National College of Ireland

# IoT Botnet Detection using Machine Learning Techniques

MSc Internship
Cybersecurity

Suhas Jagannath
Student ID: x19113781@student.ncirl.ie

School of Computing
National College of Ireland

Supervisor: Mr Vikas Sahni

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

| | | | |
|---|---|---|---|
| **Student Name:** | Suhas Jagannath | | |
| **Student ID:** | X19113781 | | |
| **Programme:** | MSc in Cybersecurity | **Year:** | 2019-2020 |
| **Module:** | MSc Internship | | |
| **Supervisor:** | Mr Vikas Sahni | | |
| **Submission Due Date:** | August 17 | | |
| **Project Title:** | IoT Botnet Detection using Machine Learning Techniques | | |
| **Word Count:** | 6636 | **Page Count** | 20 |

I hereby certify that the information contained in this (my submission) is information pertaining 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 the rear 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.
I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

**Signature:** Suhas Jagannath

**Date:** 17.08.2020

**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): | |

# IoT Botnet Detection using Machine Learning Techniques

Suhas Jagannath

x19113781

## Abstract

The widespread use of the internet of things (IoT) has led to serious security problems such as the denial of service (DoS) attack caused by a large group of compromised IoT devices. The IoT devices can easily be compromised as they do not have a good security posture. IoT devices have full internet access without any packet filtration in place, making them suitable to be a part of the zombie network. Although many researchers have proposed various IoT botnet detection techniques, many challenges remain unaddressed. In this paper, various machine learning techniques are proposed to effectively identify the presence of IoT botnet. The detection models predict the IoT botnet based on the network traffic information. The proposed model uses feature selection to achieve a faster detection rate with less false positive. The Random Forest classifier model outperformed the other machine learning models and deep learning model with an accuracy of 94.47% with a lesser detection time. Therefore, this model can be considered as an appropriate solution to effectively detect the IoT botnet.

***Keywords: Internet of things (IoT), IoT botnet, IoT botnet detection, dimensionality reduction, feature selection, random forest, AdaBoost, KNN, dense neural network.***

# 1 Introduction

There are millions of smart devices that are connected to the Internet, ranging from mobile phones to computers, home thermostats, video surveillance cameras, and coffee makers [20], [21]. The proliferation of IoT devices will continue and accelerate substantially as countless devices are getting connected to the internet every day. Internet of things (IoT) technology is constantly evolving as these devices are implemented in various other fields such as healthcare monitoring, energy management, automobiles, smart home/office, and smart cities [1]. However, the usage of a large number of IoT devices over different types of services, technologies, devices, and protocols have made the management of IoT network a cumbersome process. The security design incorporated in the IoT devices is less sophisticated making them an easy target for the attackers. As there is not much room for security enhancement in the IoT devices, cyber-attacks on these devices have exponentially increased. Most of the IoT botnets such as *Mirai, Bashlite, Emotet* & *Reaper* are cleverly designed to perform various activities like *propagation, infection, C&C callback,* and *execution.* Thus, timely detection of such botnets in the IoT network has become critical to lessen the risks associated with it. This research focuses on employing machine learning techniques to effectively identify the IoT botnet using network traffic information. The below figure illustrates the idea of implementing machine learning techniques for predicting the existence of the IoT botnet using network information.

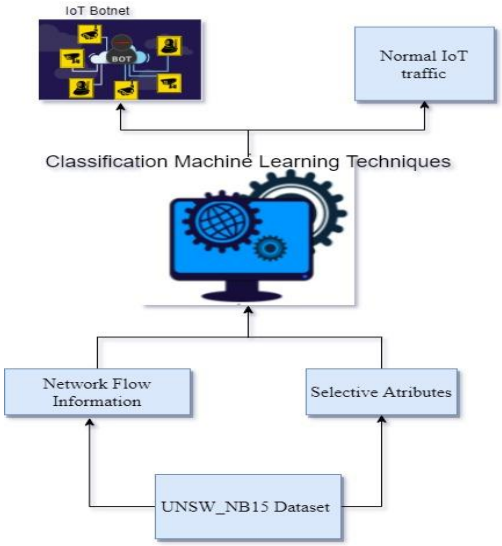


*Figure 1: IoT botnet classification using machine learning techniques*

## 1.1 Background & Motivation

According to Cisco's report, the number of devices connected to the internet would exceed 50 billion by the end of 2020, and by 2022 around 1 trillion smart devices will be connected to the internet. The below figure shows the increasing trend in the growth of the IoT devices:

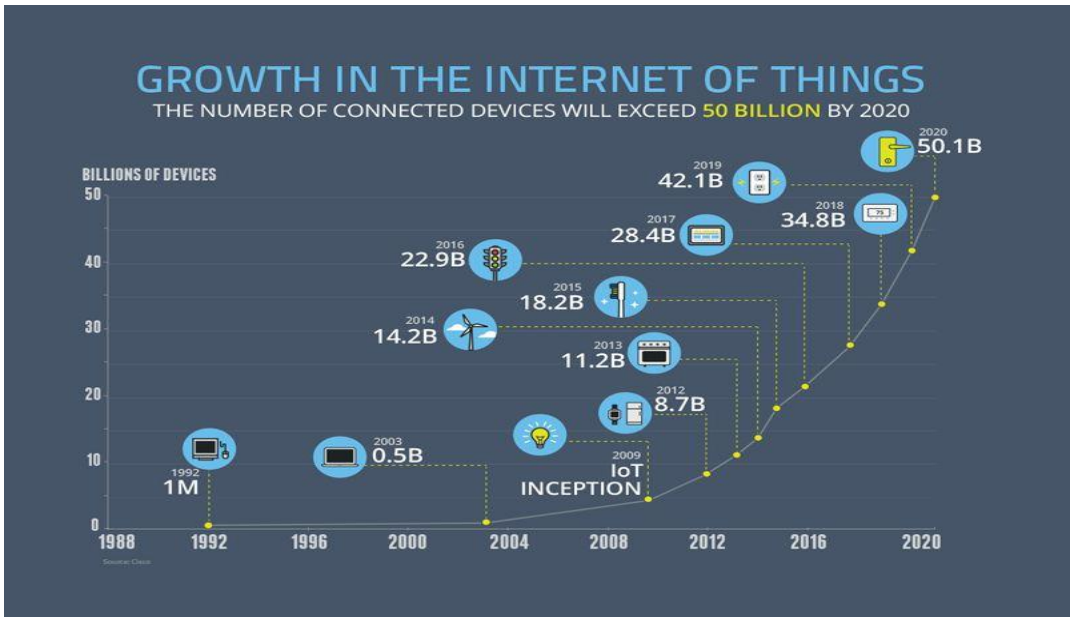


*Figure 2: The massive growth of IoT devices[18]*

As innumerable devices are getting connected to the internet the attack surface is constantly increasing, making it tough for the security professionals to secure it. The primary reason behind IoT devices being an easy victim of a cyberattack is the way these devices are configured. These IoT devices have complete internet access without any bandwidth filtration technique in place. Also, there is no room for any security enhancement in IoT devices because of the constraints of the operating system used. According to the study conducted by the American Consumer Institute, more than 8-10 home and office routers are vulnerable to hacking. Weak/default credentials, unpatched vulnerabilities both are the main reasons for the exploitation of IoT devices. Most of the malware tends to make use of the computational power available from the IoT devices for executing a large scale denial-of-service (DoS) attack. Thus,

identifying the existence of IoT botnet could drastically reduce the chances of a DoS attack. Implementing a machine learning technique to identify IoT botnet is a faster and efficient approach. This research discusses various classification models to accurately and effectively identify botnet traffic.

## 1.2 Research Question

*"Is it possible to detect a potential botnet in the IoT environment using machine learning techniques based on network traffic information?"*

The motive of this research is to answer the above-mentioned question by employing four different machine learning techniques on the UNSW_NB15 dataset. Since the UNSW_NB15 dataset has too many features, dimensionality reduction is performed to aid faster detection with lesser error rate making the model suitable for IoT botnet detection. Based on the information extracted from the feature selection such as source-destination transaction bytes, source-destination time to live, traffic transmission rate, etc. which helps in reducing the false positive and increasing the detection rate making it an ideal solution to detect IoT botnet. This research work makes use of the *UNSW_NB15* dataset that has all the relevant network traffic information required to identify the botnet activities within the network.

