UNIVERSITY OF CALCUTTA

Asutosh College

(Department of Computer Science)

Major Project Report On

Detection-of-Botnet-attack-on-IoT-devices

By

Aritra Mukhrjee

Reg. No: 012-1221-0799-13 Roll: CIS/17/04 No:**004**

Sinjhita Das

Reg. No : 012-1221-0875-13 Roll: CIS/17/04 No:**016**

Sujash Naskar

Reg. No : A03-1112-0034-14 Roll: CIS/17/04 No :**024**

Debasish Chatterjee

Reg. No : 544-1121-0701-13 Roll: CIS/17/04 No :**010**

Under the supervision of

Prof. Gautam Mahapatra
Associate Professor

Prof. Ankita Sinha

CERTIFICATE OF PROJECT COMPLETION

This is to certify that the MAJOR PROJECT report with title **Detection-of-Botnet-attack-on-IoT-devices**is being successfully submitted by **Aritra Mukherjee** (Reg No.012-1221-0799-13, Roll No.-CIS/17/04/No. 04), **Shinjita Das** (Reg No.012-1221-0875-13, Roll No - CIS/17/04/No. 016), **Sujash Naskar** Reg No.(A03-1112-0034-14) Roll No.(CIS/17/04/No. 024**)** **Debasish Chatterjee** (Reg No.544-1121-0701-13, Roll No. - CIS/17/04/No.010) under the guidance and supervision of **Prof. Gautam Mahapatra** and **Prof. Antika Sinha**, as a partial fulfillment of Four Semester/Two Years Masters of Science in Computer Science Course, University of Calcutta, from the Dept. of Computer Science, Asutosh College, 92, S.P.Mukherjee Road, Kolkata,Pin-700026.Undersigned are wishing their every success in life.

[illegible] 21/07/2019

Prof.Gautam Mahapatra (GUIDE)

[illegible]

Prof.Antika Sinha (CO.GUIDE)

[illegible]

Head

Dept. of Computer Science

Examiners:

1. [illegible]
2. [illegible]
3. [illegible]
4. [illegible]

Acknowledgement

No project is created by any individual. Many people have helped to create this project and each of their contribution has been valuable. The timely completion of this project is possible mainly due to proper guidance and persuasion of the project guide Prof. Gautam Mahapatra and Prof. Ankita Sinha.

We extend our sincere thanks to our project guides, for continuously helping us throughout the project and without their guidance, this project would have been an uphill task.

We would like to thank other individuals in M.Sc Computer Science Department for their valuable ideas, and creativity.

Date: 24/04/2019

Sujash Naskar

Preface

A Bot is a type of malware that allows an attacker to take control of infected machine. The Botnet is a network of bots. A Bot infected machine is often called as zombie and Cyber-criminals who control these bots are called Botherders or Botmasters. Bots are often spread themselves across internet by searching for vulnerable machines to expand. The way the bots are controlled depends upon architecture of botnet Command and Control(C&C) mechanism which may be based on Internet Relay Chat (IRC) or HTTP or Peer to Peer (P2P). Botnet is widely used to carry out malicious activities like Distributed Denial of Service (DDoS) attacks, sending spam mails and click frauds. In recent years, botnet based attacks have become more sophisticated and can bypass all security safeguards.
Botnet detection techniques are broadly based on either setting up of a honeypot to collect bot binaries or developing intrusion detection system. The intrusion detection system (IDS) identify botnet traffic by monitoring network and system logs. It can be based on anomaly behavior or signature or DNS. The Netow analyzer is popular tool for detecting botnet anomaly based detection. The Snort, Suricata, Ntop, Bothunter are other tools which are based on signatures of botnet. The DNS based botnet traffic is monitored by Wireshark. The BotMiner tool uses clustering algorithm to detect botnet.

We have identified some botnet attack based on UCI data using SVM.Due to memory limitation we were unable to use the whole dataset.Thus, resulting low accuracy. Sampling techniques can be used to increase the accuracy ,hence better result.

Contents

Detection of Botenet Attack on IoT Devices

1.1 Introduction :

A Bot is an autonomous program automatically perform task without knowing to a real user. A collection of machines which run such autonomous bot is called as botnets. Bot is remotely controlled by command and control server. The black-hat developers created highly sophisticated malwares that are difficult to detect and remove. Bot program is stealthy during its whole life cycle. They had generated relatively small network footprint and most of time remains ideal for stealing information. The concept of remote-controlled computer bot originated from Internet Relay Chat (IRC). It provides one to many communications channels and support very large number of concurrent users**. Eggdrop was first bot developed in 1993**.

As internet connects billions of computers, tablets, smart phones together to share the Information across the globe, peoples are relying on these technologies to share their personal as well as business information. The black hat hackers used its vulnerabilities to perform attacks. The initial intention of these cyber criminals was just to gain fame but over the period they are doing criminal activities to earn money.

1.2 Lifecycle of a Bot :

The bot lifecycle consists of following phases shown in following figure.

Creation: Firstly, botmaster develop his software mostly by extending previous code or by adding new features. This is very well tested in isolated environment.

Infection: There are many ways for infecting victim’s machine through software vulnerabilities, email attachments and trojan horse. Once victim’s machine is infected by this software then it is called zombie.

Rallying: After infection, zombie machine attempts first and try to contact command and control machine. This process is called Rallying. In centralized botnet topology, this could be IRC or HTTP servers whereas in P2P topology zombie tries to locate peer machine and join the network. Bot program contains multiple addresses of servers. Some C&C servers are configured in such a way that it immediately reply to bots initial request.

