VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Machhe, Belagavi, Karnataka 590018

A Project Report
on
"Anomaly detection in wi-fi network using machine learning"

Submitted in partial fulfillment of the requirement
for the award of the degree of

Bachelor of Engineering
in
Information Science & Engineering
by

Bhumika J (1BG17IS009)
K Vinuth babu (1BG17IS020)

Under the Guidance Of

Dr. Shashikala
Prof. & Head, Dept. of ISE
B.N.M.I.T

Vidyayāmruthamashnuthe

B.N.M. Institute of Technology

Approved by AICTE, Affiliated to VTU, Accredited as grade A institution by NAAC.
All UG branches –CSE, ECE, EEE, ISE & Mech.E accrediated by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021

Post box no. 7087, 27th cross, 12th Main, Banashankari II Stage, Bengaluru- 560070, INDIA
Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

Department of Information Science and Engineering
2020 – 2021

B.N.M. Institute of Technology

Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.
All UG branches – CSE, ECE, EEE, ISE &Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021
Post box no. 7087, 27th cross, 12th Main, Banashankari 2nd Stage, Bengaluru- 560070, INDIA
Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Vidyayāmruthamashnuthe

CERTIFICATE

Certified that the project work entiled **Anomaly detection in Wi-Fi network using machine learning** is carried out by **Bhumika J(1BG17IS009),K Vinuth babu(1BG17IS020)** the bonafide student of **B.N.M Institute of Technology** in partial fulfillment for the award of **Bachelor of Engineering** in **Information Science & Engineering** of the **Visvesvaraya Technological University**, Belagavi during the year 2020-2021. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated in the report. The Project Report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Dr.Shashikala
Prof.& Head,
Dept. of ISE
BNMIT

Dr. Krishnamurthy G N
Principal,
BNMIT

Name of the Examiners **Signature with date**

1.

2.

ACKNOWLEDGEMENT

I consider it a privilege to express through the pages of this report, a few words of gratitude to all those distinguished personalities who guided and inspired me in the completion of this project.

I would like to thank **Shri. Narayan Rao R Maanay**, Secretary, BNMEI, Bengaluru for providing excellent academic environment in college.

I would like to thank **Prof. T.J. Rama Murthy**, Director, BNMIT, Bengaluru for having extended his support and encouragement during the course of work.

I would like to express my gratitude to **Prof. Eishwar N Maanay**, Dean, BNMIT, Bengaluru for his relentless support, guidance and encouragement.

I would like to thank **Dr. Krishnamurthy G.N.**, Principal, BNMIT, Bengaluru for his constant encouragement.

I would like to thank **Dr. Shashikala**, Professor and Head of the department of Information Science and Engineering, BNMIT, Bengaluru for her support and encouragement towards the completion of the project work.

I would like to express my gratitude to my guide **Dr. Shashikala**, Professor and Head of the department of Information Science and Engineering, BNMIT, Bengaluru who has given me all the support and guidance in completing the project work successfully.

I would like to thank project coordinator **Mr. Manjunath G.S.,** Assistant Professor, department of Information Science and Engineering, BNMIT, for being the guiding force towards successful completion of the project.

Bhumika J

K Vinuth Babu

Table of Contents

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

Due to advancements in Internet technologies and the concomitant rise in the number of network attacks, network intrusion detection has become a significant research issue. In spite of remarkable progress and a large body of work, there are still many opportunities to advance the state of-the-art in detecting and thwarting network-based attacks.

An intrusion attempts or a threat is a deliberate and unauthorized attempt to

(i) access information,

(ii) manipulate information,

(iii) render a system unreliable or unusable.

For example,

- Denial of Service (DoS) attack attempts to starve a host of its resources, which are needed to function correctly during processing;
- Worms and viruses exploit other hosts through the network;
- Compromises obtain privileged access to a host by taking advantages of known vulnerabilities.

The term anomaly-based intrusion detection in networks refers to the problem of finding exceptional patterns in network traffic that do not conform to the expected normal behaviour.

Intrusion is a set of actions aimed to compromise the security of computer and network components in terms of confidentiality, integrity and availability. This can be done by an inside or outside agent to gain unauthorized entry and control of the security mechanism. To protect infrastructure of network systems, intrusion detection systems (IDSs) provide well-established mechanisms, which gather and analyse information from various areas within a host or a network to identify possible security breaches. Intrusion detection functions include (i) monitoring and analysing user, system, and network activities, (ii) configuring systems for generation of reports of possible vulnerabilities, (iii) assessing system and file integrity (iv) recognizing patterns of typical attacks (v) analysing abnormal activity, and (vi) tracking user policy violations. An IDS uses vulnerability assessment to assess the security of a host or a network. Intrusion detection works on the assumption that intrusion activities are noticeably different from normal system activities and thus detectable.

As human, our brains are always tuned to spotting something out of the "normal" or the "usual stuff." In short, some anomaly that does not fit with the usual pattern. With the abundant growth of data, data science tools are also looking for anomalies that do not subscribe to the normal data flow. For example, an "unusually high" number of login attempts may point to a potential cyberattack, or a major hike in credit card transactions in a short period could potentially be a credit card fraud. At the same time, detecting anomalies in the face of a continuous stream of unstructured data from various sources has its own challenges. An example of a challenge is to assume that a majority of credit card transactions are legitimate and proper while looking for major deviations in a few transactions that fall outside the "normal" range. Machine learning algorithms can be deployed to define data patterns that are normal and using ML models to find deviations or anomalies.

1.1 Motivation

Recently a number of proposals have been suggested to detect anomalous behaviour in network traffic using a wide variety of techniques such as SVM, MCA, FCM-ANN, etc. However, these techniques are inefficient because of their reduced accuracy and high false positive alarms.

1. Security and data integrity have been a necessity due to proliferation of information technology and expansion of the internet in everyday life.
2. Wi-fi has played a vital role in this expansion and dependency on internet for most of the communication.
3. Providing a safe and secure wi-fi network has been the number one priority for major telecommunication companies.
4. Intrusion detection systems (IDS) and machine learning techniques can be used to automate major steps.
5. Faster and more efficiently way of intrusion detection at a lower cost.

1.2 Problem Statement

Wireless propagation characteristics is less secure and more vulnerable to attack so the packets are easily intercepted when they are propagating from source address to destination address, there are four types of anomalies which are normal record, injection attack, impersonation attack, flooding attack and they make up the larger part of the network anomalies.

1.3 OBJECTIVES

The aim of the project:

- To provide a wi-fi defense system which will be able to detect anomalies more efficiently.
- Provide a user-friendly interaction with the network so the transmission can occur without any glitches.
- Provide a more secure and reliable connection between users.
- Prevention of loss of data packets during transmission.
- Periodically updating the reference dataset to keep the system updated.

CHAPTER 2

LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

2.1. INTRODUCTION

A literature survey in a project report represents the study done to assist in the completion of a project. A literature survey also describes a survey of the previous existing material on a topic of the report.

The focus of a literature survey is in the following order:

- Existing theories on a topic with universal acceptance across the board.
- Books on the subject acting as a reference for the concepts that project uses whether they are specific or generic.
- Current research concerning the field of the project from the oldest to latest. Research papers might be a reference for theories nut most cases require a critical comparison to establish the purpose of the project and improvement.
- You may also include another project report and what helped you.
- Challenges for the project and by ongoing work if it is available.

Literature surveys provide brief overviews or a summary of the current research on topics. The structure written requires to be in a way that it seemed logical. It needs to chronologically represent a development of the ideas in the field that is being researched. The length of a literature survey depends much on whether the purpose of the project report is to complete a college assignment or submitting for journal publication. It can review a few research papers on a topic or be a full-length discussion on the significant work in the field until that date.