## 1.3 Research Objective

The following objectives are considered in this research work to meet the above-mentioned criteria:

- Performing feature engineering to extract the important feature thereby achieving the dimensionality reduction of the dataset.
- Feature selection is achieved by implementing the *Spearman's correlation* technique and feeding these selective features to the classifier models to predict the IoT botnet.
- Random Forest classifier model, Adaboost model, K-nearest neighbors (KNN) model, and Dense neural network model are implemented on the feature engineered data.
- Cross-validation, optimization, and comparison of the classifier models to perform the evaluation and faster prediction.

This research work is designed as follows: Section 2 portrays the relevant work completed in the detection of the IoT botnet using machine learning, Section 3 outlines the method employed in this research, Section 4 summarizes the three-stage design workflow incorporated for this research, Section 5 discusses the various machine learning and deep learning models implemented, Section 6 shows the results and evaluations of each model and lastly, the research work is concluded in the section 7.

# 2 Related Work

This section addresses numerous research studies related to IoT botnet detection employing different machine learning techniques. A comparative study of various network traffic analysis detection methods and how various machine learning techniques have been employed to enhance the IoT botnet detection system.

## 2.1 Dataset Description

The researchers [2] have addressed the major research challenge of the unavailability of the comprehensive modern network traffic datasets. Countering this challenge, the researcher has created the UNSW NB15 data with the combination of traditional and modern network traffic scenarios using the IXIA tool, which helped in creating the synthetic environment generating the normal IoT and botnet traffic scenarios. The generated NB15 dataset is compared with the

existing benchmark dataset which has been bounded by certain limitations such as outdated attacks and the number of information packets. Hence, in contrast with the other network intrusion detection datasets such as KDD98, KDDCUP99, and NSLKDD, the newly generated NB15 dataset demonstrates higher benefits which can be expected to be useful for this research.

Evaluating the network intrusion detection system using the existing benchmark NIDS datasets does not yield satisfactory results. So in the work proposed by [3] has discussed the issues with the dataset and to address this issue the NB15 dataset is considered and demonstrated 3 aspects such as statistical analysis, feature correlation, and implemented multiple classifiers to evaluate the complexity of the dataset. The findings are compared with the KDD99 dataset and results showed that NB15 is a comprehensible modern network intrusion dataset than KDD99 which has several drawbacks. In the second part of the study, the author has discussed the characteristics of each classifier performed on the dataset. Among all the classifiers, Decision tree and logistic regression with an accuracy of 86% and 81% respectively, surpassed the Naïve Bayes, Artificial neural network, and Expectation maximization algorithm. The study [4] demostrated a two-stage random forest classifier on the UNSW NB15 dataset to produce better performance than other classifiers used in a two-stage approach. As UNSW NB15 dataset was developed with real-time IoT network traffic details, which tends to be having the most influential features responsible for the IoT botnet detection. In another study [5] UNSW NB15 data was used to perform anomaly detection using the trapezoidal area estimation on large neural networks. Moustafa et al [5] extracted 13 network features using principle component analysis (PCA) and achieved an accuracy of 92.8% with 5.1% of false positives. Important features like time-to-live, source load, destination load, transaction rate, duration of transaction were not considered in [5], thereby not making complete use of the UNSW NB15 dataset.

## 2.2 Machine Learning Techniques

As discussed in the previous sections, the evolution of sophisticated IoT botnets has made it necessary to develop a detection technique that is distinct from the traditional signature-based detection techniques. To tackle this issue many researchers have made use of various machine learning techniques to achieve network forensics. Koroniotis et al [6] demonstrated the role of machine learning techniques in detecting the suspicious activities of the botnets. This study implemented ARM, ANN, Naïve Bayes, and decision tree machine learning techniques on the UNSW-NB15 dataset. The experimental results showed that the decision tree ML technique in aggregation with flow identifiers of source/destination IP addresses and protocols could effectively detect the botnet.

In another study [7] honeynet was deployed to provide intrusion activity logs along with the network traffic dump in the form pcap files. The data collected from the honeynets was converted into a network flow dataset with the help of ARGUS and Radium, which was used to apply the machine learning technique to detect the botnet within the network. The experimental results showed that the random forest classification model was consistent and performed better than the other machine learning models. R. Doshi et al [8] showed that IoT specific network behaviors such as the limited number of endpoints and regular time intervals between packets implemented with feature selection technique resulted in higher accuracy. KNN, linear SVM, Decision tree, Random Forest, and ANN machine learning techniques were used to classify normal IoT packets from DoS attack packets. The experimental results showed that all five algorithms achieved an accuracy of 99.99% however, the random forest algorithm outperformed the other four algorithms and gave a stable output. Another study [9] discussed that the random forest algorithm was implemented in the botnet detection model because high accuracy of prediction is required. Singh et al [9] demonstrated that the random forest classifier achieved an accuracy of 99.7% for 84,030 different occurrence of diversified traffic data.

PSI rooted subgraph-based feature for IoT botnet detection using the random forest algorithm discussed in the paper[10] showed that the random forest model yielded better results than all other classifiers with 98% true positive rate. Results also showed that the random forest classifier had

the best ROC-AUC value of 97%. The random forest model with a maximum depth of trees, the number of features, and the number of trees were used as hyperparameters [4], [22]. The advantages of implementing the random forest algorithm are that it handles the large database, reduces the overfitting, fewer model parameters [11]. Also, in the random forest algorithm, the variance of the model is indirectly proportional to the number of trees with the bias remaining constant. Chen et al [12] described a model that could handle large scale DNS query flows using the random forest algorithm. The study showed that the random forest classification model built on the spark was able to achieve 0.0% FPR and 4.36% FNR making it suitable for real-time detection. As the UNSW_NB15 dataset uses real-time IoT network traffic information using the random forest algorithm has resulted in higher accuracy with high precision.

Many papers are the witnesses that the random forest algorithm performs better than most of the machine learning models. Therefore, the random forest machine learning technique was implemented in our research to effectively detect the IoT botnet traffic with lesser training sessions and lesser detection time. Also, our research has employed the dense neural network that yielded better accuracy than the neural networks employed in [13], [14], [5].

## 2.3 Feature Selection

Feature selection is a technique used to remove input variables when developing a predictive model. This dimensionality data reduction for classification is highly desirable for overcoming the computational cost of modeling and also improving the predictive analysis of the model. Since the classification machine learning technique was implemented to perform predictive analysis on the network traffic data present in the dataset, the feature selection technique helps in improving the accuracy of the IoT detection model by removing unnecessary features.

The UNSW_NB15 dataset used for the IoT detection model has 45 features in total. Out of these 45 features, only 17 important features were taken into consideration for applying the classification model. Koroniotis et al [6] employed Information Gain (IG) for feature selection as it one of the simplest feature selection methods. For the experiment, ten features with the highest information gain (IG) were selected. Bhasi et al [1] demonstrated a dimensionality reduction based IoT detection system using machine learning. Their approach was mainly focused on minimizing the number of features required to detect the IoT botnet. Fischer's score aiming to consider only the top 3 features which yielded higher accuracy using decision tree in contrast to KNN. Followed by the discriminatory power of feature was computed to identify the best contributing features and performed analysis on which feature induces the highest accuracy for the classifier. The results obtained indicated 3 out of 6 features induced higher accuracy classifiers. Also, they investigated the impact of data balancing on the classifiers, by creating two sets of data one as the original dataset with an equal number of records of all the classes and the other one with unbalanced skewed data. And their findings showed that there is no significant variation in the accuracy of the classifiers when the model was applied on both the datasets.

In the study presented in [15], dimensionality reduction was achieved with the help of the autoencoder model and essential information for the input was obtained by compression of the code layer between encoders and decoders. Palmieri et al [16] proposed a distributed approach to network anomaly based on the independent component analysis. The network traffic features were extracted using the independent component analysis that helps in extracting unknown hidden features from multivariate data. The study showed that the information carried by one component should not be inferred by the others hence independent component analysis was implemented.

Nomm and Bahsi [1] has presented IoT botnet detection with the emphasis on feature selection using the unsupervised model also by training an individual model on all the IoT devices rather than a dedicated model for each IoT devices in achieving resource optimization. The proposed solution focused on multiple methods of feature selection such as Hopkins statistics-based

feature selection, entropy-based, and variance-based. In addition to this sampling of the original unbalanced dataset was done to obtain the balanced data. As a result, SVM performed better on the unbalanced dataset with notable accuracy whereas isolation forest with entropy-based feature selection performed superior to all the other combinations of feature selection and models.

