Ensemble Techniques to Enhance Wireless Intrusion Detection System In IoT

Research Project
Msc Cybersecurity

Viknesh Aditya Rajendran
Student ID: 19216343

School of Computing
National College of Ireland

Supervisor: Imran Khan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Viknesh Aditya Rajendran		
Student ID:	19216343		
Programme:	Msc Cybersecurity	**Year:**	2021 -2022
Module:	Research Project		
Supervisor:	Imran Khan		
Submission Due Date:	15/08/2022		
Project Title:	Ensemble Techniques to Enhance Wireless Intrusion Detection System In IoT		
Word Count:	**7632 Page Count 21**		

Signature: Viknesh Aditya Rajendran

Date: 15.08.2022

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 keep a copy on computer.	□

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

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

Ensemble Techniques to Enhance Wireless Intrusion Detection System In IoT

Viknesh Aditya Rajendran

X19216343

Abstract:

The Internet of Things (IoT) is rapidly expanding to have a greater impact on anything from everyday life to enormous industrial activities. The fast expansion of the internet of things has resulted in serious security issues, such as cyber-attacks conducted by incredibly large botnets made up of IoT devices. In order to monitor the network flow on IoT networks intrusion detection systems are very much critical, because they establish a protected traffic condition and safeguards against malicious traffic packets. However, it's still challenging to detect whether the packet is being malicious or benign and to classify the botnets types, if they are malicious. Most studies are restricted to using IP addresses to categorize just well-known botnets like Mirai, Bashlite etc, But the drawbacks are that the IP addresses can be spoofed. In this research paper, an ensemble model-based intrusion detection system (IDS) was built to determine whether packets are malicious or benign and to categorize botnet types based on md5 hash values. However, evaluation was performed to analyze whether the hash value is dangerous or benign, as well as categorize the kind of botnets and other malware, by integrating the outputs of Light GBM classifier and CatBoost classifier via soft voting classifier. Key metrics such as accuracy, F1-score, Recall and False Alarm Rate were evaluated. 99 percent accuracy has been achieved in classifying whether the packet is benign or malicious and 93 percent accuracy has been attained in classifying the botnet types and the rate of false alarm has been attained to 0.5 percent.

Keywords: IoT Botnets, LightGbM, CatBoost, Svirtu-AA, Soft Voting classification, Mirai

1. Introduction:

The Internet of Things (IoT) is the concept of connecting any device to the internet and other devices. All network devices connect with one another in order to acquire and share data. Numerous gadgets are connected to the internet via the Internet of Things (IoT) technology, which is altering civilizations, industries, and social units on a global basis. IoT has connected more than 50 billion devices, and the number has been growing exponentially. Globally, IoT networks with coerced assets have been shown to have highly compressed connectivity(David,2021). According to the International Data Corporation, there will be 41.6 billion linked IoT devices and 79.4 zettabytes of data generated in 2025, compared to an anticipated 8.1 billion individuals(Atlam and Wills, 2020). IoT integration with constantly growing and evolving software, like big data and 5G, is also receiving a lot of attention. The Internet of Things (IoT) ecosystem is growing in breadth, applications, connectivity, and complexity, and as a result, it is more crucial than ever to safeguard IoT components at all tiers, from the physical to the user(Alhowaide, Alsmadi and Tang, 2021).The ubiquity of susceptible Internet of Things (IoT) devices with high computing capabilities makes them an accessible and enticing target for attackers aiming to hack these devices and use them to establish large botnets.

Botnet is self-replicating malware capable of autonomously detecting and attacking Internet-connected susceptible machines. Devices that have been compromised become part of a botnet, or network of controlled devices. Various IoT devices, such as routers, IP cameras, smart home appliances, etc., can become compromised by an IoT botnet. The hacked devices are also referred to as "IoT bots" in the case of the dispersed bots, which are controlled by a command-and-control server and engage in peer-to-peer (P2P) communication with one another. A botnet may be made lucrative by a variety of techniques, including cryptocurrency mining, identity theft, and DDoS attacks. Scanners and loaders are used by bots to carry out their operations. First, bots will search for IoT devices with susceptible ports, then the vulnerable devices will be frisked by swapping messages to ensure they are vulnerable. In order to mislead the susceptible device into installing the malware that was received from the malware distribution server, the botnet herder will attempt to transmit the command to the bots through C2 server or to the server of the loader. The compromised IoT device will then be connected to the C2 server and wait for instructions (Trajanovski and Zhang, 2021).

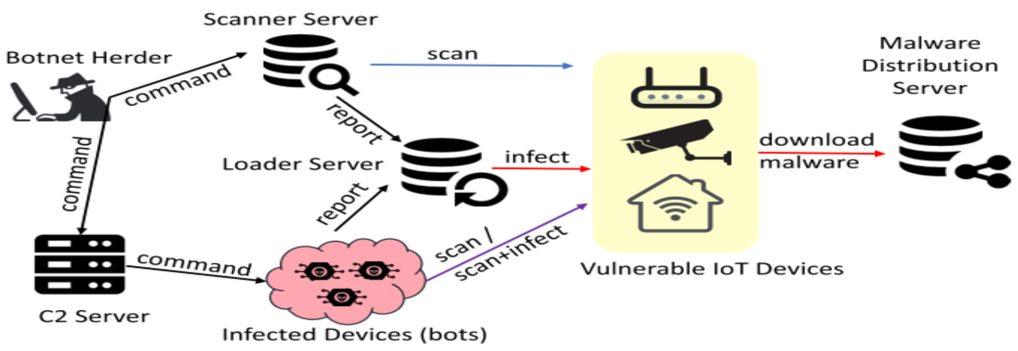


Figure 1: Working of Botnet

However, IoT botnets have the potential to quickly expand in size and carry out highly damaging operations. In recent times, The Satori botnet grabbed 280,000 bots in a period of 12 hours using Mirai source files. It is believed that between 800,000 and 2,500,000 IoT devices throughout the world have been infected by the Mirai IoT botnet and its variations. Researchers discovered another Mirai variation called Okiru, whose primary goal is to infect ARC processors and has the capacity to penetrate up to 1.5 billion devices (*IoT Botnets Are Evolving – How Big Can They Get? - Secplicity - Security Simplified*, no date). Large websites including Twitter, Netflix, Reddit, and GitHub went offline for many hours in October 2016 as a result of a DDoS attack launched by the Mirai botnet targeting the services dynamic DNS (Kolias *et al.*, 2017). Trend Micro found 122,069 hacked IP cameras worldwide after the Persirai botnet targeted 1,000 vulnerable camera types.

1.1 Working of Mirai botnet:

Step1: Brute Force attack will be initiated by the bot over to the new victim of the IoT device, this is done in order discover the vulnerable device's default passwords. Secondly shell will be created after identifying the correct password. Through a separate port, the bot transmits characteristics of the devices to the report server.

Step2: By getting connected through C & C, the botmaster will inspect on the status of the botnet via report server. Secondly, the botmaster will send an infect command to the loader which comprises of IP addresses, architecture etc.

Step 3: Over in the target device the loader gets connected and it instructs to install and execute the malicious binary. The freshly acquired bot instance may accept attack instructions from C&C. The programme resolves a hardcoded domain name. Hence, the botmaster acquired the privilege of spoofing the IP address

Step4: The botmaster instructs all bot instances to attack the target server by delivering a command via the C&C server with the kind and duration of attack and the bot instances' and target server's IP addresses. (Bertino, 2017).

1.2 Effects of Botnets:

Mirai-GH: Performs injection in a remote process using CreateRemoteThread, calls to the same API frequently in an effort to slow down analysis

Svirtu-AA: performs DDOS and It essentially compromises the compromised system by carrying out orders from a remote, malicious user.

