Network Intrusion Detection System using Ensemble Learning

MSc Academic

Cyber Security

Samruddhi Basagouda Patil

Student ID: x18202667

School of Computing

National College of Ireland

Supervisor: Prof. Micheal Pantridge

National College of Ireland

MSc Project Submission Sheet

Name: Samruddhi Basagouda Patil

Student ID: X18202667

Programme: MSc Cybersecurity **Year:** 2019-2020

Module: Academic Internship

Supervisor: Mr. Micheal Pantridge

Submission Due Date: 17/08/2020

Project Title: Network Intrusion Detection System using Ensemble Learning

Word Count: 7372 **Page Count:** 29

School of Computing

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: Samruddhi Patil……

Date: 17th August 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):	

Network Intrusion Detection System using Ensemble Learning

Samruddhi Patil

X18202667

1. Abstract

Cyber-attacks have always been a great threat to the IT sector. Network intrusion also known as APT is a major concern to all the global scale industries and government sectors. APT in a brief refers to persistent long term multi-stage attack whose main goal is to infiltrate the target network and gather sensitive information anonymously which can lead to great financial losses. These attacks are generally launched with an intention to steal data rather than to cause damage. This thesis aims at developing a system that can detect an APT attack using the concept of machine learning. The thesis aims at boosting the overall accuracy of the prediction by making use of an ensemble learning algorithm. An ensemble-based algorithm is a hybrid algorithm that is formed by a combination of two or more sets of models.

Keywords: **Network Intrusion Detection System, Ensemble Learning, Machine Learning, XGBoost, KNN, Logistic regression, Stacking, Hybrid Algorithm, Classifiers**

2. Introduction

The development of the Internet has been a revolution for humankind and a great break-through in the field of technology. From advertisements to the marketing of the company, everything went online. Every business makes use of computers and the internet for its daily work. Sensitive information is encrypted and stored online. In the current world of technology, information has become a key asset for the success or failure of any state or organization. Today's age is the age of information. Large private or government organizations spend heavily on the security of their sensitive information. With the development of technology, there is also the need for a more developed version of security. There has been a huge increase in cyber-attacks in current years. These cyber-attacks are more complex, in large volume and variety. Cybercriminals are constantly increasing their skill set and it is getting harder and harder to deal with them with usual antiviruses, firewalls, and intrusion detection and prevention systems (IDPSs). Damage done by network intrusion attacks in 2015 was about $3 trillion. According to Morgan, the annual damage cost from APT a kind of network intrusion attack can go up to $6 trillion till the year 2021 [22]. There is a common understanding of people is that if their security is good enough, the attackers will ignore their network and will move on to find a more vulnerable network. This is not true in the current world scenario. The attackers are persistently trying to intrude on a more secure system if the prize is great [13]. Now more cybercriminals are targeting selective organizations with all their efforts put into hacking a single selected organization. A very common kind of network intrusion attack is APT.

2.1 Advanced Persistent Threat (APT)

APT in itself is a very broad term. An APT attack is used to describe a series of actions taken by an attacker. APT involves performing intrusion, establishing an illicit, staying on the network for a long amount of time while maintaining anonymity, gathering the information, on and the last which is covering your tracks. The APT attacks are committed with a very high level of

planning and a good amount of reconnaissance done on the target. The attack is not committed with an intension to break something but to gain sensitive information anonymously. This is also a reason why such kind of attacks is very dangerous. The victim will never realize what hit them even after the damage is done.

Damages done by APT attacks include:

- Intellectual information loss. (trade secrets)
- Sensitive information loss. (users private data)
- The sabotaging of critical organizational infrastructures (e.g., database deletion)
- Complete website takedown

The cost of executing an APT is very high. It is at least more than executing a standard attack. The attacks are usually committed by a team of very skilled and experienced cybercriminals. Even governments execute these attacks in order to gain leverage over enemy country's companies.

Difference between an APT attack and general cyber-attacks:

- APT attacks are significantly more complex.
- These attacks are not of hit and run kind. The goal is to infiltrate and commit espionage on the target.
- These attacks are launched with an aim to infiltrate an entire network rather than targeting a specific part.

Steps involved in Network intrusion Attacks:

1. Infiltration:
   Infiltration takes place generally from compromising of one of these three assets web assets, network resources, or employees with high authorities. This can be done through some SQL injections or other malicious uploads like RFI. Spear phishing can also be used for infiltration. DDoS attacks can be launched simultaneously in order to distract network officials. As soon as the infiltration is successful the hackers install a backdoor malware.

2. Expansion:
   As soon as a foothold is established in the network, the attacker tries to move up the hierarchy by compromising employees with higher authorities. By this, the attacker is able to gather some serious sensitive information about the organization. The attacker now has control of the system. He can manipulate data as he feels. He can even delete an entire database.

3. Extraction:
   While the attack is being conducted, the attacker is gathering information at a place. This information has to be extracted once enough data has been collected. DDoS or white noise tactics can be used by the attacker to extract the information without getting noticed.

2.2 Some recent network intrusion based attacks are

- An APT malware known as Sykipot made use of the vulnerabilities in Adobe Reader and Acrobat to perform a very long-running cyberattack on UK and US organizations. Which mainly included government agencies, defense contractors, and telecommunication agencies. It was first detected in 2006 and attack continued till 2013. The hackers used

spear-phishing attacks along with APT sending their users malicious links and attachments having zero-day target emails.

- A cyber intrusion attack dubbed as GhostNet which originated from China in the year 2009 used various spear-phishing emails having various malicious attachments and links. These malicious attachments compromised computers from about 100 countries. The attack was highly focused on gaining sensitive government information related to ministries and embassies. The attackers compromised various devices which included recording devices, CCTV cameras, etc.

- In 2010, a worm name Stuxnet which was used to attack Iran's nuclear program was detected. It was used for gaining knowledge on Iran's SCADA (supervisory control and data acquisition) nuclear projects. The USA and Israel were found to be linked in the development of this worm. Both countries never confessed. The malware is still considered to be the most advanced form of network intrusion attack. It was spread using IO devices like flash drives.

2.3 Aim:

This thesis will make use of a hybrid machine-learning algorithm to tackle network intrusion attacks. A model will be build based on the dataset. Feature extraction and pre-processing will be done. The model generated will then be used to detect a network intrusion based cyber-attack. Machine Learning is a way in which the system is given a set of features based on the input data. Using these sets of features, the model predicts whether a network intrusion attack is happening or not. Machine Learning can produce better results as compared to other tools developed based on logics as Machine learning can identify some pattern that occurs while the attack is happening in the learning process. The model will learn that pattern and immediately raise an alarm when that pattern reappears (testing phase).

2.4 Research Question

- The aim of this thesis is to make use of an Ensemble-based machine learning algorithm to detect network intrusion attacks with higher accuracy as compared to the traditional machine learning algorithm.
- To perform a comparative study of multiple machine-learning algorithms to compare the results of traditional algorithms and a hybrid algorithm.

2.5 Structure of the Document

Chapter 2: Literature Review:
This section will be consisting of a discussion on the literature based on the network intrusion attacks. This section will analyze the previous work done in this field. Their pros and cons will be discussed.

Chapter 3: Methodology:
This section will consist of a detailed discussion of the algorithms used and all the requirements for the development of this thesis.

Chapter 4: Design:

This section will discuss the implementation phase of the thesis. We will discuss the code structure of each algorithm and methods used.

Chapter 5: Results:
This section will discuss the results obtained after the implementation of the above thesis. A discussion will be done based on the comparison of the results obtained by different algorithms.

Chapter 6: Conclusion & Future:
This section will discuss the conclusion made after the implementation of this thesis. The research question will be answered in this section. Furthermore, future scope will be discussed.

3. Literature Review

3.1 Introduction

This section will consist of a discussion about the literature studied by me for the development of this thesis. A brief discussion will be done regarding the pros and cons of the literature. Finally, a literature discussion will be done.

