IOTD : Internet of Things Detector

Minor Project - 1

Submitted by :

Ansh Goyal (9919103056)

Rohan Singh (9919103038)

Raghav Singh (9919103091)

Under the supervision of :

Dr. Charu Gandhi

Department of CSE/IT

Jaypee Institute of Information Technology University, Noida

December 2021

Table of Contents

ACKNOWLEDGEMENT

I would like to place on record my deep sense of gratitude to Dr. Charu Gandhi, Associate Professor and Departmental Coordinator, Jaypee Institute of Information Technology, India for her generous guidance, help and useful suggestions. I express my sincere gratitude to Dept. of CSE/IT, JIIT, for providing us the opportunity to work on this project.

I also wish to extend my thanks to my friends, Team member and other classmates for their insightful comments and constructive suggestions to improve the quality of this project work.

Signature(s) of Students

Ansh Goyal (9919103056)

Rohan Singh (9919103038)

Raghav Singh (9919103091)

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place: Jaypee Institute of Informationn Technology

Date: 01-12-2021

Name: Ansh Goyal

Enrolment No.: 9919103056

Name: Rohan Singh

Enrolment No.: 9919103038

Name: Raghav Singh

Enrolment No.: 9919103091

CERTIFICATE

This is certify that the work title “IoTD” submitted by Ansh Goyal, Rohan Singh, Raghav Singh students of B.Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of any other degree or diploma.

Digital Signature of Supervisor

Name of Supervisor : Dr. Charu Gandhi

Designation : Associate Professor and Departmental Coordinator

Date : 01-12-2021

Abstract

As the number of low-cost Internet-of-Things (IoT) devices increases dramatically in recent years, they have become ideal targets for flooding the network with junk traffic. Mostly every IoT Device is shipped to the market with default IP passwords. Operating systems of many IoT devices are often outdated or not well-configured. These device have very low vendor support so they rarely get software update or patches and they become vulnerable with the time because of outdated software. As a result these devices can be used by the advers adversary to flood the network with runt packets and this will lead to degradation of the network which affect all the other devices that are connect to that network. Now the main issue for administrator is that how to identify that this traffic is generated by which device, whether it is a IoT Device or a normal device. To solve this issue we built a tool using machine learning, called *IoT Detector (IoTD)* to detect and classify the IoT Device on the network by analysing packet capture (pcap), this tool will also provide you a real time graphical representation of mainly three type of traffic named TCP (Transmission Control Protocol), UDP (User Datagram Protocol) & ICMP (Internet Control Message Protocol). Experimental results shows that IoTD can accurately detect and identify IoT Devices. The accuracy evaluation among 12 devices of this study shows that 8 IoT Devices are true positive and the network sniffer is perfectly working on capturing network traffic.

List of Tables

List of Figures

Introduction

Internet-of-Things (IoT) devices that provide diverse applications in various fields have become more and more popular and important in recent years. A forecast indicates that the number of them will reach 25 billion in 2020.

Numerous electronic manufacturers provide various types of IoT products, such as network cameras, smart TV sets, smart fridges, smart power socket, and etc. As naturally low-end products, many of them are vulnerable to cyber-attacks because of rough configurations, default or weak passwords, and unpatched vulnerabilities. For example, a study [2] shows that cameras and speakers often transmit plain data over networks while they are operating; hence, attackers can eavesdrop the network traffic to retrieve the images, videos, and voices of victims. Another study [3] shows that power-switches and smart-bulbs can be compromised easily as well, due to poor authentication controls. Once IoT devices are compromised and organized, they may be used to conduct various malicious activities, including Flood attacks, E-mail spamming, vulnerability mining, and information stealing. However, it is a challenging work currently to protect all IoT devices against all cyber-attacks; hence, inevitably hosts are facing tremendous attacks coming from various compromised IoT devices.

The Server-Side detection of IoT Device is specially focused by this research. Many Network defenders are also implemented, spends plenty of resources to deal with detection of IoT and Flood attack. According to the study from proof-point, it reports that IoT attack activity in 2020 dramatically surpassed the combined volume of IoT activity observed by IBM Security X-Force in 2019. A recent example to put attention toward IoT device is that **The infamous Mirai botnet** that unleashed massive distributed denial-of-service (DDoS) attacks on major websites using millions of compromised devices in 2017 stands as a stark reminder of the power of IoT attacks, a power that continues to increase. In September 2020, IBM X-Force reported that IoT attacks we observed from October 2019 through June 2020 rose 400% when compared to the combined number of IoT attacks in the previous two years. Mozi has been extremely active in the last 18 months and continues to rank as the number one most active Mirai-type variant.

The Mozi botnet currently controls approximately 438,000 hosts, which is determined by the count of unique Mozi URLs we are tracking.

Each compromised host is instructed to hunt for new victim IoTs to infect while they await further instructions from the botnet's command and control.

IoTD is developed based on the following observation:

- Extracting feature from the raw data pcap dump of the network traffic using sniffer like : IPLength, IPHeaderLength, TTL, Protocol, Sourceport, DestinationPort, Acknowledgement number, sequencenumber, WindowSize etc.

- Machine learning model training using after preprocessing the data to predict out the result on the selected algorithm for Classification.

- Additional functionality is added by implementing Sniffer to get graphical representation of network.

What follows are the major contibution of IoTD :

- Machine Learning Model of IoTD is capable of classification of IoT Device at a server side. The IoT device we have taken into consideration for this project are : Aria, D-LinkHomeHub, D-LinkWaterSensor, EdimaxPlug2101W, D-LinkCam, D-LinkDayCam, D-LinkDoorSensor , D-LinkSensor, D-LinkSiren, D-LinkSwitch, EdimaxCam1, EdimaxCam2, EdimaxPlug1101W, EdnetCam1, EdnetCam2, EdnetGateway , HomeMaticPlug, HueBridge, HueSwitch ,Lightify, MAXGateway, TP-LinkPlugHS100, TP-LinkPlugHS110, WeMoInsightSwitch, WeMoInsightSwitch2, WeMoLink, WeMoSwitch, WeMoSwitch2, Withings.
- Along with IoTD a Network Sniffer is also deployed and hosted on server so that administrator can monitor the network traffic in real time.

Background Study

While building this project we have gained knowledge about some tools like Wireshark (network protocol analyzer), tcpdump, Weka. We also get insite knowledge about Machine learning and algorithms, we also get the touched some concepts of Web Development as we have used flask for our Frontend development.

What Is Wireshark and How Is It Used?

Few tools are as useful to the IT professional as Wireshark, the go-to network packet capture tool. Wireshark will help you capture network packets and display them at a granular level. Once these packets are broken down, you can use them for real-time or offline analysis. This tool lets you put your network traffic under a microscope, and then filter and drill down into it, zooming in on the root cause of problems, assisting with network analysis and ultimately network security. This free Wireshark tutorial will teach you how to capture, interpret, filter and inspect data packets to effectively troubleshoot.

