National College of Ireland

Honeypots and the Use of AI in keeping the IoT Systems Secure

MSc Research Project
Cyber Security

Venkat Goud Goundla

Student ID: x23152397

National College of Ireland

Supervisor: RAZA UI MUSTAFA

National College of Ireland
Project Submission Sheet

Student Name:	Venkat Goud Goundla
Student ID:	x23152397
Programme:	Cyber Security
Year:	2024
Module:	MSc Research Project
Supervisor:	Raza ul Mustafa
Submission Due Date:	16/09/2024
Project Title:	Honeypots and the Use of AI in keeping the IoT System Secure
Word Count:	8972
Page Count:	26

I hereby certify that the information contained in this (my submission) is informationpertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at therear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Venkat Goud Goundla
Date:	16th September 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keepa copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placedinto the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Honeypots and the Use of AI in keeping the IoT Systems Secure

Abstract

Honeypots play a crucial role in a comprehensive cybersecurity strategy by offering early detection, useful intelligence, and improved reaction capabilities. They also increase the overall security posture of a business. This study centers on implementing a sophisticated security solution for IoT using honeypots. The honeypots are monitored using the most efficient machine learning model to identify illegal access and deploy honeypots in a dynamic manner. The system utilizes models such as LightGBM, which have shown to be highly accurate and efficient, to accurately detect threats while decreasing the occurrence of false positives. This strategy enhances memory efficiency by selectively activating honeypots just in high-confidence threat scenarios, hence minimizing superfluous resource utilization. Machine learning integration improves the ability to identify and respond to threats in real-time, offering a security solution that is adaptable, effective, and strong, specifically designed for critical IoT settings. This solution is then deployed in the real time monitoring of the iOT devices.

Keywords: Cybersecurity, Machine Learning, LGBM, Access Monitoring

Chapter 1: Introduction

Fuelled by the exponential growth, IoT devices have stepped up the connectedness and have led to the industrialization of smart cities, manufacturing, agriculture, healthcare, and many more. Elaborating to define its products as starting from smart thermostats, wearable fitness trackers, industrial sensors, self-driving cars, it allows interaction and networking for a perfect automation which makes it easy and efficient.

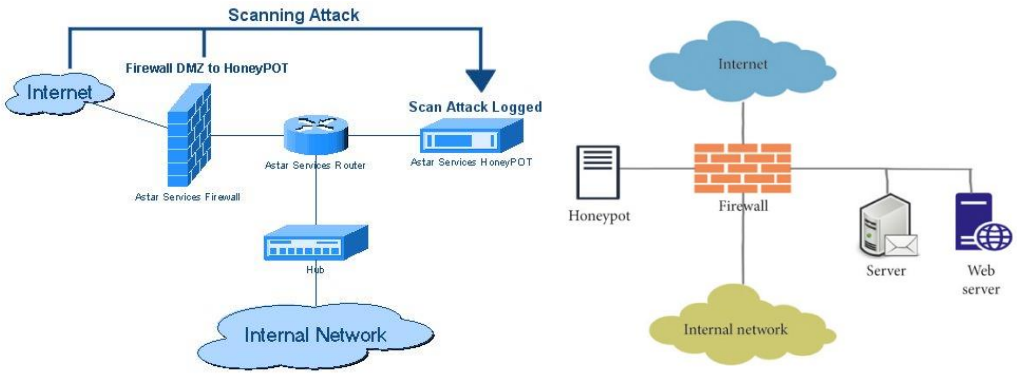

Figure 1: An example of how the honeypots is present in the firewall of the iOT systems **(Source: Wikipedia and Researchgate**)

But this explosion of IoT implementation also presents major cybersecurity problems. IoT devices are prime candidates for cyberattacks because of their natural traits including limited processing capability, varied operating settings, and often insufficient security measures. IoT ecosystems expand along with the channels for possible assaults—data breaches, distributed denial of service (DDoS) attacks, and illegal access to private systems (**Ghorbani, H. and Ahmadzadegan, M., 2017**). Designed for more homogeneous and resource-rich environments, traditional cybersecurity solutions can fall short in properly protecting IoT networks. This calls for the creation of customised security plans fit for the requirements of IoT systems (**Makhdoom, I., Abolhasan, M., Abbas, H. and Ni, W., 2019**). Since the

cost of the data privacy and the emergence of the data driven system whether iOT or any other kinds of systems arises, the security to enable a restrictive method from the attackers has increased a lot.

1.1 Motivation

Honey pots offer a calculated means of improving cybersecurity. These fake devices are meant to look to attackers vulnerable and appealing. Honeypots draw hostile actors by replicating real-world devices and networks, therefore enabling security experts to track and examine their activities in a contained context. This clarifies not just the strategies, tools, and techniques (TTPs) utilised by attackers but also helps create appropriate countermeasures (**Gharbi, C., Hsairi, L. and Zagrouba, E., 2021**). Honeypots can be used in the framework of IoT to replicate different IoT devices and networks, so offering important information on the particular hazards aiming at these systems (**Yang, Y., Wu, L., Yin, G., Li, L. and Zhao, H., 2017**). Collecting data on attack patterns and techniques, a honeypot can be set to replicate a smart thermostat or an industrial control system. Designing strong defence systems and enhancing the security of real IoT devices depend on this knowledge, which is also quite valuable (**Sicari, S., Rizzardi, A., Grieco, L.A. and Coen-Porisini, A., 2015**).

1.2 Aim of the Research

Thus, artificial intelligence (AI) in cybersecurity appears as a rising transforming tool with enhanced opportunities for threat identification, assessment, and confrontation. especially the machine learning (ML) and Deep learning (DL) artificial intelligence techniques are best suitable for analyzing large amount of data and identifying peculiarities that would lead to the indication of hostile activity (**Arora, S., Kumar, M., Johri, P. and Das, S., 2016**). It has been ascertained that AI solutions enhance the provision of security in contexts of IoT since such environments are characterized by large and complex data (**Yu, W., Liang, F., He, X., Hatcher, W., Lu, C., Lin, J. and Yang, X., 2017**). AI in IoT security is involved in anomaly detection, algorithms that scans and identify whether the traffic or the behavior of the devices is unusual, threat prediction, which deals with algorithms that analyze the previous records and predicts any potential oncoming attacks, automated response through taking immediate action eliminating the threats, and Behavioral analysis which is able to distinguish and classify the traffic into normal or malicious through constant learning (**Xu, L., He, W. and Li, S., 2014**).

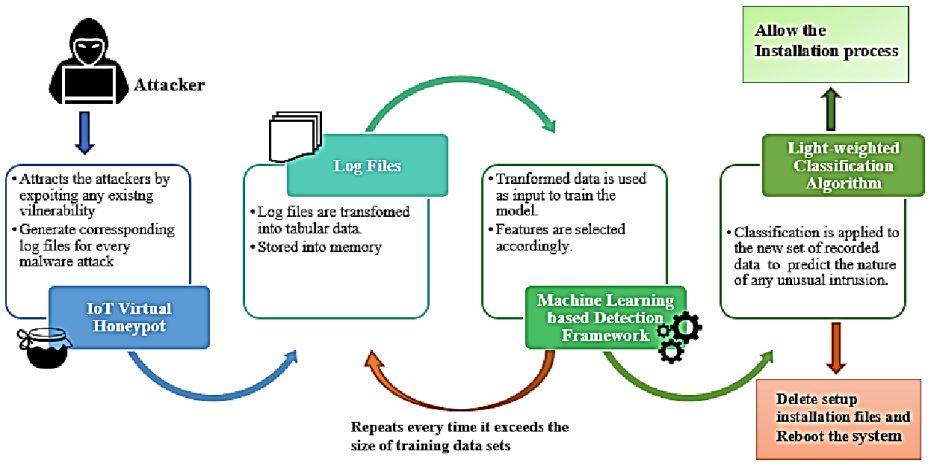

Figure 2: Process flow of the Machine Learning based Honey Pots for detecting the severity of the hacking (**Source: Semantic Scholar**)

Combing honey pots with AI-based technologies makes IoT systems' security based on honey pots a strong synergy. AI applied to honey pots can enhance their effectiveness by on the fly generating relevant honey pots that match the dynamic threat landscape, thus deceiving the adversary, and gathering the relevant data, that consequently, can be used to identify the subtle patterns of the attacks and to create the useful intelligence (**Sarhaddi, F., et. al., 2021**). AI also help in responding to the

detected threats through honeypots through means such as locking down the device or blocking the IP address. compromised devices or blocking. Actionable real-time information and alerts concerning continuous threats as a result of more effective monitoring through the incorporation of artificial intelligence assist to strengthen the security architecture even further (**Baykara, M. and Das, R., 2015**).

Through this research we intend to make a generic system using the machine learning and data acquisition techniques for the vulnerability detection in the iOT system

1.3 Research Objective

In this case it is clear that many advantages are achievable by the combination of artificial intelligence and honey pots but it also have some difficulties. Artificial intelligence driven honeypots may be a problem for some companies to deploy and manage since they require specific expertise. IoT devices often have limitations in the computational power, which hinders the applicability of AI algorithms. Like every other system, AI systems sometimes go wrong, and this leads to false positive alarms and possible interruption in some activities. First, cyberattacker implies the new approaches and ways for searching and avoiding honeypots, so the security measures must be permanent and evolving (**Mokhtari, S., Abbaspour, A., Yen, K. and Sargolzaei, A., 2021**).

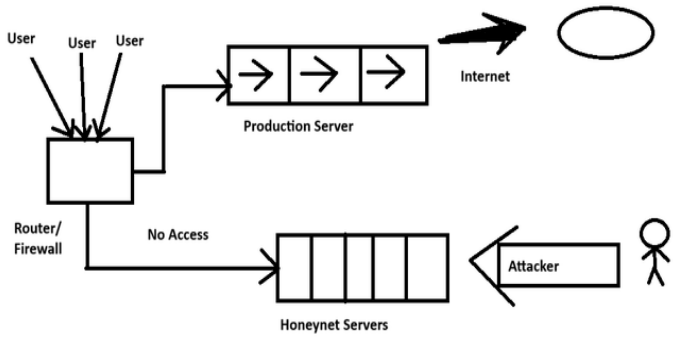

Figure 3: How Honeypots helps in mitigating the attacker into the system (**Source: GeeksforGeeks**)

Honey pots and artificial intelligence working together has significant potential to improve IoT system security. By using the best parts of both technologies, businesses can learn more about online threats and come up with better ways to protect themselves. The need of strong and flexible security measures will only become more significant as IoT keeps developing. This study article is to investigate the present developments, advantages, and difficulties of combining honeypots with artificial intelligence in IoT systems, so stressing their capacity to protect our ever-linked world.

