficiency of model training and increase the accuracy and speed of implementation.

4.4 Model Selection

Selecting a model plays a crucial step in machine learning. The type of model highly affects its performance. Model selection includes identifying the most suitable algorithm or configuration based on the dataset's nature, the task's complexity and our desired outcome. Here the project is to develop a robust classification model that can perform in high-dimensional space and with the imbalanced dataset, for which several models were employed and the best among the models was identified. In this project, techniques like Decision tree, Random forest, XGBoost, and LightGBM were used. The final model is picked from that based on the performance of each model.

Initially, the dataset is cleaned and preprocessed with steps like declaring the target variable and clearing the null values. Then it is followed by feature selection using ANOVA F-value. Since the dataset is imbalanced, and if we are making a model with that, it might lead to biasing, so ANOVA F-value selects the relevant features by finding F value which is a ratio of variance with mean class and variance within each class. Based on these F values, the features are selected from the imbalanced dataset.

$$F = \frac{variance\ between\ class\ means}{variance\ within\ each\ class}$$

The selected features of the dataset are taken and it is split as training and test data (80%-20%). For handling the imbalance in the dataset ADASYN is used, which identifies the minority and majority classes and adds some synthetic points to the minority class. It does not add random points instead it finds the nearest point and interpolates them to get a new point this imbalance in the dataset can be taken care of. If the dataset is imbalanced then the model might have a large bias towards the majority class so it is important to balance the dataset. The data is first subjected to a Decision Tree here it works like a flow chart the algorithm tries all the data and makes all the possible splits and the split with the purest node is selected. This process stops when the max_length is reached. Here hyperparameter tuning is done to find the best parameter for the model.GridSearch is used to find the best parameter which tries with all the given points and find the most suitable one. After completing the model it is subjected to various evaluation metrics like accuracy, F1-score, recall, precision, and ROC-AUC score. Decision tree uses only a single tree that splits the data based on the feature so if the number of trees is increased we can further increase the performance of the model, which can be done using a Random forest.

Random Forest is an ensemble learning algorithm that uses multiple decision trees and combines their predictions to increase the accuracy of the model. It first creates several random subsets of the training data by a method called bootstrap sampling, where each tree is trained on a different part of the data. During the formation of each tree, a random subset of features is used as each split, which due to which it avoids the model from making the same mistakes. Each tree is done fully without pruning, and once all trees are done, their predictions are combined, using majority voting for classification tasks. The accuracy can be further increased by using XGBoost which builds trees sequentially, where each tree corrects the errors of the previous one, which makes it better than any other mode. It uses techniques like parallelisation and tree pruning to train models. It also supports distributed computing, allowing it to scale better across multiple machines. This can be further improvised by using LightGBM technique, this also uses parallel technique but it is done leafwise and also uses histogram bases technique due to which it is faster than other methods but also is high in efficiency. Hyperparameter tuning was done for all the models using GridSearch.

Among all the models evaluated, LightGBM achieved the highest accuracy, due to its efficient training, native handling of categorical features, and ability to scale well with large datasets.

4.5 Proposed Model

The study presents a machine-learning model using LightGBM for classification. The dataset was imported using pandas and preprocessed, where the target variable was converted to binary values (0 for Natural and 1 for Attack), and infinite or missing values were handled using median imputation. To improve the stability and the model’s performance, a Yeo-Johnson power transformation was used. Feature selection was performed using the ANOVA F-test, which retains the top 40 most relevant features seven with an unbalanced dataset. To reduce dimensionality, Principal Component Analysis (PCA) was applied, which preserved 95% of the original data variance. The dataset was then split into training and testing sets in an 80:20 ratio. To balance the class imbalance, ADASYN (Adaptive Synthetic Sampling) was used which creates synthetic samples for the minority class. An LGBM classifier was trained on the resampled data, hyperparameters were tuned using GridSearchCV and optimal values were found. Cross-validation was performed with 5-fold, and an ROC-AUC score was obtained. The best model was then evaluated on the test set using accuracy, precision, recall, F1-score, and ROC AUC, along with visualizations such as the confusion matrix and ROC curve. To assess the model’s generalization throughout,5-fold cross-validation was used. Finally, the feature importance of the principal components was analyzed and visualized, along with the explained variance ratio from PCA to understand the contribution of each component to the model.