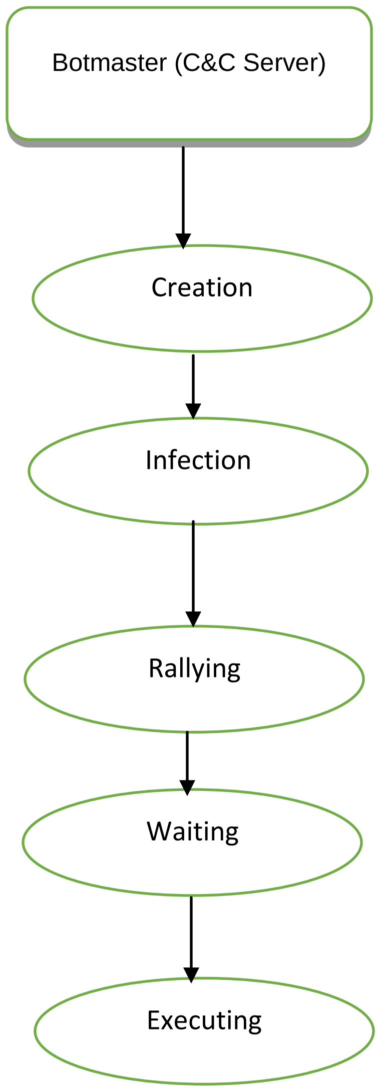


Fig: Bot Lifecycle

Waiting: After joining to network, bot waits for command from C&C server. Duringthis phase very little traffic is found between bot and its master.

Executing: Once the bot received command from its master, it starts executing it.After execution it sends result to bot master via C&C network. Typical commandsare: scanning for new victims, sending spam, and sending DoSflood.

1.3.Botnet Architecture:

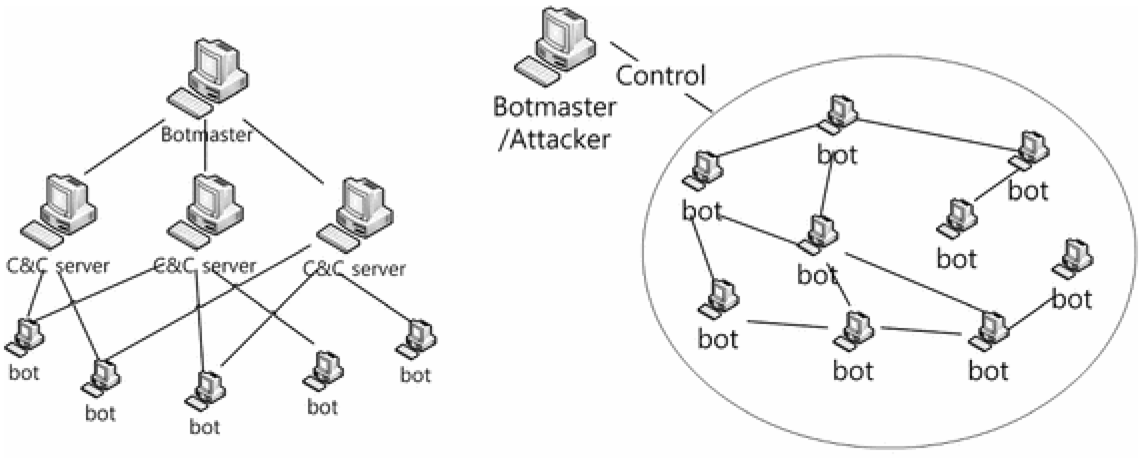
There are two main botnet topologies: **centralized** and **peer to peer** (P2P). In centralized botnets, IRC is still pre dominant protocol of C&C channel. Now this trend is decreasing and new bots come with HTTP for their C&C channel. The major drawback of centralized botnets is single point failure. If centralized entity is removed the entire network is unusable. But, modern botnets overcome this problem by using fast-ux DNS techniques.
In fast flux DNS techniques it is very difficult to trace central entity. The compromised hosts are used as proxies to hide identities of true C&C servers. These hosts constantly alternates DNS configuration to resolve one hostname with multiple IP addresses. Popular examples of IRC bots are Agobot, Spybot, and Sdbot.

- **Client-Server Botnet**

Botnets were originallyconstructed and operated using a Client-Server model. The infected clients connect to an infected server awaiting commands from the botmaster. Once the botmaster sends commands to the infected server, each client retrieves and executes those commands and reports back their results of actions to the infected server.

- **P2P Botnet**

The problem with client-server botnets is the single point of failure. Therefore, to avoid this issue, new botnets fully operate over Peer-to-Peer (P2P) networks, where each peer acts simultaneously both as a client and as a server. Despite this kind of structure, which operates without a centralized point that makes it hard to be blocked by IP address, botnets are still blockable by ports. Therefore, a combination of HTTP and P2P botnet is used, called HTTP2P botnet, which uses HTTP as the communication protocol and often employs port 80, a method that makes it impossible to be blocked by ports.



(a) Centralized IRC/HTTP botnet (b) Decentralized P2P botnet

1.4 Motivation:

Botnet means an organized automated army of zombies which can be used for creating a **DDoS attack as well as spammy actions of flooding any inbox or spreading the viruses**. Actually, this army consists of a large number of computers. Attackers use this army for malicious purposes and generally, zombies are not even aware of that they are used for malicious purposes.

Zombies have been used extensively to send spam mail; as of 2005, an estimated 50–80% of all spam was sent by zombie computers worldwide. This allows spammers to avoid detection and presumably reduces their bandwidth costs since the owners of zombies pay for their own bandwidth. General structure about botnet attacks is given below.

This process is carried out by a centralized entity called C&C, which is also called a botmaster. A botmaster is an entity that coordinates to initiate, manage, or suspend attacks on all infected machines (bots). Therefore, the aim of the C&C mechanism is to increase the number of zombie machines and to coordinate those machines for so many destructive operations. The difference between a botnet and other types of network attacks is the existence of C&C in the network. In addition, the bots receive instructions from C&C and act upon those instructions. The instructions/commands range from initiating a worm or spam attack over the Internet to disrupt a legitimate user request.