1.4 Research Questions

RQ: How can the integration of artificial intelligence and honeypots enhance the detection and mitigation of cyber threats in IoT ecosystems?

Research Solution: To develop a framework which can help in mitigation of cyber threats in IoT ecosystems. For AI (Artificial Intelligence) and DS (Data Science Platform) we will use Python and for the simple framework we will use Streamlit connected over the Server. We will analyse not only the system performance in analysing the vulnerability but also the RAM usage and the storage of the system.

1.5 Thesis Structure

In the next chapter, we will further discuss the work done under the arch in terms of the various research undertaken for the purpose of unique identification of the research works. In the chapter on methodology and implementation we will elaborate on the tasks that are needed to do this proposed

implementation. In the results we will outline the findings of this research and evaluate the findings of this study.

Chapter 2: Literature review

By using honeypot data to produce labelled datasets for machine learning and deep learning algorithms (**Ahmed, Y., Beyioku, K. and Yousefi, M., 2024**), this research advances the subject of IoT security. By utilising these algorithms, the research exhibits strong threat identification abilities.

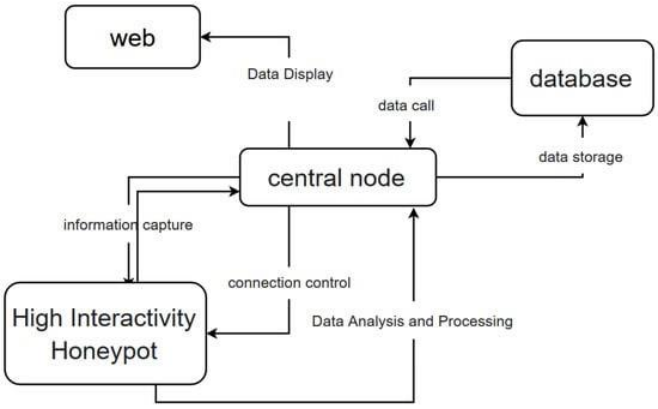

Figure 4: An enhanced and intelligent with interactive Honeypot based approach to th network threat management (**Ahmed, Y., Beyioku, K. and Yousefi, M., 2024**)

The best indexing algorithm was decision trees with accuracy ranging from 99%. 26% and 99. around 92% of the time on different datasets and iterations. Further research should collect more datasets with IoT honeypot label, and investigate applications which are related to healthcare, smart grid, and smart home domains, while ensuring the scalability and the real-time intelligence of these security systems.

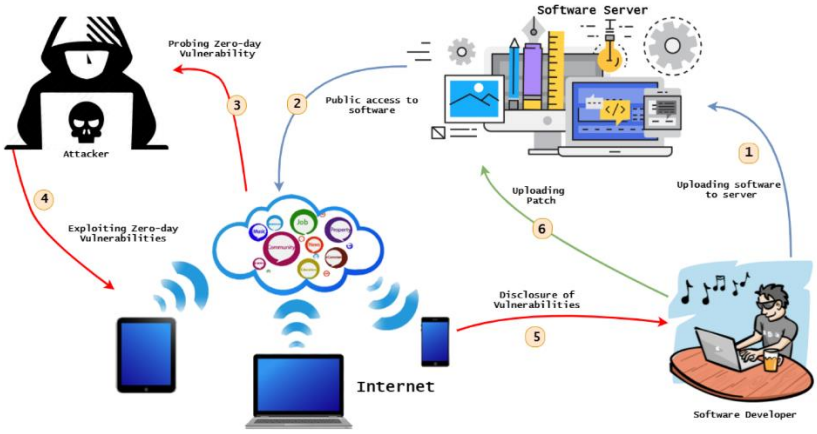

Figure 5: Zero day attacks detection using the enhances ML methods (**Lanka, P., Gupta, K. and Varol, C., 2024**)

As it attracts and ensnars attackers, honeypots are useful for security because of the information it provides about methods used by the attackers. In order to raise their efficiency (**Lanka, P., Gupta, K. and Varol, C., 2024**), every type of honeypots, including low-interaction, high-interaction, and the system that combines them both, has to be deployed intentionally and purposefully. If compliance and privacy are to be maintained, then the legal and ethical considerations have to be scrutinized properly. Honeypots are effectively used to enhance threat detection and to get an early warning of malicious activities. Thus, with the help of AI, honeypots benefit from fewer false alarms and automation of replies, being thus invaluable tools for cybersecurity. The necessity for efficient detective and

preventive measures is increasing because of the constantly rising number of cyber threats targeting critical infrastructures and IoT networks (**Morozov, D.S., et. al., 2023**). The cyber deception systems and sound honeypot solutions are the key and reasonable techniques that deliver comprehensive understanding of the methods and tools of threats. Subsequent researches are aimed at creating a credible Honypot Io T network to collect data on attack behaviors and courses. Thus, incorporating machine learning technology into this network would improve the identification and analysis of attackers, thus improving IoT security in general.

As technology changes, so do the ways that we protect it (**Sharma, D. et. al., 2023)**. Honeypots and honeynets are two types of trickery technology that have been created to fix holes in Internet of Things networks. This change will become more important for emerging technologies like 5G as the Internet of Things grows. As the 5G network grows, better security measures will be needed to protect the privacy of data and deal with the new problems that come up because of the better connections. From its inception to its most current evolution, we have examined the history of honeypot technology in great detail (**Schmitt, M., 2023**). You should weigh the benefits and drawbacks of each honeypot kind. Honeypots will probably become increasingly valuable as automation and AI continue to improve. As cyber threats change, honeypots will become even more useful for finding and stopping attacks. People and businesses need to buy honeypot technology to keep their resources safe and stay one step ahead of hackers.

The attack area has grown a lot since IoT and cyber-physical systems have brought the digital and physical worlds together. Because people are connecting more and more, we need systems that can find threats intelligently using machine learning and deep learning. With the help of artificial intelligence, security systems can block hackers and other threats more effectively. Businesses, government agencies, and important assets must add AI-powered security features to their current systems in order to protect themselves from complex cyber threats. A highly interactive honeypot threat management system can be developed using a modular design approach (**Yang, X., Yuan, J., Yang, H., Kong, Y., Zhang, H. and Zhao, J., 2023**).

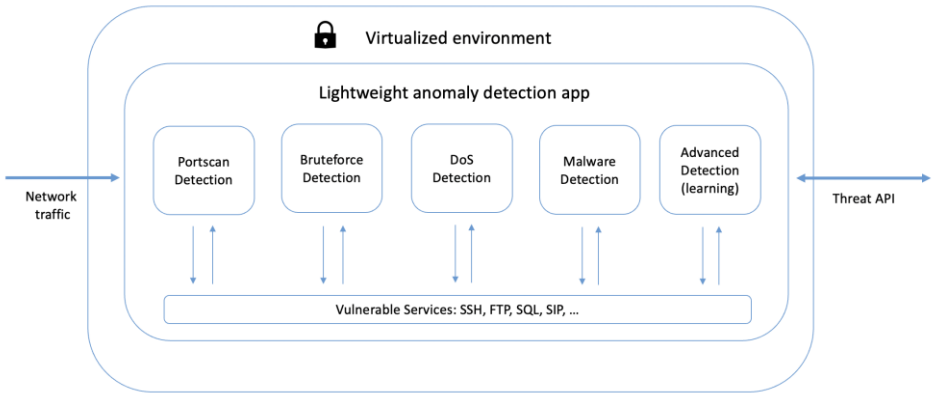

Figure 6: A virtualised honeypots to protect the iOTs (IoATC) (**Yang, X., Yuan, J., Yang, H., Kong, Y., Zhang, H. and Zhao, J., 2023**)

This method refines the idea of honeypots and makes the utilization of honeypots clearer as well as easier for SMBs and MSEs. The system integrates good UI and employs high-interaction honeypots in information collection. Subsequent researches will continue to explore enhancing honeypot corresponding security technologies and appliances and extending honeypot to other areas such as the defense against DoS and anti-phishing. As a proof of the design in a real world environment, a hybrid honeynet model that will utilize the Dock technology has been implemented (**Amal, M.R. and Venkadesh, P., 2023**). This methodology applies fuzzy rule bases to improve the security of IoT networks while, at the same time, monitoring the number of observed occurrences of the host

ransomware action. Moreover, the identified H-DOCTOR technology is capable of reducing the ransomware effects by avoiding the loss of data as the suggested technology is far accurate in the detection process in comparison with the current advanced technologies available in the market. Therefore, the primary areas for improvement in approaching the main iterations will be tied to the direct application of skills in relation to transfer learning, which will allow the maximum use of resources. Therefore, honeypots contribute a great deal to expanding the amount of information that businesses can gather about danger landscapes (**VS Devi Priya and S Sibi Chakkaravarthy, 2023**). Although they are mainly oriented for surveillance of the activity aimed at the honeypot, the data collected by them are invaluable for analytical description of tendencies of assault. The system configurative offered of all-inclusive data analytical capability and workability was established to be useful in detecting DDoS threats in live settings. In the future, attention will be focused on the enhancement of obstruction appraisal/interference detection and prevention and the knowledge essential for creating early diagnosis for obstructions and on the vulnerabilities of the system.

2.1 Research Summary

The objectives of the study are to enhance the security of the IoT systems employing machine learning and honeypot data, and deep learning techniques. Decision trees were considered to be very efficient whereby they yielded accuracy of between 99. 26% to 99. reliability of 92% has been obtained for using the proposed framework across the various datasets and multiple times. As for the follow-up studies, more IoT honeypot datasets are expected to be obtained to develop in several directions, such as healthcare, smart grids, and smart home ones. It is crucial to analysis also the predictability and practically usage of these security solutions. Addressing honeypots is an inevitable part of catching intruders since they help in obtaining valuable information about their actions. It was identified that by applying different forms of honeypot, including low-interaction, high-interaction, and low and high hybrid honeypot systems, it is likely to enhance the endeavour of cybersecurity, to reduce the noise rates or false alarms, and to integrate or automate reaction procedures. In this context the specifics of study include the analysis of cyber deception systems and complex honeypot solutions – in particular IoT honeynets – in a method for data collection on attacks and in integrating machine learning technology.

The changes in the threats identified in the research show the dynamics and constant evolution, meaning that there is always a need to enhance the developments of honeypot technology. Its use in AI and deep learning, especially in identifying threats in the future Connecticut IoT and cyber-physical system architectures, is deemed necessary. All in all, the research contributes the priority of honeypots and advanced ML approaches in enhancing the protective Ost of IoT by enhancing the knowledge and presence of risks in the network environment.