2.2. PREVIOUS WORK DONE

[1]: As mentioned by the **Jing Ran, Yidong Ji, and Bihua Tang** in their paper **A Semi-Supervised Learning Approach to IEEE 802.11 Network Anomaly Detection.**

- With the development of communication technology and Internet technology, Wi-Fi technology has gained wide recognition with its own advantages of low cost and mobility.
- Wi-Fi network with wireless propagation characteristics is relatively less secure and more vulnerable to attack.

- Packets are easily intercepted and tampered when they are propagating from source address to destination address.
- In this paper, a deep learning approach based on ladder network which self-learns the features necessary to detect network anomalies and perform attack classification accurately was proposed.
- Using focal loss as a loss function to enhance the discriminative ability of the model to classify difficult samples.
- Experiments on Aegean Wi-Fi Intrusion Dataset (AWID) public data-set, the network records were classified into 4 types: normal record, injection attack, impersonation attack, flooding attack.
- The paper achieved the classification accuracies of these four types of records are 99.77%, 82.79%, 89.32%, 73.41% respectively, and achieved an overall accuracy of 98.54%.

[2]: As mentioned by **Sahil Garg, Kuljeet Kaur, Neeraj Kumar, Georges Kaddoum, Albert Y. Zomaya and Rajiv Ranjan** in their paper on **A Hybrid Deep Learning based Model for Anomaly Detection in Cloud Datacentre Networks**: A Case Study intrusion detection in a cloud datacentre network.

- With the emergence of the Internet-of-Things (IoT) and seamless Internet connectivity, the need to process streaming data on real-time basis has become essential.
- The existing data stream management systems are not efficient in analysing the network log big data for real-time anomaly detection.
- However, the existing anomaly detection approaches are not efficient because they cannot be applied to networks, are computationally complex, and suffer from high false positives. So in order to find an efficient solution in this paper a hybrid data processing model for network anomaly detection is proposed that leverages Grey Wolf Optimization (GWO) and Convolution Neural Network (CNN).
- To enhance the capabilities of the proposed model, GWO and CNN learning approaches were:
    (i) enhanced with improved exploration, exploitation and initial population generation abilities
    (ii) revamped dropout functionality. These extended variants are referred to as Improved-GWO (ImGWO) and Improved CNN (ImCNN).

The proposed model exhibits an overall improvement of 8.25%, 4.08% and 3.62% in terms of detection rate, false positives, and accuracy, respectively; relative to standard GWO with CNN.

[3]: As mentioned by **Vrizlynn L, L. Thing** in their paper **IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach.**

Despite the significant advancement in wireless technologies over the years, IEEE 802.11 still emerges as the de-facto standard to achieve the required short to medium range wireless device connectivity in anywhere from offices to homes. So, IEEE 802.11 security has thus become a key concern over the years.

In this paper, we analysed the threats and attacks targeting the IEEE 802.11 network and also identified the challenges of achieving accurate threat and attack classification.

- Especially in situations where the attacks are novel and have never been encountered by the detection and classification system before.
- So, they proposed a solution based on anomaly detection and classification using a deep learning approach. The deep learning approach self-learns the features necessary to detect network anomalies and is able to perform attack classification accurately.
- They have considered the classification as a multi-class problem and achieved an overall accuracy of 98.6688% in classifying the attacks through the proposed solution.

[4]: As mentioned by **Nguyen Thanh Van, Tran Ngoc Thinh, Le Thanh Sach** in his paper **An anomaly-based Network Intrusion Detection System using Deep learning**

Recently, anomaly-based intrusion detection techniques are valuable methodology to detect both known as well as unknown/new attacks, so they can cope with the diversity of the attacks and the constantly changing nature of network attacks

There are many problems need to be considered in anomaly-based network intrusion detection system (NIDS) such as:

- Ability to adapt to dynamic network environments.
- Unavailability of labeled data.
- False positive rate.

In this paper, they have used Deep learning techniques to implement an anomaly-based NIDS. These techniques show the sensitive power of generative models with good classification, capabilities to deduce part of its knowledge from incomplete data and the adaptability.

Experiments with KDDCup99 network traffic connections show that our work is effective to exact detect in anomaly-based NIDS and classify intrusions into five groups with the accuracy based on network data source.

[5]: As mentioned by **Zhaomin Chen, Chai Kiat Yeo, Bu Sung Lee, Chiew Tong LauJr** in their paper on **Autoencoder-based Network Anomaly Detection.**

An autoencoder-based network anomaly detection method is able to capture the non-linear correlations between features so as to increase the detection accuracy.

- They also apply the Convolutional Autoencoder (CAE) here to perform the dimensionality reduction.
- As the Convolutional Autoencoder has a smaller number of parameters, it requires less training time compared to the conventional Autoencoder.
- By evaluating on NSL-KDD dataset, CAE-based network anomaly detection method outperforms other detection methods.
- The evaluation results show that the Convolutional Autoencoder based anomaly detection method outperforms other detection methods.

[6]: As mentioned by **Uthara E. Prakash, Athira Thankappan, Vishnupriya K. T, Arun A. Balakrishnan** in their paper **IP Network Anomaly Detection using Machine Learning.**

This paper focuses on discovering viable anomalies that have a significant potential to be associated with abnormal network behavior.

- Three approaches based on ML (Machine Learning) have been proposed to detect suspicious network behavior.
- These methods are an extension of the techniques discussed as part of the introduction below. The developed approaches in the current paper are evaluated by testing their efficiency against a real-time network attack using available open-source network tools.
- The results of the experiment demonstrate successful identification of anomalous instances from the telemetry data with a low false alarm rate. Further, they believe that their approaches can be directly deployed in a real-time environment

The current paper uses three approaches which are an extension and enhancement of the approaches mentioned in different papers-

1. Multivariate Gaussian Probability Distribution
2. K-means
3. Supervised Long-Short Term Memory.

The paper presents approaches to detect and flag anomalous instances in IP Network traffic data. The future scope may include the possible exploration to enhance reducing the number of false alarms.

CHAPTER 3

SYSTEM REQUIREMENTS AND SPECIFICATIONS

CHAPTER 3

SYSTEM REQUIREMENT AND SPECIFICATION

System Requirements Specification (SRS) also known as Software Requirements Specification is a document or group of documents that outlines the characteristics and behaviour of a system or software application. A combination of issues and possibilities is usually required to motivate a new system. Non-functional requirements such as performance goals and descriptions of quality traits, as well as functional requirements, are documented in an SRS.

3.1 Hardware requirements

The hardware requirements are the requirements of a hardware device.
Most hardware only has operating system requirements or compatibility. The following requirements is required for the development or running the project.

- OS Name: Windows 10(64 bit)
- RAM: 8 GB
- Hard disk: Minimum 20 GB
- Processor: Intel® Core i5-6500 CPU@ 3.20 GHz

3.2 Software requirements

The software requirements for a system is the description of the software or application that are used for implementation of the project. The following are the software requirements:

- Jupyter Notebook 4.3.0(64 bit).
- Python 3.0
- Anaconda navigator.
- Scikit-learn

3.3 Functional requirements

A functional requirement document defines the functionality of a system or one of its subsystems. It also depends upon the type of software, expected users and the type of system where the software is used.
The proposed system has the following functional requirements:

- This code is for detecting network anomalies in wi-fi

- Intrusion detection systems are placed alongside firewalls in Wi-Fi network.
- The model is dependent on all the feature columns given in the dataset.

3.4 Non- Functional requirements

Non-functional requirements are the requirements that do not directly show the specific functions of the system. They may specify system performance and maintainability and security.

- The output should be more accurate and should have a low false positive rate.
- Detecting and identifying hackers will prevent any losses to the companies hosting the networks

3.5 Languages, Tools and Platform

Anaconda

Anaconda is a free and open-source distribution of the python and R programming languages for data science and machine learning related applications, that aims to simply package management and deployment. Package versions are managed by the package management system conda. The Anaconda distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux and MacOS.

Anaconda distribution comes with more than 1,000 data packages as well as the Conda package and virtual environment manager, called Anaconda Navigator, so it eliminates the need to learn to install each library independently.