Mirai-APD: It will also performs DDOS and it will allows for illegal access to the infected machine by a remote user. (*Threat Encyclopedia | FortiGuard*, 2022)

In order to overcome this problem, there are certain established solutions, such as antivirus software, data-encryption methods, firewalls, and IDS, etc., to lessen the impact of the IoT botnets malicious conduct. An intrusion detection is a system that observes a network and alerts the user if it detects any harmful packets. IDS tracks the system activity, thus they will look into application weaknesses and find customary behaviour as well as data injection in a network. IDS's primary goal is to detect harmful network traffic and unauthorized network use, which is impossible with a conventional firewall. When an intrusion is discovered, the system alerts the administrator by raising an alarm. In order to achieve this, it regularly gathers data from various systems and network sources, examines the data, and looks for any potential security flaws. It contrasts regular traffic with clearly laid out rules. IDS has the capacity to not only identify harmful activity but also to stop it. IDS are of various types Signature based, Anomaly based and Hybrid based. The signature-based IDS works with the set of pre-defined patterns which will be utilized to compare whether the activity in the IoT system is malicious or benign. Anomaly-based IDS tracks the network behaviour and compares with the ongoing traffic if they don't match then it will be considered as a malicious activity. Hybrid-based IDS can detect both known and unknown attack. There are certain disadvantages with IDS which is the false alarm rate because causing excessive alert for unlikely events that might cause the administrator to face major problems (Garg *et al.*, 2021). And secondly, the majority of studies are restricted to using IP addresses to categorize just well-known botnets like Mirai and Bashlite. The IP address may, however, be fabricated. In order to over these issues ensemble model-based intrusion detection system (IDS) to classify whether the packet benign or malicious based on md5 hash values and will also classify the botnet types Mirai-GH, Mirai-APD, Svirtu-AA which is a Gagfyt botnet and other malwares as well. The Outputs of LightGbM and CatBoost was combined into a soft voting classification model to determine whether the packets are malicious or benign based on md5 hash values and various botnets types were classified through this methodology. Ensemble models have less risk than individual models. One classifier can't tackle complicated issues. Combining Light GBM and cat boost models to build ensemble learning smoothest boundary learning. Giving each choice a confidence probability reduces IDS false alarms, boosting the ensemble model's performance and resilience. Accuracy, precision, low false alarm rate, recall and f1score are evaluated for the proposed model (Alghamdi and Bellaiche, 2022b). However, accuracy of 99% was achieved in classifying whether the packets are benign or malicious and 93 % percent of accuracy was achieved in classifying the botnet types. The rate of the false attained 0.5 percent. The main purpose of this research is to determine how the ensemble techniques-based detection would be efficient in detecting the botnet attacks whether it is malicious or benign as well as to classify the botnet attack types in IoT.

2. Literature Review:

2.1 Botnet detection using various Methodologies:

The authors(Okur and Dener, 2020) conducted research on identifying botnet assaults on IOT in this study. They detected regular network traffic and attack traffic with great accuracy using the J48 supervised algorithm and expectation maximization for unsupervised learning. They used the 737E security camera data set, which is part of the N-BaIoT dataset, to conduct the research. The authors' acquired dataset with 115 features was divided into 10 measures. The Weka program was used to perform the feature extraction. During training, the 10fold option was favored in the cross validation section. Apart from Mirai and Bashlite, the writers may have classed the more variety of botnet assaults.
The authors(Soe *et al.*, 2020) suggested a sequential detection architecture, machine learning-based methodology for detecting botnet attacks. However, an efficient feature selection technique was applied to provide a lightweight detection system with high performance. The authors employed a correlated-feature selection technique to make the system lightweight and to include the proper features and capabilities for each kind of assault, such as FS-1 and FS-2, in order to limit the number of useless characteristics. The model builder module, which handled data collection, categorization feature selection, model training, and selection, and the attack detector module make up the author's suggested technique. The retrieved features will be subsequently sent through each of the sub-engines to detect attacks. If any sub-engine identifies the incoming traffic pattern as malicious, the system will issue an alarm. This study made use of N-BaIoT data sources. After that, the model is evaluated against the Naive Bayes, J48, and ANN. Furthermore, in the author's suggested detection technique using the feature selector, the detection accuracy of all three classifiers is almost comparable. The suggested systems could have been evaluated using testbeds, and the usual traffic patterns on other IoT device types could have been examined to expand the capabilities of the anomaly-sub-engine for successfully identifying unknown attacks.

The authors (Sridharan, Maiti and Tippenhauer, 2018) developed a machine learning-based AIDS. The detector may distinguish between legal and malicious traffic. Furthermore, the researchers designed an attack classifier for wireless systems. The authors' main purpose in creating the attack classifier was to differentiate between malicious traffic packets detected by the anomaly analyzer. Their algorithms are developed and tested using malicious traffic data collected at the wireless data connection layer. The models were tested on Wi-Fi-enabled IP cameras, smart lighting, and ZigBee-enabled smart bulbs. A random forest, a bagging technique, was utilized for classification, while an autoencoder, an unsupervised machine learning method, was employed for anomaly detection. Their approach identifies and classifies assaults on IP Cameras with good percentage of accuracy. During the anomaly detection stage, this research work has several restrictions, such as the fact that they built separate encoders for each IoT device, which means that anytime a new IoT device is introduced, new encoders are required to be created.

The authors (Bagui, Wang and Bagui, 2021) analyzed the traffic of the bonnets from the IoT environment. The have utilized three machine learning classifiers in their research which are Linear regression, Support vector machine and Random Forest and they classified botnet for about nine devices. N- BaIoT dataset were been utilized in their research. The data in this dataset were acquired from nine IoT systems compromised by Mirai and Bashlite-based botnets, and were then evaluated using three machine learning classifiers. The authors calculated the accuracy, True Positive, False positive, False Negative, True Negative, Precision, Recall and F1 score and they have achieved good results from all the three classifiers. There is a limitation in this research, It would have been advantageous to carry out feature selection and identify the characteristics that each attack and gadget responds to the best. It would also be helpful to thoroughly examine the characteristics.

A method of botnet identification presented by the authors (Choi *et al.*, 2008)included watching DNS traffic in order to identify the presence of botnets. The activity of DNS queries, on the other hand, is gathered from the continual requests given by the scattered bots. In this study, an anomaly-based intrusion detection system was used. First and foremost, the authors classified the patterns of botnet DNS traffic from regular DNS traffic. The authors then described the group activity by evaluating the essential attributes of the DNS traffic. Third, the authors created a botnet detection algorithm that classifies botnet DNS queries by employing the group properties provided by the authors in the previous step. The approach leverages IP headers to identify botnets that employ SSH or other encryption methods. The authors created the Botnet detection technique in three categories: Insert DNS query, Delete DNS query, and Detect DNS query. To begin, DNS requests will be kept in Array A, along with the domain name and source IP address. Whenever a new query arrives in sense, an analysis will be done to determine whether or not the new query already exists in A. If a new domain is added, the IP address is checked against the domain's IP list. Delete-DNSQuery removes duplicate DNS queries. Finally, the Detect-BotDNS-Query will detect the Botnet DNS query. There are certain limitations in this paper; for example, hash tables might be used to handle IP address lookups.

The author (Binkley, 2021) developed an anomaly based algorithm in order to detect the IRC bots. The approach combines a TCP scan identification strategy known as the TCP work weight with an IRC network recognition element. The proposed solution was used in PSU's

DMZ and shown to be successful in lowering botnet clients. However, the algorithm combines the IRC processing element with an IP host-specific syn-scanner identification mechanism. TCP packets are analyzed by the IRC processing system to identify an IRC channel or collection of IP addresses. They are then compared with IP host information from today's sampling data to establish if an IP channel host was a scanner. The top suspect IRC channels are labelled as potentially dangerous. The collected data from the front end captures innocuous IRC, botnet, and scanner list tuples. Syntuples of IRC and TCP are employed. The scanner gathers tuples throughout its 32nd sampling session. All tuples are then processed by the backend. The probe reduces the whole campus TCP syn tuple set to a smaller subset of particularly high locations. The TCP syn scanner tupes has IP src address which has the logical key. SYN, FINS and RESET represents the TCP control packets. SYNS are counts of IP SYN packets, whereas SYNACKS are those with the ACK flag enabled. RESETS are tracked by passing them to src IP. PKTSENT tracks the source IP packets whereas the PKTSBACK tracks the returned IP packet. The authors established a measure known as work weight to calculate by the probe per IP. The anomaly-based detection system proposed in this research is vulnerable to minor cipher use in order to encrypt IRC orders, which may result in the loss of mesh connection information.