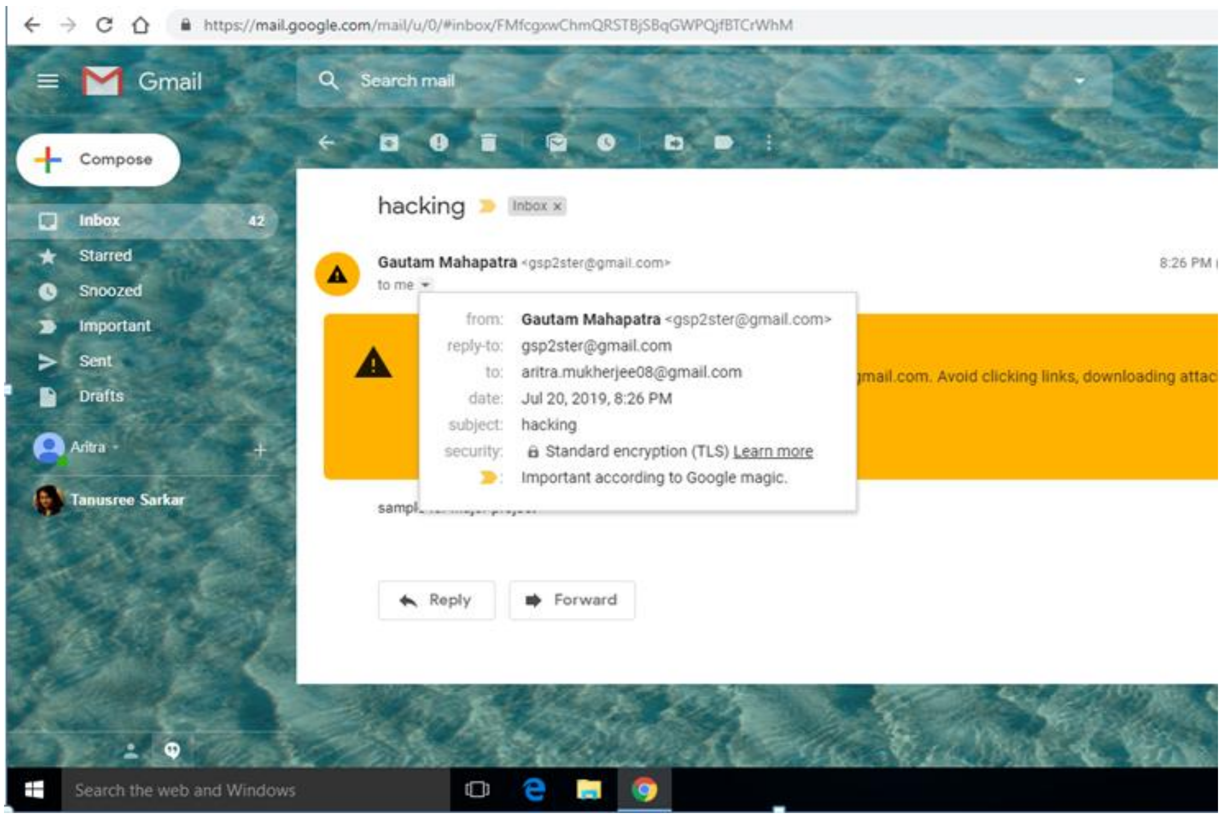
A botnet can do anything which you can imagine by the use of many computers connected to a network. Distributed power resources are the key points of the power of botnets.

Botnets are network of compromised hosts and remotely controlled computer system. In recent years, the diversity of malware has grown almost exponentially. The main goal of botnet master is to gain _nancialpro_t from the activities they allow and other include political or even military interests. In the last few years, some applications related to botnets have taken a leading role which motivates researchers to resolve these issues.

1.5.The major applications based on botnets are listed below:

1.5.1. Identity Theft:

Botmaster automatically extract users data and credentials from infected hosts. Its main targetsinclude passwords for various services like e-mail accounts, web shops, banking platforms or social networking platforms.


https://mail.google.com/mail/u/0/#inbox/FMfcgxwChmQRSTBjSBqGWPQjfBTCrWhM

hacking Inbox ×

Gautam Mahapatra <gsp2ster@gmail.com> 8:26 PM
to me

from: Gautam Mahapatra <gsp2ster@gmail.com>
reply-to: gsp2ster@gmail.com
to: aritra.mukherjee08@gmail.com
date: Jul 20, 2019, 8:26 PM
subject: hacking
security: Standard encryption (TLS) Learn more
: Important according to Google magic.

gmail.com. Avoid clicking links, downloading attac

1.5.2. Spam Email:

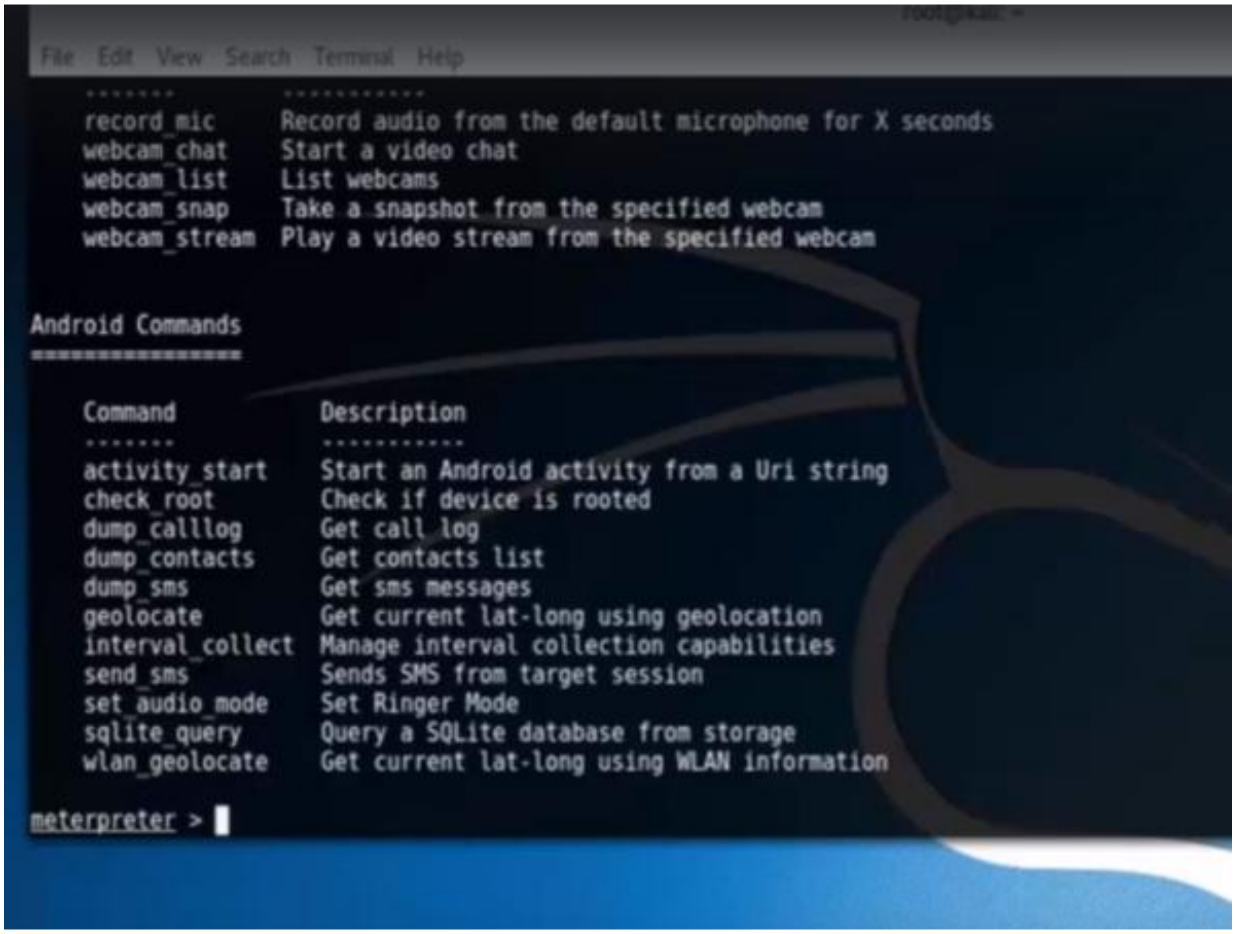
The popular use of botnets is for unsolicited mass mailing, also known as spamming. Recently, spammers are attract towards botnets which own high computation power network of compromised computers.