The open-source data packages can be individually installed from the Anaconda repository with the conda install command or using the pip install command that is installed with Anaconda. Pip packages provide many of the features of conda packages and in most cases, they can work together.

Anaconda Navigator

Anaconda Navigator is a desktop GUI that comes with Anaconda Individual Edition. It makes it easy to launch applications and manage packages and environments without using command-line commands. It is popular because it brings many of the tools used in data science and machine learning also uses the concept of creating environments so as to isolate different libraries and versions. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS

and Linux. Navigator is automatically include with Anaconda version 4.0.0 or higher. The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glueviz
- Orange
- Rstudio
- Visual Studio Code

Jupyter notebook

Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document. It provides you with an easy-to-use, interactive data science environment across many programming languages that doesn't only work as an IDE, but also as a presentation or education tool. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ". ipynb" extension. jupyter Notebook provides a browser-based REPL built upon a number of popular open- source libraries:

- IPython
- Tornado
- jQuery
- Bootstrap (front-end framework)
- MathJax

Jupyter Notebook can connect to many kernels, to allow programming in many languages. As of the 2.3 release (October 2014), there are currently 49 Jupyter- compatible kernels for as many programming languages, including Python, R, Julia and Haskell. The Notebook interface was added to IPython in the 0.12 release (December 2011), renamed to Jupyter notebook in 2015. Jupyter Notebook is similar to the notebook interface of other programs such as Maple, Mathematica and SageMath.

Jupyter kernels

A Jupyter kernel is a program responsible for handling various types of request, and providing a reply. Kernels talks to the other components of Jupyter using ZeroMQ over the network, and thus can be on the same or remote machines. Usually Kernels are implemented and allow execution of a single language with a couple of exceptions.

Jupyter hub

Jupyter Hub is a multi-user server for Jupyter Notebooks. It is designed to support many users by spawning, managing, and proxying many singular Jupyter Notebook servers.

Jupyter lab

Jupyter Lab is the next-generation user interface for Project Jupyter. It offers all the familiar building blocks of the classic Jupyter Notebook in a flexible and powerful user interface.The first stable release was announced on February 20, 2018.

Python3

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. It supports automatic garbage collection and can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python 3.0 (also called "Python 3000" or "Py3K") was released on December 3, 2008. It was designed to rectify fundamental design flaws in the language—the changes required could not be implemented while retaining full backwards compatibility with the 2.x series, which necessitated a new major version number. The guiding principle of Python 3 was: "reduce feature duplication by removing old ways of doing things".

Python 3.0 was developed with the same philosophy as in prior versions. However, as Python had accumulated new and redundant ways to program the same task, Python 3.0 had an emphasis on removing duplicative constructs and modules, in keeping with "There should be one— and preferably only one —obvious way to do it".

Nonetheless, Python 3.0 remained a multi-paradigm language. Coders still had options among object-orientation, structured programming, functional programming and other paradigms, but within such broad choices, the details were intended to be more obvious in Python 3.0 than they were in Python 2.x.

Python 3, projects requiring compatibility with both the 2.x and 3.x series were recommended to have one source (for the 2.x series), and produce releases for the Python

3.x platform using 2to3. Edits to the Python 3.x code were discouraged for so long as the code needed to run on Python 2.x. This is no longer recommended as of 2012 the preferred approach is to create a single code base that can run under both Python 2 and 3 using compatibility modules.

Features

Some of the major changes included for Python 3.0 were:

- Changing print so that it is a built-in function, not a statement. This made it easier to change a module to use a different print function, as well as making the syntax more regular. In Python 2.6 and 2.7 print () is available as a built-in but is masked by the print statement syntax, which can be disabled by entering from future import print function at the top of the file.
- Removal of the Python 2 input function, and the renaming of the raw input function to input. Python 3's input function behaves like Python 2's raw input function, in that the input is always returned as a string rather than being evaluated as an expression.
- Adding support for optional function annotations that can be used for informal type declarations or other purposes.
- Unifying the str Unicode types, representing text, and introducing a separate immutable byte's type and a mostly corresponding mutable byte array type, both of which represent arrays of bytes.
- Removing backward-compatibility features, including old-style classes, string exceptions and implicit relative imports.
- A change in integer division functionality: in Python 2, 5/2 is 2; in Python 3, 5/2 is 2.5. (In both Python 2 (2.2 onwards) and Python 3, 5//2 is 2).

SciKit-learn

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k- means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits-. learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy. The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa,

Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from INRIA took leadership of the project and made the first public release on February the 1st 2010. Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November2012.

Numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. By using them in Python programming, they can be used with two simple commands:

```
>>> import numpy
```

The core functionality of NumPy is its "ndarray", for n-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure (which, despite the name, is a dynamic array), these arrays are homogeneously typed: all elements of a single array must be of the same type.

Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

Library features are:

- DataFrame object for data manipulation with integrated indexing.
- Toolsfor reading andwriting data between in-memory datastructures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.
- Data structure column insertion and deletion.

- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional
- Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging.

Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of matplotlib.

Matplotlib was originally written by John D. Hunter, has an active development community, and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012, and further joined by Thomas Caswell.

As of 23 June 2017, matplotlib 2.0.x supports Python versions 2.7 through 3.6. Matplotlib 1.2 is the first version of matplotlib to support Python 3.x. Matplotlib 1.4 is the last version of matplotlib to support Python 2.6

CHAPTER 4

SYSTEM DESIGN

CHAPTER 4

SYSTEM DESIGN

4.1 Data flow diagram:

Data flow diagram is a graphical representation of the flow of data through an information system or a model. DFD is often used as preliminary step to create an overview of the system.

DFD 0:

DFD 0 is a general overview of all the operations required and the detailed description of the individual process is given separately.

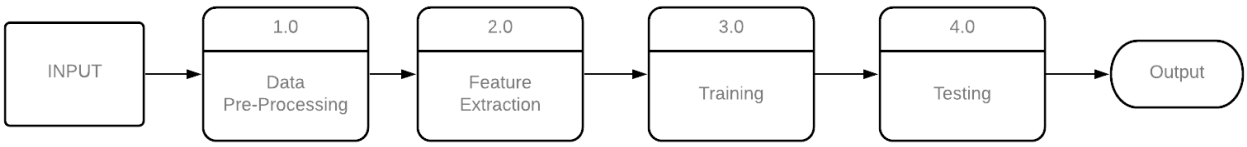

Fig 4.1 DFD 0

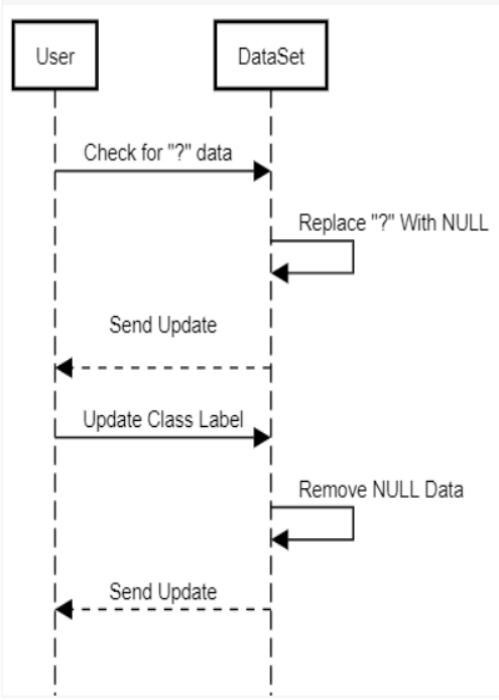

Fig 4.2 Data pre-processing sequence diagram.

Data Pre-processing of the AWID dataset include checking for attributes with “?” value and replacing them with NULL values, “?” values are removed because attributes with “?” values don’t make any relevance in the dataset and “?” values cannot be used in the further process because algorithms only take integer value.