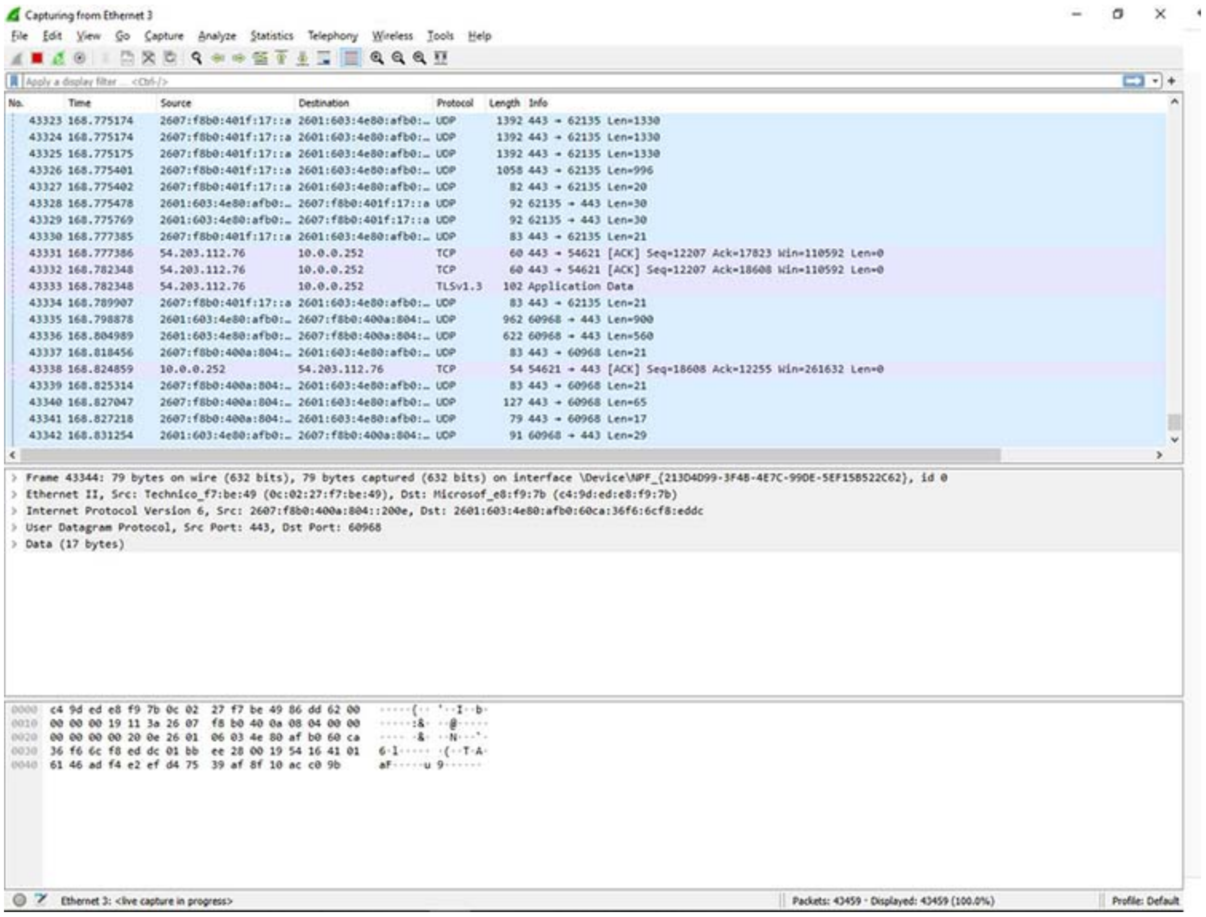
Wireshark is a network protocol analyzer, or an application that captures packets from a network connection, such as from your computer to your home office or the internet. Packet is the name given to a discrete unit of data in a typical Ethernet network.

Wireshark is the most often-used packet sniffer in the world. Like any other packet sniffer, Wireshark does three things:

- Packet Capture: Wireshark listens to a network connection in real time and then grabs entire streams of traffic – quite possibly tens of thousands of packets at a time.

- Filtering: Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain just the information you need to see.

- Visualization: Wireshark, like any good packet sniffer, allows you to dive right into the very middle of a network packet. It also allows you to visualize entire conversations and network streams.


Capturing from Ethernet 3
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.   Time       Source                Destination           Protocol Length Info
43323 168.775174 2607:f8b0:401f:17::a  2601:603:4e80:afb0:…  UDP      1392 443 → 62135 Len=1330
43324 168.775174 2607:f8b0:401f:17::a  2601:603:4e80:afb0:…  UDP      1392 443 → 62135 Len=1330
43325 168.775175 2607:f8b0:401f:17::a  2601:603:4e80:afb0:…  UDP      1392 443 → 62135 Len=1330
43326 168.775401 2607:f8b0:401f:17::a  2601:603:4e80:afb0:…  UDP      1058 443 → 62135 Len=996
43327 168.775402 2607:f8b0:401f:17::a  2601:603:4e80:afb0:…  UDP        82 443 → 62135 Len=20
43328 168.775478 2601:603:4e80:afb0:…  2607:f8b0:401f:17::a  UDP        92 62135 → 443 Len=30
43329 168.775769 2601:603:4e80:afb0:…  2607:f8b0:401f:17::a  UDP        92 62135 → 443 Len=30
43330 168.777385 2607:f8b0:401f:17::a  2601:603:4e80:afb0:…  UDP        83 443 → 62135 Len=21
43331 168.777386 54.203.112.76         10.0.0.252            TCP        60 443 → 54621 [ACK] Seq=12207 Ack=17823 Win=110592 Len=0
43332 168.782348 54.203.112.76         10.0.0.252            TCP        60 443 → 54621 [ACK] Seq=12207 Ack=18608 Win=110592 Len=0
43333 168.782348 54.203.112.76         10.0.0.252            TLSv1.3   102 Application Data
43334 168.789907 2607:f8b0:401f:17::a  2601:603:4e80:afb0:…  UDP        83 443 → 62135 Len=21
43335 168.798878 2601:603:4e80:afb0:…  2607:f8b0:400a:804:…  UDP       962 60968 → 443 Len=900
43336 168.804989 2601:603:4e80:afb0:…  2607:f8b0:400a:804:…  UDP       622 60968 → 443 Len=560
43337 168.818456 2607:f8b0:400a:804:…  2601:603:4e80:afb0:…  UDP        83 443 → 60968 Len=21
43338 168.824859 10.0.0.252            54.203.112.76         TCP        54 54621 → 443 [ACK] Seq=18608 Ack=12255 Win=261632 Len=0
43339 168.825314 2607:f8b0:400a:804:…  2601:603:4e80:afb0:…  UDP        83 443 → 60968 Len=21
43340 168.827047 2607:f8b0:400a:804:…  2601:603:4e80:afb0:…  UDP       127 443 → 60968 Len=65
43341 168.827218 2607:f8b0:400a:804:…  2601:603:4e80:afb0:…  UDP        79 443 → 60968 Len=17
43342 168.831254 2601:603:4e80:afb0:…  2607:f8b0:400a:804:…  UDP        91 60968 → 443 Len=29
```

```
> Frame 43344: 79 bytes on wire (632 bits), 79 bytes captured (632 bits) on interface \Device\NPF_{213D4D99-3F48-4E7C-99DE-5EF15B522C62}, id 0
> Ethernet II, Src: Technico_f7:be:49 (0c:02:27:f7:be:49), Dst: Microsof_e8:f9:7b (c4:9d:ed:e8:f9:7b)
> Internet Protocol Version 6, Src: 2607:f8b0:400a:804::200e, Dst: 2601:603:4e80:afb0:60ca:36f6:6cf8:eddc
> User Datagram Protocol, Src Port: 443, Dst Port: 60968
> Data (17 bytes)
```

```
0000  c4 9d ed e8 f9 7b 0c 02  27 f7 be 49 86 dd 62 00   ·····{·· '··I··b·
0010  00 00 00 19 11 3a 26 07  f8 b0 40 0a 08 04 00 00   ·····:&· ··@·····
0020  00 00 00 00 20 0e 26 01  06 03 4e 80 af b0 60 ca   ···· ·&· ··N···`·
0030  36 f6 6c f8 ed dc 01 bb  ee 28 00 19 54 16 41 01   6·l····· ·(··T·A·
0040  61 46 ad f4 e2 ef d4 75  39 af 8f 10 ac c0 9b      aF·····u 9······
```

```
Ethernet 3: <live capture in progress>    Packets: 43459 · Displayed: 43459 (100.0%)    Profile: Default
```

Fig [1] Wireshark

Multi-Label Classification

Multi-label classification refers to those classification tasks that have two or more class labels, where one or more class labels may be predicted for each example.

Consider the example of photo classification, where a given photo may have multiple objects in the scene and a model may predict the presence of multiple known objects in the photo, such as "*bicycle*," "*apple*," "*person*," etc.

This is unlike binary classification and multi-class classification, where a single class label is predicted for each example.

It is common to model multi-label classification tasks with a model that predicts multiple outputs, with each output taking predicted as a Bernoulli probability distribution. This is essentially a model that makes multiple binary classification predictions for each example.

Classification algorithms used for binary or multi-class classification cannot be used directly for multi-label classification. Specialized versions of standard classification algorithms can be used, so-called multi-label versions of the algorithms, including:

- Multi-label Decision Trees
- Multi-label Random Forests
- Multi-label Gradient Boosting

Another approach is to use a separate classification algorithm to predict the labels for each class.We can use the make_multilabel_classification() functionto generate a synthetic multi-label classification dataset. The example below generates a dataset with 1,000 examples, each with two input features. There are three classes, each of which may take on one of two labels (0 or 1).