2.2 Research Niche

This study will focuses on deploying a smart honeypot model on IoT that will be overseen by the best machine learning algorithm to detect unauthorized access and or install honeypots. The employed models like LightGBM that have high accuracy ensure that dangerous threats are noticed while minimizing false signals. This strategy improves the functionality of the memory by only engaging the honeypots in the high-confidence threat situation; thus, avoiding wastage of resources. Thus, machine learning increases the potential of real-time threat identification and response. This leads to something like securing our IoT environment in a way that is malleable, efficient, and robust and which targets the severest situations.

Chapter 3: Methodology

3.1 Research Resource: Dataset for training the Machine Learning Models

3.1.1 Data Collection and Preprocessing

The gathering and preliminary analysis of honeypot data are essential stages in this research. In order to draw in and seize harmful activity, honeypots mimic susceptible Internet of things devices. This creates a controlled environment in which to gather attack data. The data is made clean, standardised, and analysis-ready through preprocessing. This is an important step since, prior to applying machine learning algorithms, raw data from honeypots frequently contains noise, missing numbers, and inconsistencies that must be resolved. Preprocessing makes trustworthy analysis and model training possible by guaranteeing data consistency and quality, which is necessary for precise threat identification and classification.

Following are the datasets that are used for the analysis,

a. Standard Datasets after a thorough literature review and web searching to prove the hypothesis

b. Real Time Dataset to check if the application using a webapp works fine along the deployed best machine learning model.

3.1.1.1Dataset 1: DDS Dataset Creation - Link - https://research.aalto.fi/en/datasets/iot-devices-captures

Description: The file is an original compressed CSV file in the tar/gzip format taken from the dataset of AWS honeypots. The information data provided into the system is a compressed CSV file containing the domain name and its category. The classification includes two primary categories: Collecting two kinds of domains, one is the domain generated by algorithm 'dga' (short for domain generated algorithmically) and the other is the legitimate one, 'legit'. Further, the “legit” category is subclassified into types that are, namely, “cryptolocker,” “gox,” and “newgoz” ".

3.1.1.2 Dataset 2: IoT devices captures - Link - https://research.aalto.fi/en/datasets/iot-devices-captures

Description: This dataset involves the communication for the setup of the 31 Internet of Things (IoT) devices that are present in smart homes. These devices fall into 27 different types, and four of these types include two of the named devices each. Each configuration exhibited was performed a minimum of twenty times for each sort of device.

3.1.1.3 Dataset 3: Threat_Research - Link - https://github.com/JonathanPhillips/Threat_Research

Description: A centralised repository is needed to store threat research data collected from my network of honeypots.

3.1.2 Exploratory Data Analysis

The analysis of the honeypot data entails the exploration of the characteristics and behaviors, which can only be determined by EDA. EDA helps researchers in understanding the nature of relations between the variables or some of them, trends or/and outliners with the help of the statistical measures, visualizations, and assessments of the distributions. Specifically, it reveals white and black behaviour of IoT devices in the context of IoT security. In this way, it is possible to understand these patterns to select the corresponded machine learning algorithms and do the feature engineering. Thus, decisions about the overall approach to model development as well as the more detailed data preprocessing steps are also affected by EDA so as to ensure that the chosen approaches are appropriate given the characteristics of the data and the objectives of the study..

3.1.3 Feature Selection

In order to create machine learning models for IoT security that are both fast and effective, feature selection is essential. Feature selection strategies aid in determining the most pertinent aspects that help differentiate between benign and malevolent behaviour in the context of honeypot data, which may contain a multitude of variables. Predictive accuracy is increased, computational complexity is decreased, and interpretability of the model is improved by relevant features. Researchers can create more reliable models that are able to precisely identify and mitigate cyber threats in Internet of Things environments by concentrating on the most useful elements.

3.1.4 Data Balancing using SMOTE, ADASYN, and Stratification

In cybersecurity applications, imbalanced datasets—where the proportion of normal behaviour to attack instances is large—are frequently encountered. In order to balance the dataset, two oversampling strategies called SMOTE and ADASYN create artificial samples of the minority class, or assaults. Maintaining this balance will help machine learning models identify rare but serious security threats more accurately by preventing them from becoming biassed in favour of the majority class. By maintaining class proportions in training and testing datasets, stratification protects the integrity of measures used in model evaluation. When combined, these methods improve the efficacy and dependability of machine learning models in Internet of Things security applications, guaranteeing strong performance in a variety of attack scenarios.

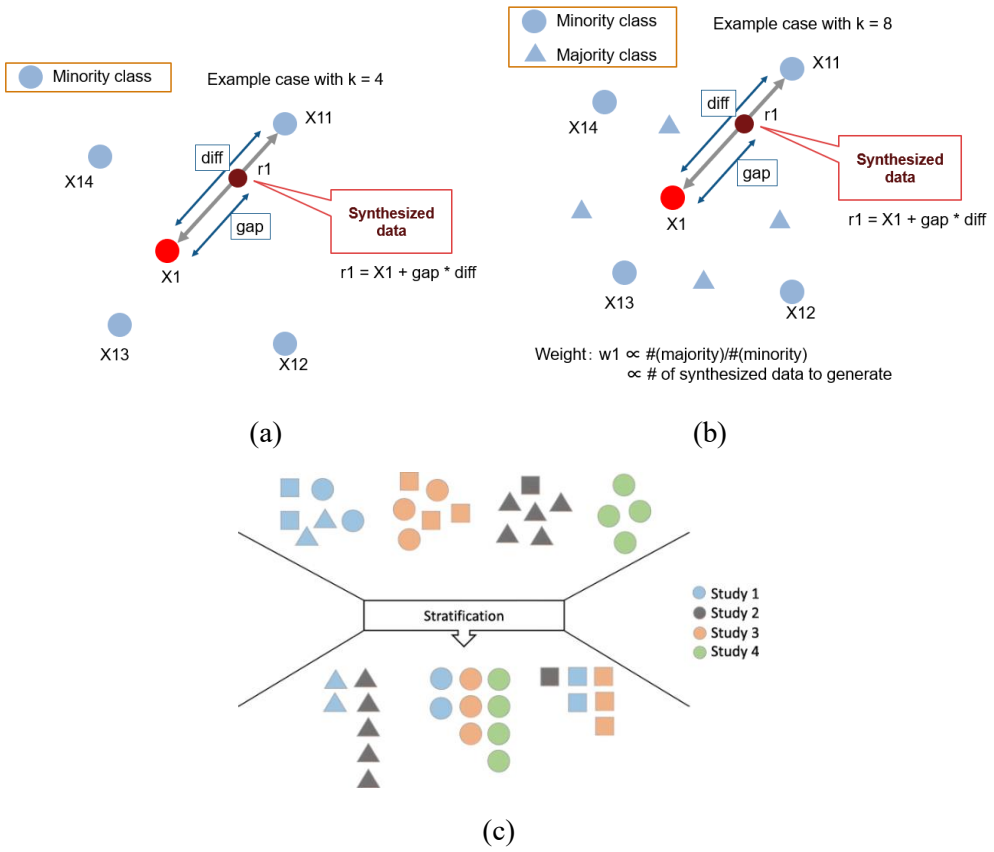

Figure 7: (a) The working of SMOTE, (b) The working of ADASYN and (c) The working of Stratification (**Source: AnalyticsVidya**)

Integrated there are two possibilities toward which the data balancing can be carried out. A one is oversampling while the other is undersampling. In the case of SMOTE which stands for Synthetic Minority Oversampling Technique, to begin with, the value of oversampling observations, N, is determined. In most cases, it is applied to guarantee that the distribution of classes is equal; in this case,

50:50 for the binary class. Nevertheless, with regard to the degree of tuning it can simply be stated that this parameter can be tuned to the degree required. As to the next step, it randomly selects a positive class instance to begin the loop. Next, the immediate top K neighbours (default is 5) for that given instance are obtained. Last, N out of these Kinstances are selected with the view of creating new synthetic instances through interpolation. To achieve this, an Lp distance measure is used to express the divergence between the feature vector and its next neighboring vectors. Now, this disparity is multiplied by a random number between 0 and 1 but excluding 0 and is then added on the previous feature vector (**Devi Priya, V.S. and Chakkaravarthy, S.S., 2023**).

ADASYN is another version of SMOTE where the former is an expanded version of the latter. This method also aims at advancing the numbers in the minority class by enumerating as many fake cases as possible towards that class. Nevertheless, it differs from the mentioned approaches by regards to the density distribution, *ri* with using it to define the number of synthetic examples to generate for difficult samples to learn. Thus, it helps in achieving the flow of the parameters of the judgment limits based on the samples that are difficult to learn. This constitutes the first imperative difference when comparing to SMOTE.

They ensure that in the sampling process every subclass in the population is well represented in the sample by the use of Stratified Sampling. This strategy allows attaining greater precision in estimating the model parameters in cases when the population can be subdivided into fairly homogeneous subsamples. Consequently, simple random sampling is quite useful in cases where the section of population is known not to be compartmentalized into different subgroups by virtue of the many variations in the population.

3.2 Research Resource: Machine Learning Algorithms

3.2.1 Decision Trees

Determining decision rules from data features, Decision Trees are non-parametric supervised learning models (**Devi Priya, V.S. and Chakkaravarthy, S.S., 2023**). They work especially well at capturing intricate connections and interactions between different aspects in IoT security datasets. Decision Trees generate a tree-like structure with each internal node representing a feature, each branch representing a decision rule, and each leaf node representing a class label (attack or non-attack) by recursively splitting the data into subsets based on the most important features. These models may produce human-readable rules that explain how particular traits help identify cyber dangers in Internet of Things environments. They are also interpretable.

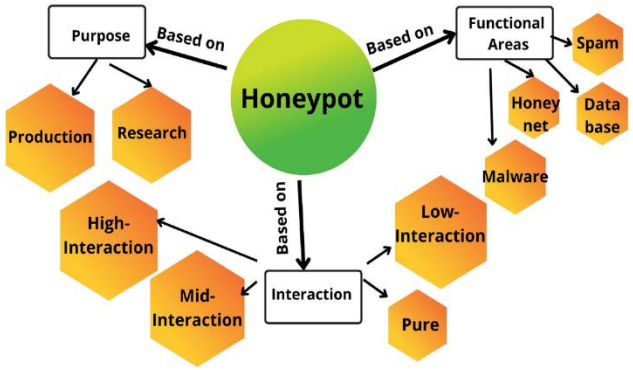

Figure 8: The decision nodes focusing upon the detection of the Malware using the Honeypots

Researchers have effectively developed decision tree classifiers by utilizing data obtained from honeypots, which are decoy devices strategically placed on a network to attract attackers, as well as data from regular network operations. The investigation results in the formulation of decision criteria for detecting malicious activities (**Grégio et al., 2007**).