One of the primary aspects of a detection model in the cybersecurity field is to identify the threats faster. Therefore, in this research work, feature selection is implemented to attain faster and accurate detection using various machine learning techniques. Feature selection helps in adopting potentially important and relevant attributes from the dataset making the machine learning models perform efficiently.

# 3 Research Methodology

This section discusses the research methodology employed in predicting the presence of IoT botnet using various machine learning techniques on the UNSW NB15 dataset. This section also gives a brief idea about the data analytics techniques and data mining techniques. This research is implemented based on the KDD data mining technique. KDD stands for Knowledge Discovery in Database which is an iterative and interactive data mining technique comprising nine steps. This methodology starts by determining the KDD objective and ends with the implementation of the discovered knowledge [17]. The different stages of KDD methodology are 1) Data Acquisition, 2) Data Integration, 3) Data Selection, 4) Data Transformation, 5) Data mining, 6) Pattern Evaluation, 7) Knowledge Representation[1]. The below figure illustrates the KDD methodology incorporated in this research.

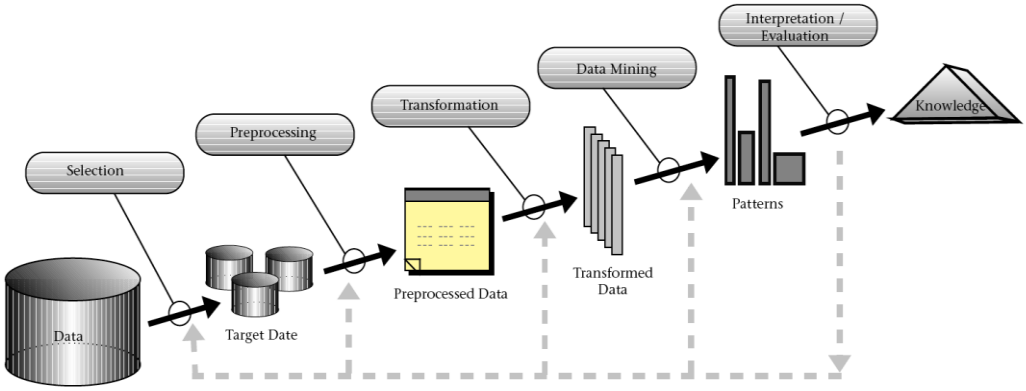


*Figure 3: KDD Methodology incorporated in IoT botnet detection.[19]*

## 3.1 Data Acquisition

In this stage, the unwanted data and noisy data was removed to prepare a meaningful data that can be used in the IoT botnet detection. Noisy data or the missing data makes the prediction models weak and slow therefore data cleaning was performed on the UNSW NB15 dataset[2] to achieve a better prediction of the model. In this research, the XL tool was used to perform data cleaning. The dataset is available publicly and the *UNSW_NB15* dataset was initially created by the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for generating a hybrid of real modern normal activities and synthetic contemporary attack behaviors [2],[21].

[1] https://www.geeksforgeeks.org/kdd-process-in-data-mining/

[2] https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/index.php

## 3.2 Data Integration

Data Integration refers to the aggregation of data from multiple sources to form one single data with all the information required for the prediction model. The dataset used in this research was prepared by merging *UNSW_NB15_training-set.csv* and *UNSW_NB15_testing-set.csv.* The *UNSW_NB15_training-set.csv* dataset consists of 82332 rows x 45 columns and the *UNSW_NB15_testing-set.csv* dataset consists of 175341 rows x 45 columns, these both datasets are combined to achieve *finaldataset.csv* with 257673 rows x 45 columns. The finaldataset.csv consists of 44 different features and label as the target column where "1" indicates the presence of IoT botnet and "0" indicate the normal traffic.

## 3.3 Data Selection

Here, the data relevant to the IoT botnet predictive analysis model is retrieved from the dataset prepared in the previous step. To extract the important features, the inbuilt class feature_importance of tree classifier is implemented. The main idea of using the ExtraTressClassifier is to fit several randomized decision trees into the data, making sure that the model does not overfit the data. To identify the important features in the finaldataset, wrapper-based feature selection was implemented.
Later, these important features were fed into a data frame, and the correlation matrix was plotted using the seaborn library. To find the relationship between the important feature extracted from the previous steps, the Spearman correlation method was incorporated.

### 3.3.1 Feature Engineering

The important network features extracted from the dataset by incorporating ExtraTreeClassifer is shown below. This method yielded 10 important network features namely sinpkt, sttl, sbytes, rate, swin, dur, dload, dttl, spkts along with their respective variance.

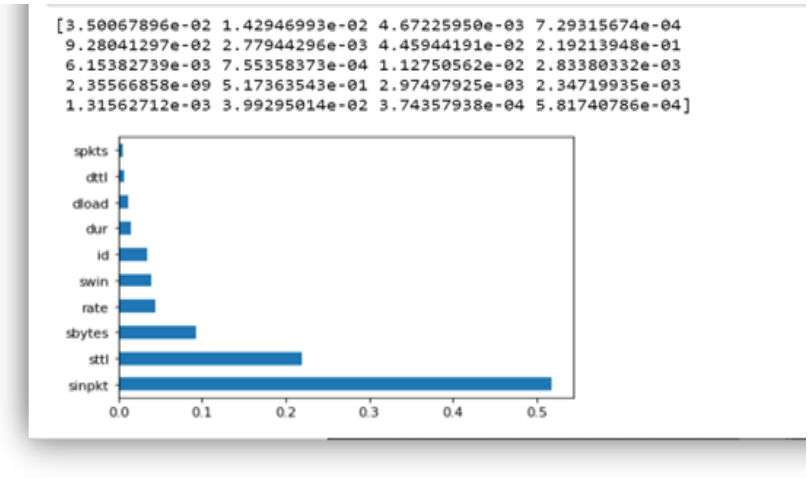


*Figure 4: Tree classifier based feature selection*

The heatmap graph illustrates the correlation between the variables obtained using the Spearman correlation technique. Out of 40 network traffic features only 17 highly correlated features were selected based on the correlation matrix. From the plot, dark blue indicates that the features are strongly correlated and light blue indicates that the features are weakly correlated.

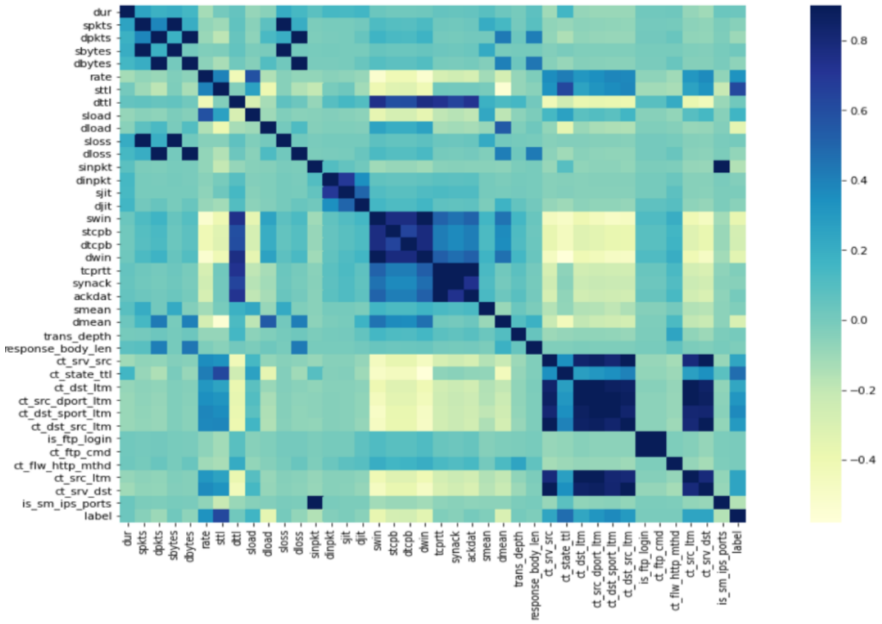


*Figure 5: Spearman's correlation plot*

Although in [6]only 10 network features were selected using the Information Gain Ranking Filter (IG) and those 10 features were fed into the machine learning models. However, in this research work, the number of selected features was increased thereby increasing the model accuracy.