Chapter 5
Results and Future Works

5.1 Results and Cross-Validation

To evaluate and optimize the performance of the classification model, hyperparameter tuning and cross-validation have been used. This section presents the optimal parameters derived and the obtained performance metrics for each model.

5.1.1 Hyperparameter tuning results

Hyperparameter tuning helps to improve model performance by adjusting parameters that influence the model structure. Key parameters in tree-based models are - max_depth which controls tree complexity and min_samples_split and min_samples_leaf or min_child_samples(in LightGBM) which regulates the node splitting to prevent overfitting, n estimators is the number of trees involved in the model training and learning_rate controls the contribution of each tree in adjusting the model. Sub_sample and colsample_bytree improve generalization by introducing randomness while num_leaves in LightGBM sets tree complexity. The best value obtained for each parameter that yields the maximum accuracy for each model is given below

TABLE 5.1 Hyperparameter Tuning results

MODEL	PARAMETERS	VALUES CHECKED	BEST VALUE	ACCURACY
Decision Tree	max_depth	5,10,20,None	None	89.44%
	min_samples_split	2,5,10	10	
	min_samples_leaf	1,2,4	2	
Random Forest	n_estimators	100,200	200	94.27%
	max_depth	10,20,None	None	
	min_samples_split	2,5	2	
	min_samples_leaf	1,2	1	
XGBoost	n_estimators	100,200	200	93.66%
	max_depth	3,6,9	9	
	learning_rate	0.01,0.1,0.2	0.2	
	subsample	0.6,0.8,1.0	1	
	colsample_bytree	0.6,0.8,1.0	1	
LightGBM	n_estimators	100,200	200	94.77%
	max_depth	10,20,-1	-1	
	min_child_samples	2,5	2	
	num_leaves	31,63	63	
	colsample_bytree	0.6,0.8,1.0	0.6	

Table 5.1 highlights the influence of hyperparameters on the tree-based model's performance. For all models, increasing the number of n_estimators(trees) improved performance. The max_depth when set to flexible values like None for RF or -1 for LightGBM, indicates no restriction on tree depth, allowing the models to grow deeper trees to capture more patterns. The relatively small values (2 & 1) for min_samples_split and min_samples_leaf or min_child_samples enable the model to make finer distinctions without overfitting. The moderate learning rate balances stability and speed of learning. The values of subsample and colsample_bytree (1) indicated all rows and features are used for every tree. The larger value of num_leaves(63) benefits LighGBM to further create a more complex tree, thereby helping to fit intricate data better.

5.1.2 Performance Metrics Analysis

TABLE 5.2 Performance Metrics of each model

MODEL	Accuracy	Precision	F1-Score	Recall	ROC-AUC	Cross-Validation Accuracy
Decision Tree + Hyperparameter tuning	0.8944	0.9601	0.9300	0.9018	0.8975	0.6172 (±0.0263)
Random Forest +Hyperparameter tuning	0.9427	0.9674	0.9630	0.9587	0.9818	0.6794 (±0.0289)
XGBoost +Hyperparameter tuning	0.9366	0.9709	0.9588	0.9470	0.9692	0.6508 (±0.0309)
LightGBM +Hyperparameter tuning	0.9477	0.9701	0.9663	0.9625	0.9759	0.6690 (±0.0299)