3.2.2 Random Forest

To increase accuracy and resilience, Random Forest is an ensemble learning technique that constructs several decision trees and aggregates their predictions (**Ali, J., Khan, R., Ahmad, N. and Maqsood, I., 2022**). Unlike individual decision trees, each decision tree in the forest is trained individually using a subset of features and data, which reduces the likelihood of overfitting. Random Forests are particularly good in IoT security because they can handle noisy datasets and high-dimensional data, which are common in honeypot-based cybersecurity applications. By integrating the prediction strength of several decision trees, they improve generalisation performance and offer trustworthy cyber threat detection and categorization.

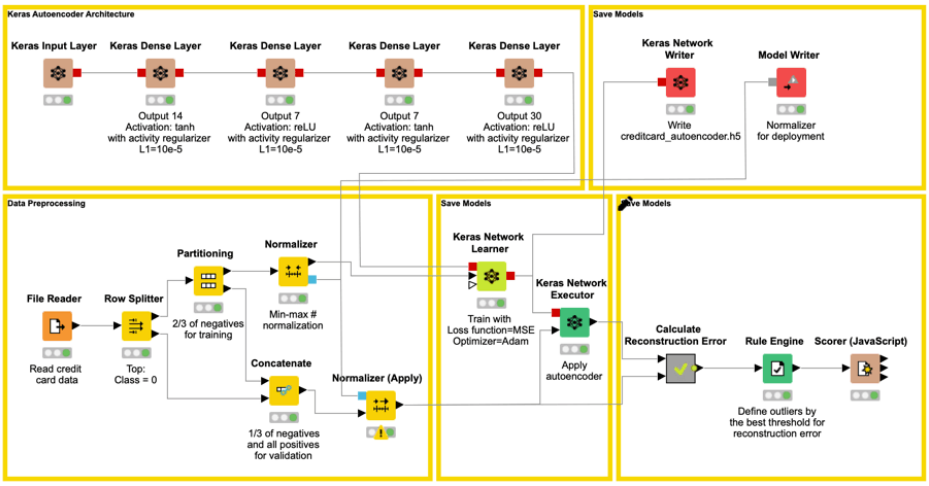

Figure 9: A Random forest based malware detection with the implementation of the Honeypots (**Devi Priya, V.S. and Chakkaravarthy, S.S., 2023**)

3.2.3 Support Vector Machines (SVM)

These robust supervised learning models are applied to regression and classification problems alike (**Ali, J., Khan, R., Ahmad, N. and Maqsood, I., 202**). SVMs are useful in IoT security for decomposing large, complicated datasets into smaller groups by identifying the best hyperplane to maximise the margin between attack and non-attack classes. They can employ kernel functions, including radial basis function (RBF) kernels, to manage non-linear connections between features. SVMs offer excellent accuracy and reliable performance in identifying cyber threats based on honeypot data, making them especially appropriate in situations where the decision border between attack and non-attack cases is not linearly separable.

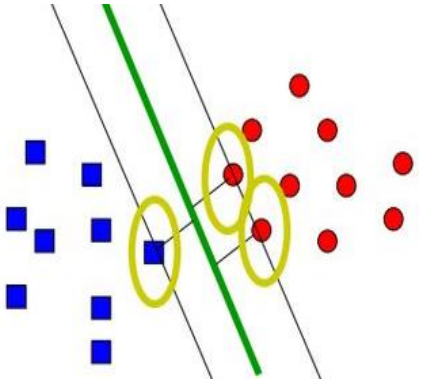

Figure 10: The hyperplane showcasing how to detect the intrusions from the true datapoints. In case of the intrusion or the suspicious attack, the models's hyperplane in the SVM modelling can differentiate easily (**Evgeniou, T. and Pontil, M., 2001**)

3.2.4 XGBoost

The gradient boosting technique known as XGBoostexcels in processing structured data because of its effectiveness (**Chen, T. and Guestrin, C., 2016**). The process iteratively constructs a group of ineffective learners, usually decision trees, and maximises model performance by reducing prediction mistakes. XGBoost works well with imbalanced datasets that are frequently found in honeypot-based cybersecurity applications for IoT security, where attack occurrences are much less than regular instances. By concentrating on regions of the dataset where misclassifications happen, it improves the identification and classification of cyber threats with the fewest false positives and reaches high accuracy.

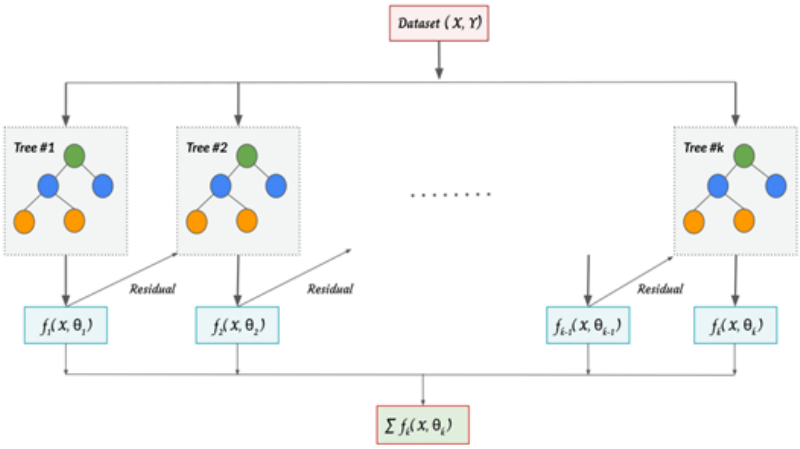

Figure 11: The image showcasing how XGBoost helps in differentiating the fraudulent or suspicious activity in a network (**Chen, T. and Guestrin, C., 2016**)

3.2.5 LightGBM

Another gradient boosting framework, LightGBM (**Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y., 2017**), makes use of tree-based learning methods, which are renowned for their effectiveness and speed. It uses methods like Exclusive Feature Bundling and Gradient-based One-Side Sampling to handle large-scale data and deliver great performance. Because of its quick training pace and capacity for handling huge datasets with numerous features, LightGBM is very useful in the field of IoT security. It is a good contender for use in Internet of Things environments due to its ability to precisely detect and classify cyber threats in real-time applications.

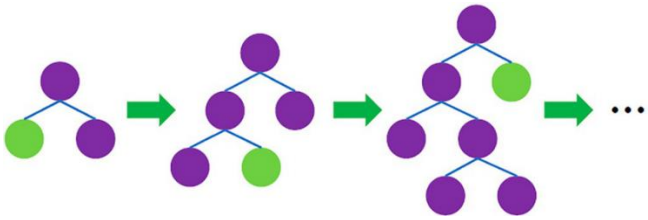

Figure 12: LGBM in the detection of the suspicious activity (**Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y., 2017**)

3.2.6 Neural Networks

Multi-Layer Perceptron in particular are a class of neural networks that are highly effective at identifying intricate patterns and relationships in data[24].

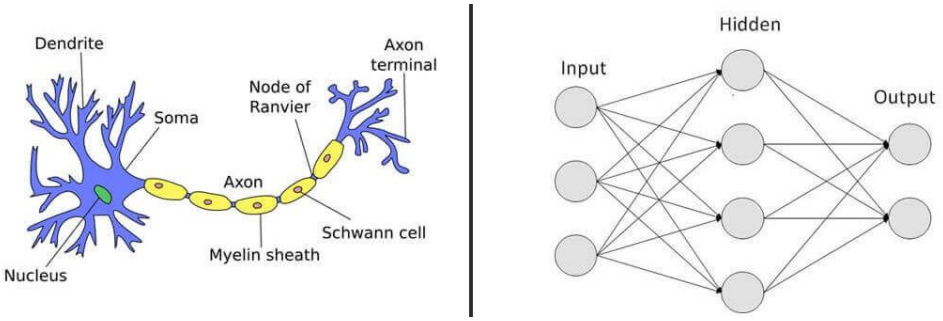

Figure 13: The left side of this figure depicts the biological neuron responsible for carrying the signals forward. The right side is the Artificial Neural Network (**Popescu, M.-C., Balas, V., Perescu-Popescu, L. and Mastorakis, N., 2009**)

The mathematics behind the working of the ANN is as $f(x) = (g * h)(x) = g[h(x)]$ then $f'(x) = g'[h(x)].h'(x)$, this is the impact of the chain rule. When it comes to IoT security, NNs are able to analyse vast amounts of honeypot data and discover complex feature relationships in order to identify cyber risks. Their capacity to adapt to many kinds of attacks and generalise successfully on unknown data makes them very valuable. But they need a lot of labelled data for training, along with a lot of processing power. In IoT environments, NNs can detect and mitigate cyber threats with high accuracy and robust performance, despite these challenges.

3.3 Research Resource: HoneyPot-Net Framework

A honeypot (**Yang, X., Yuan, J., Yang, H., Kong, Y., Zhang, H. and Zhao, J., 2023**) is a cybersecurity measure that establishes a simulated trap to entice and capture cyber assailants. Organizations can employ a honeypot to identify, divert, or analyze efforts to illicitly access information systems. Integrating machine learning (ML) into honeypot systems can improve their efficacy by facilitating immediate identification of threats, adaptive reactions, and more comprehensive understanding of assault patterns. Essential Elements of Honeypot Framework Setting up the Honeypot are as below:

a. *Decoy Systems*: Emulate susceptible systems to lure in attackers

b. *Network Services*: Operate commonly targeted services, such as SSH, FTP, HTTP, and others.

c. *Logging and Monitoring*: Gather information on every contact with the honeypot.

d. *Alerting*: Notify security teams immediately upon detection of a potential assault. Dynamic Adaptation involves modifying honeypot configurations in response to certain attack types, such as altering IP addresses or simulating various vulnerabilities.

e. *Threat Intelligence*: Disseminate knowledge to wider security networks to enhance collective defense.

f. *Model Evaluation*: Regularly evaluate the effectiveness of machine learning models by measuring measures such as accuracy, recall, and F1 score.

g. *Feedback Loop*: Utilize recently acquired data from identified assaults to retrain and enhance the models.