3.2 Machine Learning

Machine Learning a branch of AI is also known as the science which gets the computers to act without the involvement of any logical code. [12] The concept of machine learning has lead to the development of some amazing gadgets like self-driving cars, practical speech recognition, effective web search, and a vastly improved understanding of the human genom. [24] The idea behind machine learning is to generate a system that can take decisions on its own and make a prediction based on the set of inputs provided to it. These inputs can be passed to the data. In case of dealing with cybersecurity aspects, machine learning tends to produce better results. [7]

Types of machine learning approach:
1. Supervised Learning
2. Unsupervised Learning

Supervised Learning:
In a supervised form of machine learning the model is given both, the input and the output data as a dataset. Different features are given as input data. The job of a supervised machine learning algorithm is to use these given features and identify any relationship between the input data and output data. In the supervised form of learning the model already knows the final result. It just needs to learn the path for reaching that result. Here for example, if the algorithm must figure out if the email is spam or not. These 0 and 1 based Boolean problems come under supervised learning. In a broad sense, the algorithms that come under Supervised Learning are Classification algorithms and Regression. [12]

Unsupervised Learning:
Unsupervised machine learning is a type of learning in which only input data is given to the algorithm and the output is unknown. Here, the algorithm has no supervision regarding the steps it should take. The algorithm has to analyse some pre-existing hidden patterns in the input dataset and make predictions or find a solution. This is the hardest learning method. The algorithm has to follow the try and error kind of approach to reach a conclusion. Using Unsupervised Machine Learning, new methods can be developed to tackle any problem. Hence this kind of approach is also known as the knowledge discovery approach. [12]

3.3 Literature

Ghafir et al developed a system MLPLAT to prevent intrusion attacks. They made use of the Random Forest classification algorithm for detecting an intrusion. They classified the entire detection process into multiple steps. [15]

Step 1 (Intelligence Gathering): This step cannot be detected as there is no intrusion taking place at this step.

Step 2 (Point of Entry): Malicious domain name and exe files are detected at this phase.
DEFD(Disguised exe file Detection): At this step, any kind of file which is entering the system is scanned. The scanning process is to check if the file entering consists of exe content and is trying to behave as a non-exe file.

Step 3: This step involves the detection of a malicious IP address.
Malicious Domain Name Detection (MDND): This module is used to detect any connection to a malicious domain name. It is based on a blacklist of malicious domain names. DNS traffic is filtered, all DNS requests are analysed and the queries are matched with the blacklist.

Step 4: The system keeps an eye on incoming and outgoing traffic.
MFHD (Malicious File Hash Detection): The system keeps an eye on the traffic going out and coming in. The module detects any malicious file downloads from a network host.

Results obtained by the implementation of MLPLAT consisted of an accuracy of about 84.8% with FPR (False Positive Rate) about 4.5% and TPR (True Positive Rate) of about 81.8%. It consists of four network intrusion detection steps, thus increases the overhead of the system. The accuracy obtained is not very high and certainly can be improved.

Brogi and Tong developed a system called Terminaptor. The system made use of IDS for data collection. The IDS records data from the network and returns them in the chronological order of happening of the events recorded. Whenever an attack is detected an alert is raised by the IDS. Each alert is used to indicate a list of events. An attack is interpreted as a list of events occurring in the system. [6]

Type of event	Input objects	Output objects
Create process	current process	new process
Execute file	filename	current process
Change permissions	current process	filename
Change owner	current process	filename
Create file	current process	filename
Read file	filename	current process
Write file	current process	filename
Receive packet	remote socket	local socket current process
Send packet	local socket current process	remote socket

List of Events [6]

These events are treated as a flow of information through the system. Terminaptor keeps a watch on the flow of information throughout the system. This flow of information is used by it to find any existing links between elementary attacks of a network intrusion lifecycle.
The author demonstrated the working on Terminaptor on only two scenarios which are definitely not enough.

Chandra et al developed a system to detect APT intrusion attacks when done using spear-phishing tactics. The approach solely depends on the mathematical and computational analysis of spam emails. The words contained in the email are tokenized and pre-processed. Machine learning algorithms are applied to these bags of words and a model is created. This algorithm uses the NLP process to check for spear phishing attacks. [8]
This kind of process is completely dependent on the word match system and it is a completely phishing oriented solution.

Sigholm and Bang came up with an approach to detect network intrusion attacks using Data Leakage Prevention. The system developed solely focus on detecting the last phase of the APT attack which is data exfiltration. A Data Leakage Prevention algorithm is used to detect any data leakage present in the system. It creates a fingerprint of the entire network for the presence of any data leakage. It makes use of sensors for detecting data leaks in the system. The developed system focus on only one aspect of the entire APT cycle. It also has to wait for the sensor to send the data back to it which definitely can be time-consuming. [28]

Halimaa & Sundarakantham developed a system to detect network intrusion using machine learning algorithms. [17] Furthermore, they studied the work of Al-Yaseen, et al. who developed an intrusion detection system using Machine learning algorithms like SVM, ELM, and K-means. The best accuracy was obtained with ELM of about 95%. The dataset was divided into smaller parts and it was observed that SVM worked well with smaller datasets while ELM produces best results for a larger dataset. The larger dataset here refers to the dataset of the size of about 2 lakhs. [1]

The algorithms used were Support Vector Machine and Naïve Bayes. In order to build the model pre-processing of the dataset was done. All symbolic and non-numeric features were removed as these features were of no use to the classifier. The four main features used were Normal, DoS, Probe, R2L. Comparative analysis between SVM and Naïve Bayes was done in order to analyze their accuracy Misclassification rate. Methodologies like CFS subset evaluation were performed for the reduction of features. The dataset was processed in the WEKA tool for binary conversion of features.

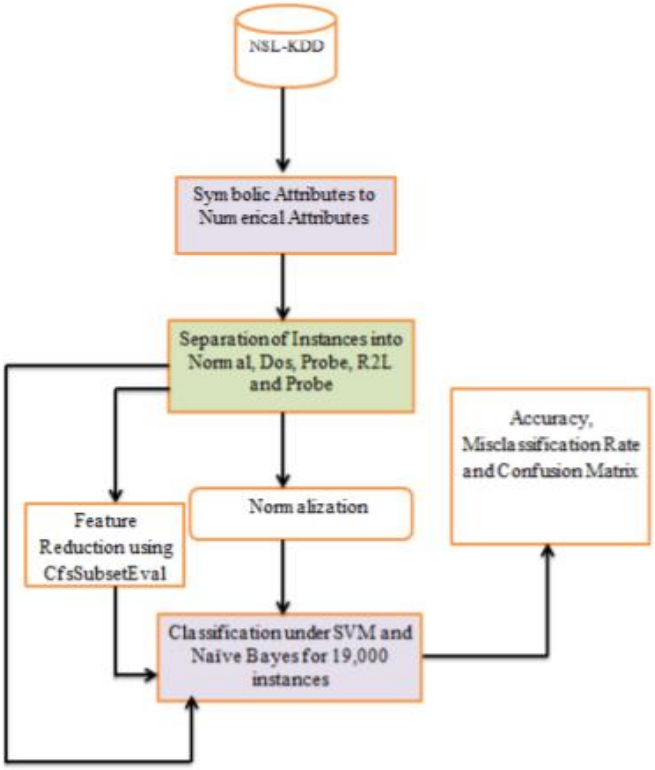

Fig 3.1: Block Diagram [17]

Govindarajan & Chandrasekaran developed a system to detect network intrusion using K-NN algorithm. KNN is a supervised form of a machine learning algorithm. The data points are classified based on the features of that data. The classifiers do not use any model to fit and only based on memory. Given a query point, we find K number of objects or (training points) closest to the query point. The classification is using majority vote among the classification of the K objects. Any ties can be broken at random. K-Nearest neighbor algorithm used neighborhood classification as the prediction value of the new query instance. [16]

Algorithm Used:

Algorithm: K-Nearest Neighbor.

```
Input :
        T // Training data
        K  // Number of neighbors
        t   // Input tuple to classify
Output:
        c  // class to which t is assigned
            KNN Algorithm
        N = 0;      // Find set of neighbors, N for t
For each d e T do
```

```
if | N | = K, then
    N = N U {d}
 else
    if ∃ u e N such that
      sim(t,u) = sim (t,d) then
       begin
          N = N – {u};
          N = N U {d};
       end
// Find class for classification
c = class to which the most u e N are classified;
```

They also evaluated the idea of the K-NN algorithm and used a hybrid form of K-NN developed by them. Feature selection and model selection are done simultaneously by here. In order to optimize the overall working, other methods are used along with K-NN like comparative cross-validation. Execution of K-NN is done with a bagging concept hence termed as a hybrid. The results obtained by executing this model showed that the run time reduces drastically with a high decrease in the error rate. [16]

Literature Review Discussion

MLPAT system developed by Ghafir, et al. performed well for a larger dataset. The problem with MLPAT was that with so much functionality, the overhead of the system increases by a huge amount, thus reducing the overall speed of the model. [15]

A system developed called Terminaptor by Brogi & Tong made use of IDS to monitor the flow of information throughout the network. The accuracy obtained was not very high and according to the authors the attack detection test was performed for only two types of attacks. [6]

The system developed by Sigholm & Bang to detect APT attacks using data leakage seems to be a good approach as the system monitors all the data that leaves the network. But, the problem with this system is that it is very slow. The developed system focuses on only one aspect of the entire APT cycle. It also has to wait for the sensor to send the data back to it which definitely can be time-consuming. [28]
The system developed by Govindarajan & Chandrasekaran showed good results but only in terms of the K-NN algorithm. The system claimed that the results obtained by its hybrid K-NN are better than previous ones. [16]

This system will make use of the K-NN algorithm as it showed better results in the case of Govindarajan & Chandrasekaran to compare its results with a hybrid classifier. Other ensemble algorithms that will be used for comparisons are XGBoost and Stacking classifier. This thesis will compare algorithms used in the literature review that produced good results for the hybrid ensemble used here. [16]

4. Methodology

4.1 Introduction

This section discusses the algorithms, dataset used for the development of this thesis. The section will also consist of a brief discussion about the requirements (hardware and software) for the development of this thesis.