```
# example of a multi-label classification task
from sklearn.datasets import make_multilabel_classification
# define dataset
X, y = make_multilabel_classification(n_samples=1000, n_features=2, n_classes=3, n_labels=2,
random_state=1)
# summarize dataset shape
print(X.shape, y.shape)
# summarize first few examples
for i in range(10):
	print(X[i], y[i])
```

Fig[2] Multi Label Classification Task

Running the example first summarizes the created dataset showing the 1,000 examples divided into input (*X*) and output (*y*) elements.

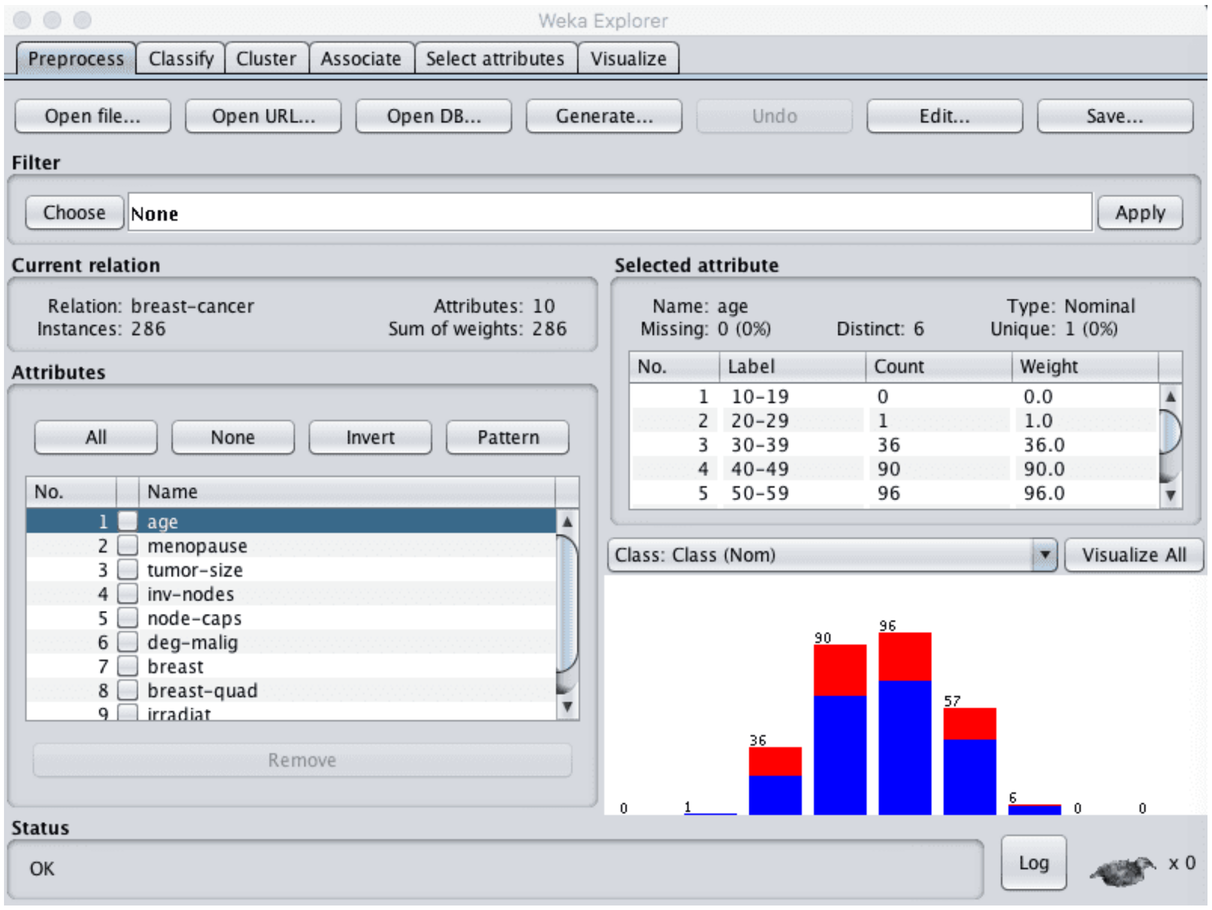
Next, the first 10 examples in the dataset are summarized showing the input values are numeric and the target values are integers that represent the class label membership.

```
(1000, 2) (1000, 3)