3.4 Research Resource: Evaluation Criteria for the deployment

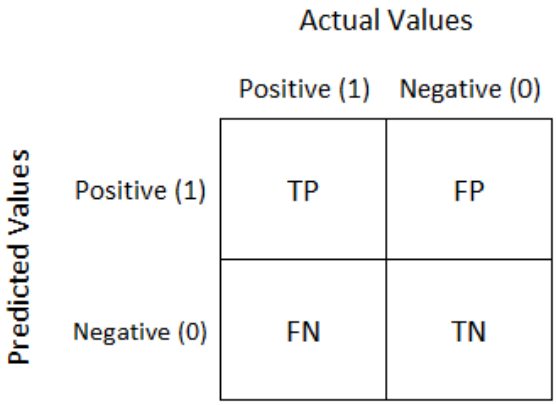

Figure 14: Confusion Matrix depicting the True Positive, False Positive, False Negative and True Negative (**Source: TowardsDataScience**)

True Positive means our algorithm has predicted positive and the actual label or class is also positive. Similarly, the true negative. The False Negative which is also the Type 2 error (Type II) is when the model predicted negative and its wrong while the False positive which is also the Type 1 error (Type I) is when the model predicted positive and its false.

$$Recall = \frac{TP}{TP + FN}$$

The above equation can be explained by saying, from all the positive classes, how many we predicted correctly. Recall should be high as possible.

$$Precision = \frac{TP}{TP + FP}$$

The above equation can be explained by saying, from all the classes we have predicted as positive, how many are actually positive. Precision should be high as possible. Accuracy is from all the classes (positive and negative), how many of them we have predicted correctly. Accuracy should be high as possible.

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

It is difficult to compare two models with low precision and high recall or vice versa. So to make them comparable, we use F-Score. F-score helps to measure Recall and Precision at the same time. It uses Harmonic Mean in place of Arithmetic Mean by punishing the extreme values more.

3.5 Research Resource: Cloud Framework

Currently deploying this solution in a local framework. This honeypot using the machine learning based solution can be made available in the cloud server like AWS. The whole response system which comprises of two parts i.e. the trained ML algorithm and the Honeypot algorithm can be used an API and can be integrated to any iOT devices as required. This will help in mitigating the entry of any unauthorised entry.

In this the criteria for success is to check the performance of the model in detecting the vulnerability and the storage of the solution and the RAM usage while the solution works.

Chapter 4: Implementation

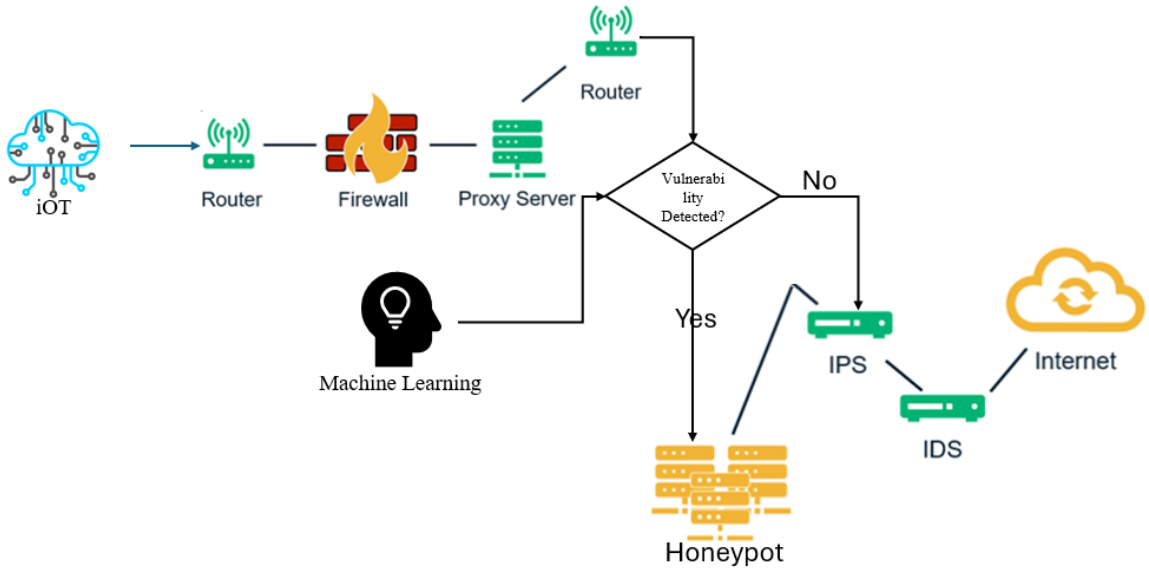

Figure 15: Proposed Framework for the automated deploying the Honey pot when ever there is a probability of the intrusion detection

4.1 Algorithm Flow

Step 1: Dataset Acquisition – The data is fixed after the detailed literature review so that those features can be easily extracted from the real time data.

Step 2: Data Exploration and Data Understanding – Understanding the data and to take actions on the data cleaning. Most important are,

a. Missing Values Imputation: Understanding the analysis of the missing values and imputing them We have used median for the continuous and mode for the categorical
b. Outliers Imputation

Step 3: Data Sampling – Even though the data has an evenly poised target variable, different feature engineering techniques like SMOTE, ADASYN and stratification is implemented. For this dataset the stratification is used. The code is made generic so that if the data is not evenly balanced the best algorithm from SMOTE and ADASYN is implemented.

Step 4: Training and Testing Set – In this case Holdout method is used for test ration 20%

Step 5: Machine Learning Modelling – The best model is selected and saved as the pickle file.

Step 6: Real Time honeypots Deployment – The trained machine learning model detects the real time data and in case the detection is found to be suspicious, honeypot is deployed.

Step 7: Set up Honeypot Environment – The decoy systems and services using the python is deployed. This ensures comprehensive looging and monitoring.

Step 8: ML saved model Integration – The saved model is deployed in the honeypot environment that enables real time data analysis and threat detection.

Step 9: Automated Responses – As the above flow talsk about te logging and generating the report for the suspicious attack.

Step 10: Continuous Monitoring – The system monitors and trains the ML models based on the learning and updates the saved model.

4.2 Dataset Description

The IoT DDoS Honeypot Dataset (Source Link: (https://data. mendeley. com/datasets/8dns3xbckv/1) which is a compilation of information to enhance the understanding of the researches on the DDoS attack exclusively targeting IoT devices. This dataset should comprised of a great deal of data such as; network traffic logs, packet captures and system level statistics. This information is created as IoT honeypots are run. Paragraph one of the dataset seeks to record all the methods and protocols used by evil people to compromise the IoT traffic and flood it with undesired traffic hence causing a denial of service. The collected data includes the information from the Internet of Things (IoT), collected by honeypot – the intentionally built system to becomes the target of cyberattacks to expose actions, methods and procedures used by the attackers. The main purpose of this data gathering is to carry out observations of the attack traffic and identify potential threats as well as the techniques and approaches used by the attackers. Actually, the data is stored on this program in the PCAP data format, and after that, the data is converted to the CSV format for further convenient retrieval of the data. The dataset has 10 essential attributes: dt, duration in usec, duration in nsec, total duration, packet per second, protocol, port no, transmitted kbps, received kbps, total kbps and name. Here, the specifics of the features used are outlined,

Table 1: Dataset description

<class 'pandas.core.frame.DataFrame'>			
RangeIndex: 103839 entries, 0 to 103838			
Data columns (total 23 columns):			
#	Column	Non-Null Count	Dtype
---	------	--------------	-----
0	dt	103839 non-null	float64
1	switch	103839 non-null	int64
2	src	103839 non-null	object
3	dst	103839 non-null	object
4	pktcount	103839 non-null	int64
5	bytecount	103839 non-null	int64
6	dur	103839 non-null	float64
7	dur_nsec	103839 non-null	int64
8	tot_dur	103839 non-null	float64
9	flows	103839 non-null	int64
10	packetins	103839 non-null	int64
11	pktperflow	103839 non-null	float64
12	byteperflow	103839 non-null	float64
13	pktrate	103839 non-null	float64
14	Pairflow	103839 non-null	int64
15	Protocol	103839 non-null	object
16	port_no	103839 non-null	int64
17	tx_bytes	103839 non-null	int64
18	rx_bytes	103839 non-null	int64
19	tx_kbps	103839 non-null	float64
20	rx_kbps	103839 non-null	float64

21	tot_kbps	103839 non-null	float64
22	label	103839 non-null	float64
dtypes: float64(10), int64(10), object(3)			
memory usage: 18.2+ MB			

In the above DDoS dataset, the label is the target variable in the same, the distribution of the both classes are as below. We can see that although the classes are not highly imbalance, we will use different class imbalance techniques to learn and execute.

Table 2: Class distribution for the both class. The 0 is correct and no attack while 1 means attack probable

0.0	63335
1.0	40504
Name: count, dtype: int64	

Missing values in each column:

Table 3: Missing values finding for each of the features

dt	0
switch	0
src	0
dst	0
pktcount	0
bytecount	0
dur	0
dur_nsec	0
tot_dur	0
flows	0
packetins	0
pktperflow	0
byteperflow	0
pktrate	0
Pairflow	0
Protocol	0
port_no	0
tx_bytes	0
rx_bytes	0
tx_kbps	0
rx_kbps	0
tot_kbps	0
label	0
dtype: int64	

In the above table we find that there is no missing values in the datset and hence we don't need any kind of missing values imputation.