## 3.4 Data Mining Techniques

Various data mining techniques were applied to the extracted network traffic feature to accurately classify the IoT botnet traffic from the normal traffic. Since the data was used to predict a categorical variable, the classification machine learning technique was employed. In this research, the label "1" indicates the presence of an IoT botnet traffic within the network, and label "0" indicates the presence of normal traffic. Hence, classification algorithms such as Random Forest, AdaBoost classifier, K-Nearest Neighbor, and Dense Neural Network were implemented for reliable and faster prediction. These different classification models gave different outputs for the same data. Further, the Random Forest model yielded better results when compared to other predictive models such as Adaboost, K-Nearest Neighbor (KNN), and Dense Neural networks. The evaluation of each model is discussed in detail in the coming sections.

## 3.5 Evaluation

The confusion matrix illustrates how the classification model gets confused when the model makes a prediction, the following details can be inferred from the confusion matrix:

***True Positive (TP)-*** The total number of occasions the model correctly recognised as a subset of the IoT botnet class.

***False Positive (FP)-*** The total number of occasions the model incorrectly recognised as a subset of the IoT botnet class.

***False Negative (FN)-***The total number of occasions the model incorrectly recognised as not a subset of the IoT botnet class.

***True Negative (TN)-***The total number of occasions the model correctly recognised as not a subset of the IoT botnet class.

*Table 1: Confusion Matrix*

| | Predicted Normal Traffic | Predicted IoT Botnet Traffic |
|---|---|---|
| Actual Normal Traffic | TN | FP |
| Actual IoT Botnet Traffic | FN | TP |

# 4 Design Specification

The three-stage design is incorporated in this research work comprising of following:

- Data preparation stage- This stage consists of data extraction, cleaning, merging, feature selection using the Spearman correlation technique. The original data source is of CSV format and stored in Jupyter notebook using python.
- Modeling stage- It comprises of multiple machine learning and deep neural network frameworks such as random forest, K-nearest neighbor, decision tree, dense neural network. These techniques/architecture are implemented on the UNSW NB15 dataset, model optimization using AdaBoost and SGD, appropriate evaluation metrics such as confusion matrix, accuracy, precision, recall, F1-score, and AUC-ROC were chosen to evaluate the models.
- Visualization stage- The result obtained in the previous stage is represented in the form of graphs, plots for a better understanding.

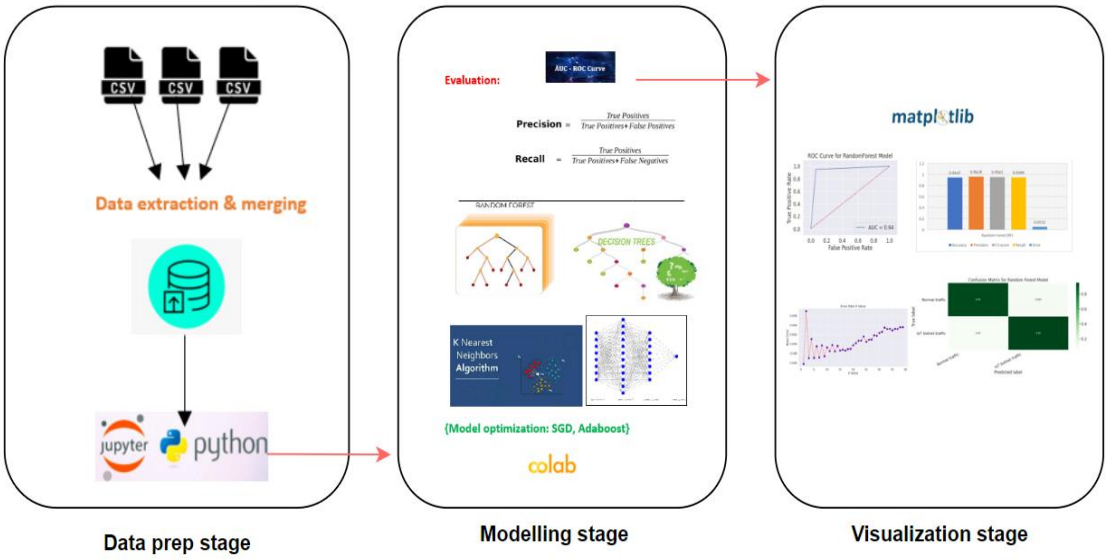


*Figure 6: IoT botnet detection- design flow*

# 5 Implementation

This section explains the implementation of IoT botnet detection using various machine learning technologies. The research work was carried out in three stages: Data Preparation, Cross-Validation, and Model Refinement with the help of hyperparameters. The idea behind

using hyperparameters is to improve the prediction rate of the models. The outputs of different detection models are compared to determine the appropriate predictive analysis technique.

## 5.1 Dataset Preparation

As discussed in the previous sections, the 49 features dataset was transformed into 17 features dataset using feature engineering techniques. So at the modeling stage, a dataset with 257673 rows and 18 columns was prepared. This dataset was loaded into X and y variables, where X had only the network traffic information without the target variable and y had both the information. Later X and y were split into 70 % training and 30 % testing set. The training set was used to train the model using various classifier algorithms.

Dimensionality reduction was achieved with the help of Spearman's correlation technique. Spearman's correlation coefficient is a statistical measure of the strength of the monotonic relationship between the paired data. Spearman's correlation coefficient is denoted by $r_s$ that is defined as

$$-1 \leq r_s \leq 1$$

Unlike Pearson's correlation, Spearman's correlation does not require normality and therefore it is considered as a nonparametric statistic. Based on the correlation coefficients obtained from the correlation matrix, the dataset with 44 features was converted into a dataset with 17 important features using python that was sufficient for a successful prediction of IoT botnet traffic. The below graph provides a better understanding of the network traffic information taken into consideration for this research work.

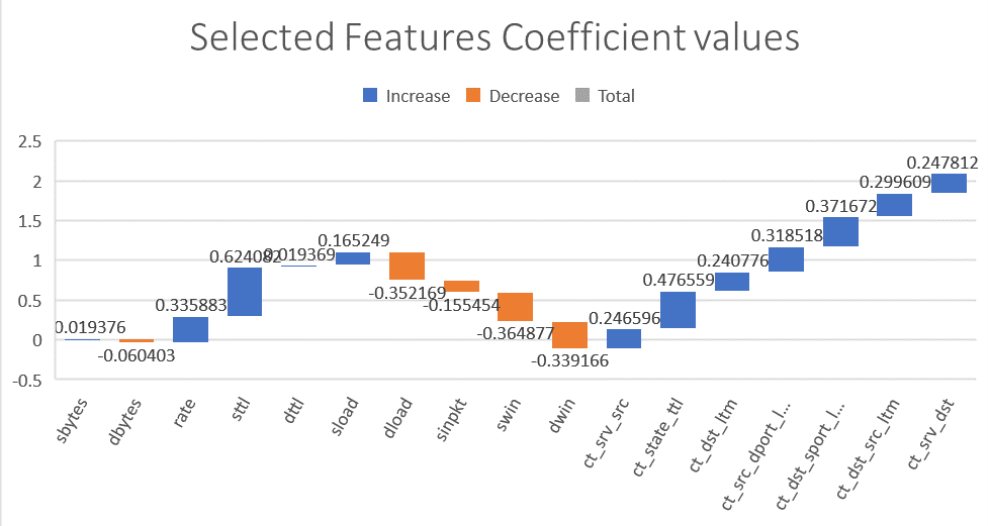


*Figure 7: Plot showing the extracted features.*

The below table shows the description of the extracted features along with their respective correlation coefficient values. To identify the significant network features of the dataset a tree-based feature selection was also implemented. However, the results showed that Spearman's correlation feature selection method provided more insights into the network traffic attributes. Therefore, all the 17 extracted features were used to feed the IoT botnet detection model for accurate and faster detection.