[18. 35.] [1 1 1]
[22. 33.] [1 1 1]
[26. 36.] [1 1 1]
[24. 28.] [1 1 0]
[23. 27.] [1 1 0]
[15. 31.] [0 1 0]
[20. 37.] [0 1 0]
[18. 31.] [1 1 1]
[29. 27.] [1 0 0]
[29. 28.] [1 1 0]
```

What is Weka and Why to use it?

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization.



Filter: None

Current relation

Relation: breast-cancer Attributes: 10
Instances: 286 Sum of weights: 286

Selected attribute

Name: age Type: Nominal
Missing: 0 (0%) Distinct: 6 Unique: 1 (0%)

No.	Label	Count	Weight
1	10-19	0	0.0
2	20-29	1	1.0
3	30-39	36	36.0
4	40-49	90	90.0
5	50-59	96	96.0

Attributes

No.	Name
1	age
2	menopause
3	tumor-size
4	inv-nodes
5	node-caps
6	deg-malig
7	breast
8	breast-quad
9	irradiat

Class: Class (Nom)

0 1 36 90 96 57 6 0 0

Status

OK

Fig[3] Weka

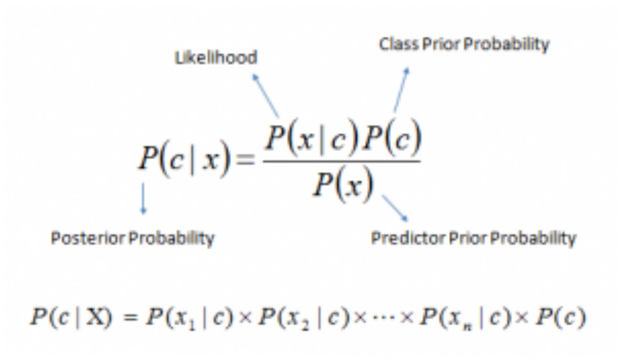
What is Naive Bayes algorithm?

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

$$P(c \mid \mathbf{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Above,

- *P*(*c*|*x*) is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- *P*(*c*) is the prior probability of *class*.
- *P*(*x*|*c*) is the likelihood which is the probability of *predictor* given *class*.
- *P*(*x*) is the prior probability of *predictor*.

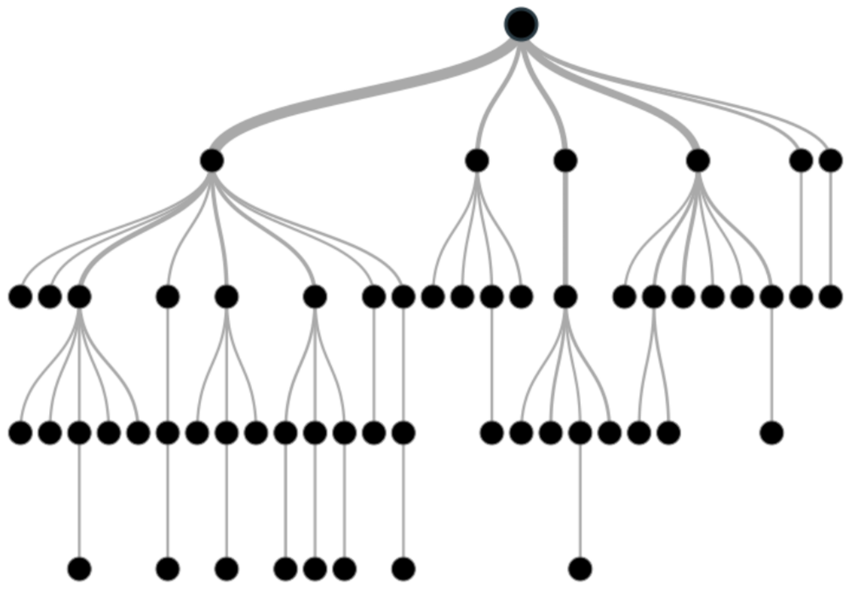
Applications of Naive Bayes Algorithms

- Real time Prediction**:** Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
- Multi class Prediction**:** This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
- Text classification/ Spam Filtering/ Sentiment Analysis: Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)
- Recommendation System**:** Naive Bayes Classifier and Collaborative Filtering together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not

Decision Tree

What is a Decision Tree?

It is a tool that has applications spanning several different areas. Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.



Fig[4] Decision Tree

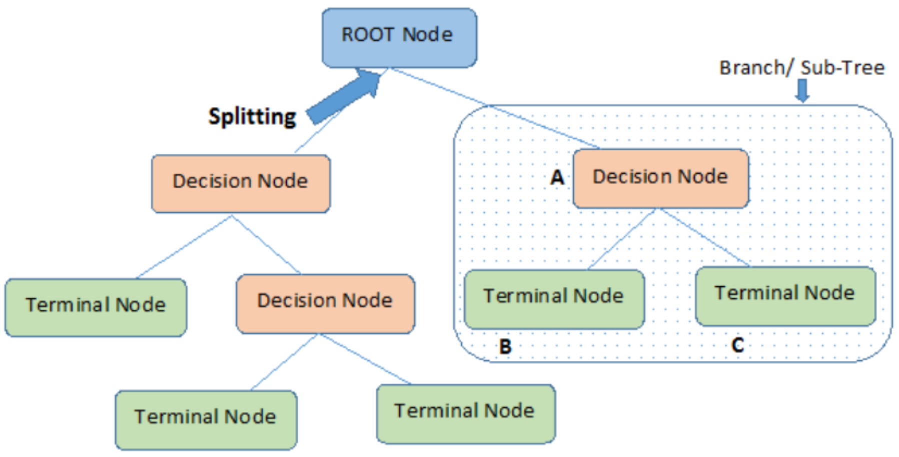
Root Nodes – It is the node present at the beginning of a decision tree from this node the population starts dividing according to various features.

Decision Nodes – the nodes we get after splitting the root nodes are called Decision Node

Leaf Nodes – the nodes where further splitting is not possible are called leaf nodes or terminal nodes

Sub-tree – just like a small portion of a graph is called sub-graph similarly a sub-section of this decision tree is called sub-tree.

Pruning – is nothing but cutting down some nodes to stop overfitting.



Fig[5] Decision Tree Component

Flask

Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks.

Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

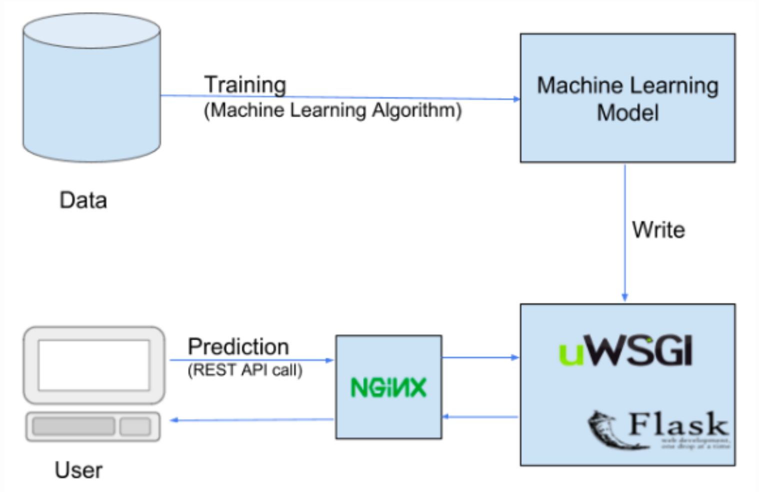


Fig [6] Flask Integration

HTML CSS

Hypertext Markup Language and CSS (Cascading Style Sheets) are two of the core technologies for building Web pages. HTML provides the structure of the page, CSS the (visual and aural) layout, for a variety of devices. Along with graphics and scripting, HTML and CSS are the basis of building Web pages and Web Applications.

What is CSS?

CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language. The separation of HTML from CSS makes it easier to maintain sites, share style sheets across pages, and tailor pages to different environments. This is referred to as the separation of structure (or: content) from presentation.

HPING

hping is a command-line oriented TCP/IP packet assembler/analyzer. The interface is inspired to the ping(8) unix command, but hping isn't only able to send ICMP echo requests. It supports TCP, UDP, ICMP and RAW-IP protocols, has a traceroute mode, the ability to send files between a covered channel, and many other features.

While hping was mainly used as a security tool in the past, it can be used in many ways by people that don't care about security to test networks and hosts. A subset of the stuff we can do using hping:

- Firewall testing
- Advanced port scanning
- Network testing, using different protocols, TOS, fragmentation
- Manual path MTU discovery
- Advanced traceroute, under all the supported protocols
- Remote OS fingerprinting
- Remote uptime guessing
- TCP/IP stacks auditing
- hping can also be useful to students that are learning TCP/IP

Requirement Analysis

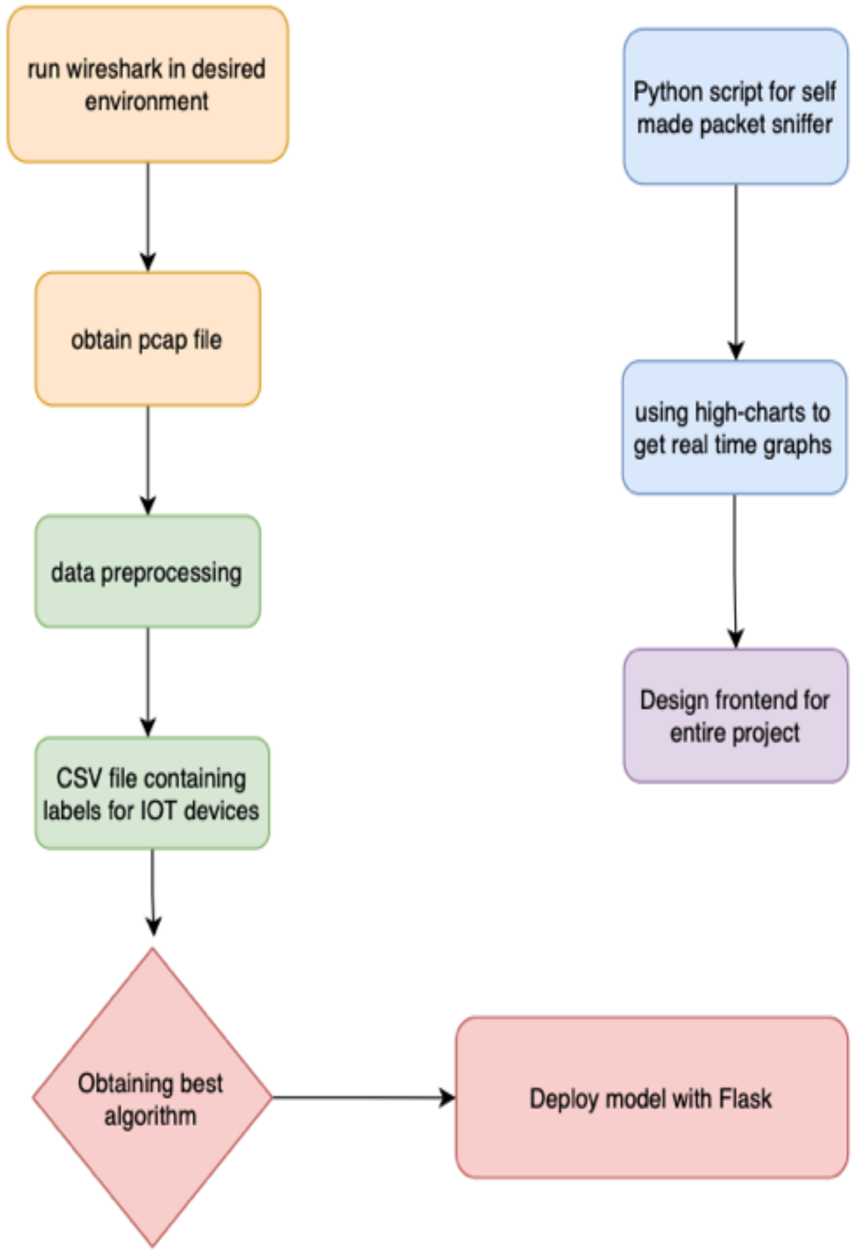
The Internet of Things (IoT) also called the Internet of Everything is a system of smart interconnected devices. The smart devices are uniquely identifiable over the network and perform autonomous data communication over the network with or without human-to-computer interaction. These devices have a high level of diversity, heterogeneity, and operates with various computational capabilities. It is highly necessary to develop a framework that allows to classify the devices into different categories from effective management, security, and privacy perspectives. Various solutions such as network traffic analysis, network protocols analysis, etc. have been developed to solve the problem of device classification.

The Importance of Network Traffic Monitoring

Network traffic monitoring and analysis is an essential component of network performance monitoring. It helps you determine whether your network's bandwidth is working correctly. If there are issues, network traffic monitoring can help you solve the problem fast. When implemented correctly, network traffic monitoring can help you identify and resolve network bottlenecks, find your network top talkers, and boost your security—an unusually high amount of network traffic could be a sign of a cyberattack.