4.2 Language

I have used Python as our main language for the development of this thesis. Python is a very much machine learning-oriented language. It consists of a variety of machine learning libraries that can be used directly with an import. Developers across the world usually choose python to work with machine learning due to its long list of machine learning libraries. Python has great community support and hence new features keep on getting added with every update.

4.3 Feature Selection

Even with the great problem-solving abilities that machine learning possesses, one of the greatest problems that can occur is the overfitting of the dataset due to the use of a large set of features. The greater the number of machine learning features, the larger the dataset gets and the more complex model is developed. In the era of technology, information gaining has become much easier and hence the dataset created usually consists of a higher number of features. The training time also increases with an increase in the number of features which ultimately increases the overhead of the system. [5]

Feature selection is used to overcome this problem. With the help of feature selection, the correlation among different features can be obtained. The correlation describes the degree of dependency among the set of features. On the basis of this dependence or correlation, the features are selected.

Aldehim and Wang researched feature selection. They made comparisons of the results obtained with the help of a full dataset and partial dataset with a selected set of features. The results obtained from their research showed that the selected set of features produced better results as compared to the full set of features.

SelectFromModel feature selection method is used along with Extra Tree classifier for feature selection in this thesis.

Extra Tree Classifier:

Extra tree classifier also known as extremely Randomized Tree classifier is a kind of ensemble learning approach. Extra tree classifier consists of multiple decision trees. The results obtained from each decision tree are aggregated to obtain the final results. At each node of the decision tree classifier, it has to select k set of features for further processing. These k-set of features is obtained by calculating the gini value for each feature.

The generation of each tree in an extra tree classifier takes place with the help of training data. A set of random feature data from test data is provided to the trees during each decision point or node. From these provided sets of features, each decision tree must select a feature for performing a further split. This split is basically performed using some mathematical formulae like typically gini value. This random sample of features leads to the creation of multiple de-correlated decision trees.

The algorithms used for the development of this thesis

1. XGBoost
2. KNN
3. Logistic Regression
4. Stacking (XGBoost + KNN) as a base classifier (Random Forest) as meta classifier

4.4 XGBoost

XGBoost is an ensemble-based algorithm. It makes use of decision trees for building a model. XGBoost is most effective when dataset size ranges from medium to small. XGBoost has been developed on the idea of Gradient boosting. IT can also be considered as an advanced version of gradient boosting. The XGboost algorithm consists of a more optimized system and more enhance form or algorithm structure as compared to the GVM framework. [21]

Chen & Guestrin presented the XGBoost algorithm for the first time at SIGKDD Conference in 2016 and since then it has created a great impact on the Machine Learning community around the world, from winning Kaggle competitions to solving major problems. [9]

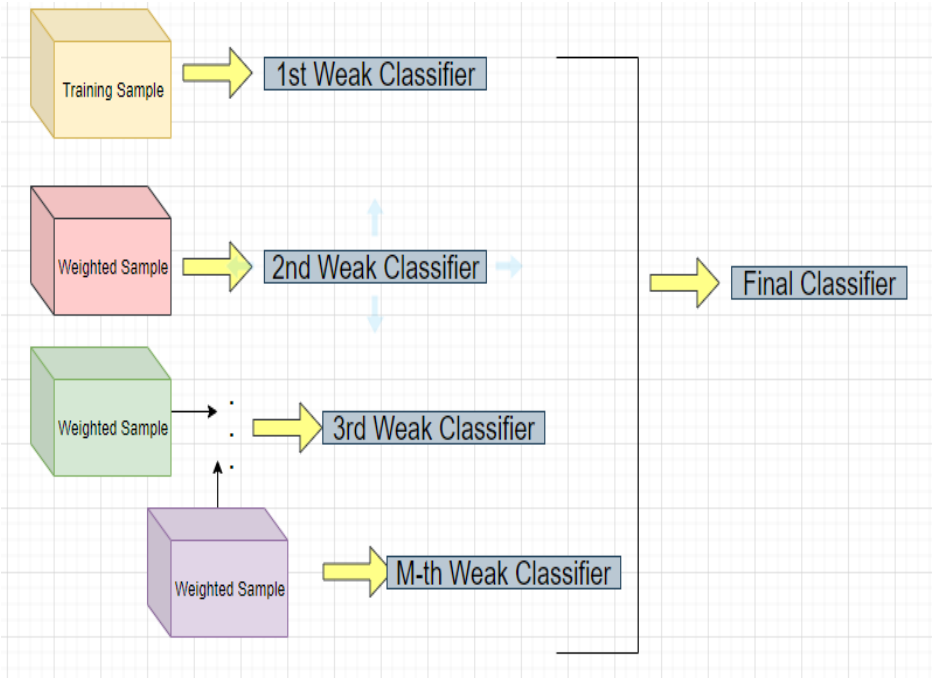

Fig 4.1 XGBoost Classifier Algorithm Flow Chart [26]

Algorithm:

1. A subset is selected from the training dataset.
2. Now the model is trained on the dataset. The result of the first model is treated as the output for the next model in the iteration.
3. The weaker classifier is assigned the higher weight while the strong classifiers are assigned lower weights.
4. During each phase of the iteration, weight is assigned to each classifier.
5. This process iterates until the complete training data fits without any error or until it reached the specified maximum number of estimators.
6. The entire output is summed and the final prediction is achieved.

4.5 KNN (K Nearest Neighbours) Algorithm

KNN is used for both classification and regression purposes. However, it specializes more in the classification part. KNN utilizes 'feature similarity' for identifying new data points. The new data points generated will be given a measure based on the distance of the data point from the training dataset. [18]

Steps involved in the implementation of the KNN algorithm:

1. The dataset is divided into training and testing dataset and both the training and testing dataset is loaded.
2. The training dataset is divided into different clusters based on their type.
3. The k-value is selected for the data point for which the classification has to be done.
4. K closest neighbor data points are selected from all the classified clusters. A distance function is used for finding these closest neighbors.
5. The new data point is assigned to the cluster for which the number of neighbors is high.

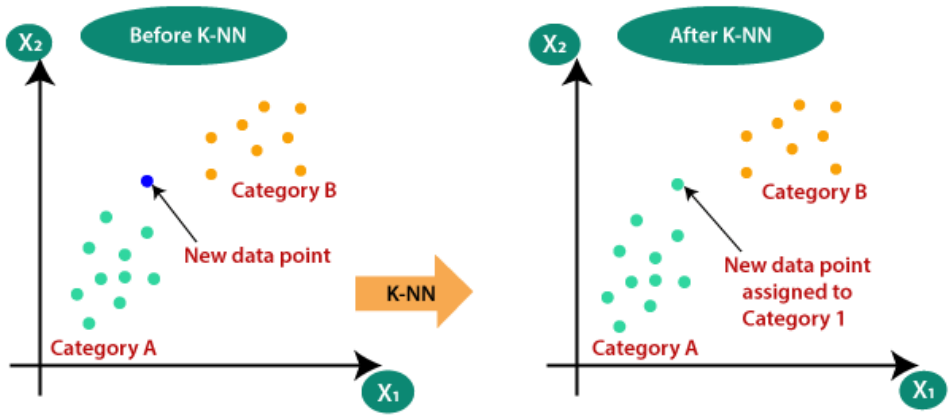

Fig 4.2: Data point classification [2]

Why is the KNN algorithm is used

Suppose we have two different categories present for example A and B and a data point. K-NN algorithm can be used to identify which category the data point belongs to.

Different decision functions can be used for calculating the closest neighbors.