(Gong, 2022) proposed a botnet detection using LightGBM algorithm, CTU-13 dataset was utilized in the research. The recommended detection method uses robot interaction patterns. The best feature subset is used as the LGBM classification source dataset. The suggested method increases detection rate while using less memory. To develop the model, the network collected packet was used as an input, and feature extraction, numerical labelling, and data under sampling were all carried out. After that, the LightGbM model was developed using the pre-processed data to determine if it was a botnet or benign. The LGBM approach was improved using Bayesian hyperparametric optimization. The detection efficiency might have been improved by using additional feature selection techniques, and data could have been gathered and trained using novel intrusion detection techniques as wavelet transform and clustering.

An extensive process was given by the authors (Islam *et al.*, 2021) to forecast the best attack detection in the IoT system. In this study, data from the NSL-KDD, DS2OS, IoTDevNet, IoTID20, and IoT Botnet were used. These 5 datasets were used to assess the performance of shallow and deep learning methodologies. The data from five datasets are combined to create the best model prediction. The data are then pre-processed, including cleaning, visualization, numericalization, and normalization. Then, the result was divided in an 80:20 ratio. The model was constructed using the data that had been trained, and it was then tested, allowing the authors to assess the model. If the detection rate satisfied the criteria used to compare the models, The authors declared that the assessed model would be considered successful. The most effective DL approach among those examined was Bi-LSTM. This study has a weakness; more analysis might have been done on the IoT network's microservices as they respond differently at different events, deviating from their usual behaviour, which may cause inconsistency.

The authors of this study, (Al-Haija and Al-Dala'ien, 2022), utilized the ensemble method in ELBA-IoT to develop a model for identifying botnet attacks. This research was based on Anomaly based network intrusion detection system. However, In order to analyze behavioural patterns of the IoT networks the proposed model was used. The authors' central idea is that an ensemble learning model can deal with data heterogeneity issues that arise often during anomaly detection. N-BaIoT dataset was used in this research. The advantage of this methodology is that actual data gathered from the IoT traffic will be pre-processed before being sent to the learning phase. Initially, the data preparation process took place where the preprocessing of the actual data from the IoT traffic will be converted into a table of the labelled features which are trained and fed as inputs in learning process module. The authors obtained the process of data hosting in order to maintain the data on stable and reliable platform. However, MATLAB computing platform was utilized by the author in order to host, train and validate the data. second, Data cleansing process has been performed after DHP. The steps for data standardization, data shuffling, and data distribution were then completed. Additionally, the authors trained and tested three ensemble models Adaboost, RUSboost, and bagged decision tree—using K-Fold Cross Validation throughout the testing phase. Voting probability was used to evaluate the best prediction model. The evaluation was carried out by the researchers using three classifiers. The evaluation findings showed that IoTs with ELBA functionality excelled earlier methods, yielding in lower interference overhead. There are specific constraints to this research, such as the possibility that the original IoT gateway would have been utilized in the ELBA-IoT for major botnet detection.

2.2 Intrusion Detection System of various techniques:

For Wi-Fi networks, the authors (Vaca and Niyaz, 2018) implemented an intrusion detection system. This study made use of the AWID Wi-Fi incursion dataset. Ensemble machine learning was used to create the model. The model was created by first importing the dataset, then removing features with zero variance and missing values that were less than 50%. Following that, they replaced the missing values with the most often occurring values, and then they replaced mac address attribute values with has MAC address boolean attribute values. Finally, they eliminated the characteristics based on time sequence and the highly linked values. The pre-processed dataset is then fed into the model development, which comprises a 4-class training set consisting of Random Forest, Bagging, Extra Trees, and XGBoost. Based on the training models, the best model was chosen. Random Forest out performed other models and the dataset has been tested with the Random Forest. Finally, based on their classifications, their model exhibited high accuracy and precision metrics in identifying traffic records. The suggested model was capable of distinguishing an impersonation attack from regular traffic. Several data in this study that were labelled as impersonation assaults were really injection attacks, which is viewed a disadvantage in this research paper.

Authors (Meidan *et al.*, 2018)pioneered anomaly-based network intrusion detection. The author's goal was to acquire network data based on behaviour and then use deep auto encoders to recognize anomalous network traffic from afflicted IoT devices. The authors got original data for use in corporate network-based anomaly detection via port duplication in the switch. A snapshot was obtained as soon as the packet received to examine the relevant host and protocol. The snapshot contains the whole data record, allowing the authors to determine if the data came from the same IP, started from the same source mac and IP, or was

transmitted between the source and destination TCP/UDP interfaces. Deep auto encoders were used to identify IoT bot attacks and tits behaviour. If the IoT device shows unexpected behaviour, it's presumed it's been compromised, which indicates the auto encoders unable to reconstruct the snapshot. The authors gathered real traffic from IoT devices infected with Mirai and Bashlite to evaluate performance. The compromised IoT devices might have been spotted earlier before accessing the corporate network, and traffic could have been anticipated theoretically and operationally which is considered as a limitation.
The authors (Tufan, Tezcan and Acartürk, 2021)performed an investigation on machine learning based anomaly IDS with the misuse based model. Ensemble learning model and CNN model were constructed and applied on data collected from the institutional production environment and UNSW-NB15 dataset. Data were gathered using tcpdump from the organization’s network interface.100,000 flows/sessions make up the final data collection, up from 481,000 before random sampling. In order to identify the alive host and portscan, the model was trained on probing-based intrusion detection. Following the packet capture, NAT was used to change the public IP for argus and tcptrace to the private IP, and formatting errors were resolved by using the tcpwrite programme. To inspect analysis packet captures using wireshark and argus, tcpwrite gave pseudo-random MAC addresses. The authors regarded the dataset's custom feature as the temporal feature. After that, feature sets are merged. Following the feature extraction, the pre-processing procedure included the removal of unnecessary data, managing missing values, one-Hot encoding, and labelling. Second, the pre-processed data were used to pick the features. Filter technique and wrapper method were used to accomplish this. The ensemble model and the CNN model were then trained and tested on the two datasets. The results revealed that the approach was sufficiently robust to assure the validity of the investigation. The ensemble model performed better than the CNN model. In a real-world institutional situation, the proposed anomaly-based IDS models beat the actual misuse-based categorisation. The authors may have sought to broaden the suggested approach to include different sorts of attacks, which is seen as a shortcoming.

2.3 Literature Gap:

Most of the researches are limited to classifying only the well-known botnets such as Mirai, Bashlite etc. by utilizing the IP address. However, the IP address could be spoofed. In order to overcome these issues, hashes can be utilized along with the IP address which can efficiently detect the botnet and its types.
In this research endeavour, botnets were detected based on the md5 hash values. With the provided method, it was possible to classify the botnets types and other malwares as well as establish whether the hash values were malicious or benign. Additionally, In order to improvise the proposed IDS, the false alarm rate was reduced.

3. Research Methodology:

3.1 LightGbM:
Gradient boosting is a classification and regression machine learning technique. It integrates patterns from many algorithms to create a new iterative one. Because of its effectiveness and

precision, gradient boosting is one of the most often used machine learning techniques.(Corporation, 2022) It began with adaptive boosting and progressed to a variety of algorithms and methods, including model-based boosting, CatBoost, XG boost, and LightGBM. IoT networks made up of diverse, resource-constrained devices. Therefore, the most effective option for any high-performance security system for IoT networks is the lightweight categorization method. Microsoft's LightGBM (Light Gradient Boosting Machine) is a gradient boosting machine that employs decision trees as its foundation training technique. Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling are the two upgraded algorithms in LightGBM. In this algorithm the decision tree grows in leaf wise. This improves the loss that produces branches. According to (M. Al-kasassbeh et al) LightGBM, on the other hand, improves training performance while decreasing memory use by employing histogram-based techniques.