Beyond these benefits, network traffic monitoring is also important because it can help with capacity planning and resource allocation—two essential processes for maintaining the performance of a network. When a slowdown occurs, network traffic analysis can give you visibility into which applications are suffering from a lack of resources and which ones are hogging them. You can use this information to distribute resources better and clear up congestion. When your network is running smoothly, network traffic monitoring can still give you useful insights into whether you've set your network up correctly. You can use this information to plan for future provisioning as your network continues to grow.

Detailed Design of IoTD


run wireshark in desired environment
obtain pcap file
data preprocessing
CSV file containing labels for IOT devices
Obtaining best algorithm
Deploy model with Flask
Python script for self made packet sniffer
using high-charts to get real time graphs
Design frontend for entire project

Fig[7] IoTD Flow-Chart

Implementation

Steps Involved in Data Exploration and pcap file feature extraction

Importing the data, from kaggle.com website, as a pcap file.

Importing all the necessary libraries for the thesis. Define a dictionary to store the string for each device to filtering

Open a csv file to store the labels and features. The header should be the name of features and the first column should be the label.

Filter out all the packets from the 13 different devices in the original pcap files and save the result into 13 new pcap files.

Create the labels and extract the features for them with the newly generated pcap files.

Steps Involve in IoT Device Detection

Import the required libraries

Read the features and class values from file less dataset

Since the last four columns of features are hexadecimal digits, they need to be converted to decimal digits.

Encode the labels for class values. Standard scale the feature values and split the dataset into training and testing set.

Define the function to print statistics metrics

Specify the classifiers

Libraries

Importing python libraries is one of the core aspects of completing any project, without the library not a single work can be done one the dataset. Even to read the dataset, the author must import a library. Many libraries were used while building the machine learning model and for data preparation. The libraries that were used by the author were given below with the description

Library	Description
Scikit	Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.
Pandas	The Pandas is one of the most important libraries while dealing with data, all the data analysis, and data manipulation can be done with the help of pandas
socket	This module provides access to the BSD socket interface. It is available on all modern Unix systems, Windows, MacOS, and probably additional platforms.
schedule	Schedule is in-process scheduler for periodic jobsthat use the builder patter for configuration.
textwrap	This module provides various time-related functions. For related functionality, see also the datetime and calendar modules.
Flask	Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. It is a lightweight abstraction that works with your existing ORM/ libraries. Flask-RESTful encourages best practices with minimal setup. If you are familiar with Flask, Flask-RESTful should be easy to pick up.
Pickle	The pickle module implements binary protocols for serializing and de-serializing a Python object structure. “Pickling” is the process whereby a Python object hierarchy is converted into a byte stream, and “unpickling” is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.
termcolor	termcolor’ module: termcolor is a python module for ANSII Color formatting for output in the terminal.

Table 1. Libraries

Dataset

The dataset used here is the packet capture of various IoT device on the network. The dataset that used for this project consist of 13 devices as the packets capture increase drastically on the network.

We will extract 18 features from each of the package from the devices. The 18 features are

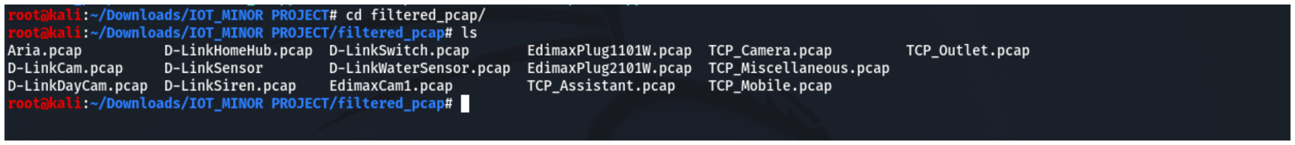
Feature name	Field name in pcap file
IPLength	ip.len
IPHeaderLength	ip.hdr_len
IPFlags	ip.flags
TTL	ip.ttl
Protocol	ip.proto
IPID	ip.id
Ipchecksum	ip.checksum
Sourceport	ip.srcport
DestPort	ip.destport
SequenceNumber	tcp.seq
AckNumber	tcp.ack
WindowSize	tcp.window_size_value
TCPHeaderLength	tcp.hdr_len
TCPFlags	tcp.flags
TCPChecksum	tcp.checksum
TCPStream	tcp.stream
TCPUrgentPointer	tcp.urgent_pointer

Table 2 Feature Vector

We have the traffic capture of 13 IoT devices on network in the form of pcap file. To extract features from the pcap files in order to represent the raw data in the vector space model. We use a tool called tshark to filter every packet to filter out all the packets from the different devices in the original pcap files and save the result into new pcap files.

Use tshark command to extract features from the currently processed file Use several -e options for each feature (field) that you want to extract from the pcap file, such as -e ip.len excute the command and save the result to the allFeatures parameter Before writing the label and features to the csv file, you need to convert the tab-separated results to the comma-separated results and breaking the results at line boundaries.

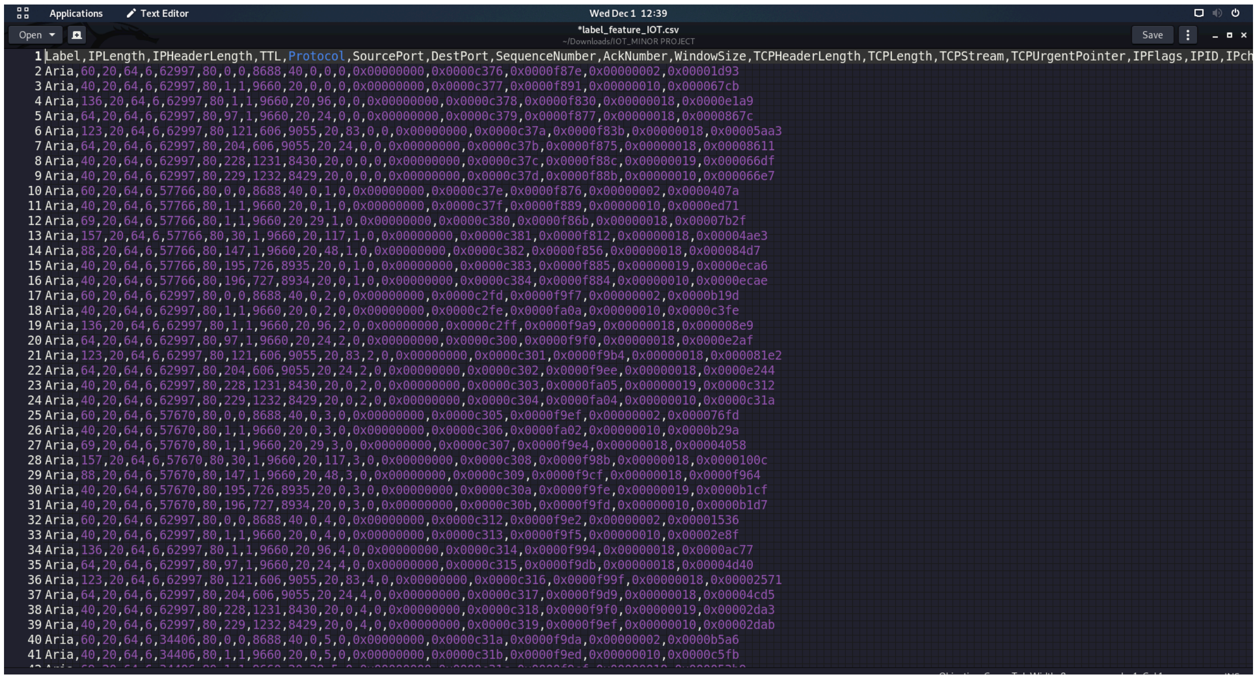
First row shows the label name (class) and feature names. Other rows show the label values and the feature values