1.5.3. Click Frauds:

As botmaster has full control on infected machine, the attacker take help of the controlled bots to visit the pages and to generate clicks on the target banners. In this case, the attacker gains money directly from the advertising company.

1.5.4. Distributed Denial of Service attack(DDoS):

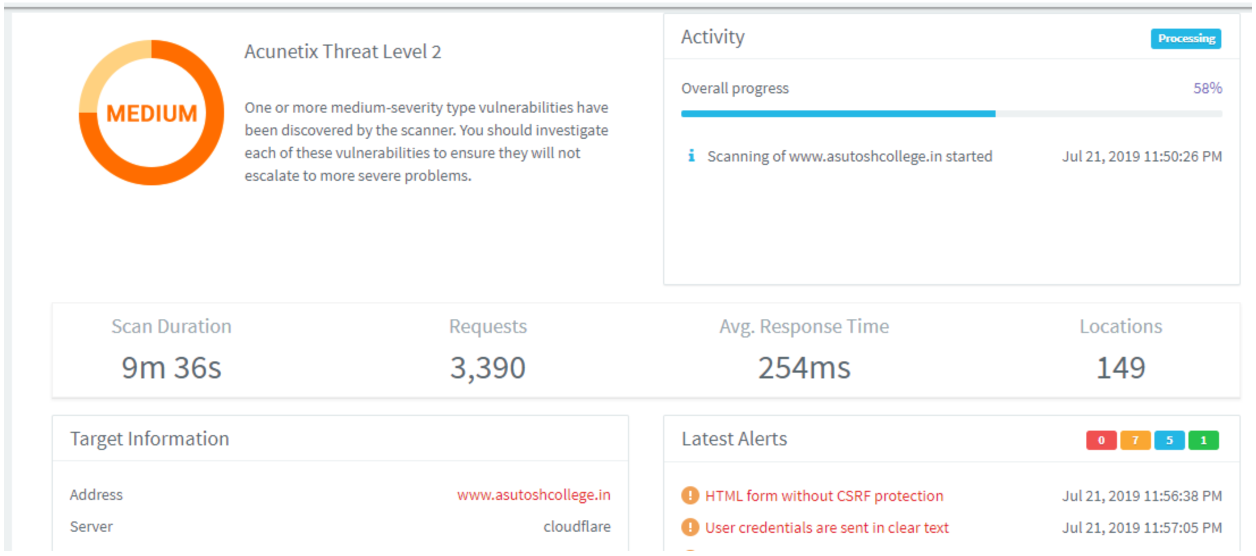
Botnets usually consist of large numbers of remote machines, their cumulative bandwidth can reach multiple gigabytes of upstream traffic per second. This enables botmasters to start targeted sabotage attacks against websites.



```
    -------        -----------
    record_mic     Record audio from the default microphone for X seconds
    webcam_chat    Start a video chat
    webcam_list    List webcams
    webcam_snap    Take a snapshot from the specified webcam
    webcam_stream  Play a video stream from the specified webcam


Android Commands
================

    Command           Description
    -------           -----------
    activity_start    Start an Android activity from a Uri string
    check_root        Check if device is rooted
    dump_calllog      Get call log
    dump_contacts     Get contacts list
    dump_sms          Get sms messages
    geolocate         Get current lat-long using geolocation
    interval_collect  Manage interval collection capabilities
    send_sms          Sends SMS from target session
    set_audio_mode    Set Ringer Mode
    sqlite_query      Query a SQLite database from storage
    wlan_geolocate    Get current lat-long using WLAN information

meterpreter >
```

1.5.5.Slow down the server :

A Large number of botnets are used to send so many TCP or UDP packets as request. Server usually not ableto handle these much requests on the same time,hence as result the server gets slow down. This helps the botmaster to find vulnerabilities.



MEDIUM

Acunetix Threat Level 2

One or more medium-severity type vulnerabilities have been discovered by the scanner. You should investigate each of these vulnerabilities to ensure they will not escalate to more severe problems.

Activity — Processing

Overall progress 58%

Scanning of www.asutoshcollege.in started — Jul 21, 2019 11:50:26 PM

Scan Duration	Requests	Avg. Response Time	Locations
9m 36s	3,390	254ms	149

Target Information

Address	www.asutoshcollege.in
Server	cloudflare

Latest Alerts — 0 7 5 1

HTML form without CSRF protection	Jul 21, 2019 11:56:38 PM
User credentials are sent in clear text	Jul 21, 2019 11:57:05 PM

1.6 Execution Channel of Botnet Attack:

1.6.1What made this attack so easy?

The answer is **Internet,** which became a necessity into our daily life.