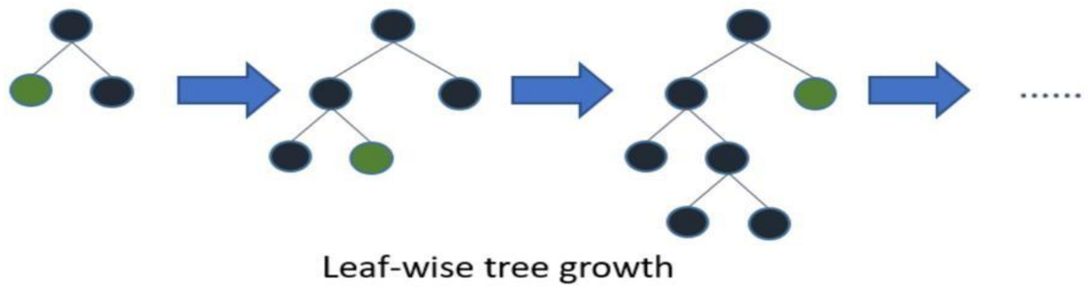


Figure 2: Leaf-wise tree growth of LightGBM

3.2 CatBoost:

Yandex developed CatBoost in 2017, a new open-source supervised learning algorithm that outperformed the other models in this testing. It uses decision trees with gradient boosting. In contrast to other gradient boosting methods, it takes less time to train. This algorithm's ability to automatically handle category features by turning them into numerical characteristics is another distinguishing feature (https://docslib.org/doc/4944088/detection-and-mitigation-of-botnet-based-ddos-attacks-using-CatBoost-machine-learning-algorithm-in-sdn-environment). However, the main reason of choosing CatBoost is that it has various features like fast prediction, improved accuracy and requires very minimal hyperparameter tunning. Moreover, unlike other ML algorithms that require substantial data training, CatBoost produces ground-breaking results without this requirement. CatBoost's ability to overlook sparsity while accelerating computation is its main advantage. According to (Alghamdi and Bellaiche, 2022a)CatBoost makes use of the Ordered Boosting method, which broadens the model's applicability.

3.3 Soft Voting Classification:

In the proposed technique, both LighGBM and CatBoost will classify whether the packet is malicious or benign based on the md5 hash values. However, it will also classify botnet types. The outputs of both LightGbM and CatBoost will be combine with the soft voting classifier. Soft voting classifiers sort input data into categories based on the likelihoods of each prediction made by each LighGBM and CatBoost. In order to apply training weights concurrently rather of just one booting model, the major goal of voting classification is to integrate both models. Three parameters were added to the model to combine it. The first two parameters are the CatBoost and LightGbM objects, and the third parameter is the training method. It has two different kinds of parameters: soft voting and hard voting. Soft voting classification has been employed in this study because the models' even numbers make it possible for their outputs to clash during classification. As a consequence, the probability from both classification outputs has been obtained using this approach, allowing the mean result to be used.

```
xtrain, xtest, ytrain, ytest <- TrainTestSplit(X,Y)
Initialize Weights{
    CatBoosting <- Train(iteration=1000)
    LightGBM <- Train()
    VotingClassification <- (CatBoost(iteration=1000, LightGBM)
   VotingClassication.method <- 'soft'
   VotingClassifcation <- Train(xtrain,ytrrain)
   prediction <- Test(xtest)
   Accuracy <- ConfusionMatrix(ytest, predition)

   print(Accuracy)
}
```

Figure 3: Snippet of the model's pseudo code

4. Design Analysis:

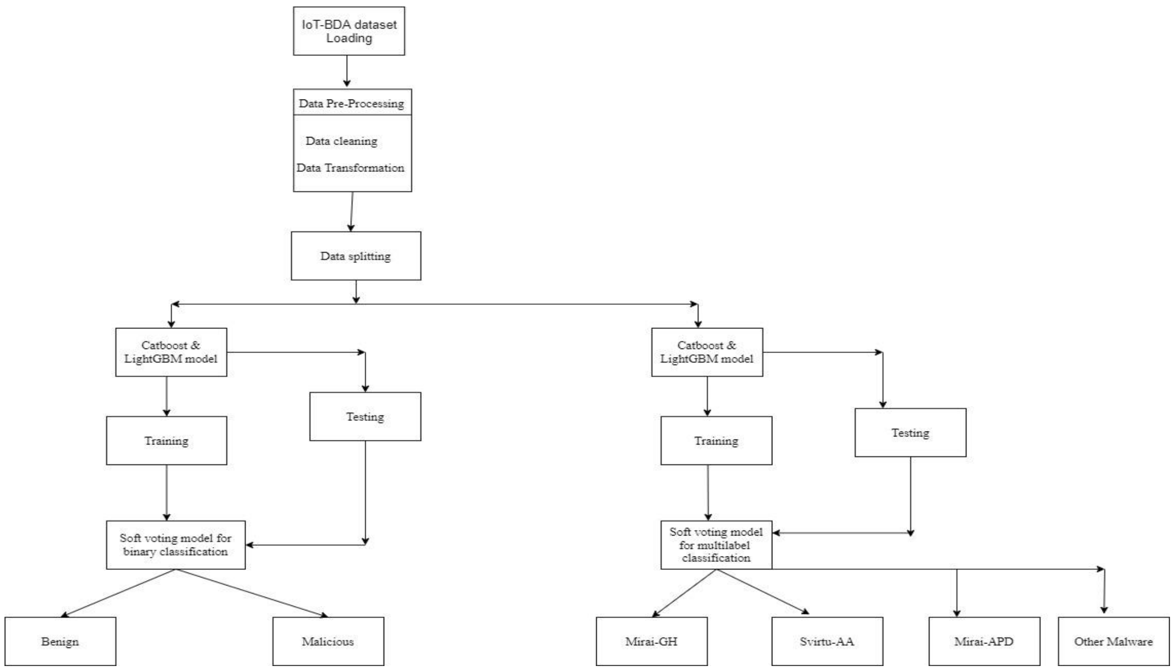


Figure 4: Design of proposed model

Phase1: In this phase the IoT-BDA dataset will be loaded, after which the data will be cleaned, features will be extracted and null values will be removed. The dataset consists of two columns at the end of the dataset. The column named ‘labels’ consists of records of malicious and benign and column named ‘final_label’ consists of the botnet types. These two columns were extracted by utilizing the md5 hashes via virus total checker. Secondly, the data transformation will take place by using the PCA. The Pre-processed data will be splitted into two models.

Phase2: For the binary classification of benign or malicious the LightGbM model and CatBoost model will be trained and tested separately in the ratio of 70:30 by using the xdata1 which contains the extracted features. However, benign and malicious records which was present in the column named ‘label’ which is considered as the ydata1. Then combined output of the LightGbM and CatBoost will be passed into soft voting classification. In this phase the soft voting model will be trained and tested again to classify whether the output is ‘0’ or ‘1’. If the output is ‘0’ it is considered as benign and if the output is ‘1’ then it is considered as malicious

Phase3: In this phase, xdata2 and yadata2 will be slitted in the ratio of 90:10. The Xdata2 consists of extracted features and ydata2 comprises of the botnet types. Then the CatBoost and LightGbM model will be trained and tested again separately. After which the output of the LightGbM and CatBoost model will be passed to the soft voting model where the soft voting model will be trained and tested again through which, the output of botnet types Mirai-GH, Svirtu-AA, Mirai APD and other malware will be classified will be obtained.