```
root@kali:~/Downloads/IOT_MINOR PROJECT# cd filtered_pcap/
root@kali:~/Downloads/IOT_MINOR PROJECT/filtered_pcap# ls
Aria.pcap          D-LinkHomeHub.pcap  D-LinkSwitch.pcap       EdimaxPlug1101W.pcap  TCP_Camera.pcap         TCP_Outlet.pcap
D-LinkCam.pcap     D-LinkSensor        D-LinkWaterSensor.pcap  EdimaxPlug2101W.pcap  TCP_Miscellaneous.pcap
D-LinkDayCam.pcap  D-LinkSiren.pcap    EdimaxCam1.pcap         TCP_Assistant.pcap    TCP_Mobile.pcap
root@kali:~/Downloads/IOT_MINOR PROJECT/filtered_pcap# 
```

Filtered pcaps of IoT devices

All the packet will be distributed according to the filter so that the file name can be used as a label



```
Label,IPLength,IPHeaderLength,TTL,Protocol,SourcePort,DestPort,SequenceNumber,AckNumber,WindowSize,TCPHeaderLength,TCPLength,TCPStream,TCPUrgentPointer,IPFlags,IPID,IPch
Aria,60,20,64,6,62997,80,0,0,8688,40,0,0,0,0x00000000,0x0000c376,0x0000f87e,0x00000002,0x00001d93
Aria,40,20,64,6,62997,80,1,1,9660,20,0,0,0,0x00000000,0x0000c377,0x0000f891,0x00000010,0x000067cb
Aria,136,20,64,6,62997,80,1,1,9660,20,96,0,0,0x00000000,0x0000c378,0x0000f830,0x00000018,0x0000e1a9
Aria,64,20,64,6,62997,80,97,1,9660,20,24,0,0,0x00000000,0x0000c379,0x0000f877,0x00000018,0x0000867c
Aria,123,20,64,6,62997,80,121,606,9055,20,83,0,0,0x00000000,0x0000c37a,0x0000f83b,0x00000018,0x00005aa3
Aria,64,20,64,6,62997,80,204,606,9055,20,24,0,0,0x00000000,0x0000c37b,0x0000f875,0x00000018,0x00008611
Aria,40,20,64,6,62997,80,228,1231,8430,20,0,0,0,0x00000000,0x0000c37c,0x0000f88c,0x00000019,0x000066df
Aria,40,20,64,6,62997,80,229,1232,8429,20,0,0,0,0x00000000,0x0000c37d,0x0000f88b,0x00000010,0x000066e7
Aria,60,20,64,6,57766,80,0,0,8688,40,0,1,0,0x00000000,0x0000c37e,0x0000f876,0x00000002,0x0000407a
Aria,40,20,64,6,57766,80,1,1,9660,20,0,1,0,0x00000000,0x0000c37f,0x0000f889,0x00000010,0x0000ed71
Aria,69,20,64,6,57766,80,1,1,9660,20,29,1,0,0x00000000,0x0000c380,0x0000f86b,0x00000018,0x00007b2f
Aria,157,20,64,6,57766,80,30,1,9660,20,117,1,0,0x00000000,0x0000c381,0x0000f812,0x00000018,0x00004ae3
Aria,88,20,64,6,57766,80,147,1,9660,20,48,1,0,0x00000000,0x0000c382,0x0000f856,0x00000018,0x000084d7
Aria,40,20,64,6,57766,80,195,726,8935,20,0,1,0,0x00000000,0x0000c383,0x0000f885,0x00000019,0x0000eca6
Aria,40,20,64,6,57766,80,196,727,8934,20,0,1,0,0x00000000,0x0000c384,0x0000f884,0x00000010,0x0000ecae
Aria,60,20,64,6,62997,80,0,0,8688,40,0,2,0,0x00000000,0x0000c2fd,0x0000f9f7,0x00000002,0x0000b19d
Aria,40,20,64,6,62997,80,1,1,9660,20,0,2,0,0x00000000,0x0000c2fe,0x0000fa0a,0x00000010,0x0000c3fe
Aria,136,20,64,6,62997,80,1,1,9660,20,96,2,0,0x00000000,0x0000c2ff,0x0000f9a9,0x00000018,0x000008e9
Aria,64,20,64,6,62997,80,97,1,9660,20,24,2,0,0x00000000,0x0000c300,0x0000f9f0,0x00000018,0x0000e2af
Aria,123,20,64,6,62997,80,121,606,9055,20,83,2,0,0x00000000,0x0000c301,0x0000f9b4,0x00000018,0x000081e2
Aria,64,20,64,6,62997,80,204,606,9055,20,24,2,0,0x00000000,0x0000c302,0x0000f9ee,0x00000018,0x0000e244
Aria,40,20,64,6,62997,80,228,1231,8430,20,0,2,0,0x00000000,0x0000c303,0x0000fa05,0x00000019,0x0000c312
Aria,40,20,64,6,62997,80,229,1232,8429,20,0,2,0,0x00000000,0x0000c304,0x0000fa04,0x00000010,0x0000c31a
Aria,60,20,64,6,57670,80,0,0,8688,40,0,3,0,0x00000000,0x0000c305,0x0000f9ef,0x00000002,0x000076fd
Aria,40,20,64,6,57670,80,1,1,9660,20,0,3,0,0x00000000,0x0000c306,0x0000fa02,0x00000010,0x0000b29a
Aria,69,20,64,6,57670,80,1,1,9660,20,29,3,0,0x00000000,0x0000c307,0x0000f9e4,0x00000018,0x00004058
Aria,157,20,64,6,57670,80,30,1,9660,20,117,3,0,0x00000000,0x0000c308,0x0000f98b,0x00000018,0x0000100c
Aria,88,20,64,6,57670,80,147,1,9660,20,48,3,0,0x00000000,0x0000c309,0x0000f9cf,0x00000018,0x0000f964