The NULL value is used because Pandas has an inbuilt function isNull() which is used to check for null values and dropna() function can be used to drop Null values, In the data pre-processing process Null values are also removed because they cannot be used in encoding and training process has algorithms cannot take Null as an input.

The feature extraction process, In this pre-processed training dataset from the data pre-processing step, is used and using Random forest algorithm important features are extracted from the dataset and these important features are used for training of the model. The importance of the particular feature is calculated by considering the effect of the particular feature on the final classification.

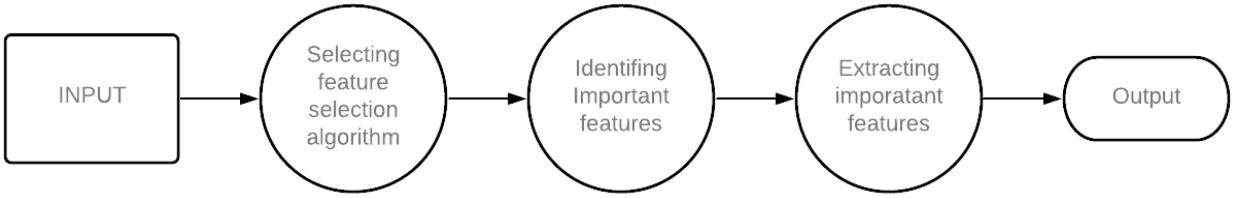

Fig.4.3. DFD 2.1

The training process of the model using ML algorithm and the training model uses the features extracted from the feature extraction process and the obtained results after the training process are graphically represented as output. The machine learning algorithms used are decision tree, Naive Bayes and Logistic regression and supervised learning is used for training the algorithms as the AWID dataset is a well-defined dataset.

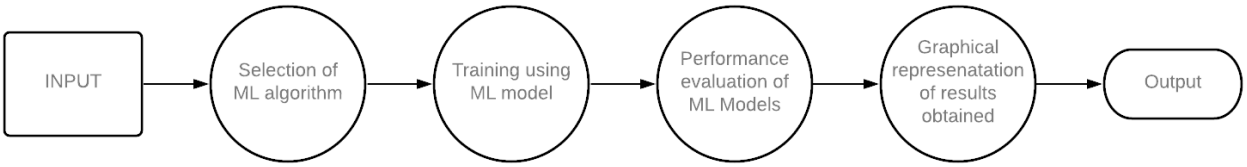

Fig 4.4. DFD 3.1

Testing process of the model, in the testing phase the model is tested using AWID-CLS-R-Tst dataset which is a well defined AWID test dataset and the algorithms used for testing the dataset are

decision tree, Naive Bayes and Logistic regression and the results obtained are cross-validated with the training results and the final result is visualized.

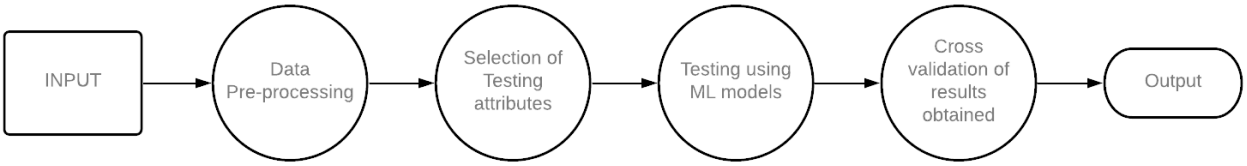

Fig 4.5. DFD 4.1

CHAPTER 5

IMPLEMENTATION

CHAPTER 5

IMPLEMENTATION

5.1 Implementation of the model

The network anomaly detection process can be divided into five parts.

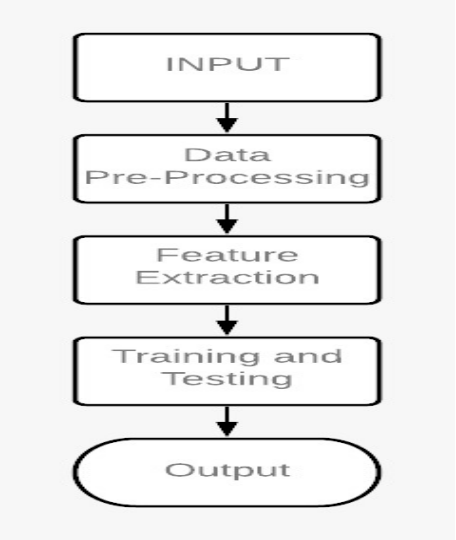

Fig 5.1 Flow Diagram

1. Input

It consists of selecting the input dataset which is required for the given problem statement. We are considering Aegean Wi-Fi Intrusion Dataset (AWID)-Dataset. Kolias et al produced AWID, a publicly available Wi-Fi intrusion dataset, using Wi-Fi network frames of normal and intrusion traffic. We are using AWID-CLS-R version in our work, a reduced dataset in terms of size and number of records. AWID-CLS-R includes separate sets for training (AWID-CLS-R-Trn) and test (AWID-CLS-R-Tst) purposes.

There are four major classes in the dataset:

1. Impersonation- An impersonation attack occurs when a hacker successfully assumes the identity of one of the legitimate parties in a system or communication channel. The purpose of a strong identification or entity authentication procedure is to reduce the likelihood that any third party would have access instead of the legitimate user.
2. Injection- Injection attacks refer to a broad class of attack vectors. In an injection attack, an attacker supplies untrusted input to a program. This input gets processed by an interpreter as part of a command or query. In turn, this alters the execution of that program. Injections are amongst the oldest and most dangerous attacks aimed at web

applications. They can lead to data theft, data loss, loss of data integrity, denial of service, as well as full system compromise. The primary reason for injection vulnerabilities is usually insufficient user input validation

3. Flooding- Flood attacks are also known as Denial of Service (DoS) attacks. In a flood attack, attackers send a very high volume of traffic to a system so that it cannot examine and allow permitted network traffic. For example, an ICMP flood attack occurs when a system receives too many ICMP ping commands and must use all its resources to send reply commands.
4. Normal- The Normal class consists of normal day-to-day general traffic among the devices, and majority of data frames belong to normal class.

The first three classes belong to intrusion attacks and the last one represents normal traffic. The dataset contains a total of 1,795,575 records for training the data. Each record in the dataset represents a Wi-Fi network frame and contains 154 features extracted from the frame.

2. Pre-processing of data

Data pre-processing is an important step to prepare the data. Data pre-processing is a required first step before any machine learning machinery can be applied, because the algorithms learn from the data and the learning outcome heavily depends on the data needed to solve a particular problem. Features can be anything from a timestamp and a number to hexadecimal digits and characters. With such a diverse number of features, it is necessary to prepare the dataset by analysing, transforming, and removing specific aspects in order to make it acceptable and compact for ML model construction. There are features in the dataset that have the same values for all entries, thus we deleted them. Furthermore, numerous features in the training dataset had missing values in more than 50% of the records; these features have been eliminated as well. We are also checking the data type values present in that data if there are any categorical or string value present, we are converting it into integer type.

3. Feature Selection

In feature selection, features that contribute significantly in output prediction are identified and selected. It reduces the number of features, model over-fitting, and processing time along with the improvement in model accuracy. Random forest classifier algorithm is used for feature selection

Feature selection using Random Forest comes under the category of Embedded methods. Embedded methods combine the qualities of filter and wrapper methods. They are

implemented by algorithms that have their own built-in feature selection methods. Some of the benefits of embedded methods are:

- They are highly accurate.
- They generalize better.
- They are interpretable.

Working of random forest classifier

Random forests are made up of 4–12 hundred decision trees, each of which is made up of a random extraction of the dataset's observations and a random extraction of the features. Because not every tree sees all of the features or all of the data, the trees are de-correlated and hence less prone to over-fitting. Each tree is also a series of yes-or-no questions based on one or more attributes. The tree separates the dataset into two buckets at each node (this is at each question), each of which contains observations that are more similar among themselves and dissimilar from those in the other bucket. As a result, the value of each feature is determined by how "pure" each bucket is.