Table 4: Dataset statistical distribution

	dt	switch	pktcount	bytecount	\
count	103839.000000	103839.000000	103839.000000	1.038390e+05	
mean	17999.454165	4.214496	52781.703165	3.801729e+07	
std	11962.227566	1.956320	52061.409178	4.874544e+07	
min	2488.000000	1.000000	0.000000	0.000000e+00	

25%	7128.000000	3.000000	801.000000	7.928200e+04
50%	11965.000000	4.000000	42557.000000	6.409692e+06
75%	29982.000000	5.000000	94683.000000	7.597222e+07
max	42935.000000	10.000000	260006.000000	1.471280e+08

	dur	dur_nsec	tot_dur	flows	\
count	103839.000000	1.038390e+05	1.038390e+05	103839.000000	
mean	322.126118	4.613734e+08	3.225151e+11	5.666339	
std	283.926141	2.771020e+08	2.838104e+11	2.951013	
min	0.000000	0.000000e+00	0.000000e+00	2.000000	
25%	128.000000	2.330000e+08	1.280000e+11	3.000000	
50%	252.000000	4.180000e+08	2.520000e+11	5.000000	
75%	413.000000	7.040000e+08	4.130000e+11	7.000000	
max	1881.000000	9.990000e+08	1.880000e+12	17.000000	

	packetins	pktperflow	byteperflow	pktrate	\
count	103839.000000	103839.000000	1.038390e+05	103839.000000	
mean	5196.261308	6365.870588	4.689472e+06	211.682248	
std	5266.631950	7413.263909	7.562023e+06	247.138265	
min	4.000000	-130933.000000	-1.464426e+08	-4365.000000	
25%	1943.000000	29.000000	2.842000e+03	0.000000	
50%	3024.000000	8304.000000	5.447360e+05	276.000000	
75%	7385.000000	10004.000000	9.727070e+06	333.000000	
max	25224.000000	19190.000000	1.495387e+07	639.000000	

	Pairflow	port_no	tx_bytes	rx_bytes	\
count	103839.000000	103839.000000	1.038390e+05	1.038390e+05	
mean	0.603916	2.337580	9.370704e+07	9.373492e+07	
std	0.489085	1.082973	1.521679e+08	1.331642e+08	
min	0.000000	1.000000	2.527000e+03	8.560000e+02	
25%	0.000000	1.000000	4.799000e+03	3.539000e+03	
50%	1.000000	2.000000	4.552643e+06	1.401134e+07	
75%	1.000000	3.000000	1.356509e+08	1.439286e+08	
max	1.000000	5.000000	1.269982e+09	9.905962e+08	

	tx_kbps	rx_kbps	tot_kbps	label
count	103839.000000	103839.000000	103839.000000	103839.000000
mean	1003.767322	1003.811420	2007.578742	0.390065
std	2428.363391	2054.887034	3144.437173	0.487767
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	4.000000	0.000000
75%	255.000000	557.000000	3838.000000	1.000000
max	20580.000000	16577.000000	20580.000000	1.000000

From the above graphs each individual plot represents the individual statistical description. This talks about the extreme values and minimum values along with how the values are distributed. This helps us understanding what kind of feature engineering needs to be handled for each of the dataset.

4.2 Data Analysis

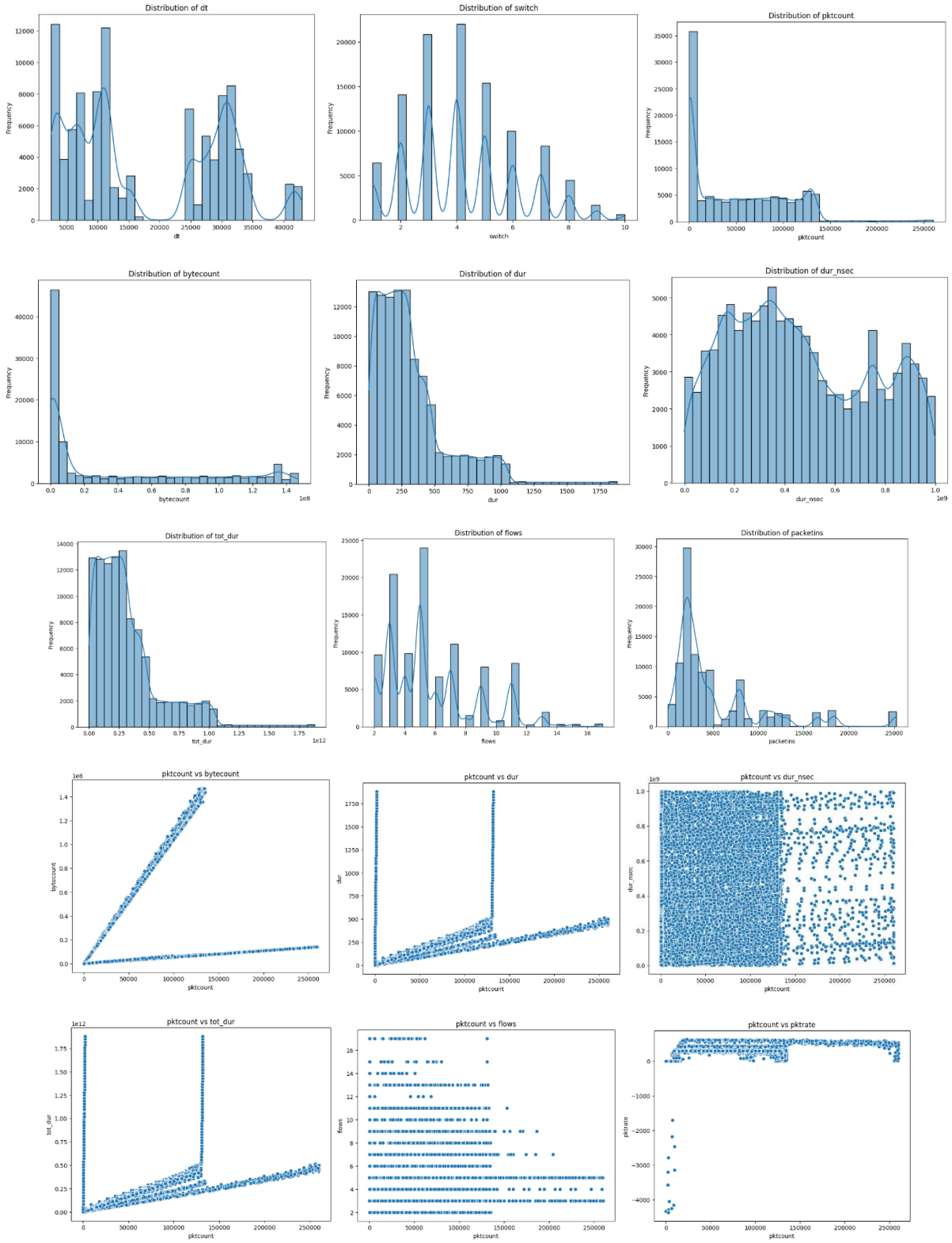

Figure 16: Features distribution graphically

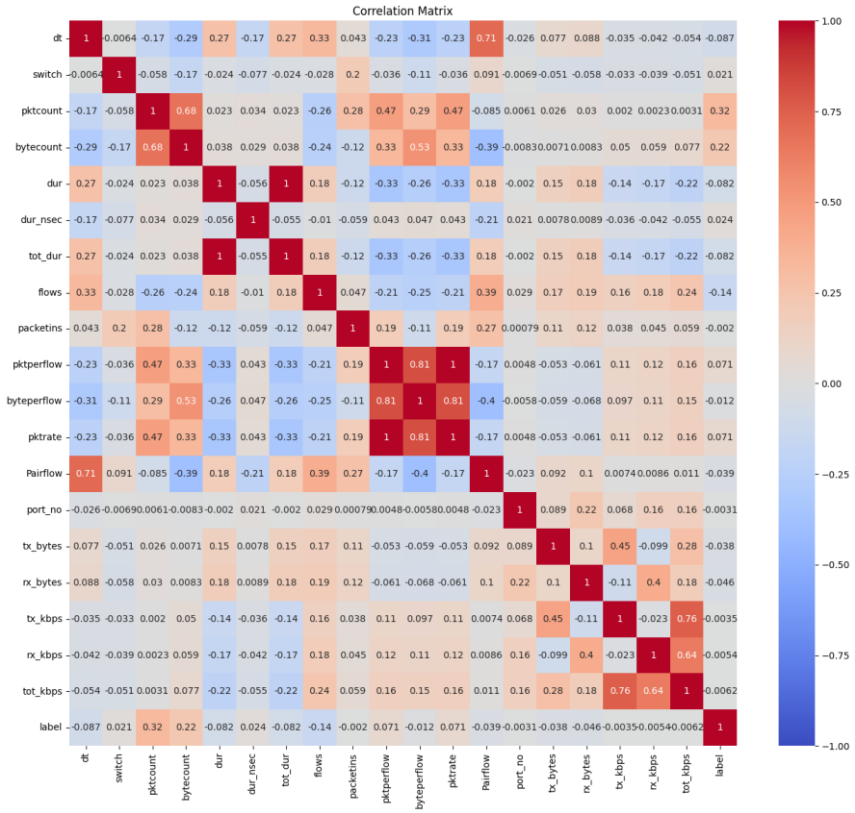

Figure 17: Correlation plots

From the above we have derived, The Pearson correlation coefficient (PCC), which is a measure of the extent to which two set of data are related to each other. The correlation coefficient is a statistic that describes the extent of the linear relationship between two variables defined as the ratio of covariance between the two variables to the product of their standard deviations. It gives a standardized value ranging from -1 to +1, to show the intensity and type of relationship. Like covariance, this measure is also limited only to show the linear relationship between the variables and cannot depict any other types of relationships and correlations. For example, when comparing a set of data points containing some primary school students, one would expect the value of the correlation coefficient between the students' age and height to be more than 0, but less than 1 (as a correlation coefficient of 1 is practically impossible).

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y}$$

where

- cov is the covariance
- σ_X is the standard deviation of X
- σ_Y is the standard deviation of Y.

$$\operatorname{cov}(X,Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)],$$

the formula for ρ can also be written as

$$\rho_{X,Y} = \frac{\mathbb{E}[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$

where

- σ_Y and σ_X are defined as above
- μ_X is the mean of X
- μ_Y is the mean of Y
- $\mathbb{E}$ is the expectation.

From the above correlation matrix, it is found that the Highly correlated feature pairs with correlation coefficient above 0.5

```
('dt', 'Pairflow', 0.7130594434882578)
('pktcount', 'bytecount', 0.6757916735819185)
('bytecount', 'byteperflow', 0.5332008174381108)
('dur', 'tot_dur', 0.9999983664257023)
('pktperflow', 'byteperflow', 0.8129399149230081)
('pktperflow', 'pktrate', 0.9999988366109656)
('byteperflow', 'pktrate', 0.8129396244162204)
('tx_kbps', 'tot_kbps', 0.757079709029326)
('rx_kbps', 'tot_kbps', 0.6355446801842544)
```

4.3 Honeypot-network

```
import logging
from datetime import datetime

# Set up logging
logging.basicConfig(filename='honeypot.log', level=logging.INFO,
format='%(asctime)s - %(message)s')

def monitor_access():
    """
    Monitor access attempts and trigger the honeypot on unauthorized
access.
    """
    authorized_users = ['user1', 'user2', 'admin']  # Replace with
actual authorized users
    access_log = [
        {'user': 'user1', 'action': 'login', 'status': 'success'},
        {'user': 'intruder', 'action': 'login', 'status': 'failed'},
        # Add more access attempts as needed
    ]

    for attempt in access_log:
        user = attempt['user']
        status = attempt['status']

        if user not in authorized_users and status == 'failed':
            logging.warning(f'Unauthorized access attempt by {user}.')
```

```
            honeypot_triggered(user)
        else:
            logging.info(f'Authorized access by {user}.')

def honeypot_triggered(intruder):
    """
    Honeypot function triggered on unauthorized access.
    """
    logging.info(f'Honeypot triggered by {intruder}. Deploying
countermeasures.')
    # Simulate successful deployment of countermeasures
    deploy_countermeasures(intruder)

def deploy_countermeasures(intruder):
    """
    Simulate the deployment of countermeasures.
    """
    response = f'Successfully deployed countermeasures against
{intruder}.'
    logging.info(response)
    print(response)

if __name__ == "__main__":
    monitor_access()
```