Aria,40,20,64,6,57670,80,195,726,8935,20,0,3,0,0x00000000,0x0000c30a,0x0000f9fe,0x00000019,0x0000b1cf
Aria,40,20,64,6,57670,80,196,727,8934,20,0,3,0,0x00000000,0x0000c30b,0x0000f9fd,0x00000010,0x0000b1d7
Aria,60,20,64,6,62997,80,0,0,8688,40,0,4,0,0x00000000,0x0000c312,0x0000f9e2,0x00000002,0x00001536
Aria,40,20,64,6,62997,80,1,1,9660,20,0,4,0,0x00000000,0x0000c313,0x0000f9f5,0x00000010,0x00002e8f
Aria,136,20,64,6,62997,80,1,1,9660,20,96,4,0,0x00000000,0x0000c314,0x0000f994,0x00000018,0x0000ac77
Aria,64,20,64,6,62997,80,97,1,9660,20,24,4,0,0x00000000,0x0000c315,0x0000f9db,0x00000018,0x00004d40
Aria,123,20,64,6,62997,80,121,606,9055,20,83,4,0,0x00000000,0x0000c316,0x0000f99f,0x00000018,0x00002571
Aria,64,20,64,6,62997,80,204,606,9055,20,24,4,0,0x00000000,0x0000c317,0x0000f9d9,0x00000018,0x00004cd5
Aria,40,20,64,6,62997,80,228,1231,8430,20,0,4,0,0x00000000,0x0000c318,0x0000f9f0,0x00000019,0x00002da3
Aria,40,20,64,6,62997,80,229,1232,8429,20,0,4,0,0x00000000,0x0000c319,0x0000f9ef,0x00000010,0x00002dab
Aria,60,20,64,6,34406,80,0,0,8688,40,0,5,0,0x00000000,0x0000c31a,0x0000f9da,0x00000002,0x0000b5a6
Aria,40,20,64,6,34406,80,1,1,9660,20,0,5,0,0x00000000,0x0000c31b,0x0000f9ed,0x00000010,0x0000c5fb
```

Fig [8] Label_Feature CSV file

Encode the labels for class values. Standard scale the feature values and split the dataset into training and testing set. LabelEncoder can transform non-numerical labels to numerical labels. StandardScaler can standardize the features by removing the mean and scaling to unit variance. train_test_split can randomly select data and split the dataset into training set and testing set according to the ratio. We set True values and predicted values are used to print the confusion matrix and other statistics metrics
Here we select logistic regression and Naive Bayes as the classifiers.We first fit them with training set and perform the prediction with testing set.Finally, print_stats_metrics() is called to print out the statistics

Now this complete model along with sniffer is integrated with flask web application.

Experimental Results and Analysis

Login Page : A authentication page is made for administrator to login into IoTD. This will protect from miss use from the ousider and unauthorized user.

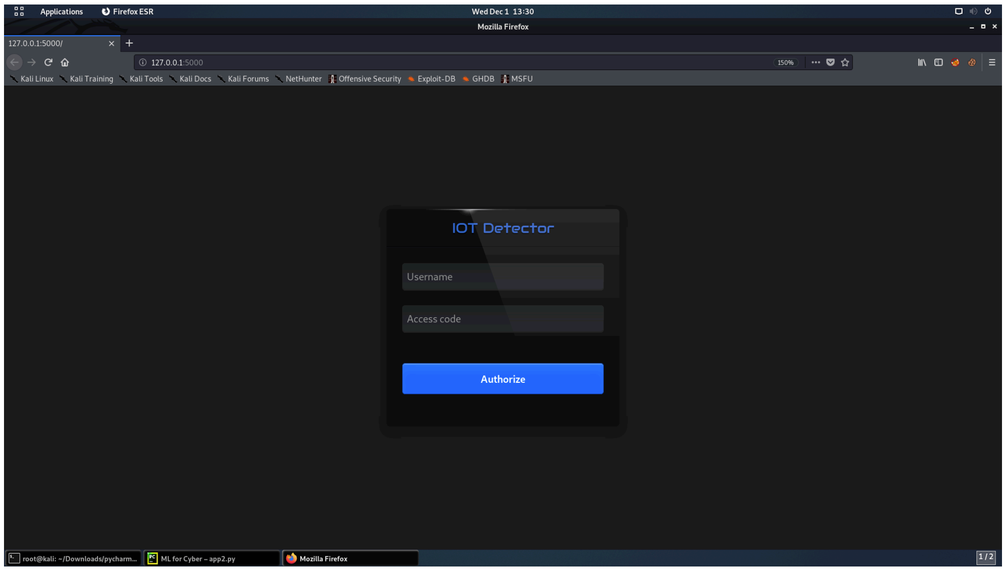
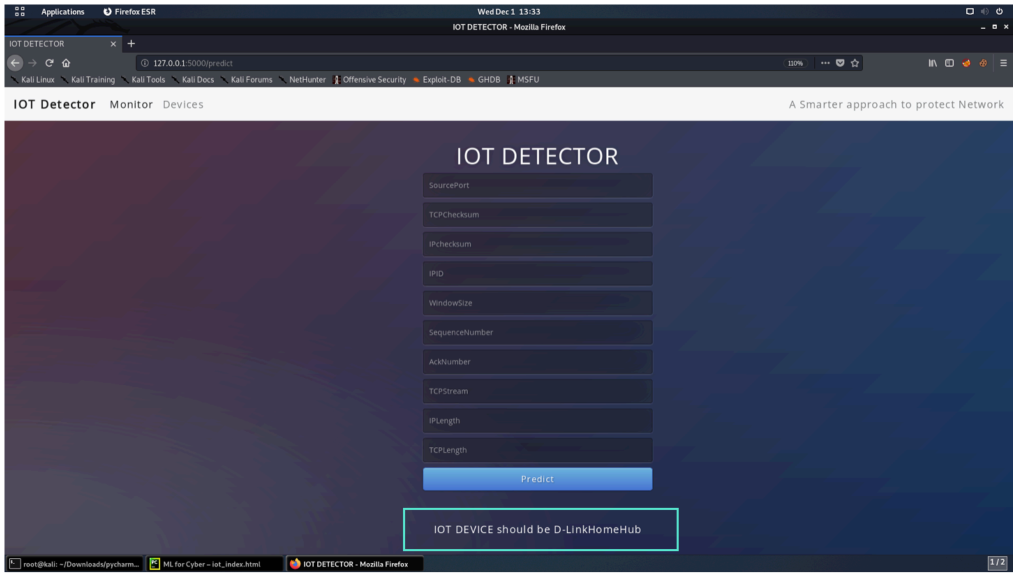


Fig [9] Login Page

Dashboard Page is created using bootstrap, HTML and CSS. Admin need to enter some parameters and this will predict the result using our trained model and display the result on the screen.



IOT DEVICE should be D-LinkHomeHub

Fig [10] Dashboard

Build a Network Sniffer which get updated in every 4 seconds and the number of packets is being plotted on the curve. This will give a idea about the volume of Flood attack, whether it is TCP(Transmission Control Protocol), UDP(User Datagram Protocol) or ICMP(Internet Control Message Protocol)

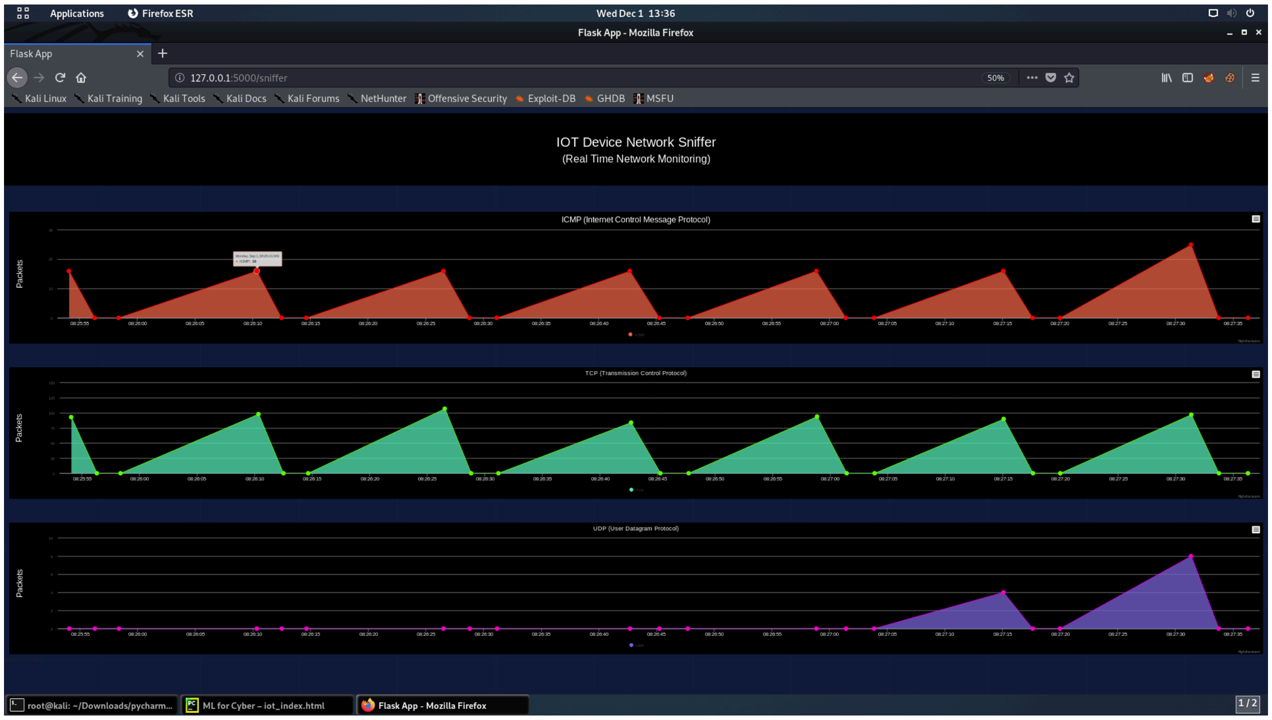
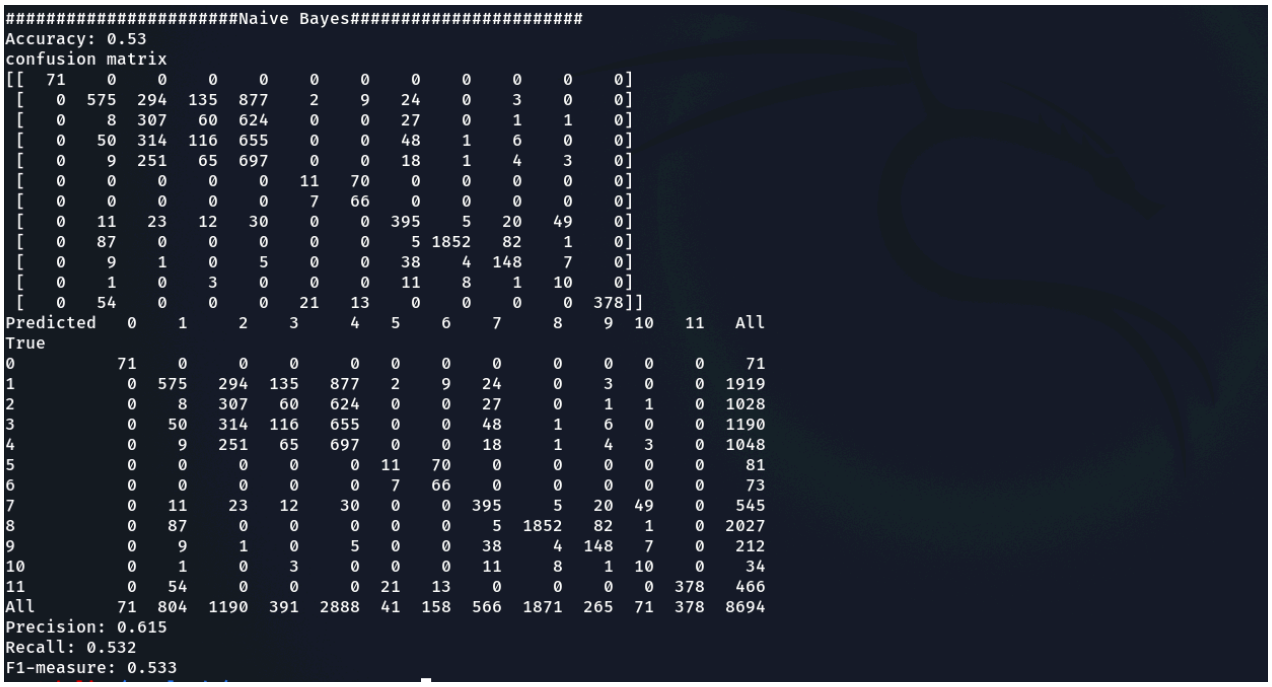

Applications
Firefox ESR
Wed Dec 1 13:36
Flask App - Mozilla Firefox
Flask App
127.0.0.1:5000/sniffer
Kali Linux
Kali Training
Kali Tools
Kali Docs
Kali Forums
NetHunter
Offensive Security
Exploit-DB
GHDB
MSFU
IOT Device Network Sniffer
(Real Time Network Monitoring)
ICMP (Internet Control Message Protocol)
Packets
TCP (Transmission Control Protocol)
Packets
UDP (User Datagram Protocol)
Packets
root@kali: ~/Downloads/pycharm...
ML for Cyber – iot_index.html
Flask App - Mozilla Firefox
1 / 2

Fig [11] Network Sniffer

Performance of the model is not that good as we expected. The accuracy of our model is 53% when Naive Bayes is used and 60% when logistic regression is used.