Distance functions

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

Fig 4.3: Distance Function [23]

4.6 Logistic Regression

Logistic regression is used for making a binary classification. For example identifying whether an email is a spam or not, whether a tumor is malignant or not. That is, Logistic regression works well for classification between 0 and 1 kind of problems. Logistic regression makes use of the logistic function thereby estimating the underlying probability for identifying the relationship between a dependent and multiple independent variables. For making the final prediction the probability values are transformed into binary measures. This is achieved with the help of a sigmoidal function also referred to as logistic function. [29]

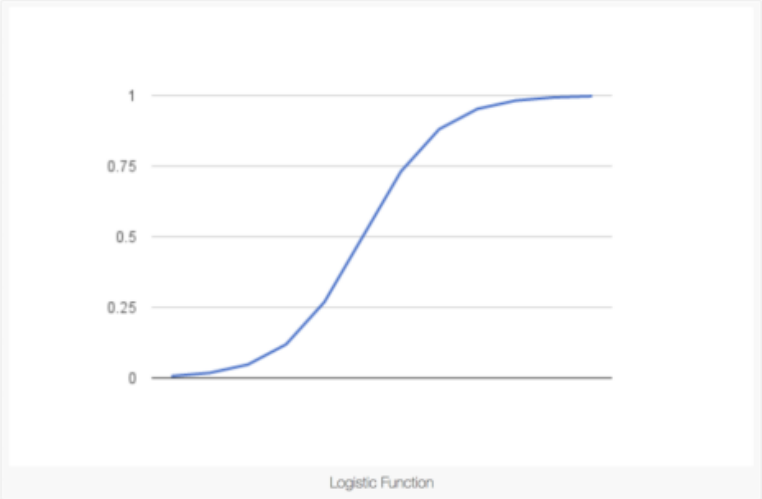

Fig 4.4: Sigmoidal function [10]

The sigmoidal function can take any value ranging between 0 and 1 as input. These values are transformed into binary values as 0 and 1 by sigmoidal function.

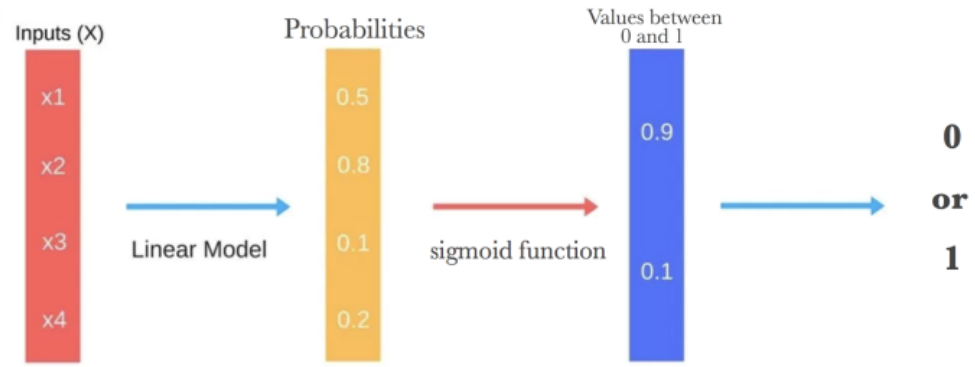

Fig 4.5: Logistic regression [10]

The above figure displays a step by step functioning of logistic regression.

4.7 Stacking Algorithm

Stacking is an ensemble-based machine learning algorithm. There are different kinds of ensemble/hybrid algorithms. The most famous form of the ensemble is bagging and boosting. Bagging comprises of multiple weak classifiers to operate in an isolation. The result of each classifier is combined in the end to obtain the final result. Boosting involves multiple classifiers to be trained in an iteration. The result of one classifier is passed to the other in the iteration. Each iteration overcomes the error produced by the previous model. Stacking is quite different from bagging and boosting. The idea behind the stacking algorithm is to attack a problem from various paths. Different types of models are trained independently in order to overcome different weaknesses. Finally, the results of the models are fed to a meta classifier which gives the final results. Stacking classifier is also known as stacking Generalization. The final model or the meta classifier is stacked over the base learners in case of stacking. [11]

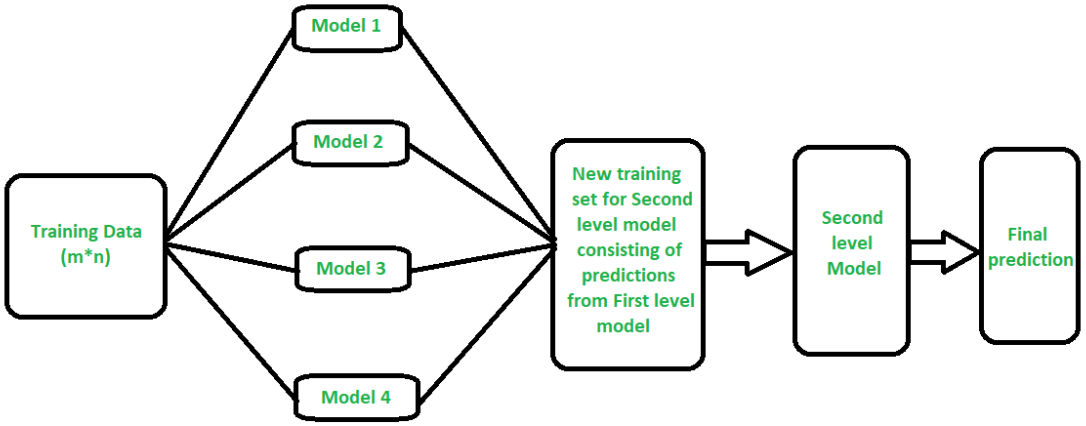

Fig 4.6: Stacking algorithm [11]

Steps involved in Stacking algorithm:

1. First, the training dataset is divided into K different parts.
2. Predictions are made for the K part by fitting a base model on K-1 parts.
3. Steps 1 and 2 are done for each K data part.
4. The performance of the test set is obtained by fitting the base model on entire training data.
5. The above 3 steps are repeated for all the base classifiers.
6. The output obtained from base-level classifiers is treated as an input for the final model.
7. The result of the prediction obtained from the final model is the final result.

4.8 Random Forest

Random Forest is a supervised form of the machine learning algorithm. Random forest is generated by the collaboration of multiple decision trees. Each decision operates independently the results obtained by each decision tree are finally averaged to generate a final result. The averaging is done by bootstrap aggregation of all the outputs obtained from multiple decision trees. [31]

Advantage of Random forest over decision trees

The decision tree operates by performing a split at each logical event. If a split occurs in a decision tree, it cannot go back to the previous logical decision. Problems like overfitting can occur in the case of a single decision tree.

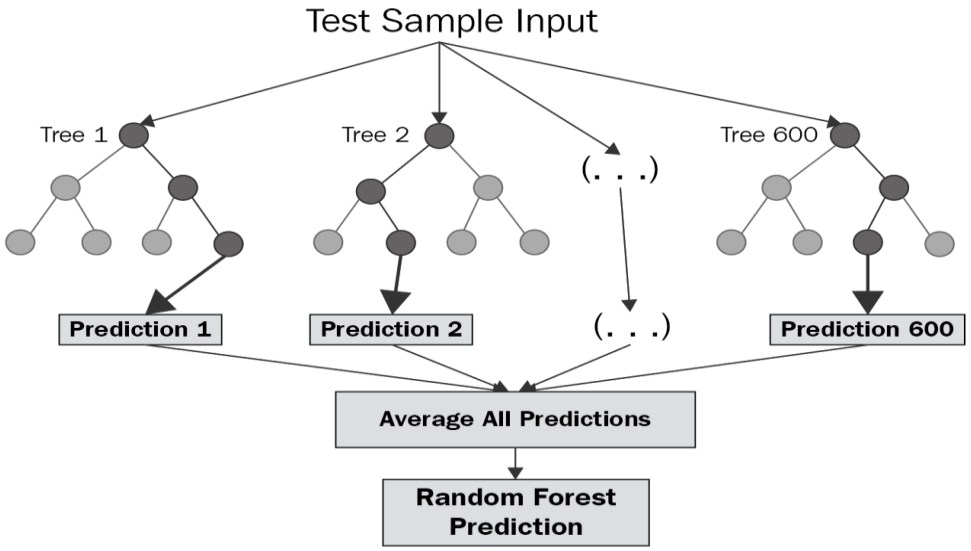

Fig 4.7 Random forest Tree

In case, of random forest algorithm, each tree involved is trained individually and parallelly. These trees do not interact with each other during the training phase. The results of all the trees are combined and a mean is obtained which serves as the final output for a random forest Regressor model.