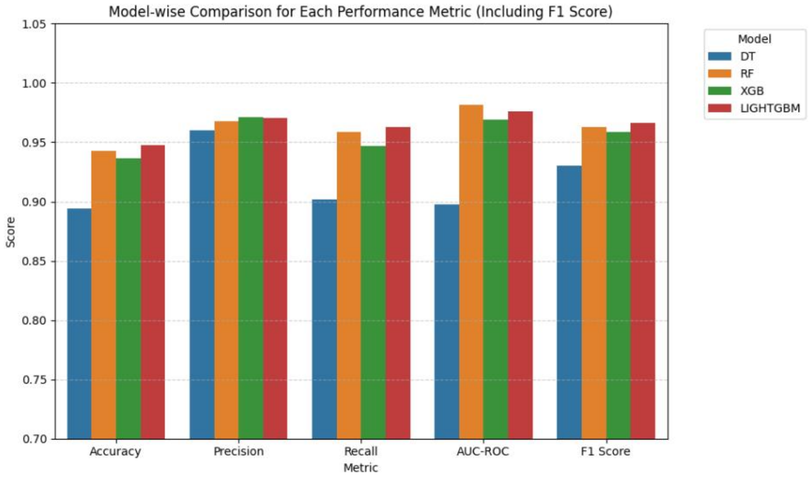


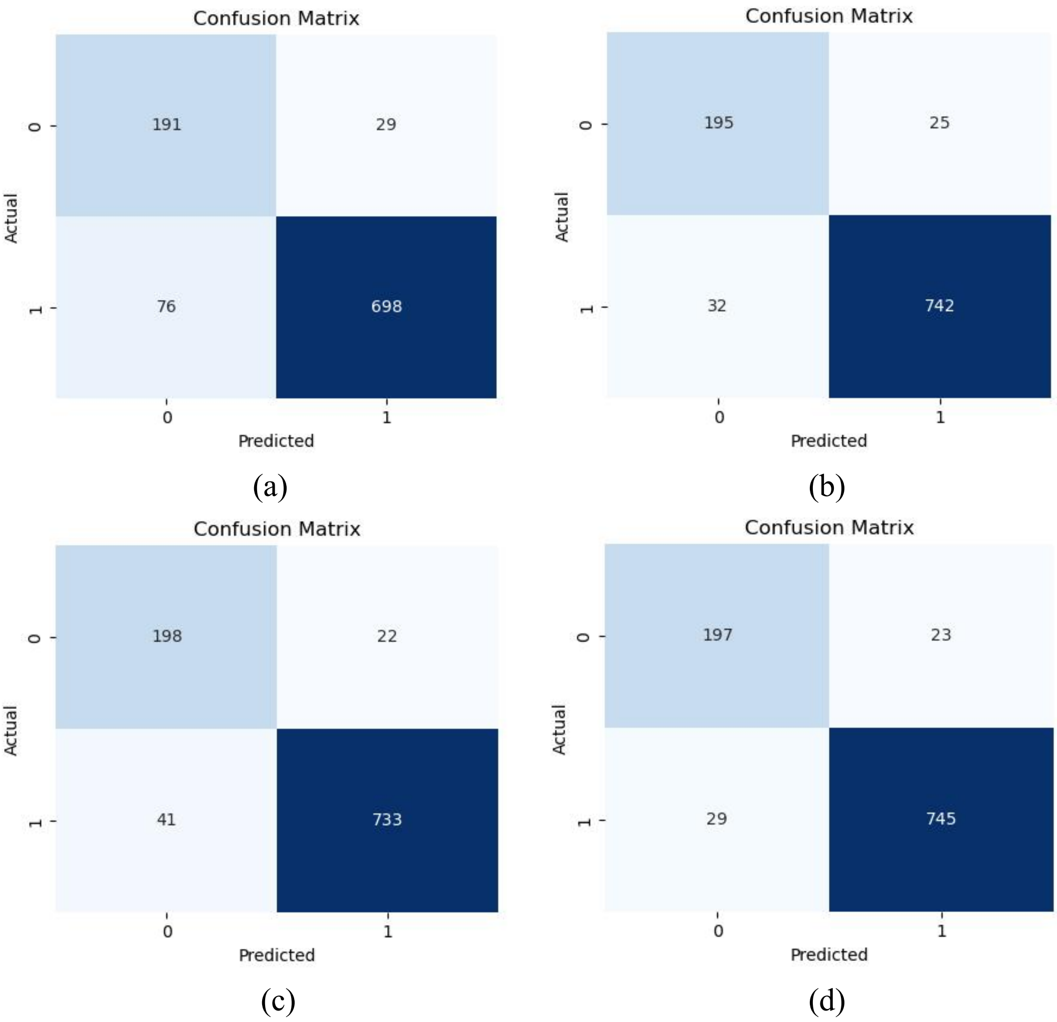
Figure 5.1 Comparison of metrics for each model