```
#######################Naive Bayes#######################
Accuracy: 0.53
confusion matrix
[[  71    0    0    0    0    0    0    0    0    0    0    0]
 [   0  575  294  135  877    2    9   24    0    3    0    0]
 [   0    8  307   60  624    0    0   27    0    1    1    0]
 [   0   50  314  116  655    0    0   48    1    6    0    0]
 [   0    9  251   65  697    0    0   18    1    4    3    0]
 [   0    0    0    0    0   11   70    0    0    0    0    0]
 [   0    0    0    0    0    7   66    0    0    0    0    0]
 [   0   11   23   12   30    0    0  395    5   20   49    0]
 [   0   87    0    0    0    0    0    5 1852   82    1    0]
 [   0    9    1    0    5    0    0   38    4  148    7    0]
 [   0    1    0    3    0    0    0   11    8    1   10    0]
 [   0   54    0    0    0   21   13    0    0    0    0  378]]
Predicted   0    1     2    3     4   5    6    7     8    9  10   11   All
True
0          71    0     0    0     0   0    0    0     0    0   0    0    71
1           0  575   294  135   877   2    9   24     0    3   0    0  1919
2           0    8   307   60   624   0    0   27     0    1   1    0  1028
3           0   50   314  116   655   0    0   48     1    6   0    0  1190
4           0    9   251   65   697   0    0   18     1    4   3    0  1048
5           0    0     0    0     0  11   70    0     0    0   0    0    81
6           0    0     0    0     0   7   66    0     0    0   0    0    73
7           0   11    23   12    30   0    0  395     5   20  49    0   545
8           0   87     0    0     0   0    0    5  1852   82   1    0  2027
9           0    9     1    0     5   0    0   38     4  148   7    0   212
10          0    1     0    3     0   0    0   11     8    1  10    0    34
11          0   54     0    0     0  21   13    0     0    0   0  378   466
All        71  804  1190  391  2888  41  158  566  1871  265  71  378  8694
Precision: 0.615
Recall: 0.532
F1-measure: 0.533
```

Fig [12] Result

Conclusion of the Report and Future Scope

We conclude that the dataset is not a complete space, and there are still other feature vectors missing from it. What we were attempting to generalize is a subspace of the actual input space, where the other dimensions are not known, and hence none of the classifiers were able to do better than 53.6% (Naïve Bayes). In the future, if similar studies are conducted to generate the dataset used in this report, more feature vectors need to be calculated so that the classifiers can form a better idea of the problem at hand and we also try to implement some more secure way of authentication.

References

Book:

[1] Samuel Greengard, “The Internet of Things”

Online:

[2] https://blog.apnic.net/2020/12/23/how-to-detect-iot-devices-in-a-network/

[3] https://www.researchgate.net/post/What_is_the_best_way_to_identify_IoT_devices

[4] https://dl.acm.org/doi/10.1145/3229565.3229572

[5] https://www.mdpi.com/1424-8220/21/8/2660