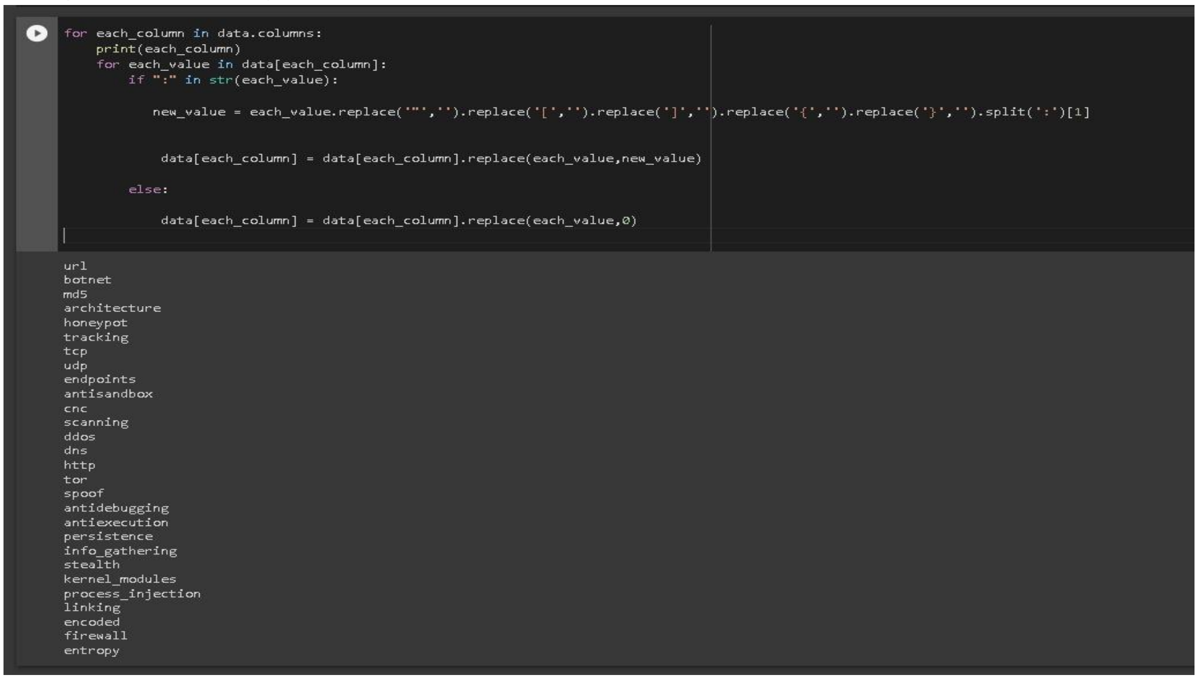
5. Implementation:

5.1 Dataset Description:

In this study, the IOT-BDA botnet analysis dataset was used which is a publicly available dataset(Trajanovski, 2021). The dataset includes the results of IoT-BDA Framework's analysis of 4077 different IoT botnet samples that were gathered using honeypots. The samples in this dataset were examined in a sandbox, where network and behavioural analysis, as well as antistatic and dynamic analysis employed by IoT botnets, were conducted to find signs of susceptibility. There are 42 features in the.csv file, covering the url, md5 hash, architecture, honeypot, tracking, TCP, UDP, endpoints, antisandbox, CNC, scanning, DDOS, DNS, HTTP, persistence, information collecting, stealth, kernel modules, process injection, linking, encoding, firewall, and so on. However, 4077 records were found for these 42 characteristics. In this dataset, there wasn’t any final labels to classify whether it is benign or malicious as well there wasn’t any labels having attack types. Records for the final columns were obtained utilizing the md5 hash values, which is done by merging two engines such as avast and kaspersky in the virus total checker and based on md5 hashes(*File Finder*, 2021), labels have been obtained for the sorts of botnets and whether they are malicious or benign. The obtained output has finally been labelled in the.csv file with the term’s "label" for benign and harmful behaviour and "final_label" for botnet kinds like.

5.2 Data Pre-Processing:

Colab was utilized in this research. First, panda's library was imported for its built-in quick function and it converts data into data frames. Second, there's no need to execute customized code for each cell, as with other libraries. Secondly warning was imported to ignore all the warnings when running cells. The IoT-BDA dataset's .csv file was imported to begin the Pre-Processing phase. Pandas was used to read the csv files. The df.head code then covered the first five rows, and the df.shape code was used to display the number of rows and columns after that. After then, the feature selection was finished. Following that, each column's overall null value count was shown. Then, every row that included a single null value was eliminated. Some of the features that were selected to address this problem included columns that used the JSON format. In order to determine which column is now being processed, the column names from the beginning to the last column of the data frame where the iteration is carried out were then presented. Most importantly, it is regarded as JSON data if there is a single ":" value that is null. However, values were gathered in order to process each column one at a time. Thirdly, after the values' extraction from the cells' JSON-formatted cells, the appropriate column's values were stored in the data frame. The values were changed to numbers in the next phase of the pre-processing using the label encoder library. The datatypes for each column were examined. The data was subsequently transformed into strings and ultimately into numbers.



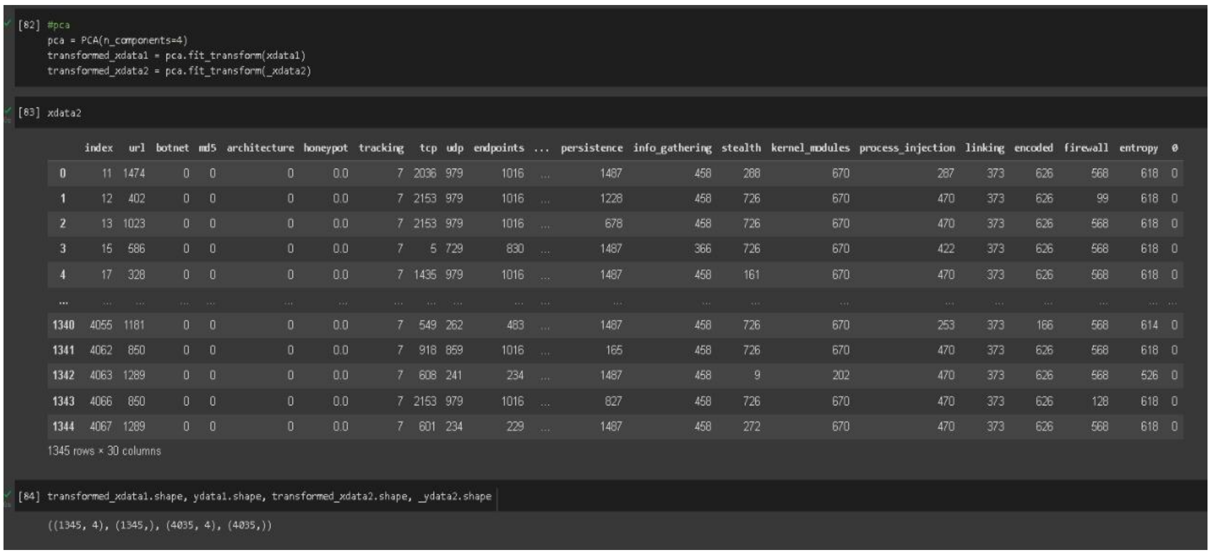
```
for each_column in data.columns:
    print(each_column)
    for each_value in data[each_column]:
        if ":" in str(each_value):

           new_value = each_value.replace('"','').replace('[','').replace(']','').replace('{','').replace('}','').split(':')[1]


            data[each_column] = data[each_column].replace(each_value,new_value)

        else:

            data[each_column] = data[each_column].replace(each_value,0)
```

```
url
botnet
md5
architecture
honeypot
tracking
tcp
udp
endpoints
antisandbox
cnc
scanning
ddos
dns
http
tor
spoof
antidebugging
antiexecution
persistence
info_gathering
stealth
kernel_modules
process_injection
linking
encoded
firewall
entropy
```