*Table 2: Selected features with respective correlation coefficients*

| Selected Features | Correlation Coefficient Values | Feature Description |
|---|---|---|
| sbytes | 0.019376 | bytes transmitted from source to destination |
| dbytes | -0.060403 | bytes transmitted from destination to source |
| rate | 0.335883 | Transmission rate |
| sttl | 0.624082 | source time to live value |
| dttl | 0.019369 | Destination time to live value |
| sload | 0.165249 | Source bits per second |
| dload | -0.352169 | Destination bits per second |
| sinpkt | -0.155454 | Source interpacket arrival time (mSec) |
| swin | -0.364877 | Source TCP window value |
| dwin | -0.339166 | Destination TCP window value |
| ct_srv_src | 0.246596 | No. of connections that contain the same service and source address |
| ct_state_ttl | 0.476559 | No.of connections of the same state and the time to live value |
| ct_dst_ltm | 0.240776 | No. of connections of the same destination |
| ct_src_dport_ltm | 0.318518 | No. of connection of the same source address and the destination port. |
| ct_dst_sport_ltm | 0.371672 | No. of connections of the same destination address and the source port |
| ct_dt_src_ltm | 0.299609 | No. of connections of the same source and the destination address. |
| ct_srv_dst | 0.247812 | No. of connection that contain the same service and destination address. |

## 5.2 Random Forest Classifier Model

Random forest classifier as the name indicates is a group of individual decision trees that operate as one. This ensemble tree-based learning algorithm aggregates the outputs of different decision trees to predict the final class of the target object. Random forest classifier with hyperparameter tuning was incorporated in this research to effectively identify IoT botnet. Model training time and model detection time is also considered for this research using the time function. In this study, the random forest model was optimized using the hyperparameters like n_estimator, max_features, max_depth, min_samples_split, min_samples_leaf, and oob_score. The below figure shows the random forest model fine-tuned with n_estimator as 200, max_features as 8, max_depth as 15, min_samples_split as 2, criterion as gini and min_sample_leaf as 1.

```
from sklearn import metrics
from sklearn import model_selection
import time
# random forest model creation
t0 = time.time()
rfc = RandomForestClassifier(n_estimators=200, max_features=8, max_depth= 15 ,random_state=0, min_samples_split = 2, min_samples_leaf = 1,  criterion='gini', oob_score = True)
rfc.fit(X_train,y_train)
t1 = time.time()
print("Model Training time:", (t1 - t0))
# predictions
rfc_predict_test = rfc.predict(X_test)
rfc_predict_train = rfc.predict(X_train)
score_rfc_train = metrics.accuracy_score(y_train, rfc_predict_train)
score_rfc_test = metrics.accuracy_score(y_test, rfc_predict_test)
t2 = time.time()
print("Model Detection time:", (t2 - t1))
print("Accuracy of random forest on train data :",score_rfc_train)
print("Accuracy of random forest on test data :",score_rfc_test)
```

```
Model Training time: 73.87213730812073
Model Detection time: 4.994750261306763
Accuracy of random forest on train data : 0.9584016532614074
Accuracy of random forest on test data : 0.9444067138837683
```

*Figure 8: Code snippet of Random Forest model*

## 5.3 Adaboost Classifier Model

The AdaBoost classifier is a boosting ensemble model used for binary classifications. This algorithm classifies by converting a group of weak classifiers into a strong one. Usually, AdaBoost is used along with simple decision trees. Furthermore, the training data that is hard to predict is given more weight and easy prediction is given less weight. This boosting algorithm can be used to boost the performance of any machine learning algorithms but it works best with the decision tree model. Therefore, a simple AdaBoost model was implemented for this study with max_depth as 5 and n_estimators as 100 as shown below.

```
[63] from sklearn import metrics
     from sklearn import model_selection
     from sklearn.tree import DecisionTreeClassifier
     import time
     # Adaboost model creation
     t0=time.time()
     ada = AdaBoostClassifier(DecisionTreeClassifier(max_depth=5),n_estimators=100)
     ada.fit(X_train,y_train)
     t1=time.time()
     print("Model Training time:", (t1 - t0))
     # predictions
     ada_predict_test = ada.predict(X_test)
     ada_predict_train = ada.predict(X_train)
     score_ada_train = round(metrics.accuracy_score(y_train, ada_predict_train) * 100, 2)
     score_ada_test = round(metrics.accuracy_score(y_test, ada_predict_test) * 100, 2)
     t2=time.time()
     print("Model Detection time:", (t2 - t1))
     print("Accuracy of adaboost on train data :",score_ada_train)
     print("Accuracy of adaboost on test data :",score_ada_test)
```

```
Model Training time: 75.32802367210388
Model Detection time: 4.427966356277466
Accuracy of adaboost on train data : 95.79
Accuracy of adaboost on test data : 94.41
```

*Figure 9: Code snippet of Adaboost classifier model*

## 5.4 KNN Classifier Model

K-nearest neighbors (KNN) algorithm a supervised machine learning technique that can be used for both classification and regression predictive problems. However, the KNN algorithm is popularly used for classification predictive problems. KNN is also known as a lethargic learning algorithm since it does not require much effort in the training phase and uses all the data for training while performing classification. Initially, the K value was assumed to be 40 later it was reduced to 15 based on the mean error rate plot.

```
from sklearn import metrics
from sklearn import model_selection
from sklearn.neighbors import KNeighborsClassifier
import time
# KNN model creation
t0 = time.time()
knn = KNeighborsClassifier(n_neighbors=40)
knn.fit(X_train,y_train)
t1 = time.time()
print("Model Training time:", (t1 - t0))
# predictions
knn_predict_test = knn.predict(X_test)
knn_predict_train = knn.predict(X_train)
score_knn_train = round(metrics.accuracy_score(y_train, knn_predict_train) * 100, 2)
score_knn_test = round(metrics.accuracy_score(y_test, knn_predict_test) * 100, 2)
t2 =time.time()
print("Model Detection time:", (t2 - t1))
print("Accuracy of KNN on train data :",score_knn_train)
print("Accuracy of KNN on test data :",score_knn_test)
```

```
Model Training time: 1.3918628692626953
Model Detection time: 23.62638783454895
Accuracy of KNN on train data : 91.01
Accuracy of KNN on test data : 90.42
```

*Figure 10: Code snippet of the KNN classifier model*

## 5.5 Dense Neural Network

Dense Neural Network is a deep learning technique where neurons in one layer are connected with the neurons in the consecutive layers. A densely connected layer provides learning features from all the combinations of the features of the previous layer. A sequential model is used when each layer has exactly one input tensor and one output tensor. The sequential model was designed with two dense layers having the *relu* activation function and one dense layer having the *sigmoid* activation function.

```
from keras.models import Sequential
from keras.layers import Dense
```

```
Using TensorFlow backend.
```

```
model = Sequential([Dense(32, activation='relu', input_shape=(17,)), Dense(32, activation='relu'), Dense(1, activation='sigmoid'),])
```

```
model.compile(optimizer='sgd', loss='binary_crossentropy', metrics=['accuracy'])
```

```
history = model.fit(X_train, y_train, batch_size=32, epochs=50)
```

*Figure 11: Code snippet of Dense Neural network*

Accuracy was defined as the Keras classification metric for accurate classification of the IoT botnet. To understand the performance of the sequential model binary cross-entropy was defined. Also, the Stochastic Gradient Descent (SGD) optimizer was used to estimate the error gradient of the current state of the model and to update the weights of the model using the backpropagation method. To yield better and faster results the batch size was 32 and iteration was 50. Characteristics of the three dense layers implemented in this study are shown in the below figure.

```
model.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_13 (Dense)             (None, 32)                576
_________________________________________________________________
dense_14 (Dense)             (None, 32)                1056
_________________________________________________________________
dense_15 (Dense)             (None, 1)                 33
=================================================================
Total params: 1,665
Trainable params: 1,665
Non-trainable params: 0
```

*Figure 12: The Sequential model summary.*

# 6 Evaluation

This section illustrates the different experiments conducted with a motive of evaluating the accuracy and performance of the classification models implemented for this research work. Random Forest (RF), AdaBoost, and K-nearest neighbors (KNN) classification algorithms were implemented to accurately identify the IoT botnet. The performance comparison of the implemented models will be based on the outputs of the confusion matrix and ROC-AUC curves.