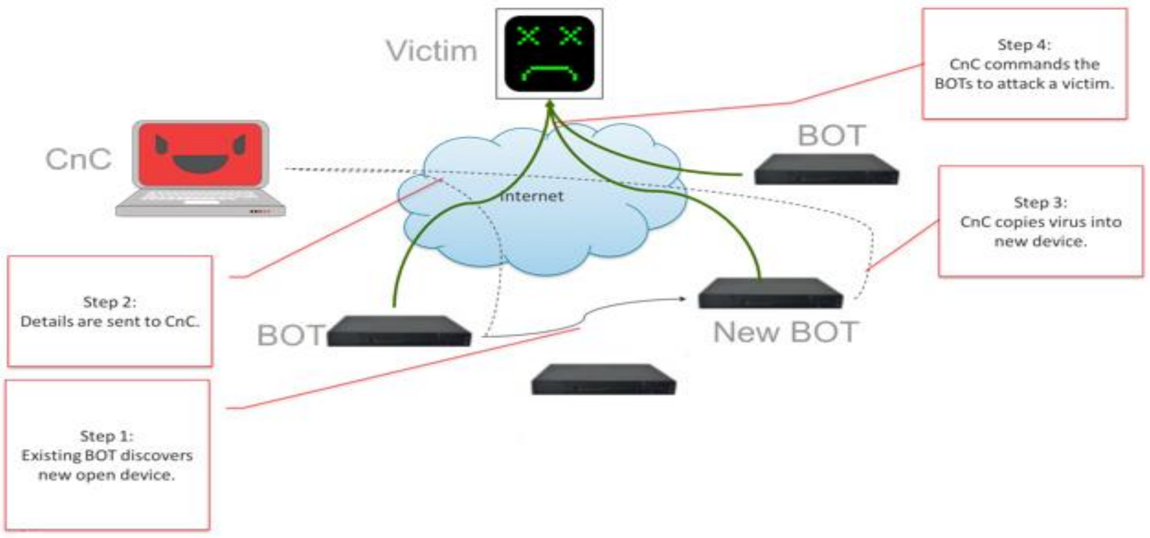

Victim
Step 4:
CnC commands the BOTs to attack a victim.
CnC
BOT
Internet
Step 3:
CnC copies virus into new device.
Step 2:
Details are sent to CnC.
BOT
New BOT
Step 1:
Existing BOT discovers new open device.

Figure 1 Mirai System

As we can see in the above picture, **internet is basically providing a channel** to establish the attack.Using internet it became very easy to identify the victim and gather information about the victim. After identifying the victim, using internet the original attack can be executed very easily.

1.6.2 Internet of Things (IoT):

The **Internet of things** (**IoT**) is the extension of Internet connectivity into physical devices and everyday objects. Embedded with electronics, Internet connectivity, and other forms of hardware (such as sensors), these devices can communicate and interact with others over the Internet, and they can be remotely monitored and controlled.

The definition of the Internet of things has evolved due to the convergence of multiple technologies, real-time analytics, machine learning, commodity sensors, and embedded systems. Traditional fields of embedded systems, wireless sensor networks, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of things. In the consumer market, IoT technology is most synonymous with products pertaining to the concept of the "**smart home**", covering devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smartphones and smart speakers.

Fig: Internet of Things

Literature Survey

As more cyber criminals are using botnet to perform sophisticated attacks, there isaneed to develop strong defense mechanism against it. Lots of research articles were available related to botnet detection. Some of the important papers are summarized below:

2.1 Botnet in DDoS Attacks: Trends and Challenges

NazrulHoque,presented comprehensive overview of DDoS attack. The paper also contains detail discussion of botnet architecture, tools developed using botnet architectures to perform **DDos attack**. This paper also summarized important issues and research challenges. In context to DDoS , there are two categories of botnet, DDoS attack using stationary bot-net and DDoS attack using mobile botnet. There are four reasons behind using botnet for performing DDoS attack:

1. Large number of zombie nodes allow generation of powerful flood attacks quickly
2. Difficulty to identify the main attacker
3. Ability to use protocols to bypass security mechanisms
4. Difficulty in real time detection

2.2 Botnet Detection Techniques: Review, Future Trends, and Issues:

Ahmad Karim, [2] presents a comprehensive review of the latest state-of-the-art techniques for botnet detection and _gures out the trends of previous and current research. The author also discuss future direction of botnet detection techniques. Researchers have developed many architectures and botnet detection taxonomies. The honeynets are used to collect information about bots for analysis such as finding botnet characteristics, finding tools used behind attack and motivation behind the attack. Intrusion detection system is a software application or hardware to monitor system services for malicious activities or policy violations and accordingly generate re-
ports.

Through the development of technology, every personal computer has the great amount of processing power (CPU, GPU) and bandwidth capacity. So, every personal computer which is joined into botnet is made botnet more powerful.

Works that require too much processing power can be done in distributed networks easily. In this type of network, the work is divided into sub works and assigned to the individual machine. The main purpose of botnet attacks is combining this multiple sources and building an incredibly powerful source. Combined sources can be

bandwidth or processing capacity. After creating botnet which has enough bots, attackers can use it in so many malicious purposes. Some examples are;

- Distributed Denial-of-Service Attacks (DDoS)
- Spamming
- Sniffing Traffic &Keylogging
- Infecting New Hosts
- Identity Theft
- Attacking IRC Chat Networks
- Hosting of Illegal Software
- Google AdSense Abuse & Advertisement Addons
- Click Fraud
- Manipulating Online Polls
- Remote Use of Computers
- Attacking Bank Computers (Atm or any others since they are also networked)
- Manipulating Games
- Exploiting Private Documents

Netflix, Slack, Imgur, HBO Now, PayPal, PlayStation Network, Yammer, Seamless, and many more services have also experienced interruptions in attack day. It is certain that Mirai is not only IoT botnet, we can witness another IoT botnet attacks in the near future.

2.3 Some popular Botnet:

Here’s a sample of some large scale IoT botnet attacks:

- **Linux.Aidra** – Also known as Linux.Lightaidra, this botnet was discovered in 2012 by security researchers at ATMA.ES. It was first noticed when researchers witnessed a large number of Telnet-based attacks on IoT devices.

- **Bashlite** – Also known as Gayfgt, Qbot, Lizkebab and Torlus, this IoT botnet was discovered in 2014 with the Bashlite source code published (with several variants) in 2015. Some variants of this botnet reached over 100,000 infected devices, serving as the precursor to Mirai (see below).