Table 5.2 compares the performance metrics obtained for all the models evaluated—Decision Tree(DT), Random Forest Classifier(RF), Extreme Gradient Boost Classifier(XGB) and Light Gradient Boost Classifier(LightGBM). The comparison reveals that ensemble-based methods notably outperform the standalone DT classifier. The Decision Tree classifier though simple and easily interpretable, shows the weakest performance among all the metrics with accuracy of (0.8944), the lowest F1-score(0.93000) and ROC-AUC score(0.9818). This shows its limited ability to handle complex data. RF classifier has a strong performance with the highest ROC-AUC score (0.9818), indicating its high efficiency in distinguishing between classes. XGBoost though has a relatively low recall and F1-score, achieves the highest precision (0.9709), demonstrating its efficiency in reducing false positives.

Among the four models evaluated, LightGBM emerged as the best model. It overperforms all other models in key metrics - accuracy(0.9477), F1-score(0.9663) and recall(0.9625). High accuracy confirms the general correctness of the predictions made. High recall shows that the model minimizes false negatives making it sensitive to capture true positive cases. A high F1-score indicates a strong balance between precision and recall and hence, does not overfit or underfit any of the classes. LightGBM also achieves a robust cross-validation accuracy of (0.6690 ± 0.0299), showing its strong generalization power with minimal variance. This ensures its reliable performance on unseen data. Hence LighGBM outperformed all other models in all the key metrics. Figure 5.1 confirms the same by providing a comparison bar graph of all metrics for all four models.

5.1.3 Confusion Matrix

Confusion Matrix provides a complete picture of the classification model by comparing actual and predicted values. Instead of relying solely on accuracy, it breaks down the predictions of true positives, true negatives, false positives and false negatives. This helps in identifying how often the model is correct as well as incorrect. This is essential particularly when errors have a huge impact on the model performance and decisions made.



(a)

(b)

(c)

(d)