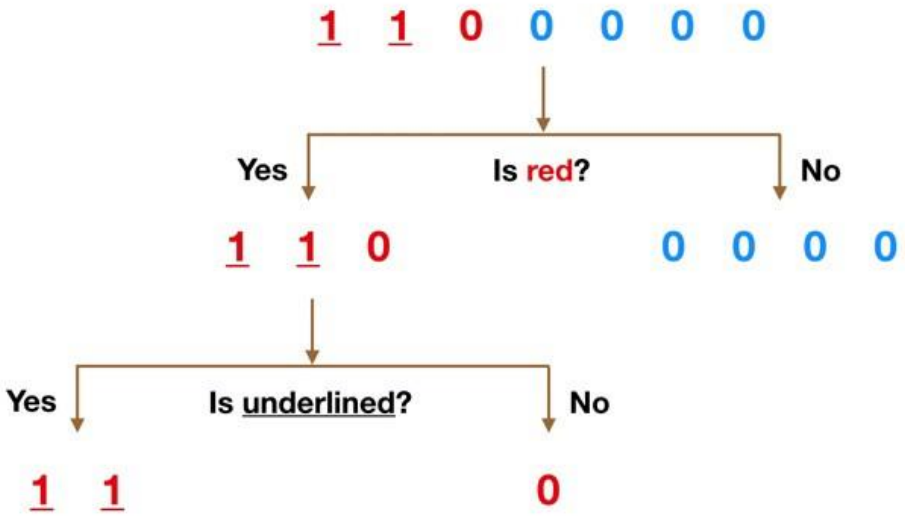

Fig 4.8: Working of decision tree [31]

Fig 4.8 shows an example of a decision tree. As we can see from the figure there are two leaf nodes present. That means two decisions are made by the algorithm to reach a conclusion. At the leaf node, one of the first decisions is made, i.e whether the given data is red or blue. If the numbers are red then the second decision is made, whether the numbers are underlined or not. When multiple decision trees like Fig 4.8 are stacked together to generate a result, then it is termed as a random forest classifier. In random forest classifier the mistake done by one algorithm may not be done by the other using this principle, analysing the result of every tree a final result is obtained.

5. Implementation

5.1 Design

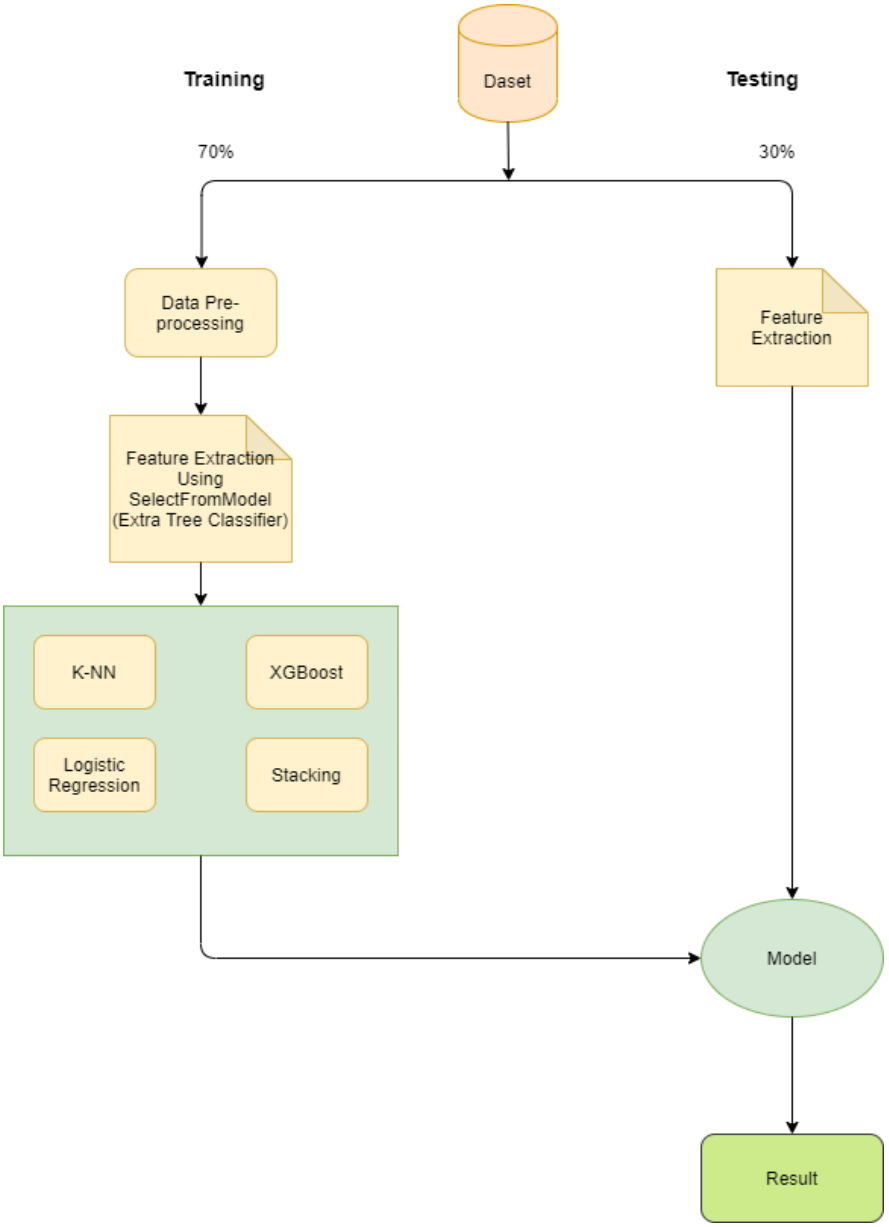

Fig 5.1: Architecture

Figure displays the architecture of this thesis. The dataset used is divided into two parts. 70% of the dataset is used for training while 30% is used for testing. The training data is subjected to pre-processing. The pre-processing involves the removal of garbage values and null values in the dataset. After pre-processing feature extraction is performed on the dataset. SelectFromModel is used along with Extra-tree classifier for extracting main features from the dataset. After feature extraction is performed the dataset is subjected to the algorithms for the training purpose and development of the model. The model developed will be used for testing purposes. Feature extraction is performed on testing data and then the dataset is fed to the model developed during the training phase. Based on its training the model predicts the final results of whether there is a network intrusion performed or not.

5.2 Dataset

The dataset used for this thesis is obtained from https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. The dataset is capable of predicting nine different types of attacks. The dataset was created at Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) using the IXIA PerfectStorm tool for generating a hybrid of real modern normal activities and synthetic contemporary attack behaviours.

5.2 List of Imports used by us for developing this thesis

```
from google.colab import drive
drive.mount('/content/drive')

import numpy as np
import pandas as pd
from sklearn import preprocessing
import xgboost as xgb
from sklearn import svm
from sklearn.metrics import classification_report,accuracy_score,roc_auc_score,average_precision_score,confusion_matrix
import sklearn
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
from yellowbrick.classifier import ClassificationReport
from sklearn.ensemble import AdaBoostClassifier,RandomForestClassifier
from sklearn.ensemble import VotingClassifier
from mlxtend.classifier import StackingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from mlxtend.feature_selection import SequentialFeatureSelector as SFS
from sklearn.externals import joblib
```

Fig 5.2: Libraries used

5.3 Libraries Used

Pandas: It is a machine learning library used in python. Pandas is an open-source library. Pandas is usually used for data analytics and dataset loading. Pandas is being used in a variety of fields for machine learning like finance, economics, etc. It is highly user friendly and can represent dataset into a tabular format for better understanding. [14]

Numpy: Numpy is a machine learning library used in python. Numpy is used for dealing arrays in python. All calculations making use of 1-d or 2-d arrays are done with NumPy. Numpy also consists of functions that can operate with linear algebra and Fourier transform. [4]

Sklearn: One of the most important libraries for machine learning in python is Sklearn. Sklearn consists of lots of tools that can perform statistical modeling and perform classification, regression, clustering, and dimensionality reduction. [19]

MAPTlotlib: MAPTlotlib is a data visualization library. It is a python open-source library used for plotting graphs from the results obtained from the model. These plots can help to better understand the scenario of the results. Different aspects of the results can be graphically formatted for better understanding. [3]

5.4 Algorithms

XGBoost Classifier

```
#XGBOOST
def XGB(X_train,y_train,X_test,y_test):
    xg_clf = xgb.XGBClassifier(max_depth=5, learning_rate=0.01, n_estimators=100, gamma=0,min_child_weight=1, subsample=0.8,
    pred = xg_clf.fit(X_train, y_train).predict(X_test)
    #xg_clf=joblib.load('/content/drive/My Drive/NIDS_ML/features_model/xgb.pkl')
    joblib.dump(xg_clf, '/content/drive/My Drive/NIDS_ML/features_model/xgb.pkl')
    pred= xg_clf.predict(X_test)
    print(classification_report(y_test, pred))
    print(confusion_matrix(y_test, pred))
    print ("XGBoost:Accuracy : ", accuracy_score(y_test,pred)*100)
```

Above shown is the implementation for the XGBoost algorithm. Various parameters like max_depth, learning_rate, n_estimators, gamma, min_child_weight, subsample are set here during model generation. The model generated is (xg_clf). The input dataset is divided into training and testing dataset as X_train, Y_train, and X_test and Y_test respectively. Finally, the model is fitted to the input train dataset. (pred = xg_clf.predict(X_test)). Here pred holds the result of the XGB classifier.