Accuracy, Sensitivity, Precision, and false-positive rate (FPR) can be calculated for each detection models using the confusion matrix and these evaluating metrics are defined as follows:

- The accuracy is defined as the number of instances the model classified all the normal and IoT botnet records correctly, i.e,

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

- The Sensitivity (SN) is defined as the percentage of IoT botnet records identified correctly, i.e,

$$SN = \frac{TP}{(TP + FN)}$$

- Precision is defined as the percentage of predicted IoT botnet records that turned out to be positive, i.e,

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

- The false-positive rate (FPR) is defined as the percentage of IoT botnet records identified incorrectly, i.e,

$$FPR = \frac{FP}{(FP + TN)}$$

- F1 score is defined as the harmonic mean of precision and recall and the AUC-ROC curve is a performance measurement for binary classification with threshold settings. The AUC-ROC curve tells how well the model differentiates between IoT botnet & normal traffic[moustafa2017].

$$F1score = 2 * \frac{Precision * recall}{precision + recall}$$

- Error Rate is calculated as the total number of incorrect predictions made by the prediction model. The prediction model is considered good if the error rate is close to 0. It is represented by the following formula.

$$Error\ Rate = \frac{(FP + FN)}{(TP + TN + FN + FP)}$$

## 6.1 Experiment with Random Forest classifier:

Random Forest (RF) bagging ensemble algorithm is a group of decision trees that are generated randomly from the training subset. This algorithm aggregates votes of randomly generated decision trees to finally decide the object class. With the Random Forest model, accuracy of 94.55% with a precision of 96.30%, recall of 95.11%, F1-score of 95.70%, and an error of 5.54% was obtained. The training time of the model is 73.87 seconds whereas the detection

time of the model is 4.99 seconds. The bar chart represents the evaluation metrics and results of the Random Forest model.

| Model | Accuracy | Precision | F1-score | Recall | Error Rate |
|---|---|---|---|---|---|
| Random Forest Classifier | 0.9455 | 0.963 | 0.957 | 0.9511 | 0.054 |

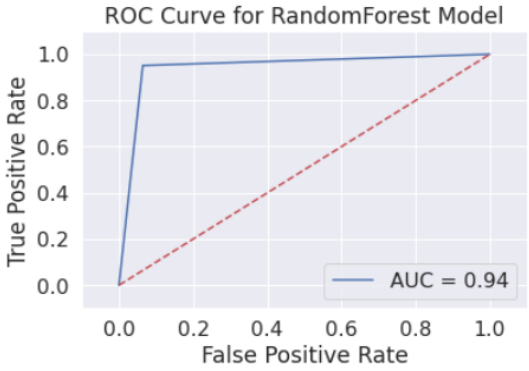


*Figure 13: ROC curve for Random forest model*

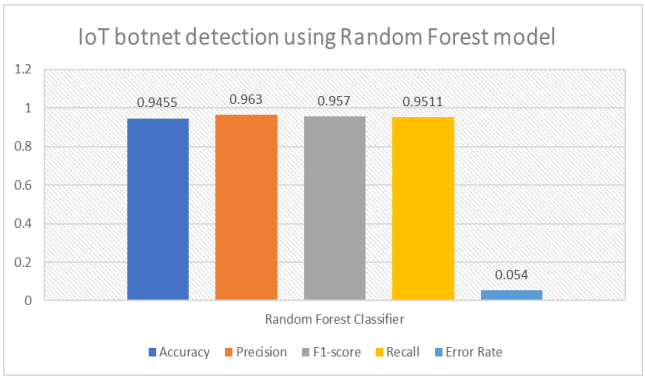


*Figure 14: Random Forest classifier model evaluation graph*

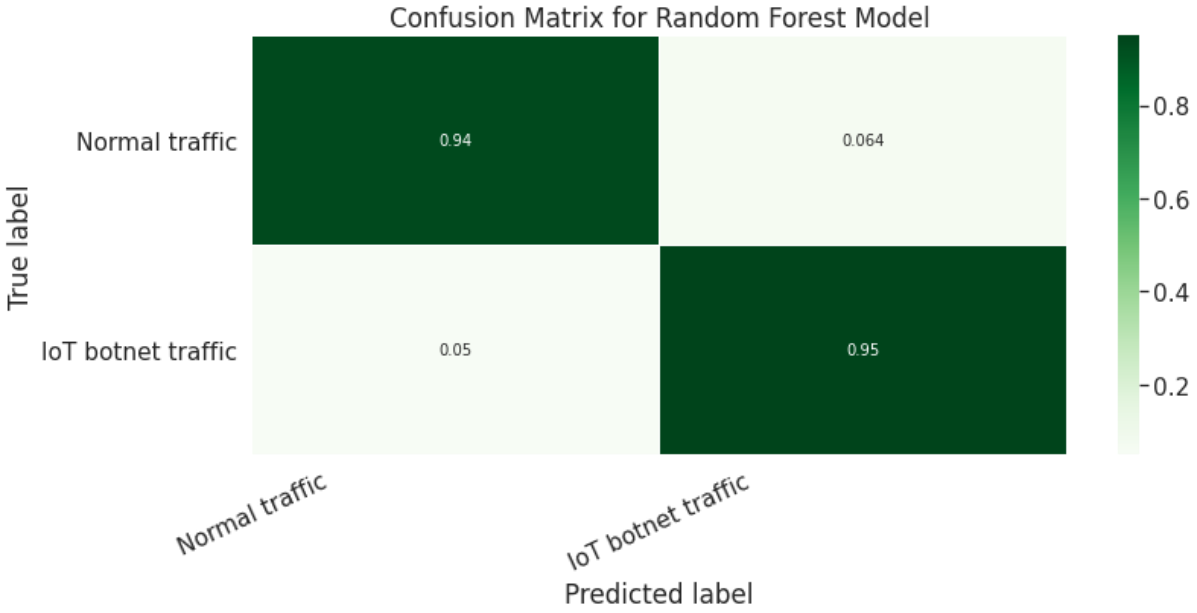


*Figure 15: Confusion matrix plot of the Random Forest model.*

## 6.2 Experiment with AdaBoost Classifier:

AdaBoost ensemble algorithm is usually used to increase the efficiency of the binary classifiers. It employs an iterative approach to understand from the mistakes of the weak classifiers and convert them into strong classifiers. AdaBoost is a boosting ensemble method used to decrease bias and increase accuracy. With AdaBoost classifier, the accuracy rate of 94.29% with a precision of 95.76%, recall of 95.28%, F1-score of 95.51%, and an error of 5.7% was attained. AdaBoost classifier model took 4.42 seconds to detect the IoT botnet with a total training time of 75.32 seconds The bar chart represents the evaluation metrics and results of the AdaBoost Classifier model.

| Model | Accuracy | Precision | F1-score | Recall | Error Rate |
|---|---|---|---|---|---|
| AdaBoost Classifier | 0.9429 | 0.9576 | 0.9551 | 0.9528 | 0.057 |