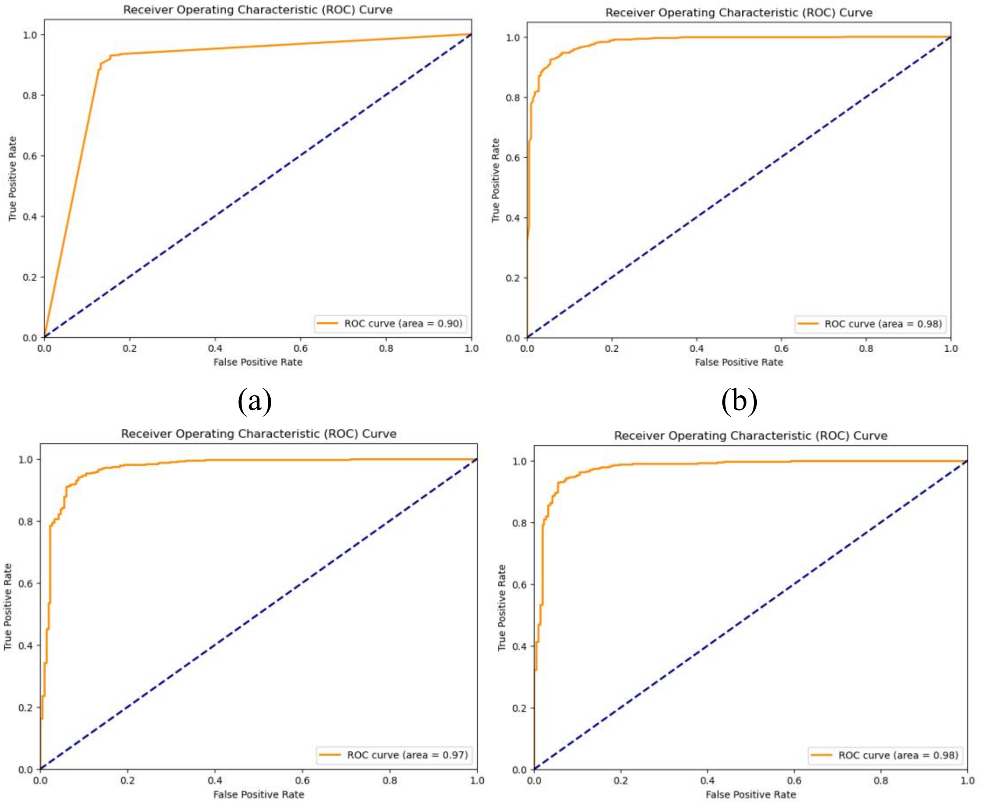
Figure 5.2 Confusion matrix

Figure 5.1 (a) shows the confusion matrix of DT. While DT captures many true positives (TP = 698) and true negatives (TN=191) it also sufferer from high false negatives (FN=76) and false positives (FP=29). The higher misclassification rates suggest that the model often misses actual positive cases making it less reliable for sensitive applications with signals prone to significant noises. Figure 5.1 (b) shows the confusion matrix of RF. Random Forest significantly reduces both FN (32) and FP (25) indicating better recall, improved precision and bringing a stronger balance of high TP (742) and TN (195). This shows the efficiency of RF in handling variance and reducing overfitting. Figure 5.1 (c) represents the confusion matrix of XGBoost. XGB shows a strong performance by achieving high TP (733) and TN (198) values and relatively fewer errors, FN (41) and lowest FP (22). Hence it balances variance and bias well. Figure 5.1 (d) shows the confusion matrix of LightGBM, highlighting the strongest performance in all metrics with the highest TP (745), high TN (197) and lowest FN (29) -indicating very good recall, and low FP (23). This shows its ability to learn complex, non-linear relationships and offer both good precision and recall. A final summary of the analysis is given below :

TABLE 5.3 Confusion Matrix Analysis

Model	TP	TN	FN	FP	Inference
DT	Relatively Low	Moderate	High	High	Prone to errors, lacks generalization
RF	High	High	Low	Low	Good precision and recall, balanced performance
XGB	High	Highest	Moderate	Lowest	Reduces False positives and balance bias and variance
LightGBM	Highest	High	Lowest	Low	Best recall and precision, fast and scalable

5.1.4 AUC-ROC Curve



(a) (b)

(c) (d)

Figure 5.3 ROC-AUC Curves

Figure 5.2 (a) DT– The ROC curve starts strong with a steep rise in the True Positive Rate at the False position rate, then flattens and approaches the diagonal. Since it is a single tree, it is prone to overfitting and makes more False Positive errors. The AUC (0.90) indicates moderate classification performance. (b) RF–Very low false positive rate for most thresholds. Handling both bias and variance effectively classifies classes. The AUC(0.98) indicates its excellent classification capability. (c) XGB–shows very high TPR and FPR across a wide range of thresholds but slightly lower than RF. AUC score (0.97) shows its high efficiency as a classifier. (d) LightGBM–maintains a very low FPR with the highest TPR, equally strong as RF in terms of discrimination. AUC score(0.98) equal to RF denotes a very good performance. Supports leaf-wise growth, which tends to reduce loss more effectively than RF or XGBoost.