- **Mirai** – Gaining worldwide attention in 2016, the Mirai botnet consisted of record-breaking DDoS attacks on Krebs, OVH and Dyn. The botnet, which targeted closed-circuit television cameras, routers and DVRs, generated traffic volumes above 1Tbps. Featuring ten pre-defined attack vectors, this botnet took down the infrastructure of service providers and cloud scrubbers. Some of the vectors include GRE floods and Water Torture attacks.

- **Linux/IRCTelnet** – Discovered in 2016 by Malware Must Die, this IoT botnet targets routers, DVRs and IP cameras. It can send UDP and TCP floods along with other methods in both Ipv4 and Ipv6 protocols.

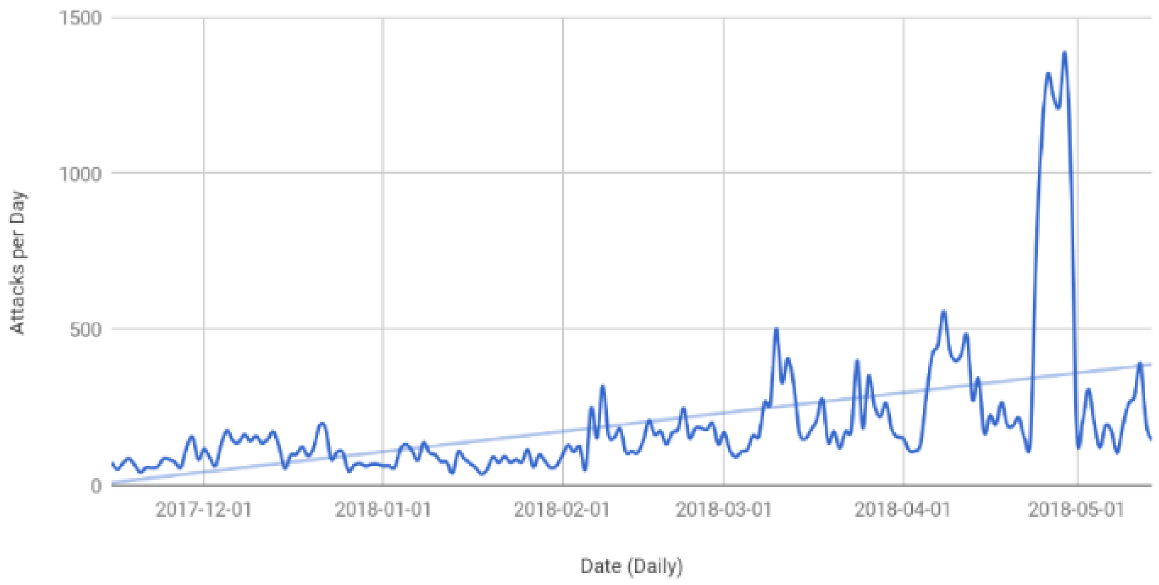


Fig: Botnet attack trend

So**, botnet detection and elimination of these botnets are very important and at the same time challenging tasks in the cyber security domain**. Big companies that have security-concern have made great effort to detect and eliminate botnets. For example, ZeuS botnet malware package that runs on Microsoft OS operated for over three years in just this matter, eventually leading to an estimated $70 million in stolen funds and the arrest of over a hundred individuals by the FBI in 2010. ZeuS was active, even when ZeuS creator was arrested. Microsoft which is suffered the most from this botnet, spent great effort to eliminate ZeuS. Eventually, in March 2012, Microsoft announced it had succeeded in shutting down the “majority” of C&C servers of ZeuS.

It has been observed that detecting a zombie machine is not an easy task. Even one of the zombie machines detected, what about the rest of the network? Detecting all network about a specific botnet is a tough task. So that, it is harder to recognize a botnet, if zombies are IoT devices.

Botnet detection is somewhat different from the detection mechanisms posed by other malware/anomaly detection systems

2.4 Botnet detection techniques :

Botnet detection techniques are classified into two broad categories**, IDSs** and**HoneyNets**.

A **honeynet** is used to collect information from bots for further analysis to measure the technology used, botnet characteristics, and the intensity of the attack. Moreover, the information collected from bots is used to discover the C&C system, unknown susceptibilities, techniques and tools used by the attacker, and the motivation of the attacker. A honeynet is used to collect bot-binaries which penetrate the botnets. However, intruders developed novel methods to overcome honeynet traps. The key component of honeynet trap is the honeywall, which is used to separate honeybots from the rest of the world.

Another botnet detection technique is based on **IDS**. IDS is a software application or hardware machine to monitor system services for malicious activities. IDS detection techniques are further classified as two types of approaches, signature-based, and anomaly-based.

In **Signature Based systems**, botnet signatures are used to give information about specific botnet behavior. But this type of techniques cannot detect unknown botnet whose signature is not created before.

Anomaly-based detection is a prominent research domain in botnet detection. The basic idea comes from analyzing several network traffic irregularities including traffic passing through unusual ports, high network latency, increased traffic volume, and system behavior indicating malicious activities in the network. Anomaly-based approaches are further divided into host- and network-based approaches. In host-based approaches, individual machines are monitored to find suspicious actions. Despite the importance of host-based monitoring, this approach is not scalable, as all machines are required to be fully equipped with effective monitoring tools.

As opposed to other techniques, **network-based approaches** analyze network traffic and gathering some meaning about botnets using machine learning techniques. Network monitoring tool examines network behavior based on different network characteristics, such as bandwidth, burst rate for botnet C&C evidence, and packet timing. It filters traffic that is unlikely to be part of botnet activity, classifies the remaining traffic into a group that is likely to be part of a botnet.

Machine learning techniques are used widely in both anomaly-based approaches; host based and network based. Some of the used machine learning techniques are Decision Trees, Neural Networks, Graph Theory, Artificial Immune System, Clustering Based techniques, Data mining Based Techniques, Correlation, Entropy etc.

Machine Learning Based Botnet Detection

3.1. Introduction to Machine Learning and Its Techniques

3.1.1. Introduction to Machine Learning:

Machine Learning is a concept which allows the machine to learn from examples and experience, and that too without being explicitly programmed. So instead of you writing the code, what you do is you feed data to the generic algorithm, and the algorithm/ machine builds the logic based on the given data

It enables the computers or the machines to make data-driven decisions rather than being explicitly programmed for carrying out a certain task. These programs or algorithms are designed in a way that they learn and improve over time when are exposed to new data. Machine Learning algorithm is trained using a training data set to create a model. When new input data is introduced to the ML algorithm, it makes a prediction on the basis of the model.