Advantages of Random Forest

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

4. Training the model

A training model is a set of data that is used to train a machine learning algorithm. The training model is used to process the input data via the algorithm in order to compare the processed output to the sample output. The model is modified based on the results of this association. Model fitting is the term for this iterative procedure. After feature selection we are training the data using different ML models

a. Decision tree
b. Logistic regression
c. Naïve bayes

a. Decision tree-

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.

It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees
- Below diagram explains the general structure of a decision tree:

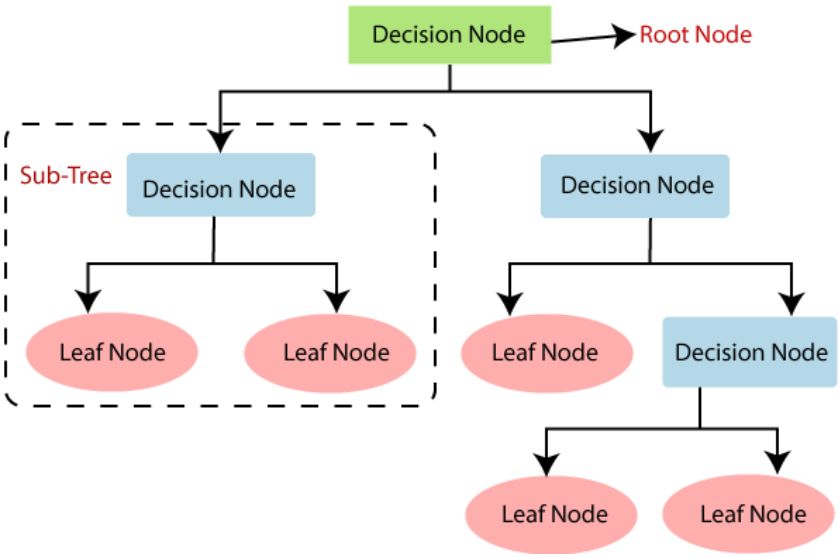

Fig 5.2 Decision Tree

Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- Branch/Sub Tree: A tree formed by splitting the tree.
- Pruning: Pruning is the process of removing the unwanted branches from the tree.
- Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree.This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree.

Input- x_train and y_train are two vectors . x_train is the train dataset and y_train is final classification

Output-It predicts the accuracy of each class.

- Step-1: Begin the tree with the root node, says S, which contains the complete dataset.
- Step-2: Divide the S into subsets that contains possible values for the best attributes.
- Step-3: Generate the decision tree node, which contains the best attribute.
- Step-4: Recursively make new decision trees using the subsets of the dataset created in step Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.

- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

A decision tree is a decision-making tool that employs a tree-like model of decisions and their potential outcomes, such as chance event outcomes, resource costs, and utility. It's one technique to show a conditional control algorithm.

b. Naïve Bayes

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in *text classification* that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of colour, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem.

- Bayes' Theorem:
- Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A|B)= \frac{P(B|A)P(A)}{P(B)}$$

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Input- x_train and y_train are two vectors . x_train is the train dataset and y_train is final classification

Output-It predicts the accuracy of each class.

Step1- Converts the dataset into a frequency table.

Step2-Creates a likelihood table for each class.

Step3- Use naïve bayes equation to calculate the probability of each class.

Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for text classification problems.

Disadvantages of Naïve Bayes Classifier:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

c. Logistic Regression

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1,

true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:

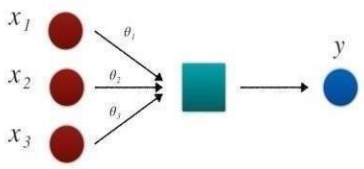

Fig 5.3 Logistic Regression

Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Input- x_train and y_train are two vectors . x_train is the train dataset and y_train is final classification

Output-It predicts the accuracy of each class.

Step1- Converts the dataset into a frequency table.

Step2-Check probability of each class.

Step3- Calculate the accuracy of the model.

Advantages of Logistic regression

- Logistic regression is easier to implement, interpret, and very efficient to train.
- It makes no assumptions about distributions of classes in feature space.
- It is very fast at classifying unknown records.

Disadvantages of Logistic regression

- If the number of observations is lesser than the number of features, Logistic Regression should not be used, otherwise, it may lead to overfitting.
- The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables.

5. Output - After obtaining the results after from training the ML model the accuracy of training of the model is used to measure the efficiency of the ML model. Higher accuracy of the model after testing will determine the higher efficiency and effectiveness of the ML models.

The accuracy is obtained using the confusion matrix. Let's us understand few terms that are used in obtaining the accuracy and confusion matrix.

5.2 Definition of terms:

Confusion matrix

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which your classification model is confused when it makes predictions. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

Here,

Class 1: True

Class 2: False

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

Fig 5.4: Confusion Matrix

- Positive(P): Observation is positive.
- Negative(N): Observation is not positive.
- True Positive (TP): Observation is positive, and is predicted to be positive.
- True Negative (TN): Observation is negative, and is predicted to be negative.
- False Positive (FP): Observation is negative, but is predicted positive.
- False Negative (FN): Observation is positive, but is predicted negative.

Accuracy

Accuracy is given by the relation

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad \text{(1)}$$

However, there are problems with accuracy. It assumes equal costs for both kinds of errors. 99% accuracy can be excellent, good, mediocre, poor or terrible depending upon the problem.

Recall

Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples. High recall means that an algorithm returned most of the relevant results.

Recall is given by the relation:

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

Precision

To get the value of precision, divide the total number of correctly classified positive examples by the total number of predicted positive examples. High Precision indicates an example labeled as positive is indeed positive (small number of FP).

Precision is given by the relation:

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

Comparative Study of Algorithms

After the comparative study of algorithms, have found that Decision Tree algorithm has more accuracy compared to other algorithms.

Table 5.1: Comparison of Algorithms

Model	Accuracy
Decision tree	99.99%
Naïve Bayes	95.97%
Logistic Regression	87.54%

Finding the accuracy for each algorithms for each iterations from starting

The basic approach, called k-fold CV, the training set is split into k smaller. The following procedure is followed for each of the k "folds". We are comparing the algorithms, starting from 10 iteration to 50 iterations and plotting the graph for each algorithms.

CHAPTER 6

TESTING

CHAPTER 6

TESTING

Software testing is conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. It involves the execution of a software component or system component to evaluate one or more properties of interest (a program or application), with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

6.1 Introduction

A system based on machine learning is primarily been associated with building models that could do numerical or class-related predictions. This is unlike conventional software development, which is associated with both development and "testing" the software. The usage of the word "testing " in relation to Machine Learning models is primarily used for testing the model performance in terms of accuracy/precision of the model. It can be noted that the word, "testing" means different for conventional software development and Machine Learning models development.

Machine Learning models would also need to be tested as conventional software development from the quality assurance perspective. Techniques such as Blackbox and white box testing would, thus, apply to Machine Learning models as well for performing quality control checks on Machine Learning models.

6.2 What is Black Box Testing?

Blackbox testing is testing the functionality of an application without knowing the details of its implementation including internal program structure, data structures, etc. Test cases for Blackbox testing are created based on the requirement specifications. Therefore, it is also called as specification-based testing. The following figure 6.1 represents the Blackbox testing.

When applied to Machine Learning models, Blackbox testing would mean testing Machine Learning models without knowing the internal details such as features of the Machine Learning model, and the algorithm used to create the model etc. These challenges however, is to identify the test oracle which could verify the test outcome

against the expected values known beforehand. This is discussed in the following section.