5.2 Limitations

While the model used in the study achieves strong classification performance across multiple models such as Decision Tree, Random Forest, XGBoost, and the LightGBM, the study is not without limitations. The use of PCA for dimensionality reduction, though effective in reducing overfitting and preserving variance, makes the interpretability of the individual features difficult. Secondly, although the ADASYN technique helped to balance the classes by generating synthetic samples, it also introduces a risk of producing noise or artificial patterns. This concern is reflected in lower cross-validation accuracies, particularly for DT and XGBoost. This indicates possible variance and a decrease in model generalizability across different data splits. Additionally, hyperparameter tuning using GridSearchCV, though increasing model accuracy is computationally expensive and time-consuming, especially for XGBoost and LighGBM. Lastly, while the ensemble-based models achieved high accuracy, they pose challenges in transparency and explainability.

5.3 Future Works

Several future advancements can be made in the presented study. One such enhancement could be integrating trained machine learning models into a real-time intrusion detection framework through which we can test our models on live data to check how well the model detects intrusions as they happen. This can help to analyse the practical capability of the model beyond static datasets. Advanced feature engineering techniques can be done with the help of deep autoencoders or embedding-based representations which can replace PCA and capture complex patterns and relationships more efficiently. Additionally, models can also use SHAP or LIME which can help to visualize and identify the impact of each feature, especially in the case of ensemble-based algorithms. Hybrid architectures combined with gradient boost algorithms with models LSTM can provide advanced detection for attach pattern that varies with time. Adaptive resampling techniques or other cost-sensitive learning techniques can be used as alternatives to ADASYN. This can handle class imbalance more effectively. Finally, the scope can be expanded to detect multiclass or multilabel attacks for more comprehensive and realistic stability detection.

Chapter 6
Conclusion

6.1 Challenges Faced

The development and assessment of the power stability detection system studied here used advanced machine learning models that presented several challenges. The primary challenge faced was the notable class imbalance in the dataset, where the volume of "Natural" (normal) traffic samples exceeded that of the "Attack" samples. This posed a serious risk of model bias toward the majority class, potentially degrading the detection performance for minority (attack) instances. To address this, ADASYN (Adaptive Synthetic Sampling) was implemented to generate synthetic minority class examples in a data-distribution-aware manner, thereby improving class balance while avoiding overfitting. With a large number of features where many were noisy, and redundant and a few irrelevant data, which posed a risk of overfitting and increased computational burden. To resolve this, a hybrid feature reduction strategy was adopted, called SelectKBest. This was used to retain statistically significant features, and PCA (Principal Component Analysis) was subsequently applied to reduce the dimensionality. However, this also introduced a trade-off between performance and model interpretability, as PCA transforms features into unrecognizable principal components. Hyperparameter tuning using GridSearchCV was resource-intensive and time-consuming. Efficient parallel processing and narrowing the parameter grid helped to resolved this to some extent. Additionally, train-test splits, multiple metrics, and consistent preprocessing pipelines across all models added to ensure fair performance increased complexity, especially when it came to explaining model decisions post PCA.

6.2 Conclusion

The study discussed here presents a machine learning-based power grid stability detection system using four advanced classifiers—Decision Tree, Random Forest, XGBoost, and LightGBM. optimized with the help of hyperparameter tuning. The dataset was thoroughly preprocessed through handling of missing values, and addressing class imbalance using ADASYN, PowerTransformer-based feature scaling, SelectKBest feature selection, and PCA for dimensionality reduction. Among the tested models, LightGBM emerged as the overall best performer with the highest accuracy of 94.77%, F1-score of 0.9663 and ROC-AUC score of 0.9759, proving its high generalization power, effectiveness in reducing errors and detecting any attack on the grid. Though the models perform well on static datasets, employing them to work on real-time live data to detect the ability to detect evolving cyber threats in practical settings remains a task for future research. Overall, the project contributes a solid foundation for the development of intelligent, data-driven intrusion detection frameworks capable of enhancing modern network security.

References

[1]S. R. Ahmed et al., “Enhancing Power Grid Efficiency using Machine Learning Algorithms,” 2024 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), pp. 1–7, May 2024, doi: 10.1109/hora61326.2024.10550851.