The prediction is evaluated for accuracy and if the accuracy is acceptable, the Machine Learning algorithm is deployed. If the accuracy is not acceptable, the Machine Learning algorithm is trained again and again with an augmented training data set.

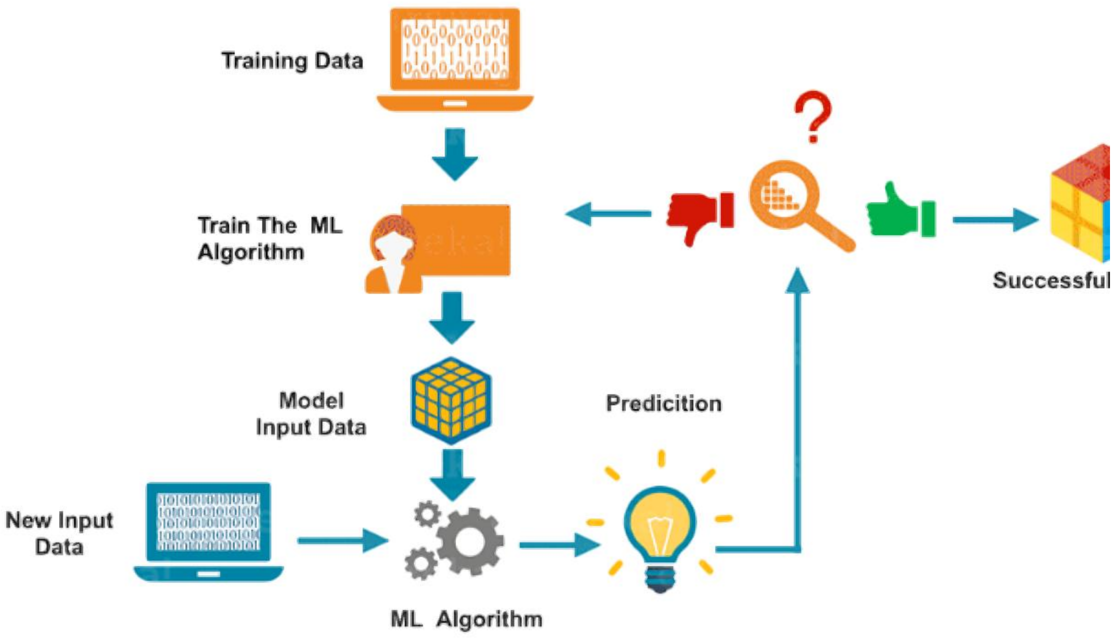
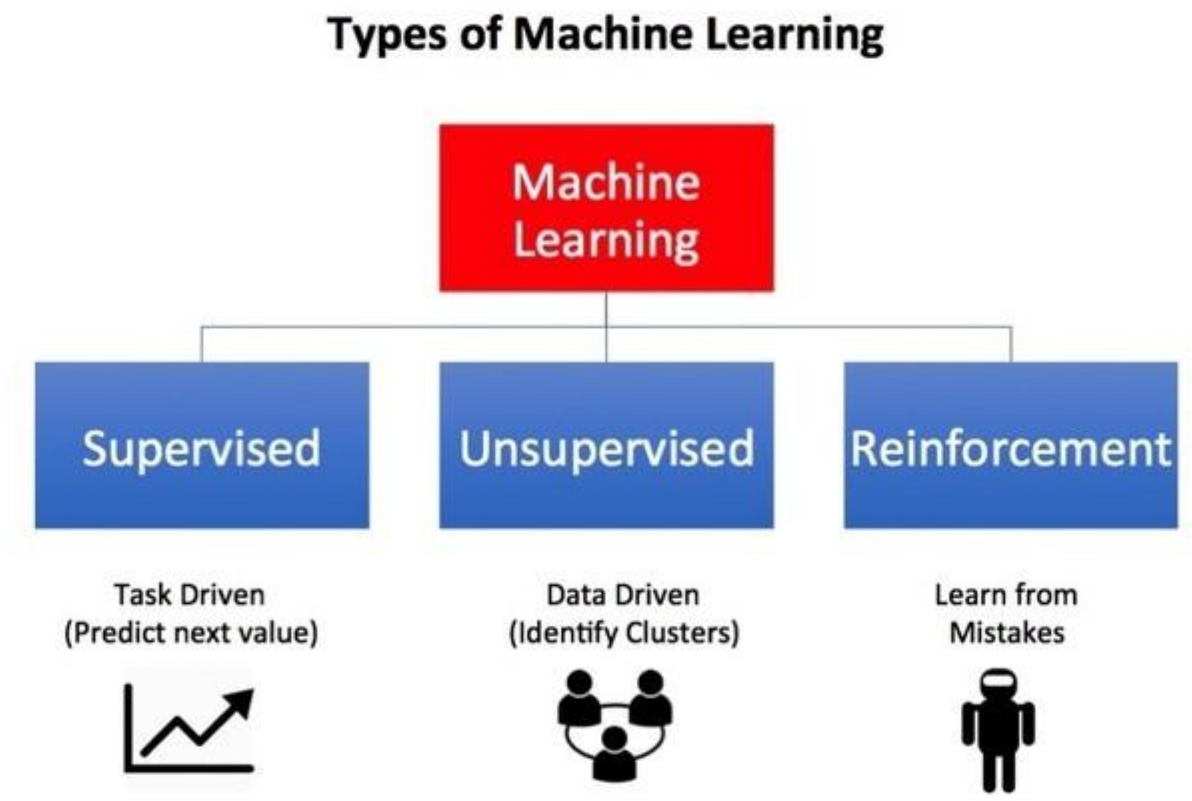


Fig: how machine learning works


Types of Machine Learning
Machine Learning
Supervised
Unsupervised
Reinforcement
Task Driven (Predict next value)
Data Driven (Identify Clusters)
Learn from Mistakes

3.1.2 Category of Machine Learning :

Machine learning is sub-categorized to three types:

- **Supervised Learning**
- **Unsupervised Learning**
- **Reinforcement Learning**

➢ Supervised Learning

Supervised Learning is the one, where you can consider the learning is guided by a teacher. We have a dataset which acts as a teacher and its role is to train the model or the machine. Once the model gets trained it can start making a prediction or decision when new data is given to it.