Code Snippet 1: HoneyPot code for access monitoring

This Python script defines a rudimentary honeypot setup that involves logging control for supervising the attempts at accessing the honeypot and triggering countermeasures in cases of unauthorized access. The program employ the services of the logging module to help it capture events and these are stored in a file known as honeypot. log. The access attempts are documented with a time stamp and the message that describes the sort of access and activities occurring next during the event. The monitor_access function acts as the focus of this system because it will be used for granting or denying access to the system. The system imitates the process of observing login attempts as through comparison of an access logs list with a prescribed list of permitted users. The access logs themselves remain as a list of dictionaries built from each username, the action that was attempted (for example, 'login'), along with the result of the attempt (for example, 'success', 'failure'). When a user that is not permitted tries to login into the system, and results a failure, then the system registers a warning and sets the honeypot. In the case of an undesirable attempt to traffic through the honeypot, the honeypot_triggered function is called. This function logs the activity in the honeypot and then puts up defenses. The deploy_countermeasures function emulates the procedure of countermeasures implementation. After executing the push operation it stores a message to show that the configuration has been pushed and successfully deployed and this message appears on the console. This is meant to be run as a separate process. When the program is run it will process the simulated access logs, logging legitimate accesses, a log failed attempts and how the program will mimic the triggering of the honeypot defenses to the attackers. Thus, this framework devotes an essential technique in honeypot functionality which includes identification of any malicious actions and a prompt reaction to them, while at the same time recording the events as a means of surveillance and assessment.

Chapter 5: Results and Analysis

5.1 Experiments

The machine learning code for different model is performing as below. In this figure we have shown how the models are performing in the part of accuracy, precision, recall, f1-score, precision for minority. In tsi case the minority is the suspicious attack.

Table 5: Comparative Analysis of different machine learning models

Model	Accuracy	Precision	Recall	F1 Score	Precision Minority
Decision Tree	0.902591	0.874261	0.876312	0.875285	0.874261
Neural Networks	0.640312	0.539992	0.525861	0.532833	0.539992
Random Forest	0.901002	0.870359	0.876805	0.873570	0.870359
LightGBM	0.904372	0.877423	0.877423	0.877423	0.877423
XgBoost	0.904083	0.876773	0.877423	0.877098	0.876773
SVM	0.611662	0.528754	0.040859	0.075857	0.528754

We found that Light Gradient Boosting algorithm performed the best. In this the accuracy was achived to be 90.43% followed by XgBoost with 90.4%. Thus saying that boosting methods performed well. But we are more interested with the mnortiy precision i.e in case of the malicious attack detected or not. In this LGBM performed the best.

Table 6: Recall and F1-Score comparison for different Models

Model	Recall Minority	F1 Score Minority
Decision Tree	0.876312	0.875285
Neural Networks	0.525861	0.532833
Random Forest	0.876805	0.873570
LightGBM	0.877423	0.877423
XgBoost	0.877423	0.877098
SVM	0.040859	0.075857

When checked the recall and f1-score the importance is given to the f1 score of the minority class detection. LGBM is the best model to be selected followed by xgboost with 87.742% and 87.7% f1 score respectively.

5.2 Confusion Matrix and Deep Dive for different models

5.2.1 Decision Tree

The Decision Tree model achieved fair accuracy for both classes, with a total of 90% accuracy; the model achieved good precision, a recall rate, and F1 scores. Specifically, the quality of the proposed model showed 88% of recall and 92% of precision for the default class (1.0) and 0. 0 for the non-default or correct data class and default or suspicious class. This therefore speaks volume on how well the Decision Tree can perform in this capacity, that is distinguishing between default and not. The performance is equally satisfying in both groups, which makes the model's ability to predict loan defaults reliable. Some of the key findings include high accuracy that suggest that the model is fit for use in cases where both default sensitivity and specificity is desired as evidenced by the models' balanced precision and recall measurements.

5.2.2 Random Forest

Similar to the Decision Tree, good accuracy of the Random Forest model was reported, with precision and recall of 92% and 88%, for non-default and default classes respectively. Also, concerning the accuracy of total number of predicted result, it was 90 percent. Less likely to overfit than a single decision tree because of randomness, Random Forest is a more reliable decision making model as it makes its decision based on several decision trees. Such an approach is most fitting in places where there is a need to minimize variation and enhance the model's ability to generalize.

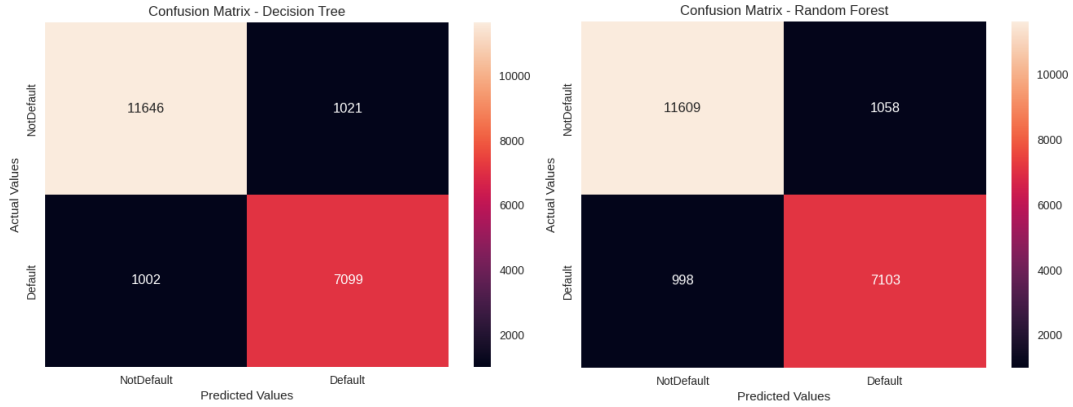

Figure 18: Confusion Matrix for Decision Trees and Random Forest

5.2.3 SVM

The accuracy of the SVM model was 61% over all the folds which is significantly higher than all the tree based models. The default class now yielded somewhat lower precision at 53%, the non-default class was again slightly higher at 61%. The recall for the default class, nonetheless, was relatively lower at 4 percent, indicating the fact that the SVM had poor ability to correctly predict the instances that belong to the default class. This understandably led to a degradation of the default class's F1-score. The difference in their performances is evident of the challenges which SVM experiences while working on imbalanced data, as it is inclined to favor the large class, which makes detection of a small class practically impossible.

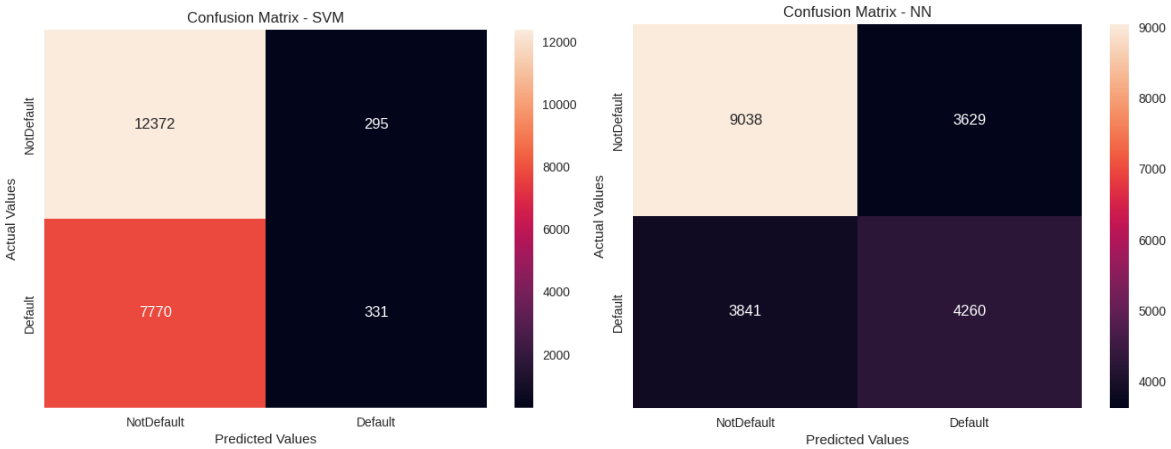

Figure 19: Confusion Matrix for SVM and NN (down)

5.2.4 XGBoost

To provide the evaluation, XGBoost model achieved the precision and recall at the level of 92% for the non-default class and 88% for the default class with the accuracy of 90% in total. Since F1-scores for both classes are high, it can be concluded that XGBoost handles class imbalance problem effectively as

well as maintaining relative high predictive accuracy. The gradient boosting method of tackling errors of previous iterations makes it the ideal system for massive stages of predictive modelling, especially in cases of over-complicated datasets that are likely to be imbalanced.

5.2.5 LightGBM

For LightGBM, thorough examination of the results also yielded an overall accuracy of 90% with a good precision and recall for the non-default class and default class at 92% and 88%, respectively. LightGBM is suitable for real time prediction because of the high accuracy rate, efficiency and speed it which operates on very large datasets. That is why LightGBM can be a rather reliable answer to the demands of applications, where high accuracy along with fast computation are required, as well as the performance is approximately the same for both classes.