[2]A. Zahedi, “Smart Grid Opportunities &amp;amp; Challenges for Power Industry to Manage the Grid More Efficiently,” 2011 Asia-Pacific Power and Energy Engineering Conference, pp. 1–4, Mar. 2011, doi: 10.1109/appeec.2011.5749149.

[3]D. Rangel-Martinez, K. D. P. Nigam, and L. A. Ricardez-Sandoval, “Machine learning on sustainable energy: A review and outlook on renewable energy systems, catalysis, smart grid and energy storage,” Chemical Engineering Research and Design, vol. 174, pp. 414–441, Oct. 2021, doi: 10.1016/j.cherd.2021.08.013.

[4]S. M. Miraftabzadeh, A. D. Martino, M. Longo, and D. Zaninelli, “Deep Learning in Power Systems: A Bibliometric Analysis and Future Trends,” IEEE Access, vol. 12, pp. 163172–163196, 2024, doi: 10.1109/access.2024.3491914.

[5]M. S. Ibrahim, W. Dong, and Q. Yang, “Machine learning driven smart electric power systems: Current trends and new perspectives,” Applied Energy, vol. 272, p. 115237, Aug. 2020, doi: 10.1016/j.apenergy.2020.115237.

[6]S. Azad, F. Sabrina, and S. Wasimi, “Transformation of Smart Grid using Machine Learning,” 2019 29th Australasian Universities Power Engineering Conference (AUPEC), pp. 1–6, Nov. 2019, doi: 10.1109/aupec48547.2019.211809.

[7]Noor. Z. Mahmood, S. R. Ahmed, A. F. Al-Hayaly, S. Algburi, and J. Rasheed, “The Evolution of Administrative Information Systems: Assessing the Revolutionary Impact of Artificial Intelligence,” 2023 7th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), pp. 1–7, Oct. 2023, doi: 10.1109/ismsit58785.2023.10304973.

[8]C. Rudin et al., “Machine Learning for the New York City Power Grid,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 2, pp. 328–345, Feb. 2012, doi: 10.1109/tpami.2011.108.

[9]S. R. Ahmed, E. Sonuc, M. R. Ahmed, and A. D. Duru, “Analysis Survey on Deepfake detection and Recognition with Convolutional Neural Networks,” 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), pp. 1–7, Jun. 2022, doi: 10.1109/hora55278.2022.9799858.

[10] et al., “Smart Grid: A Survey of Architectural Elements, Machine Learning and Deep Learning Applications and Future Directions,” Journal of Intelligent Systems and Internet of Things, pp. 32–42, 2021, doi: 10.54216/jisiot.030103.

[11]C. Zhang, O. Vinyals, R. Munos, and S. Bengio, “A Study on Overfitting in Deep Reinforcement Learning,” *arXiv:1804.06893 [cs, stat]*, Apr. 2018, Available: https://arxiv.org/abs/1804.06893

[12]Md. Akhtaruzzaman, M. K. Hasan, S. R. Kabir, S. N. H. S. Abdullah, M. J. Sadeq, and E. Hossain, “HSIC Bottleneck Based Distributed Deep Learning Model for Load Forecasting in Smart Grid With a Comprehensive Survey,” IEEE Access, vol. 8, pp. 222977–223008, 2020, doi: 10.1109/access.2020.3040083.

[13]J. Wu, Y. Chua, M. Zhang, H. Li, and K. C. Tan, “A Spiking Neural Network Framework for Robust Sound Classification,” *Frontiers in Neuroscience*, vol. 12, Nov. 2018, doi:10.3389/fnins.2018.00836.

[14]M. F. Guato Burgos, J. Morato, and F. P. Vizcaino Imacaña, “A Review of Smart Grid Anomaly Detection Approaches Pertaining to Artificial Intelligence,” Applied Sciences, vol. 14, no. 3, p. 1194, Jan. 2024, doi: https://doi.org/10.3390/app14031194.

[15]Y. Liu *et al.*, “Selective ensemble method for anomaly detection based on parallel learning,” *Scientific Reports*, vol. 14, no. 1, Jan. 2024, doi:10.1038/s41598-024-51849-3.