Figure 5: Snippet of Data pre-processing

5.3 Data transformation:

After obtaining PCA to minimise the number of columns and then performing convergent for the complete set of data into four columns, the output of the converted data was produced. However, the dimensions for the four columns of the principal component analysis were reduced using independent data. This transformed data will be used to classify the benign and malicious. The final_label will then be classified to independent data as well as with the column label. The.csv file containing the benign and malicious data and botnet types that were categorised based on md5 hash values by using the programme virustotal checker as indicated in the dataset description above were then imported in the next step. Then frequency was displayed for each value in the respective column for both the column label and final_label separately. Second, all records that had the chosen attack type name in the final_label column were filtered. After that, attack types were determined for each value in the column final_label. To hold all of the data, a list was made, to which the values were added, and it was then turned into a numpy array. Concatenation was performed for label and final_label which is considered as xdata. The column label is the one which contains the benign and malicious records and final_label contains the records of attack types. The unique values are then checked. After which dropna() function been utilized to drop null values form the xdata. The values of xdata are then considered as xdata1. The label column has been stored in ydata1. The label encoder was again utilized to transform the ydata1. In the next step, xdata1 and ydata were concatenated and get stored in xdata2. Lists were created for the final_column and then appended with the ydata2. Most importantly, PCA have been utilized and attained the transformed_xdata1 and transformed_xdata2



```
[82] #pca
     pca = PCA(n_components=4)
     transformed_xdata1 = pca.fit_transform(xdata1)
     transformed_xdata2 = pca.fit_transform(_xdata2)
```

```
[83] xdata2
```

```
      index   url botnet md5 architecture honeypot tracking   tcp udp endpoints ... persistence info_gathering stealth kernel_modules process_injection linking encoded firewall entropy 0
0        11  1474      0   0            0      0.0        7  2036 979      1016 ...        1487            458     288            670               287     373     626      568     618 0
1        12   402      0   0            0      0.0        7  2153 979      1016 ...        1228            458     726            670               470     373     626       99     618 0
2        13  1023      0   0            0      0.0        7  2153 979      1016 ...         678            458     726            670               470     373     626      568     618 0
3        15   586      0   0            0      0.0        7     5 729       830 ...        1487            366     726            670               422     373     626      568     618 0
4        17   328      0   0            0      0.0        7  1435 979      1016 ...        1487            458     161            670               470     373     626      568     618 0
...     ...   ...    ...  ..          ...      ...      ...   ... ...       ... ...         ...            ...     ...            ...               ...     ...     ...      ...     ... ...
1340   4055  1181      0   0            0      0.0        7   549 262       483 ...        1487            458     726            670               253     373     166      568     614 0
1341   4062   850      0   0            0      0.0        7   918 859      1016 ...         165            458     726            670               470     373     626      568     618 0
1342   4063  1289      0   0            0      0.0        7   608 241       234 ...        1487            458       9            202               470     373     626      568     526 0
1343   4066   850      0   0            0      0.0        7  2153 979      1016 ...         827            458     726            670               470     373     626      128     618 0
1344   4067  1289      0   0            0      0.0        7   601 234       229 ...        1487            458     272            670               470     373     626      568     618 0
1345 rows × 30 columns
```

```
[84] transformed_xdata1.shape, ydata1.shape, transformed_xdata2.shape, _ydata2.shape

     ((1345, 4), (1345,), (4035, 4), (4035,))
```

Figure 6: Snippet of Data Transformation

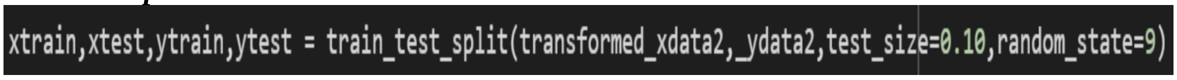
5.4 Model Training and Testing:

In the next step, LGBM classifier, CatBoost classifier, train_test_split and classification_report were imported. Label encoder has been applied to transform the ydata2. The training and testing for xdata1 and ydata1 was kept as 70:30 which is to classify whether the packets are malicious or benign and for xdata2 and ydata2 the training and the testing ratio was kept as 90:10 to classify the types of the botnets. Initial model building for xdata2 and ydata2 was performed in the CatBoost model with an iteration of 1000. The model has been built with its default parameters. Initially the accuracy for the classifying the botnet attack types was lesser and it got increased as the number of iterations increases. The cat boost model was th tested.The same types of approach in performed on the LightGbM model.

The obtained results of LightGbM and Cat boost and combined to together into a single model via soft voting classification. Secondly, the soft voting classification were then trained with 1000 iterations and then its tested. This will help to classify the types of botnets in a much better rate and it will reduce the misclassified samples as well. The main reason of training and testing the model in the ratio of 90:10 because, the model has to get trained on maximum number of samples. However, the dataset size was bit small. Suppose if 70/30 or 80/20 is used in sense, the size of the dataset would get reduced. In order to avoid that 90:10 ratio was utilized in this research for classifying the types of botnets Mirai.GH, Svirtu-AA, Mirai-APD, other malwares. The same approach is used in xdata1 and xdata2 for classifying the packets whether they are malicious or benign but the training and testing ratio is performed in 70:30, If the output is ‘1’ it is malicious and if the output is ‘0’ it is considered as benign.

5.5 Training and Testing of xdata2 and ydata2 (For IoT botnets classification):

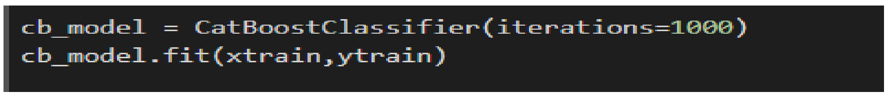
Train test Split:



```
xtrain,xtest,ytrain,ytest = train_test_split(transformed_xdata2,_ydata2,test_size=0.10,random_state=9)
```

Figure7: Train and Test Split for xdata2 and ydata2

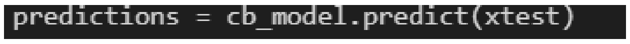
CatBoost Training:



```
cb_model = CatBoostClassifier(iterations=1000)
cb_model.fit(xtrain,ytrain)
```

Figure8: Training of Catboost

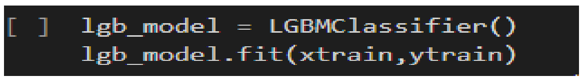
CatBoost Testing:



```
predictions = cb_model.predict(xtest)
```

Figure9: Testing of Catboost

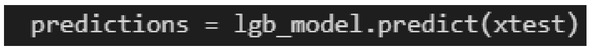
LightGbM Training:



```
[ ]  lgb_model = LGBMClassifier()
     lgb_model.fit(xtrain,ytrain)
```

Figure10: Training of LightGBM

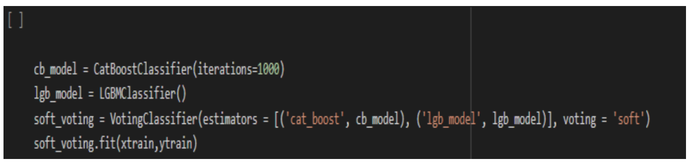
LightGbM Testing:



```
predictions = lgb_model.predict(xtest)
```

Figure11: Testing of LightGBM

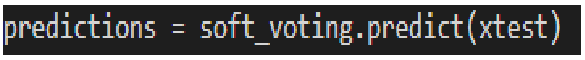
Soft voting classification Training:



```
[ ]

    cb_model = CatBoostClassifier(iterations=1000)
    lgb_model = LGBMClassifier()
    soft_voting = VotingClassifier(estimators = [('cat_boost', cb_model), ('lgb_model', lgb_model)], voting = 'soft')
    soft_voting.fit(xtrain,ytrain)
```

Figure12: Training of Soft voting classification

Soft voting classification Testing:



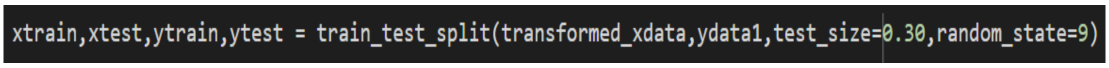
```
predictions = soft_voting.predict(xtest)
```

Figure13: Testing of Soft Voting classification

5.6 Training and testing the xdata1 and ydata1(for classifying benign or malicious):

The methods used to train and test xdata1 and ydata1 are the same as those used for xdata2 and ydata2. Testing and training, however, are split 70:30, as was mentioned above in the "train and split" section. The output of Xdata1 and Ydata1 are trained and tested to see whether it is malicious or benign. If the output is ‘0’, it is regarded benign; if it the out put is ‘1’ it is considered malicious.