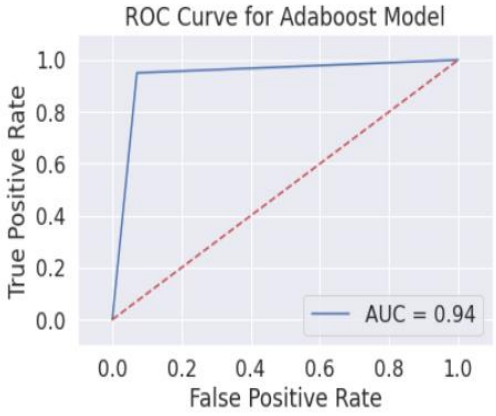


*Figure 16: ROC Curve for AdaBoost Model*

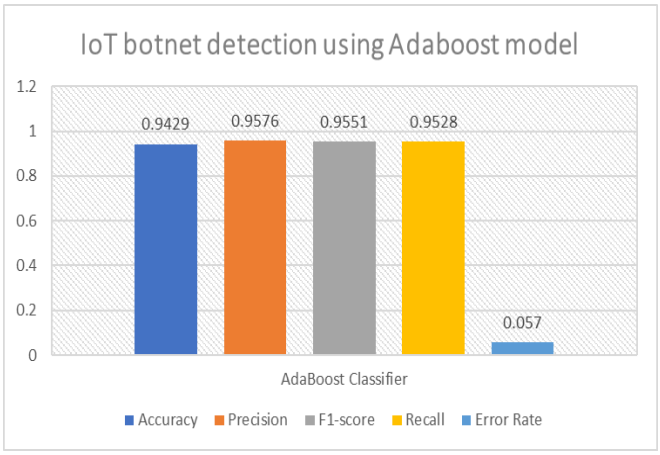


*Figure 17: AdaBoost classifier model evaluation graph*

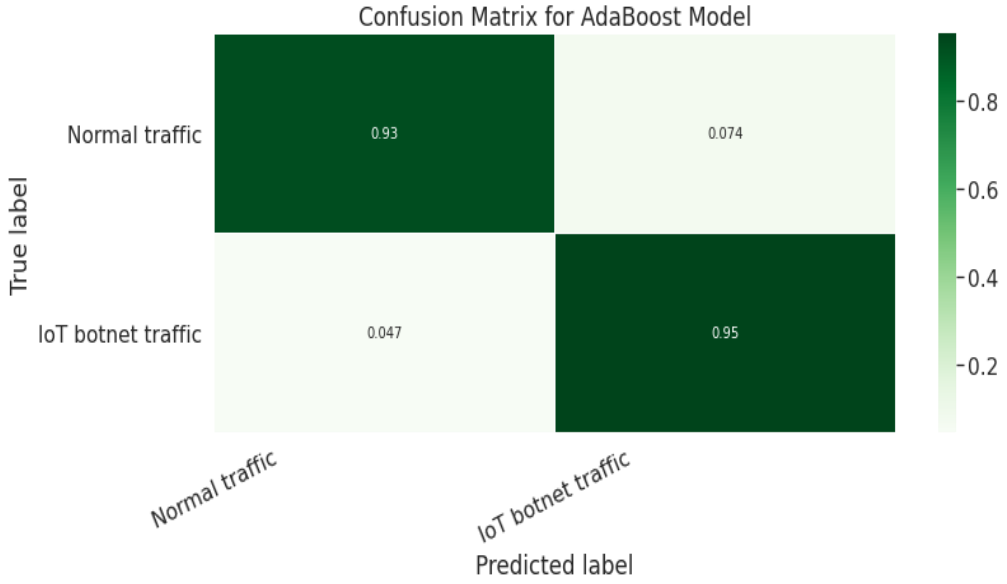


*Figure 18: Confusion matrix plot of the AdaBoost classifier model*

## 6.3 Experiment with KNN Model:

The third classification model employed for this project is the KNN classification algorithm. K-nearest neighbors (KNN) is a simple and versatile supervised ML algorithm that makes use of feature similarity to predict values. With the KNN model, an accuracy of 91.27% with a precision of 92.77%, recall of 93.61%, F1-score of 93.18 %, and an error of 8.72% was attained. The training session time reported is 1.39 seconds and 23.62 seconds to detect the IoT botnet. The bar chart represents the evaluation metrics and results of the KNN Classifier model.

| Model | Accuracy | Precision | F1-score | Recall | Error Rate |
|---|---|---|---|---|---|
| KNN Classifier | 0.9127 | 0.9277 | 0.9318 | 0.9361 | 0.0872 |

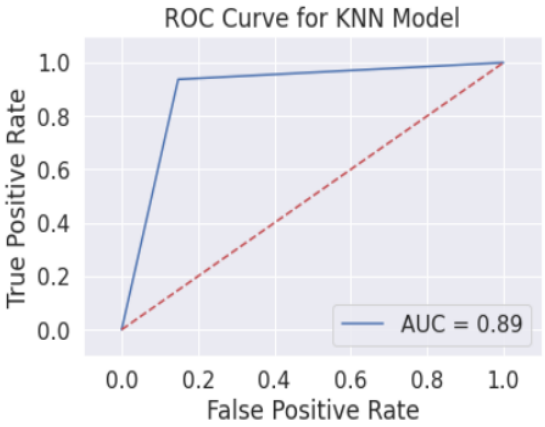


*Figure 19: ROC Curve for KNN Model*

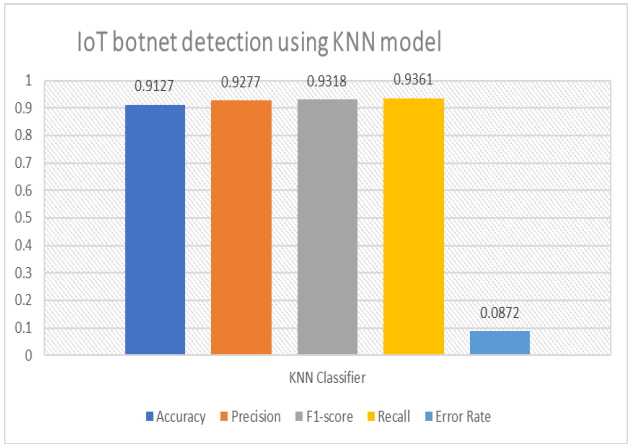


*Figure 20: KNN classifier model evaluation graph.*

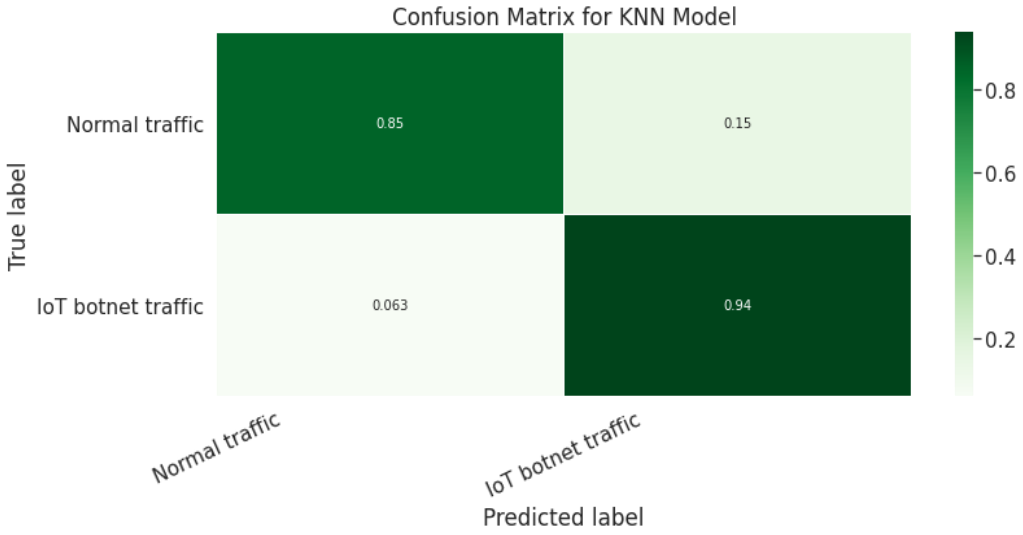


*Figure 21: Confusion matrix plot of the KNN classifier model*

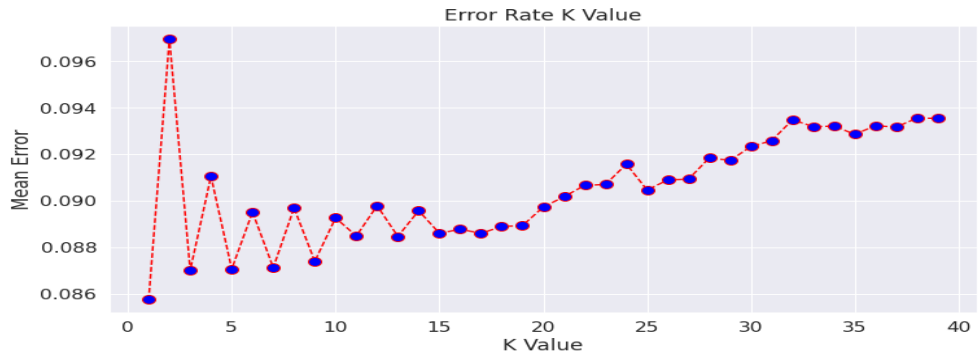


*Figure 22: Comparison of error rate with the K-value*

## 6.4 Experiment with Dense Neural Network:

The final model implemented in this research work is the Dense Neural Network, it is the part of machine learning methods based on the artificial neural network used to provide learning features from all the combinations of the features of the previous layers. The dense neural network employed in this research gave an accuracy of 91.14% with a validation loss of 17.58% obtained after 50 epochs. The below plots summarizes the model accuracy and model loss

obtained during the experiment. From the below plot, it is indicated that the training accuracy is increasing steadily until 50 epochs whereas the test accuracy is fluctuating between 90.8% to 91.7% which can be considered trivial. So the final accuracy achieved at the 50$^{th}$ epoch is 91.14%.

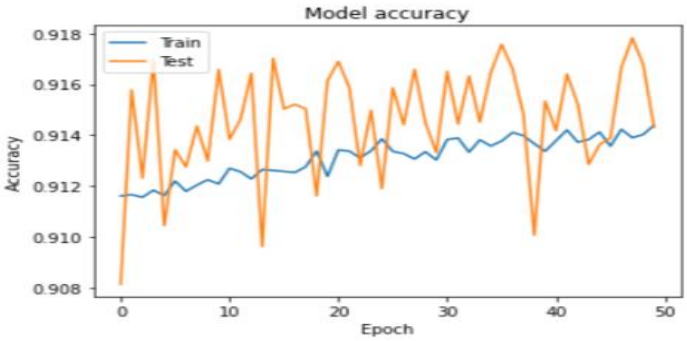


*Figure 23: Dense Neural Networks model accuracy graph.*

The below model loss plot depicts that the training loss is gradually decreasing and the testing loss shows major fluctuations throughout all the epochs and reduces to 0.1758.