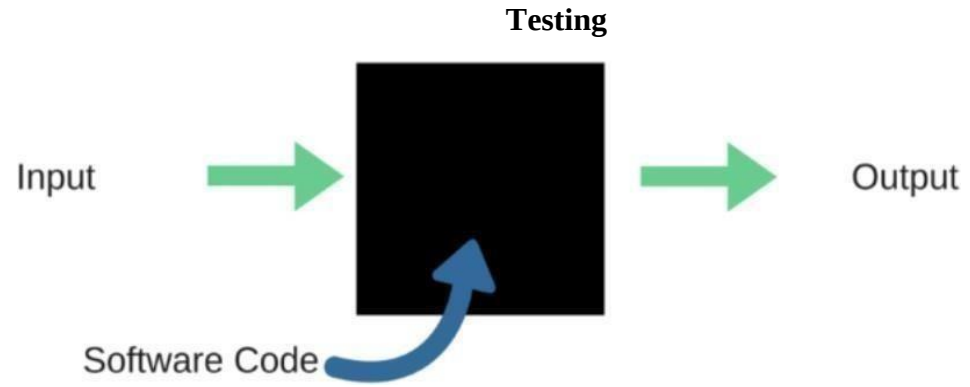

Figure 6.1: Blackbox testing

Normally in a machine learning process, data is divided into training and test sets; the training set is then used to train the model and the test set is used to evaluate the performance of a model. However, this approach may lead to variance problems. In simpler words, a variance problem refers to the scenario where our accuracy obtained on one test is very different to accuracy obtained on another test set using the same algorithm.

Steps involved in testing

1. Input the Testing data
2. Testing the data
3. Calculate the error rate

1. Input the Testing Data: The AWID-CLS-R-Tst data is loaded into the program. It consists of different attributes along with null values and categorical data. We are using dropping null values from the dataset and using LabelEncoder() to convert the categorical data into numerical data .

2. Testing the data

A test model is a set of data that is used to test a machine learning algorithm. The training model is used to process the input data via the algorithm in order to compare the processed output to the sample output. The model is modified based on the results of this association. We are using the same algorithms to test the model they Decision tree, Logistic regression and Naïve bayes. The accuracy and confusion matrix are calculated for each model.

3. **Calculate the Error rate:** Error rate (ERR) is calculated as the number of all incorrect predictions divided by the total number of the dataset. The best error rate is 0.0, whereas the worst is 1.0.

$$ERR = \frac{FP + FN}{TP + TN + FN + FP} = \frac{FP + FN}{P + N}$$

Table 6.1 Error rate for each algorithm

Algorithms	**Error rate**
Decision Tree	0.0
Naïve Bayes	0.0291
Logistic Regression	0.2888

CHAPTER 7

RESULTS AND DISCUSSION

CHAPTER 7

RESULTS AND DISCUSSION

This chapter discusses the results obtained by implementing different modules of the proposed system. It contains the snapshots of the results obtained from different modules of the system.

7.1 Snapshots

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib
        import seaborn as sns
        import sklearn
        from sklearn.preprocessing import LabelEncoder
        from sklearn.metrics import accuracy_score, classification_report
        from sklearn.linear_model import LogisticRegression
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.tree import DecisionTreeClassifier
        import matplotlib.pyplot as plt
        import warnings
        warnings.filterwarnings('ignore')

        print("pandas : {0}".format(pd.__version__))
        print("numpy : {0}".format(np.__version__))
        print("matplotlib : {0}".format(matplotlib.__version__))
        print("seaborn : {0}".format(sns.__version__))
        print("sklearn : {0}".format(sklearn.__version__))

pandas : 1.0.5
numpy : 1.18.5
matplotlib : 3.2.2
seaborn : 0.10.1
sklearn : 0.24.2
```

Fig 7.1 Setting environment variables

Fig 7.1 represents setting up the required environments variables which are required for the program.

```
awid.head()
```

	frame.interface_id	frame.dlt	frame.offset_shift	frame.time_epoch	frame.time_delta	frame.time_delta_displayed	frame.time_relative	frame.len	frame.cap_len	f
0	0	?	0.0	1.393661e+09	0.000000	0.000000	0.000000	261	261	
1	0	?	0.0	1.393661e+09	0.024271	0.024271	0.024271	185	185	
2	0	?	0.0	1.393661e+09	0.001631	0.001631	0.025902	185	185	
3	0	?	0.0	1.393661e+09	0.055325	0.055325	0.081227	159	159	
4	0	?	0.0	1.393661e+09	0.000415	0.000415	0.081642	54	54	

5 rows × 155 columns

Fig 7.2 Loading AWID- Train dataset

Fig 7.2 represents the loading of AWID-train dataset. The different attributes along with their values is shown in this table.

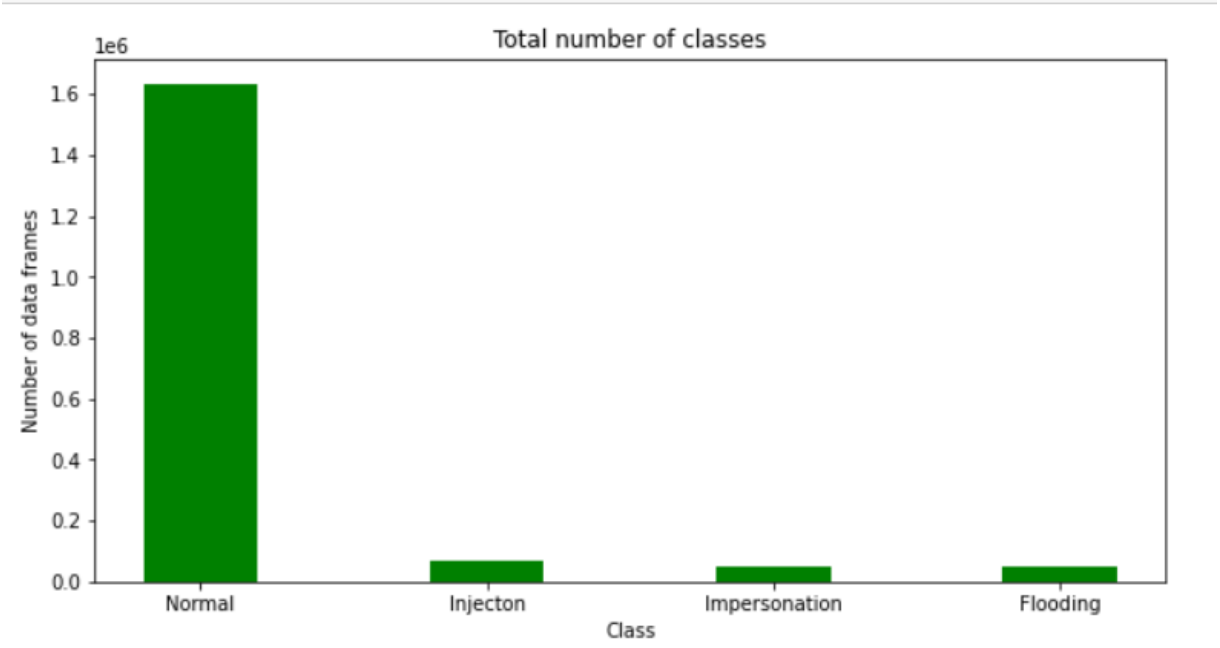

Fig7.3 Number of attacks in dataset

Fig 7.3 represent the different types of attacks in the dataset like normal attack, injection attack, impersonation attack and flooding attack.