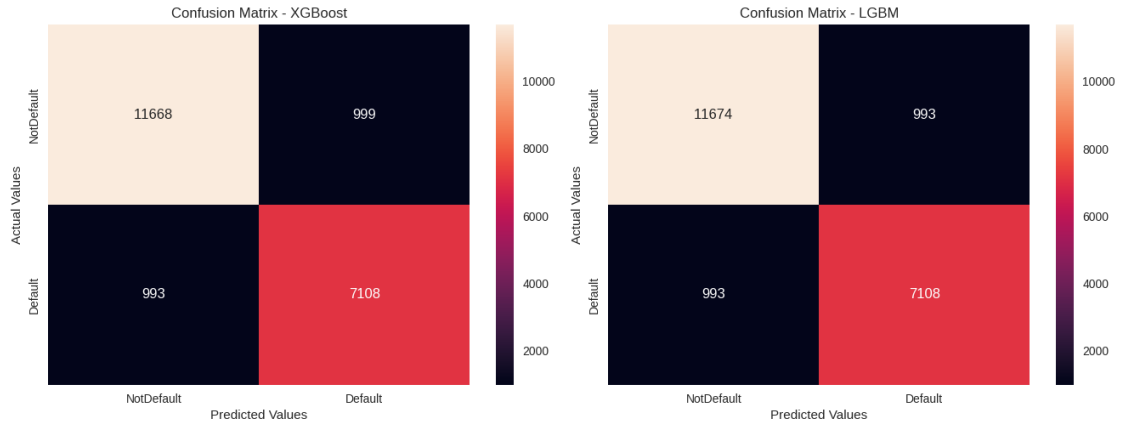

Figure 20: Confusion Matrix for XGBoost and Light Gradient Boosting

5.2.6 Neural Networks

With precision and recall scores of 70% and 71% for the non-default class and 54% and 53% for the default class, the Neural Network model yielded an overall accuracy of 64%. The neural network demonstrated a reasonable capacity to predict both classes, despite its performance being inferior to that of the boosting techniques and tree-based models. Complex patterns in data can be effectively processed by neural networks, but the architecture and training procedure have a significant impact on the network's effectiveness. The comparatively poorer performance in this instance can indicate that more complex architectures or fine-tuning of the network's settings are required to increase its capacity to handle unbalanced input.

5.3 Deployment of Honeypots

In case an authorised access is attempted, the following error is shown,

```
WARNING:root:Unauthorized access attempt by intruder.
Successfully deployed countermeasures against intruder.
```

Figure 21: Warning Alert in case of an authorised access

In order to check the model performance in the real time, we will deploy the trained model in the real time.

5.3.1 Case Study 1: In case of the Normal attempt

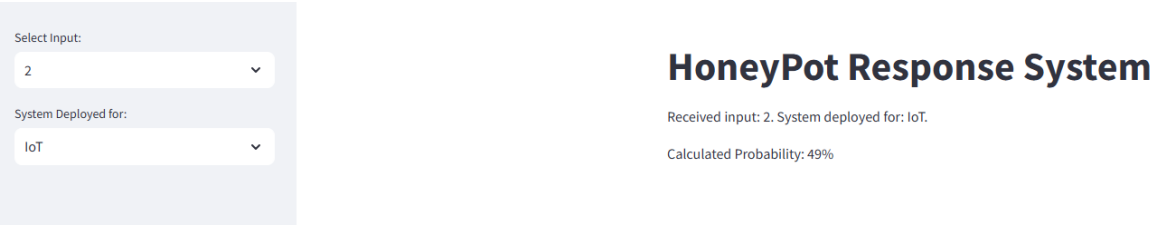

Figure 22: Deployment of the solution when the Normal solution is detected

In the scenario when the normal solution is detected, as can be seen above, no Honeypot was deployed since the probability is less than 50%. We have kept the threshold as 75% so that only the confident scenarios where the solution is deployed.

5.3.2 Case Study 2: In case of an unauthorised access

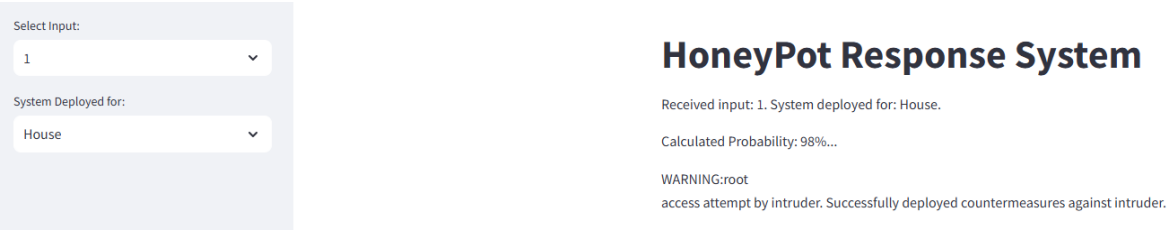

Figure 23: Deployment of the solution when the unauthorised access is detected

In the scenario when the unauthorised access is detected, as can be seen above, Honeypot was deployed since the probability is 98%. We have kept the threshold as 75% so that only the confident scenarios where the solution is deployed. In this case the LGBM which is the best machine learning model that has been deployed as for the detection purpose identified the vulrenability and hence the protection measure is deployed. If we check the task manager not much of the space is utilized, hence this solution is very beneficial.

Chapter 6: Conclusion and Future Work

This paper shows that honeypots play a central role in improving the security of the Internet of Things (IoT) using advanced machine learning and deep learning methods. The study also carried out numerous tests and realized that LightGBM as well as XGBoost are the best-suited models to detect honeypot. These models produced high levels of accuracy, precision, and recall with the evaluated documents. Such models help in providing accurate results in terms of unauthorized access and the positioning of honeypots, thereby solving the problems of random resource misuse and high false positives. Honeypots are significant parts of the modern security strategies. They attract and capture hostile activity, which helps analyze the actions and intents of the attackers. This level of intelligence is crucial for primary determine and fight menace factors within the society. In addition, honeypot helps to reduce the number of false alarms that are produced by the security devices which in turn increases the overall efficiency and accuracy of security solutions, providing legal activities do not interfere. This work argues that the various kinds of honeypots low, high and hybrid; should be placed appropriately to enhance their operations. When placed correctly, honeypots enhance the ability to identify threats, decrease the number of false alarms, and automatically outline the response measures, which makes honeypots valuable tools in the sphere of cybersecurity.

Future research needs to focus on getting more datasets with classification as IoT honeypots and exploring further opportunities for their usage in varying fields: healthcare, smart electric grids, smart homes, etc. Besides, the large scale functionality and real-time effectiveness of such solutions should

also be considered for their easy acceptance and integration. The applied research of integration of the honeypot systems with AI/ML techniques reveal the promising perspective to enhance IoT devices security and counteract with the constantly emerging cyber threats. In conclusion, this study implies the usefulness of the honeypots, developed and tuned with modern machine learning algorithms, for the aim of providing an efficient response to the possible security threats in the IoT environment. Other continued advancements and most importantly their deployment will be of great importance in safeguarding of critical infrastructures against increasingly sophisticated cyber threats.

References

Ahmed, Y., Beyioku, K. and Yousefi, M., 2024. Securing smart cities through machine learning: A honeypot-driven approach to attack detection in internet of things ecosystems. IET Smart Cities, 2024.

Ali, J., Khan, R., Ahmad, N. and Maqsood, I., 2022. Random forests and decision trees. International Journal of Computer Science Issues (IJCSI), 9.

Amal, M.R. and Venkadesh, P., 2023. H-doctor: Honeypot based firewall tuning for attack prevention. Measurement: Sensors, 25, p.100664.

Arora, S., Kumar, M., Johri, P. and Das, S., 2016. Big heterogeneous data and its security: A survey. [online] Available at: <link> [Accessed 11 August 2024].

Baykara, M. and Das, R., 2015. A survey on potential applications of honeypot technology in intrusion detection systems. Journal Name, 2, pp.10.

Chen, T. and Guestrin, C., 2016. Xgboost: A scalable tree boosting system. In: Proceedings of the Conference Name, pp.785–794.

Devi Priya, V.S. and Chakkaravarthy, S.S., 2023. Containerized cloud-based honeypot deception for tracking attackers. Scientific Reports, 13(1), p.1437.

Evgeniou, T. and Pontil, M., 2001. Support vector machines: Theory and applications. In: Proceedings of the Conference Name, volume 2049, pp.249–257.

Gharbi, C., Hsairi, L. and Zagrouba, E., 2021. A secure integrated fog cloud-iot architecture based on multi-agents system and blockchain. In: Proceedings of the Conference Name, pp.1184–1191.

Ghorbani, H. and Ahmadzadegan, M., 2017. Security challenges in internet of things: survey. [online] Available at: <link> [Accessed 11 August 2024].

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y., 2017. Lightgbm: A highly efficient gradient boosting decision tree. [online] Available at: <link> [Accessed 11 August 2024].

Lanka, P., Gupta, K. and Varol, C., 2024. Intelligent threat detection—ai-driven analysis of honeypot data to counter cyber threats. Electronics, 13(13), p.2465.

Makhdoom, I., Abolhasan, M., Abbas, H. and Ni, W., 2019. Blockchain's adoption in iot, 2019: The challenges, and a way forward. Journal of Network and Computer Applications, 125, pp.251–279.

Mokhtari, S., Abbaspour, A., Yen, K. and Sargolzaei, A., 2021. A machine learning approach for anomaly detection in industrial control systems based on measurement data. Electronics, 10, p.407.

Morozov, D.S., Vakaliuk, T.A., Yefimenko, A.A., Nikitchuk, T.M. and Kolomiiets, R.O., 2023. Honeypot and cyber deception as a tool for detecting cyber attacks on critical infrastructure. In: Proceedings of the Conference Name, pp. (if available).

Popescu, M.-C., Balas, V., Perescu-Popescu, L. and Mastorakis, N., 2009. Multilayer perceptron and neural networks. WSEAS Transactions on Circuits and Systems, 8, pp.07.

Sarhaddi, F., Azimi, I., Labbaf, S., Niela-Vilen, H., Dutt, N., Axelin, A., Liljeberg, P. and Rahmani, A.M., 2021. Long-term iot-based maternal monitoring: System design and evaluation. Sensors, 21, p.2281.

Schmitt, M., 2023. Securing the digital world: Protecting smart infrastructures and digital industries with artificial intelligence (ai)-enabled malware and intrusion detection. Journal of Industrial Information Integration, 36, p.100520.

Sharma, D., Singh, A., Chitkara, S. and Sharma, T., 2023. Honeypot networks in deception technology for iot devices. In: Proceedings of the Conference Name, pp.1156–1162.

Sicari, S., Rizzardi, A., Grieco, L.A. and Coen-Porisini, A., 2015. Security, privacy and trust in internet of things: The road ahead. Computer Networks, 76, pp.146–164.

Xu, L., He, W. and Li, S., 2014. Internet of things in industries: A survey. IEEE Transactions on Industrial Informatics, 10, pp.2233–2243.

Yang, X., Yuan, J., Yang, H., Kong, Y., Zhang, H. and Zhao, J., 2023. A highly interactive honeypot-based approach to network threat management. future internet, 15(4), p.127.

Yang, Y., Wu, L., Yin, G., Li, L. and Zhao, H., 2017. A survey on security and privacy issues in internet-of-things. IEEE Internet of Things Journal, 4(5), pp.1250–1258.

Yu, W., Liang, F., He, X., Hatcher, W., Lu, C., Lin, J. and Yang, X., 2017. A survey on the edge computing for the internet of things. IEEE Access, pp.1–1..