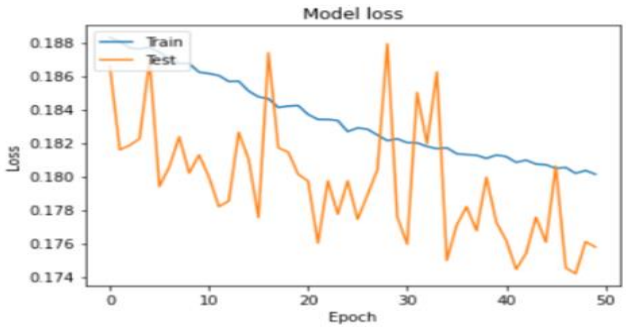


*Figure 24: Dense Neural Networks model loss graph.*

## 6.5 Discussion

The proposed study demonstrated the role of machine learning techniques in successfully identifying the IoT botnet using network traffic information. An empirical analysis of the Random forest, KNN, AdaBoost, and Dense neural networks was conducted on the IoT Intrusion dataset. Metrics such as training time, detection time, accuracy, F1-score, precision, and false alarm rate were taken into consideration and the results revealed that the Random forest classifier has the potential to outperform the other machine learning techniques and deep learning technique. Also, dimensionality reduction was implemented with the help of Spearman's correlation technique thereby achieving feature selection. Feature selection was implemented to reduce the false positive rate and detection time without the accuracy being compromised. Although all the four ML techniques achieved good accuracy, random forest explicitly outperformed with a lesser detection time. Also, the implementation of the dense neural network showed that the model was overfitting for the selected data. The below table shows the comparison of all the four machine learning models.

*Table 3: Comparison of all the models*

| Classification Models | Training Time in seconds | Detection Time in seconds | False Positive Rate | True Positive Rate |
|---|---|---|---|---|
| Random Forest Classifier | 73.87 | 4.99 | 6.40% | 95% |
| AdaBoost Classifier | 75.32 | 4.42 | 7.40% | 95% |
| K-Nearest Neighbor | 1.39 | 23.62 | 14% | 93% |
| Dense Neural Network | 600 | 650 | 17.58% | 91.43% |

# 7 Conclusion and Future Work

In this research work, the performance of the Random Forest classifier in detecting the IoT botnet was assessed. Also, the training time and detection time required for the various machine learning models were evaluated. The Random Forest algorithm executed in 4.99 seconds and achieved 94.47 % accuracy with a precision of 96.28%. The computational resources required to implement this research work are low thereby making it a suitable choice in the IoT environment.

*Limitations of this Work:* Due to time and resources constraints it was not possible to generate our own IoT dataset by simulating an IoT network. Also, not many open source simulators supported IoT networking. Research on Contiki OS (Cooja simulator), IoTify, Mininet, and Cupcarbon simulator was conducted to understand the working of IoT devices. However, extracting the network details was not feasible therefore UNSW_NB15 dataset was used in this research work.

In the future, IoT network traffic details can be extracted using open source IoT simulators, and later that data can be used to train the machine learning models to effectively identify the IoT botnet.

# 8 Acknowledgment

I would like to thank my supervisor, Mr Vikas Sahni, for his dedicated support and guidance. Mr Vikas Sahni continuously encouraged and was always willing and enthusiastic to assist in any way he could throughout the research project. I would also like to thank my friends and family for their constant support and guidance. Finally, many thanks to all my data analytics friends who helped me during the rough phase of the implementation and a big thanks to the School of Computing for this wonderful opportunity.

# References

[1] S. Nomm and H. Bahsi, "Unsupervised Anomaly Based Botnet Detection in IoT Networks," in *Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018*, Jan. 2019, pp. 1048–1053, doi: 10.1109/ICMLA.2018.00171.

[2] N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection systems (UNSW-NB15 Network Data Set)." [Online]. Available: https://cve.mitre.org/.

[3] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal*, vol. 25, no. 1–3, pp. 18–31, Apr. 2016, doi: 10.1080/19393555.2015.1125974.

[4] W. Zong, Y. W. Chow, and W. Susilo, "A two-stage classifier approach for network intrusion detection," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 11125 LNCS, pp. 329–340, doi: 10.1007/978-3-319-99807-7_20.

[5] N. Moustafa, J. Slay, and G. Creech, "Novel Geometric Area Analysis Technique for Anomaly Detection Using Trapezoidal Area Estimation on Large-Scale Networks," *IEEE Transactions on Big Data*, vol. 5, no. 4, pp. 481–494, Jun. 2017, doi: 10.1109/tbdata.2017.2715166.

[6] N. Koroniotis, N. Moustafa, E. Sitnikova, and J. Slay, "Towards Developing Network forensic mechanism for Botnet Activities in the IoT based on Machine Learning Techniques."

[7]. "2 Internet of Things."

[8] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer internet of things devices," in *Proceedings - 2018 IEEE Symposium on Security and Privacy Workshops, SPW 2018*, Aug. 2018, pp. 29–35, doi: 10.1109/SPW.2018.00013.

[9] K. Singh, S. C. Guntuku, A. Thakur, and C. Hota, “Big Data Analytics framework for Peer-to-Peer Botnet detection using Random Forests,” *Information Sciences*, vol. 278, pp. 488–497, Sep. 2014, doi: 10.1016/j.ins.2014.03.066.

[10] H. T. Nguyen, Q. D. Ngo, D. H. Nguyen, and V. H. Le, “PSI-rooted subgraph: A novel feature for IoT botnet detection using classifier algorithms,” *ICT Express*, Jun. 2020, doi: 10.1016/j.icte.2019.12.001.

[11] A. L. Buczak and E. Guven, “A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection,” *IEEE Communications Surveys and Tutorials*, vol. 18, no. 2, pp. 1153–1176, Apr. 2016, doi: 10.1109/COMST.2015.2494502.

[12] L. Chen, Y. Zhang, Q. Zhao, G. Geng, and Z. Yan, “Detection of DNS DDoS Attacks with Random Forest Algorithm on Spark,” in *Procedia Computer Science*, 2018, vol. 134, pp. 310–315, doi: 10.1016/j.procs.2018.07.177.

[13] C. Yin, Y. Zhu, J. Fei, and X. He, “A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks,” *IEEE Access*, vol. 5, pp. 21954–21961, Oct. 2017, doi: 10.1109/ACCESS.2017.2762418.

[14] B. Ingre and A. Yadav, “Performance Analysis of NSL-KDD dataset using ANN,” 2015.

[15] D. Breitenbacher and Y. Elovici, “N-BaIoT-Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders.” [Online]. Available: http://archive.ics.uci.edu/ml/datasets/detec-.

[16] F. Palmieri, U. Fiore, and A. Castiglione, “A distributed approach to network anomaly detection based on independent component analysis,” *Concurrency Computation Practice and Experience*, vol. 26, no. 5, pp. 1113–1129, Apr. 2014, doi: 10.1002/cpe.3061.

[17] “fayyad1996.”

[18] "HuffPost is now a part of Verizon Media", Huffpost.com, 2020. [Online]. Available: https://www.huffpost.com/entry/cisco-enterprises-are-leading-the-internet-of-things. [Accessed: 16- Aug- 2020]

[19] M. Mayo, "The Data Science Process, Rediscovered - KDnuggets", KDnuggets, 2020. [Online]. Available: https://www.kdnuggets.com/2016/03/data-science-process-rediscovered.html. [Accessed: 16- Aug- 2020]

[20] "DDoS Hackers Using IoT Devices to Launch Attacks - Corero", Corero, 2020. [Online]. Available: https://www.corero.com/blog/ddos-hackers-using-iot-devices-to-launch-attacks/. [Accessed: 16- Aug- 2020]

[21] T. Sureda Riera, J. Bermejo Higuera, J. Bermejo Higuera, J. Martínez Herraiz and J. Sicilia Montalvo, "Prevention and Fighting against Web Attacks through Anomaly Detection Technology. A Systematic Review", Sustainability, vol. 12, no. 12, p. 4945, 2020.

[22] D. Burgio, "Reduction of False Positives in Intrusion Detection Based on Extreme Learning Machine with Situation Awareness", NSUWorks, 2020. [Online]. Available: https://nsuworks.nova.edu/gscis_etd/1093/. [Accessed: 16- Aug- 2020]