```
awid.drop(null_column, axis=1, inplace=True)
awid.dropna(inplace=True)
```

```
awid.shape
```

```
(885744, 85)
```

```
awid.head()
```

	frame.interface_id	frame.offset_shift	frame.time_epoch	frame.time_delta	frame.time_delta_displayed	frame.time_relative	frame.len	frame.cap_len	frame.marl
17	0	0.0	1.393661e+09	0.002270	0.002270	0.217518	1524	1524	
18	0	0.0	1.393661e+09	0.000621	0.000621	0.218139	124	124	
20	0	0.0	1.393661e+09	0.000144	0.000144	0.229118	153	153	
28	0	0.0	1.393661e+09	0.021711	0.021711	0.367359	124	124	
31	0	0.0	1.393661e+09	0.000058	0.000058	0.380933	124	124	

5 rows × 85 columns

Fig 7.4 Dropping Null values in dataset

Fig7.4 shows if there are any values present in the dataset, they are automatically dropped from the dataset. Null Values are nothing but flooding attack. Therefore all the Flooding attacks are dropped from the dataset

.

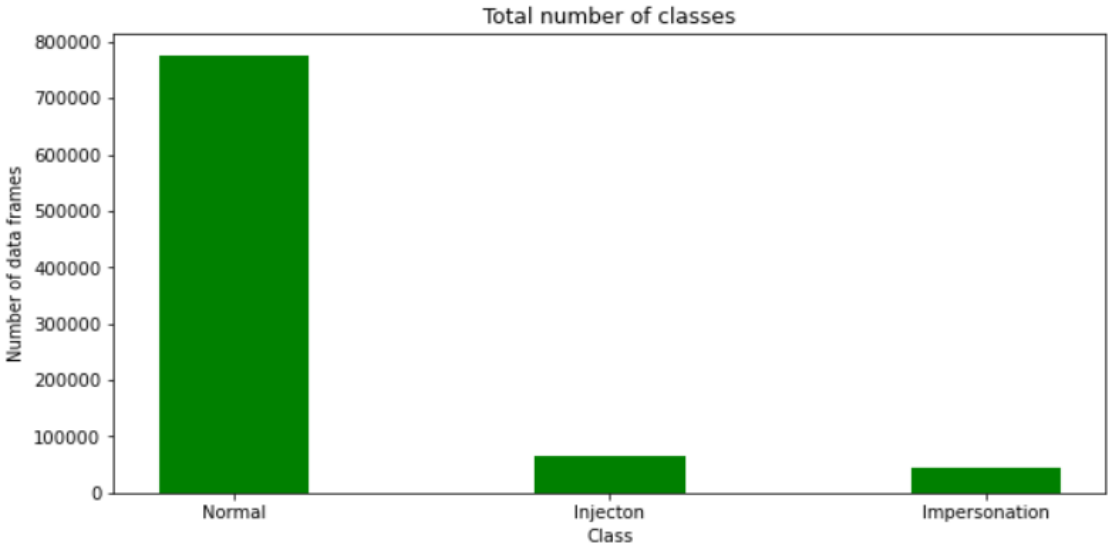

Fig 7.5 No of attacks in dataset after dropping null values

Fig 7.5 represents the no of attacks in the dataset which is flooding attack has been removed from dataset has it contained a lot of null values.

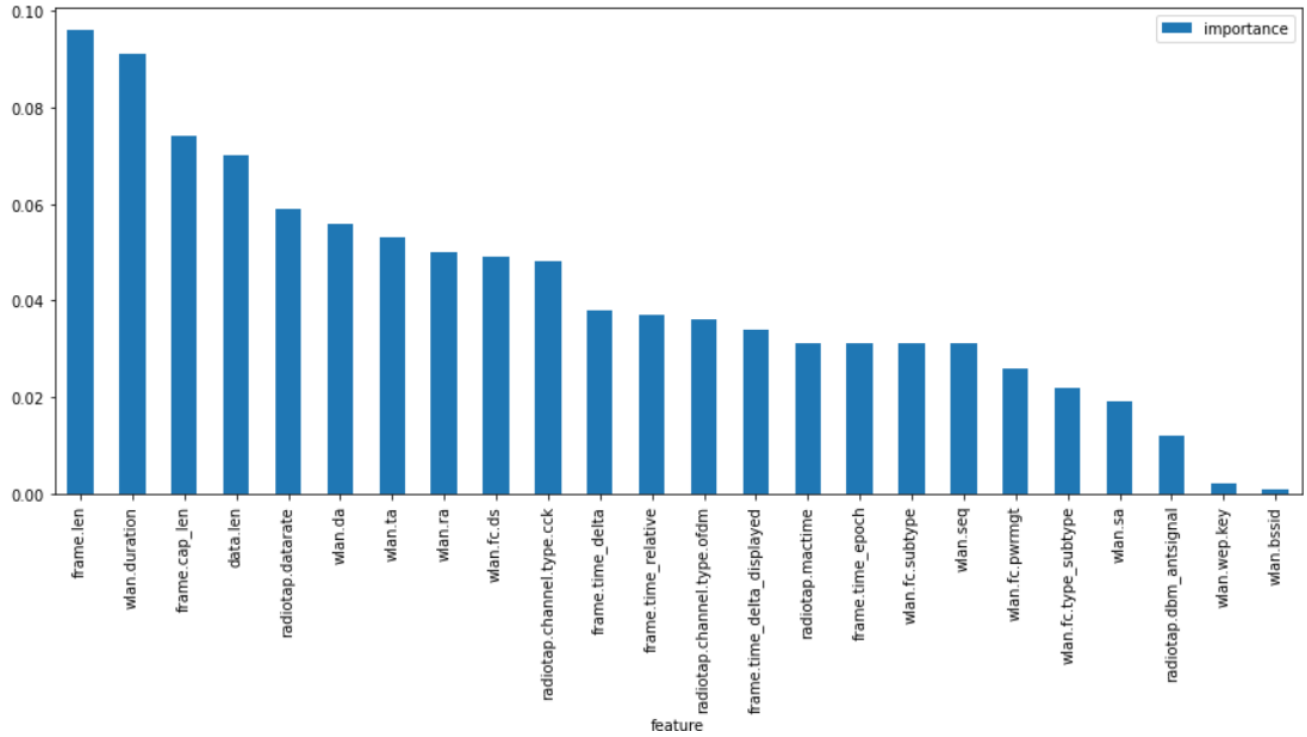

Fig 7.6 Feature selection

Fig 7.6 represents the important features which is selected from the dataset. An algorithm called random forest classifier is used to select the important features.

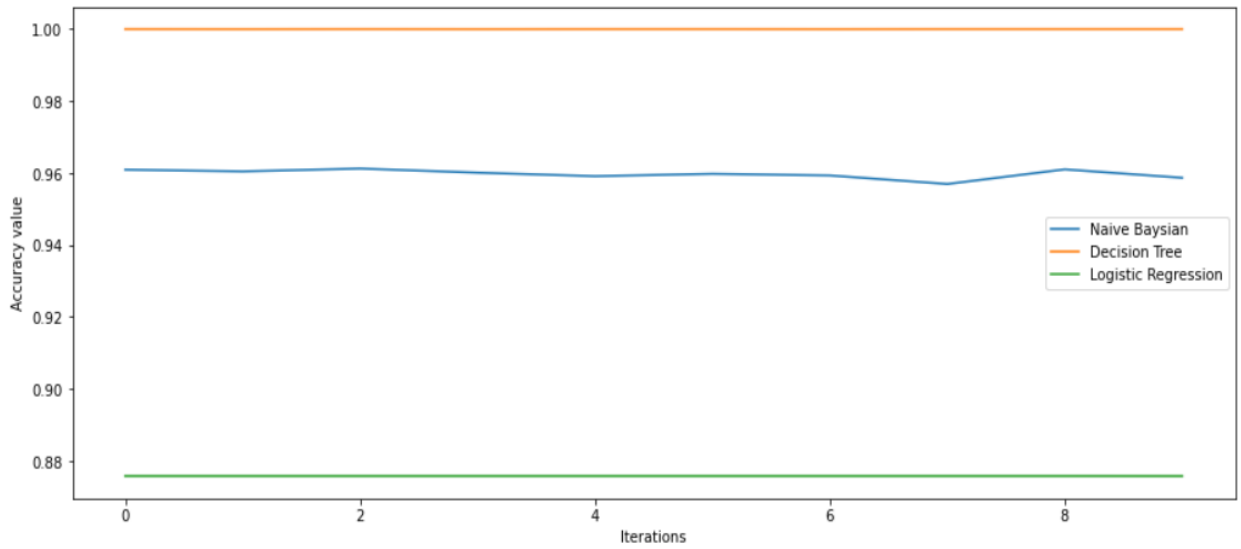

Fig 7.7 Accuracy value of 10 iterations for each algorithm

Fig 7.7 represents the accuracy of 10 iterations for each algorithm. The different algorithms used here is naïve bayes, decision tree and logistic regression. Decision tree has most accuracy compared to other algorithms.

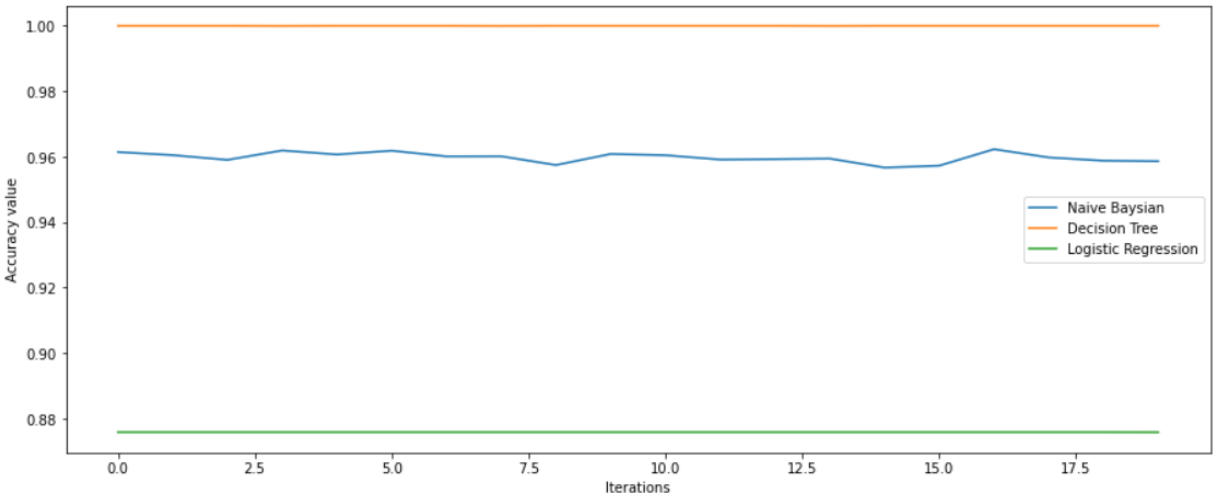

Fig 7.8 Accuracy value of 20 iterations for each algorithm

Fig 7.8 represents the accuracy of 10 iterations for each algorithm. The different algorithms used here is naïve bayes, decision tree and logistic regression. Decision tree has most accuracy compared to other algorithms.

	Iterations	DT	NB	LGR
0	10	0.999995	0.959777	0.875841
1	20	0.999983	0.959775	0.875841
2	30	1.000000	0.959792	0.875841
3	40	0.999995	0.959777	0.875841
4	50	0.999995	0.959777	0.875841

Fig 7.9 Mean Accuracy

Fig 7.9 shows the mean accuracy for each algorithm for 10 iterations starting from 10 to 50 iterations.