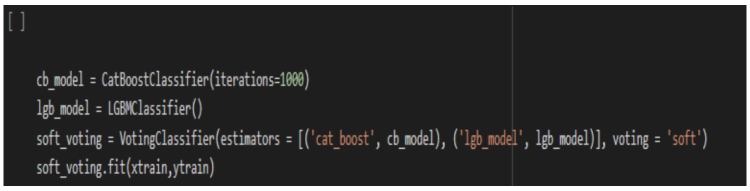
Test and Train for xdata1 and ydata1(for classifying benign or malicious):



```
xtrain,xtest,ytrain,ytest = train_test_split(transformed_xdata,ydata1,test_size=0.30,random_state=9)
```

Figure14: Test and Train for xdata1 and ydata1

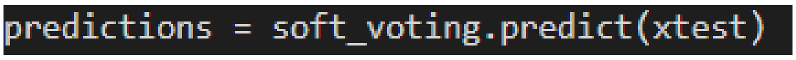
Soft voting classification training for benign or malicious:



```
[ ]

    cb_model = CatBoostClassifier(iterations=1000)
    lgb_model = LGBMClassifier()
    soft_voting = VotingClassifier(estimators = [('cat_boost', cb_model), ('lgb_model', lgb_model)], voting = 'soft')
    soft_voting.fit(xtrain,ytrain)
```

Figure15: Soft voting classification training for benign or malicious

Soft voting classification testing for benign or malicious:



```
predictions = soft_voting.predict(xtest)
```

Figure16: Soft voting classification testing for benign or malicious

6. Evaluation:

Some metrics produced by the "sklearn.metrics" library package are used to evaluate this ensemble learning.

6.1 Evaluation Metrics:

The most crucial criteria for the performance evaluation of an IDS are accuracy and false alarm rate. For the purpose of classifying an object as benign or malicious, the F1-score, precision, and recall were also evaluated.

True positive (TP): The actual packets are malicious and the predicted packets were also malicious

False Positive (FP): Actual Packets are benign and predicted packets are malicious

True Negative (TN): The actual packets are malicious while the predicted packets are benign

False Negative (FN): The actual packets are benign and the predicted packets are also benign

Accuracy: proportion of records in the dataset that were properly categorised out of all the records.

ACC = TP + TN/ TP + TN + FP + FN

Precision: The number of right identifications is measured, and the number of incorrect identifications is assessed.

Precision= TP /TP +FP

Recall: It is calculated by dividing the proportion of genuinely categorised records by the total number of anomalous records.

Recall =TP / TP + FN

F1 score: A weighted harmonic mean of recall and precision is F1.

F − measure =2 × Precision × Recall / Precision + Recall

False Positive Rate: It is calculated by dividing the proportion of inaccurately categorised records by the total number of irregular records.

FPR =fp/(fp+tn)

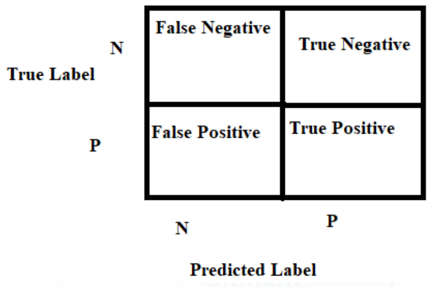
False Negative rate: It is calculated by dividing the false negatives to the sum of true positive and false negative

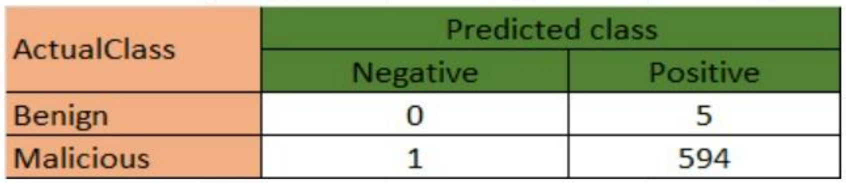
FNR = fn/(tp+fn)

False Alarm Rate:

FAR =FPR +FNR /2

6.2 Confusion Matrix:

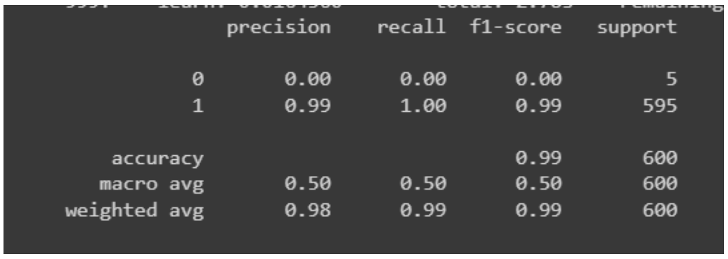




ActualClass	Predicted class	
	Negative	Positive
Benign	0	5
Malicious	1	594

Figure17: Confusion Matrix

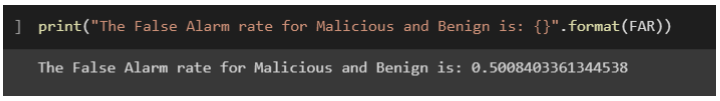
6.3 Results of classification of benign or malicious:



```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         5
           1       0.99      1.00      0.99       595

    accuracy                           0.99       600
   macro avg       0.50      0.50      0.50       600
weighted avg       0.98      0.99      0.99       600
```

Fig18: Benign or Malicious classification results

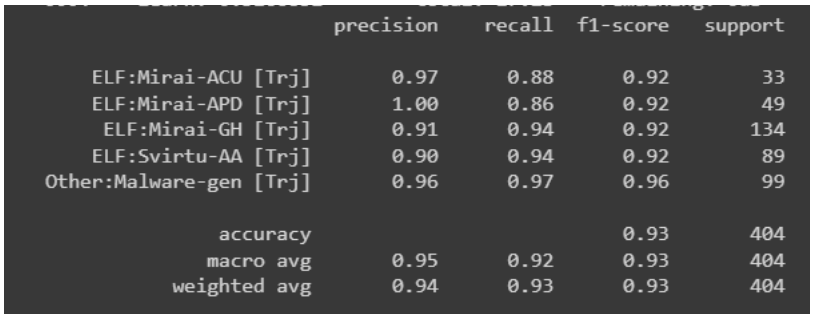
6.4 Result of False Alarm Rate:



```
] print("The False Alarm rate for Malicious and Benign is: {}".format(FAR))

  The False Alarm rate for Malicious and Benign is: 0.5008403361344538
```

Fig19: Snippet of Flase Alarm Rate result

6.5 Results of the classification of botnets types:



```
                           precision    recall  f1-score   support

      ELF:Mirai-ACU [Trj]       0.97      0.88      0.92        33
      ELF:Mirai-APD [Trj]       1.00      0.86      0.92        49
       ELF:Mirai-GH [Trj]       0.91      0.94      0.92       134
      ELF:Svirtu-AA [Trj]       0.90      0.94      0.92        89
Other:Malware-gen [Trj]       0.96      0.97      0.96        99

                 accuracy                           0.93       404
                macro avg       0.95      0.92      0.93       404
             weighted avg       0.94      0.93      0.93       404
```

Fig20: Snippet of classification of botnets types

6.6 Discussion:

An ensemble-based IDS was developed in this work to detect IoT botnets. LightGbM and CatBoost were combined to create a single vote classication that determines the likelihood of the outputs from LightGbM and CatBoost into a single output. This method, however, classifies the botnet kinds and determines if a packet is malicious or benign. The models were trained and evaluated on a ratio of 70:30, which resulted in accuracy of 99 percent and a False alarm rate of 0.5 for determining whether an incursion is malicious or benign. The models were trained and evaluated on a ratio of 90:10, and they successfully classified various forms of botnets with an accuracy of 93 percent. Other malwares were also categorised using the suggested methodologies. The analysis of the malware's characteristics and kind might have used some improvement. The main goal of the critical study was to create a smart ensemble-based IDS model that could classify botnet types and to determine if a packet was malicious or benign. However, with the aid of this model, services were offered to safeguard the systems as well as to evaluate incoming data that included attacks. The model will certainly be impacted by the change in dataset since it needs certain inputs in order to detect assaults. As a result, altering the dataset will be useful, and the model's input dimensions must be adjusted accordingly.