K-NN algorithm

```
def KNN1(X_train,y_train,X_test,y_test):
    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)
    #knn=joblib.load('/content/drive/My Drive/NIDS_ML/features_model/knn.pkl')
    joblib.dump(knn, '/content/drive/My Drive/NIDS_ML/features_model/knn.pkl')
    pred = knn.predict(X_test)
    print(classification_report(y_test, pred))
    print(confusion_matrix(y_test, pred))
    print ("KNN:Accuracy : ", accuracy_score(y_test,pred)*100)
```

The above shown is the implementation of the K-NN algorithm. The model generated is (knn). The input dataset is divided into training and testing dataset as X_train, Y_train, and X_test and Y_test respectively. Finally, the model is fitted to the input train dataset. (pred = knn.predict(X_test)). Here pred holds the result of the K-NN classifier.

Logistic Regression

```
def LR(X_train,y_train,X_test,y_test):

    vc = LogisticRegression()
    vc.fit(X_train,y_train)
    joblib.dump(vc, '/content/drive/My Drive/NIDS_ML/features_model/lr.pkl')
    #vc=joblib.load('/content/drive/My Drive/NIDS_ML/features_model/lr.pkl')
    pred = vc.predict(X_test)
    print(classification_report(y_test, pred))
    print(confusion_matrix(y_test, pred))
    print ("LogisticRegression:Accuracy : ", accuracy_score(y_test,pred)*100)
```

The above shown is the implementation for Logistic Regression algorithm. The model generated is (VC). The input dataset is divided into training and testing dataset as X_train, Y_train, and X_test and Y_test respectively. Finally, the model is fitted to input train dataset vc.fit(X_train, y_train). (pred = vc.predict(X_test)). Here pred holds the result of the Logistic regression classifier.

Stacking algorithm

```
def stacking(X_train,y_train,X_test,y_test):
    classifiers=[xg,knn]
    sc = StackingClassifier(classifiers,meta_classifier=rf)
    sc.fit(X_train,y_train)
    joblib.dump(sc, '/content/drive/My Drive/NIDS_ML/features_model/sc.pkl')
    #sc=joblib.load('/content/drive/My Drive/NIDS_ML/features_model/sc.pkl')
    pred = sc.predict(X_test)
    print(classification_report(y_test, pred))
    print(confusion_matrix(y_test, pred))
    print ("Stacking Classifier:Accuracy : ", accuracy_score(y_test,pred)*100)
```

Above shown is the implementation for the Stacking classifier algorithm. The model generated is (sc). The input dataset is divided into training and testing dataset as X_train, Y_train, and X_test and Y_test respectively. Finally, the model is fitted to input train dataset sc.fit(X_train, y_train). (pred = sc.predict(X_test)). Here pred holds the result of the Stacking classifier.

6. Result

6.1 Introduction

This section will discuss the results obtained by the implementation of the above algorithms. The section will compare the results of each algorithm in terms of the Confusion matrix, f1-score, accuracy, AUC-ROC curve measure.

6.2 Confusion Matrix

Confusion Matrix is a matrix created on the basis of the result obtained by machine learning classifiers. The confusion matrix is also sometimes referred to as the error matrix. The matrix is used to describe the overall performance of the classifier in terms of classification. It provides an easy classification between different classes of the result section. The four boxes of confusion matrix represents the counts of correctly and incorrectly classified data. The confusion matrix represents the actual state of a classifier while making the prediction. [27]

	Class 1 Predicted	*Class 2 Predicted*
Class 1 Actual	**TP**	**FN**
Class 2 Actual	**FP**	**TN**

Fig 6.1: Confusion matrix [27]

TP (True Positive): The actual data point is positive and is predicted positive by our classifier.
TN (True Negative): The actual data point is negative and is predicted negative by our classifier.
FP (False Positive): The actual data point is negative and is predicted positive by our classifier.
FN (False Negative): The actual data point is positive and is predicted negative by our classifier.

Accuracy:

Accuracy is defined by the number of correct prediction done by our classifier divided by the total prediction done by the classifier. [25]

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

Precision:

Precision is the measure of true positive obtained during the classification from all the positive results existing in the dataset. [25]

Recall:

Recall is the measure of data points which are predicted as positive from all the prediction which are termed as positive by the classifier.[25]

$$\textbf{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

$$\textbf{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

F1-score:

F1-score is the value obtained by using both precision and recall. F1-score is a harmonic mean of precision and recall.

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

Types of attacks we are looking for during testing for intrusion are classified for a better understanding of results:

Class 0: Analysis
Class 1: Backdoor
Class 2: Dos
Class 3: Exploits
Class 4: Fuzzers
Class 5: Generic
Class 6: Normal
Class 7: Reconnaissance
Class 8: Shellcode
Class 9: Worms

6.3 XGBoost Results

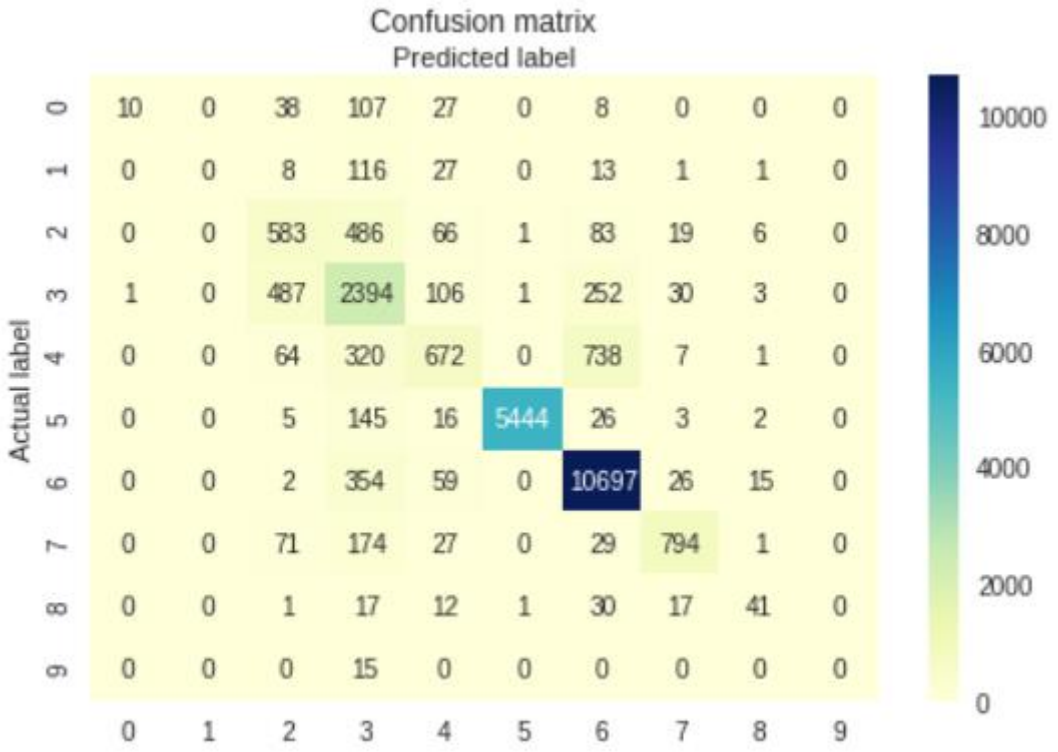

Fig 6.2 Confusion matrix for XGBoost

Above fig 6.2 represents the confusion matrix for the XGBoost classifier. Out of a total of 24700 samples, 20,635 samples were correctly classified while 4,065 samples are incorrectly classified.