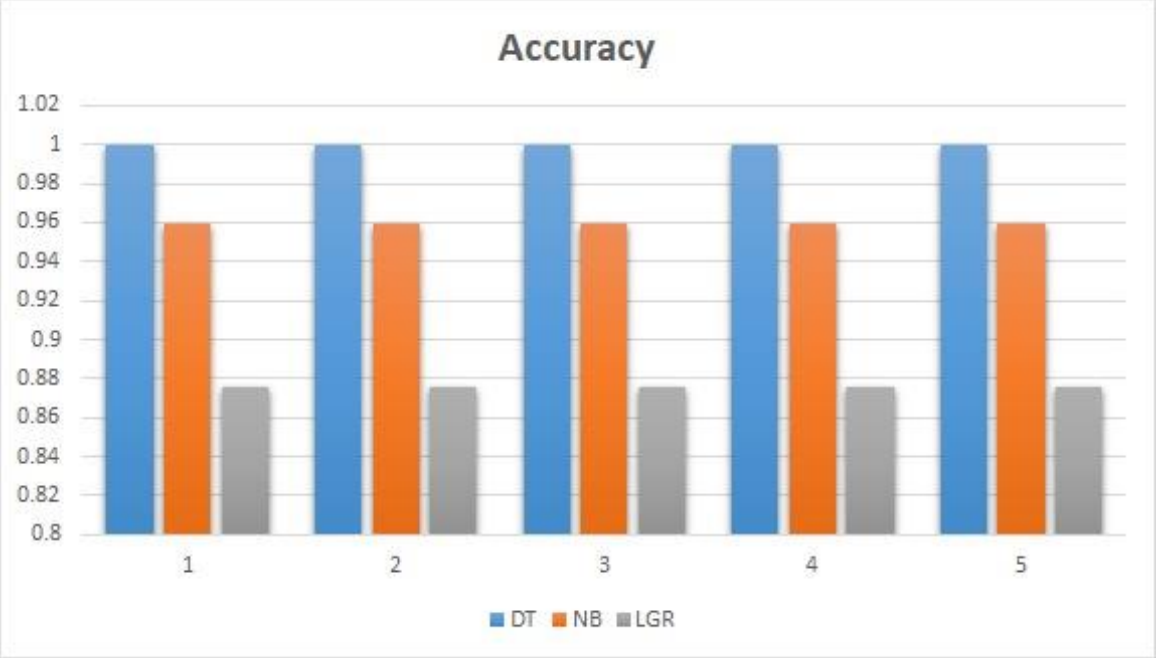

Fig 7.10 Total accuracy of the model from 10 to 50 iterations

Fig 7.10 represents the total accuracy of each algorithm from 10 to 50 iterations. Decision tree has the highest value and logistic regression has the lowest value.

CHAPTER 8

CONCLUSION AND FUTURE ENCHANMENTS

CHAPTER 8

CONCLUSION AND FUTURE ENCHANMENTS

This project compiles, visualises, and analyses a dataset based on the Wi-Fi Intrusion system. The AWID collection of datasets is well understood and used for the evaluation based on the resources. The main objective of this system is to detect anomalies in wi-fi network based on features available in the dataset. The system contains mainly of four modules, Data pre-processing module, Feature Extraction module and the training and testing module. The Data pre-processing module uses label encoders to process the data set, the feature extraction module uses random forest algorithm to identify important features and the training and testing modules uses decision tree, naive bayes and logistic regression algorithms to detect anomalies.

In the future, the system can be integrated with robust data collection modules to capture live data frames from the wi-fi network and can detect anomalies in real-time environment and the module can be can be enhanced using neural network techniques to predict attack in advance. The system can be improved to determine a wide range of anomalies such as DoS,U2R and etc.

References

1. A Semi-Supervised Learning Approach to IEEE 802.11 Network Anomaly Detection Jing Ran, Yidong Ji, and Bihua Tang
2. A Hybrid Deep Learning based Model for Anomaly Detection in Cloud Datacentre Networks Sahil Garg, Member, IEEE, Kuljeet Kaur, Member, IEEE, Neeraj Kumar, Senior Member, IEEE, Georges Kaddoum, Member, IEEE, Albert Y. Zomaya, Fellow, IEEE, and Rajiv Ranjan, Senior Member, IEEE
3. IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach Vrizlynn L. L. Thing
4. An anomaly-based Network Intrusion Detection System using Deep learning Nguyen Thanh Van, Tran Ngoc Thinh, Le Thanh Sach
5. Autoencoder-based Network Anomaly Detection Zhaomin Chen, Chai Kiat Yeo, Bu Sung Lee, Chiew Tong Lau.
6. IP Network Anomaly Detection using Machine Learning proposed by Uthara E.Prakash,Athira , Thankappan,Vishnupriya ,K.T, Arun A.Balakrishnan