7. Conclusion and Future work:

In this study, IDS was created by integrating the LightGbM and CatBoost models into a soft voting classifier that would categorise packets are benign or malicious and, this methodology will classify the botnets types Mirai-GH, Mirai-APD, Svirtu-AA, other malware-gen. The IoT-BDA dataset was used in this research project. Comparing ensemble models to individual models demonstrates lower risk. Complex problems cannot be handled by a single classifier. Models from the Cat boost and Light GBM s provide the most fluid boundary learning ensemble learning. IDS false alarms are decreased by giving each option a confidence probability, resulting in improved performance and durability of the ensemble model. The suggested model's precision, accuracy, low false alarm rate, recall, and f1score were evaluated. Accuracy of about 99 percent was achieved in classifying whether the packets are benign or malicious and false alarm rate was achieved to 0.5. However, accuracy for classifying the botnets was achieved to 93 percent. The testing and training were done in the ratio of 70:30 for the classification of benign and malicious, if the output is ‘1’ it is malicious and if the output is ‘0’ it is considered as benign and for the attack classification the training and testing was done in the ratio of 90:10 and have successfully classified the botnet attacs types Mirai-GH, Mirai-APD, Svirtu-AA botnet, Mirai-ACU. However, apart from the IDS, the model could also be developed to detect as well as to prevent the botnets, and the suggested technique might also be used to existing IoT gateway devices to give IoT networks with real-time botnet detection capabilities, which can be considered as a future work.

References

Al-Haija, Q. A. and Al-Dala’ien, M. (2022) ‘ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks’, *Journal of Sensor and Actuator Networks 2022, Vol. 11, Page 18*, 11(1), p. 18. doi: 10.3390/JSAN11010018.
Alghamdi, R. and Bellaiche, M. (2022a) ‘Evaluation and Selection Models for Ensemble Intrusion Detection Systems in IoT’, *IoT*, 3(2), pp. 285–314. doi: 10.3390/iot3020017.
Alghamdi, R. and Bellaiche, M. (2022b) ‘IoT Evaluation and Selection Models for Ensemble Intrusion Detection Systems in IoT’. doi: 10.3390/iot3020017.
Alhowaide, A., Alsmadi, I. and Tang, J. (2021) ‘Ensemble Detection Model for IoT IDS’, *Internet of Things (Netherlands)*, 16(June), p. 100435. doi: 10.1016/j.iot.2021.100435.
Atlam, H. F. and Wills, G. B. (2020) *IoT Security, Privacy, Safety and Ethics*, *Internet of Things*. Springer International Publishing. doi: 10.1007/978-3-030-18732-3_8.
Bagui, Sikha, Wang, X. and Bagui, Subhash (2021) ‘Machine Learning Based Intrusion Detection for IoT Botnet’, *International Journal of Machine Learning and Computing*, 11(6), pp. 399–406. doi: 10.18178/ijmlc.2021.11.6.1068.

Bertino, E. (2017) ‘Botnets and Internet’, *Computer*, pp. 76–79.
Binkley, J. (2021) ‘An Algorithm for Anomaly-based Botnet Detection’. Available at: https://www.usenix.org/legacy/event/sruti06/tech/full_papers/binkley/binkley_html/ (Accessed: 14 August 2022).
Choi, H. *et al.* (2008) ‘Botnet Detection by Monitoring Group Activities in DNS Traffic’, pp. 715–720. doi: 10.1109/cit.2007.90.
Corporation, M. (2022) ‘LightGbM’.
David, D. B. (2021) ‘BoT- IoT based Denial of Service Detection with Deep Learning’, pp. 221–225.
File Finder (no date). Available at: https://github.com/AmrbadryDEV/VirusTotal_Sig_Checker/find/a34674bd3b3d4d4972c9d38fbef5914a38bb2d00 (Accessed: 14 August 2022).
Garg, U. *et al.* (2021) ‘Analysis of machine learning algorithms for IoT botnet’, *2021 2nd International Conference for Emerging Technology, INCET 2021*, pp. 14–18. doi: 10.1109/INCET51464.2021.9456246.
Gong, D. (2022) ‘A Mechine Learning Approach for Botnet Detection Using LightGbM’, pp. 0–4.
IoT Botnets Are Evolving – How Big Can They Get? - Secplicity - Security Simplified (no date). Available at: https://www.secplicity.org/2018/02/20/iot-botnets-evolving-big-can-get/ (Accessed: 14 August 2022).
Islam, N. *et al.* (2021) ‘Towards Machine Learning Based Intrusion Detection in IoT Networks’, *Computers, Materials and Continua*, 69(2), pp. 1801–1821. doi: 10.32604/cmc.2021.018466.
Kolias, C. *et al.* (2017) ‘DDoS in the IoT’, *Computer*, 50(7), pp. 80–84. Available at: http://ieeexplore.ieee.org/document/7971869/.
Meidan, Y. *et al.* (2018) ‘N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders’, *IEEE Pervasive Computing*, 17(3), pp. 12–22. doi: 10.1109/MPRV.2018.03367731.
Okur, C. and Dener, M. (2020) ‘Detecting IoT Botnet Attacks Using Machine Learning Methods’, *2020 International Conference on Information Security and Cryptology, ISCTURKEY 2020 - Proceedings*, pp. 31–37. doi: 10.1109/ISCTURKEY51113.2020.9307994.
Soe, Y. N. *et al.* (2020) ‘Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture’, *Sensors 2020, Vol. 20, Page 4372*, 20(16), p. 4372. doi: 10.3390/S20164372.
Sridharan, R., Maiti, R. R. and Tippenhauer, N. O. (2018) ‘WADAC: Privacy-preserving anomaly detection and aack classification on wireless traic’, *WiSec 2018 - Proceedings of the 11th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 51–62. doi: 10.1145/3212480.3212495.
Threat Encyclopedia | FortiGuard (2022). Available at: https://www.fortiguard.com/encyclopedia (Accessed: 14 August 2022).
Trajanovski (no date) *IoT-BDA Botnet Analysis Dataset | IEEE DataPort, 2021*. Available at: https://ieee-dataport.org/open-access/iot-bda-botnet-analysis-dataset (Accessed: 14 August 2022).
Trajanovski, T. and Zhang, N. (2021) ‘An Automated and Comprehensive Framework for IoT Botnet Detection and Analysis (IoT-BDA)’, *IEEE Access*, 9, pp. 124360–124383. doi: 10.1109/ACCESS.2021.3110188.
Tufan, E., Tezcan, C. and Acartürk, C. (2021) ‘Anomaly-based intrusion detection by machine learning: A case study on probing attacks to an institutional network’, *IEEE Access*, 9, pp. 50078–50092. doi: 10.1109/ACCESS.2021.3068961.

Vaca, F. D. and Niyaz, Q. (2018) ‘An ensemble learning based Wi-Fi network intrusion detection system (WNIDS)’, *NCA 2018 - 2018 IEEE 17th International Symposium on Network Computing and Applications*. doi: 10.1109/NCA.2018.8548315.