Accuracy for XGBoost algorithm = 83.5425101214575

	precision	recall	f1-score	support
Analysis	0.91	0.05	0.10	190
Backdoor	0.00	0.00	0.00	166
DoS	0.46	0.47	0.47	1244
Exploits	0.58	0.73	0.65	3274
Fuzzers	0.66	0.37	0.48	1802
Generic	1.00	0.97	0.98	5641
Normal	0.90	0.96	0.93	11153
Reconnaissance	0.89	0.72	0.80	1096
Shellcode	0.59	0.34	0.43	119
Worms	0.00	0.00	0.00	15
Accuracy			0.84	24700
Macro avg	0.60	0.46	0.48	24700
Weighted avg	0.83	0.84	0.83	24700

Table 6.1

6.4 KNN algorithm results

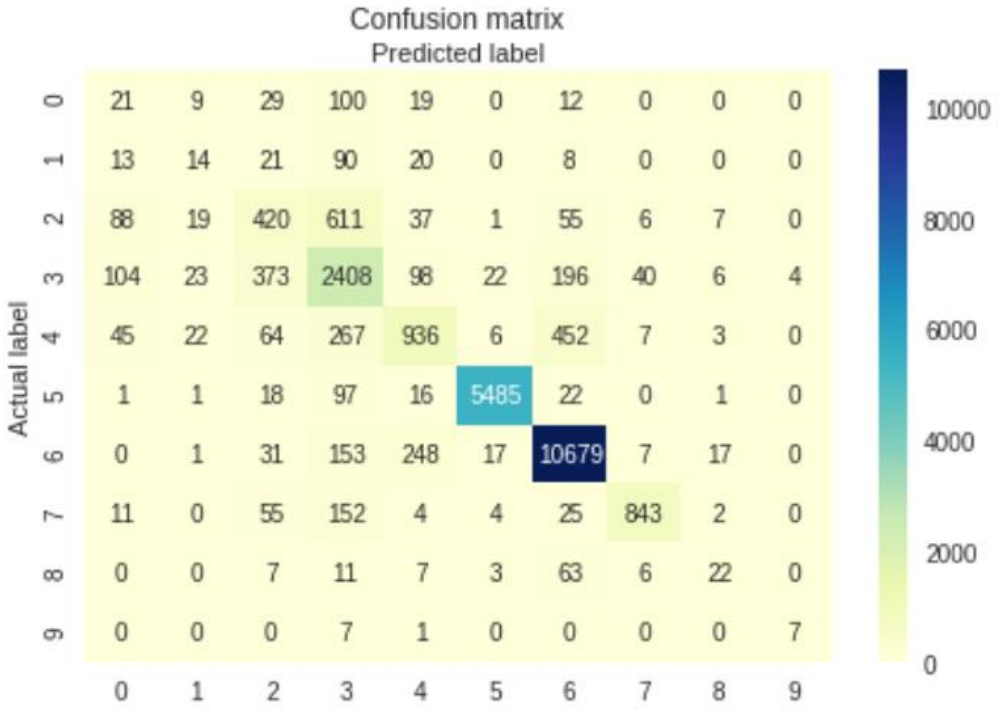

Fig 6.3 Confusion matrix for KNN algorithm

The above fig represents the confusion matrix for the KNN classifier. Out of a total of 24700 samples, 20,835 samples were correctly classified while 3,865 samples are incorrectly classified.

	precision	recall	f1-score	support
Analysis	0.07	0.11	0.09	190
Backdoor	0.16	0.08	0.11	166
DoS	0.41	0.34	0.37	1244
Exploits	0.62	0.74	0.67	3274
Fuzzers	0.68	0.52	0.59	1802
Generic	0.99	0.97	0.98	5641
Normal	0.93	0.96	0.94	11153
Reconnaissance	0.93	0.77	0.84	1096
Shellcode	0.38	0.18	0.25	119
Worms	0.64	0.47	0.54	15
Accuracy			0.84	24700
Macro avg	0.58	0.51	0.54	24700
Weighted avg	0.84	0.84	0.84	24700

Table 6.2

The accuracy obtained for K-NN algorithm is: 84.35222672064778

6.5 Logistic Regression results:

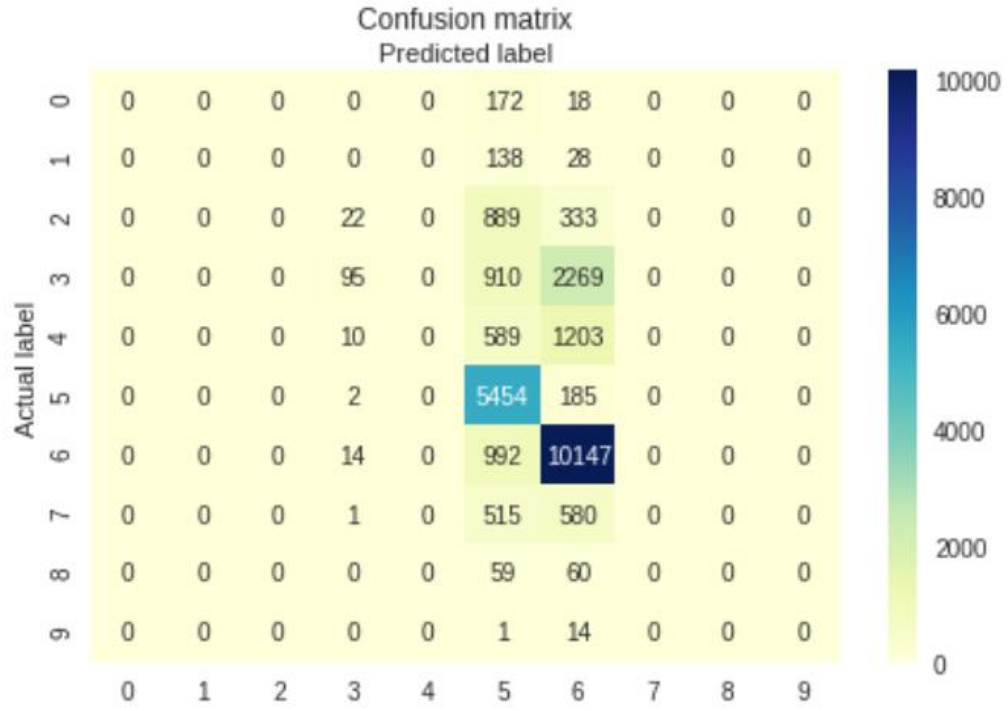

Fig 6.4 Confusion Matrix

Above fig 6.4 denotes the confusion matrix for Logistic regression. Out of 24700 test sample 15,6 96 samples are correctly classified while 9004 are incorrectly classified.

	precision	recall	f1-score	support
Analysis	0.00	0.00	0.00	190
Backdoor	0.00	0.00	0.00	166
DoS	0.00	0.00	0.00	1244
Exploits	0.66	0.03	0.06	3274
Fuzzers	0.00	0.00	0.00	1802
Generic	0.56	0.97	0.71	5641
Normal	0.68	0.91	0.78	11153
Reconnaissance	0.00	0.00	0.00	1096
Shellcode	0.00	0.00	0.00	119
Worms	0.00	0.00	0.00	15
Accuracy			0.64	24700
Macro avg	0.19	0.19	0.15	24700
Weighted avg	0.52	0.64	0.52	24700

Table 6.3

Logistic Regression: Accuracy: 63.54655870445344

6.6 Stacking Classifier results

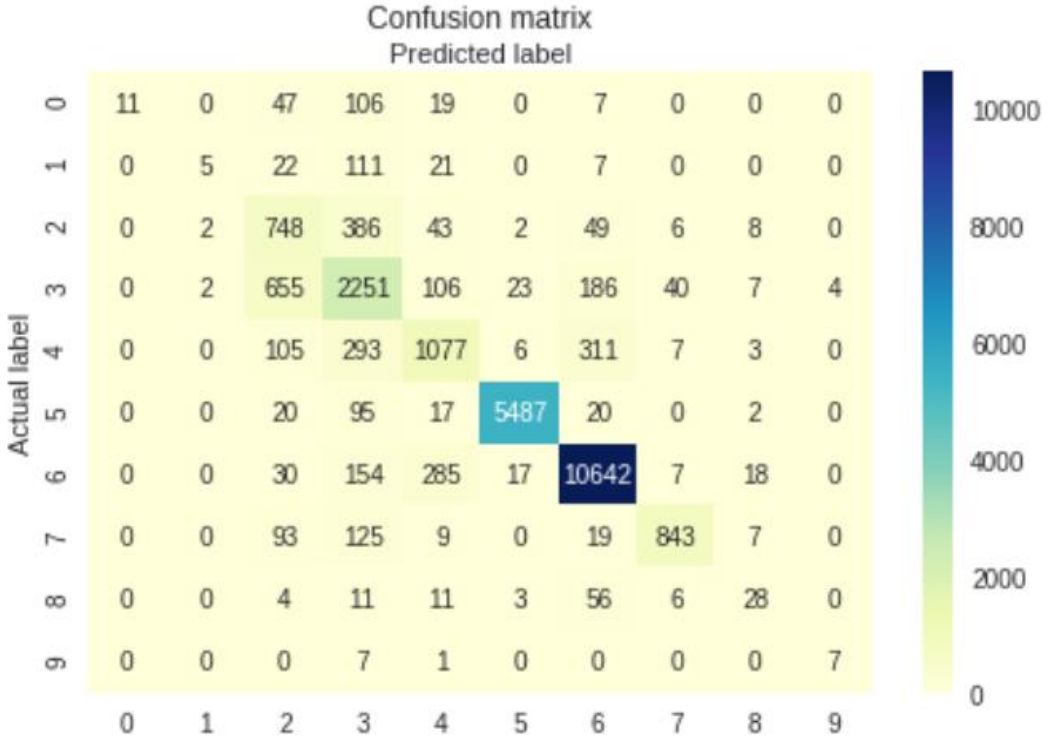

Fig 6.5 Confusion matrix

Above figure displays a confusion matrix created from the result of the Stacking classifier. Out of a total of 24700 samples, 21009 samples were correctly classified while 3691 samples are incorrectly classified.

	precision	recall	f1-score	support
Analysis	1.00	0.06	0.11	190
Backdoor	0.56	0.03	0.06	166
DoS	0.43	0.60	0.50	1244
Exploits	0.64	0.69	0.66	3274
Fuzzers	0.68	0.60	0.64	1802
Generic	0.99	0.97	0.98	5641
Normal	0.94	0.95	0.95	11153
Reconnaissance	0.93	0.77	0.84	1096
Shellcode	0.38	0.24	0.29	119
Worms	0.64	0.47	0.54	15
accuracy			0.85	24700
Macro avg	0.72	0.54	0.56	24700
Weighted avg	0.86	0.85	0.85	24700

Table 6.4

Stacking Classifier: Accuracy: 85.42105263157895

6.7 Result Discussion

	XGBoost	K-NN	Logistic Regression	Stacking
Accuracy	83.54	84	63.54	85.42
F1-score	0.83	0.84	0.52	0.85
Precision	0.83	0.84	0.52	0.86
Recall	0.84	0.84	0.64	0.85

Table 5.5

Among all the algorithms used Logistic regression has the worst accuracy. The Accuracy obtained by the hybrid algorithm (Stacking classifier) is about 86 % approximately which is greater all the accuracy we have achieved for other algorithms used. The F1-score obtained by the Stacking classifier is also high which is about 0.85 thus showing a high correlation between precision and recall values.

7. Conclusion and Future Work

7.1 Conclusion

For this thesis, we have used the dataset of size about 82,332. Out of it, 57632 has been used for training the algorithms while 24700 is being used for testing purposes. Four machine learning models were built XGBoost classifier, K-NN classifier, Logistic regression classifier, Stacking classifier.

The Stacking classifier which is a hybrid algorithm produced the best results as per the prediction made by this thesis in the beginning as compared to other general classifiers.

Hence, from the implementation of this thesis, we can conclude that a hybrid algorithm can produce a boost in accuracy when compared with general machine learning algorithms.

Furthermore, after using the stacking algorithm, we found out that the algorithm predicts with more than 80 percent accuracy for Generic, Normal and Reconnaissance attacks. Backdoor, Exploits, Fuzzers, and Worms have the accuracy rate between 50 and 80. Whereas DoS and shellcode have precision of less than 50 percent. The main reason of the differences in the accuracy is there are more number of records or more data for the NORMAL type and similarly the amount of data for Backdoor & Fuzzer is relatively less, with which the system is trained.

7.2 Future Work

The overall accuracy obtained is still not very great. Some different combinations of algorithms can be used with a different type of Hybrid approach like bagging or boosting to improve the overall accuracy. We can use different kind of dataset for more accuracy.

Furthermore, to improve the accuracy of particular kind of attacks, we need dataset which has appropriate features for the same.

The model developed in this thesis can be transformed into a proper software that can run live with the system to OS to continuously look for any APT intrusion. A mobile version of the software can be developed for the protection of android devices.

8. References

[1] Al-Yaseen, W. L., Othman , Z. A. & Nazri, M. Z. Z., 2017. "Multi-Level Hybrid Support Vector Machine and Extreme Learning Machine Based on Modified K-means for Intrusion Detection System. *ELSEVIER.*

[2] Anon., 2018. *Java POint.* [Online]
Available at: https://www.javAPToint.com/k-nearest-neighbor-algorithm-for-machine-learning

[3] Anon., 2018. *Python | Introduction to MAPTlotlib.* [Online]
Available at: https://www.geeksforgeeks.org/python-introduction-mAPTlotlib/

[4] Anon., n.d. *NumPy Introduction.* [Online]
Available at: https://www.w3schools.com/python/numpy_intro.asp

[5] Asaithambi, S., 2018. *Why, How and When to apply Feature Selection.* [Online]
Available at: https://towardsdatascience.com/why-how-and-when-to-apply-feature-selection-e9c69adfabf2

[6] Brogi, G. & Tong, V., 2016. Terminaptor: Highlighting advanced persistent threats through information flow tracking, in: New Technologies, Mobility and Security. *IEEE.*

[7] Buczak, A. L. & Guven, E., 2016. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE.*

[8] Chandra, J. V., Challa, N. & Pasupuleti, S. K., 2016. A Practical Approach to E-mail Spam Filters to Protect Data from Advanced Persistent Threat. *IEEE.*

[9] Chen, T. & Guestrin, C., 2016. XGBoost: A Scalable Tree Boosting System. *SIGKDD.*

[10] Donges, N., 2019. *The Logistic Regression Algorithm.* [Online]
Available at: https://www.experfy.com/blog/the-logistic-regression-algorithm/

[11] Dutta, A., 2019. *Stacking in Machine Learning.* [Online]
Available at: https://www.geeksforgeeks.org/stacking-in-machine-learning/

[12] Edwards, G., 2018. *Machine Learning | An Introduction.* [Online]
Available at: https://towardsdatascience.com/machine-learning-an-introduction-23b84d51e6d0

[13] Fernandes, L., 2013. *Targeted attacks and how to defend against them,* s.l.: Quocirca Ltd.

[14] Fumo, D., 2017. *Pandas Library in a Nutshell — Intro To Machine Learning.* [Online]
Available at: https://medium.com/simple-ai/pandas-library-in-a-nutshell-intro-to-machine-learning-3-acbd39ec5c9c

[15] Ghafir, I., Hammoudeh, M. & Prenosil, V., 2018. Detection of advanced persistent threat using machine-learning correlation analysis. *ELSEVIER.*

[16] Govindarajan, M. & Chandrasekaran, R., 2009. Intrusion Detection Using k-Nearest Neighbor. *IEEE.*

[17] Halimaa, A. & Sundarakantham, K., 2019. MACHINE LEARNING BASED INTRUSION DETECTION SYSTEM. *IEEE.*

[18] Harrison, O., 2018. *Machine Learning Basics with the K-Nearest Neighbors Algorithm.* [Online]
Available at: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

[19] JAIN, K., 2015. *Analytics VIdya.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2015/01/scikit-learn-python-machine-learning-tool/

[20] Matsuda, W., Fujimoto, M. & Mitsunaga, T., 2018. *Detecting APT attacks against Active Directory using Machine Leaning.* [Online]
Available at: https://scihub.wikicn.top/https://ieeexplore.ieee.org/document/8631486

[21] Morde, V., 2019. *XGBoost Algorithm: Long May She Reign!.* [Online]
Available at: https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

[22] Morgan, S., 2016. *Hackerpocalypse: A Cybercrime Revelation, Cybercrime Report,Cybersecurity Ventures,* s.l.: s.n.

[23] Muhajir, I., 2019. *K-Neighbors Regression Analysis in Python.* [Online]
Available at: https://medium.com/analytics-vidhya/k-neighbors-regression-analysis-in-python-61532d56d8e4

[24] Ng, A., 2020. *Coursera.* [Online]
Available at: https://www.coursera.org/learn/machine-learning

[25] Riggio, C., 2019. *Towards data science.* [Online]
Available at: https://towardsdatascience.com/whats-the-deal-with-accuracy-precision-recall-and-f1-f5d8b4db1021

[26] Seif, G., 2019. *A Beginner's guide to XGBoost.* [Online]
Available at: https://towardsdatascience.com/a-beginners-guide-to-xgboost-87f5d4c30ed7

[27] Sharma, A., 2020. *Confusion Matrix in Machine Learning.* [Online]
Available at: https://www.geeksforgeeks.org/confusion-matrix-machine-learning/

[28] Sigholm, J. & Bang, M., 2013. Towards Offensive Cyber Counterintelligence. *IEEE.*

[29] Swaminathan, S., 2018. *Logistic Regression — Detailed Overview.* [Online]
Available at: https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc

[30] Wu, L., Ping, R., Ke, L. & Hai-xin, D., 2011. Behavior-based Malware Analysis and Detection. *IEEE.*

[31] Yiu, T., 2019. *Understanding Random Forest.* [Online]
Available at: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[32] Yiu, T., 2019. *Understanding Random Forest.* [Online]
Available at: https://towardsdatascience.com/understanding-random-forest-58381e0602d2