➢ Unsupervised Learning

The model learns through observation and finds structures in the data. Once the model is given a dataset, it automatically finds patterns and relationships in the dataset by creating clusters in it. What it cannot do is add labels to the cluster, like it cannot say this a group of apples or mangoes, but it will separate all the apples from mangoes.

Suppose we presented images of apples, bananas and mangoes to the model, so what it does, based on some patterns and relationships it creates clusters and divides the dataset into those clusters. Now if a new data is fed to the model, it adds it to one of the created clusters.

➢ Reinforcement Learning

It is the ability of an agent to interact with the environment and find out what is the best outcome. It follows the concept of hit and trial method. The agent is rewarded or penalized with a point for a correct or a wrong answer, and on the basis of the positive reward points gained the model trains itself. And again once trained it gets ready to predict the new data presented to it.

3.1.3.Common Supervised Machine Learning Techniques:

3.1.2.1. kNN :

kNN (k-Nearest Neighbor) is one of the simplest supervised machine learning algorithms.
The idea of kNN is that it will classify the new object based on its k nearest neighbors, where k is apredefined positive integer. kNN algorithm is comprehensively described in the following steps:

- **Step 1:** Determine the parameter value k.
- **Step 2:** Calculate the distance between the new object that needs to be classified with all objectsin the training dataset.
- **Step 3**: Arrange computed distances in the ascending order and identify k nearest neighborswiththe new object.
- **Step 4:** Take all the labels of the k neighbors selected above.
- **Step 5:** Based on the k labels that have been taken, the label that holds the majority will beassigned to the new object.

3.1.2.2. Decision tree:

Decision tree is a prediction model that is a mapping from observations of a thing, or aphenomenon to the conclusions about the objective value of things, or phenomena. Decision treecreates models that allow the classification of an object by creating a set of decision rules. These rulesare extracted based on the set of characteristics of the training data. In a decision tree, leaves representclasses and each child node in the tree and its branches represent a combination of features thatlead to classification. Thus, classifying an object will begin with checking the value of the root node,and then continuing downward under the tree branches corresponding to those values. This process isperformed repeatedly for each node until it cannot go any further and touch the leaf node. For the bestmodel, the decision to select the root node and sub-node while building the decision tree is based onthe**Information Gain** (IG).

3.1.2.3. Random forest:

Random forest (RF) is a member of the chain of decision tree algorithms. The idea of this algorithmis to create some decision trees. These decision trees will run and give independent results. The answerpredicted by the largest number of decisive trees will be chosen by the random forest .To ensurethat the decision trees are not the same, random forest randomly selects a subset of the characteristics of each node. The remaining parameters are used in the random forest as those in the decision trees.

3.1.2.4. C4.5:

C4.5, which is written based on the Iterative Dichotomiser 3 (ID3) algorithm, uses a training data sets to build a decision tree in a similar way as ID3, except C4.5 utilizes the concept of gain ratio to overcome the problem of biased information entropy. The attribute of the maximum gain ratio is selected as the node to split the tree.

3.2. Botnet Detection Model Based on Machine Learning

In a network-based botnet detection strategy, the malicious traffic is captured by observing the network traffic within different parameters, including network traffic behavior, traffic patterns, response time, network load, and link characteristics. Network-based approaches are further classified into two types, active monitoring, and passive monitoring.

3.2.1.Active monitoring:

In active monitoring botnet detection policy, new packets are injected to the network in order to detect malicious activities.

3.2.2Passive monitoring:

In passive monitoring, network traffic is sniffed when the data is passed through the medium. The network traffic is analyzed by applying different anomaly detection techniques. Passive monitoring techniques employing various application models include statistical approaches, graph theory, machine learning, correlation, entropy, stochastic model, decision trees, discrete time series, Fourier transformation, group-based analysis, data mining, clustering approach, neural networks, visualization, and a combination of these technologies.

BotProb (Tokhtabayev and Skormin, 2007) is considered an active monitoring strategy, which injects packets into the network payload for finding suspicious activity caused by humans or bots. As non-human bots usually transmit commands on a predetermined pattern, which corresponds to the cause and effect correlation between C&C and the bots. Such command and response architecture can easily determine the existence of bots because the response comes from the predetermined command behavior.

3.3.Support vector machine:

Support vector machine is another simple algorithm that every machine learning expert should have in his/her arsenal. Support vector machine is highly preferred by many as it produces significant accuracy with less computation power. Support Vector Machine, abbreviated as SVM can be used for both regression and classification tasks. But, it is widely used in classification objectives.

We will take help of support vector machine in order to solve BotNet detection problem.

Support Vector Machine

4.1.Introduction:

We have seen that different machine learning algorithms can be used in the detection of **BotNet.**Here we have selected Support Vector Machine or SVM as ityields superior performance. So first thing we need to know - What is SVM?

4.1.1. Definition:

Support Vector Machine or SVM is a Supervised Machine Learning algorithm which is used to analysis of data for classification and regression.

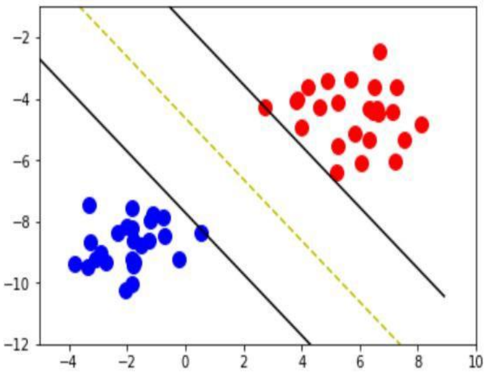
Classification is the problem of identifying one of a set of categories to which a new observation belongs , based on a set of training data containing observations whose category classification is already known. As an example we can take an email and then classify whether it is a spam email or not based on the knowledge learned from available datasets.

Regression is the problem of understanding the relationship between dependent (or criterion)and independent (or predictor) variables , more specifically how a dependent variable acts when one or more of its independent variables are varied, example: $Y = f(X, A)$. Here X,A are independent variables and Y is a dependent variable and the above notation shows a regression model that relates Y to be a function of X and A.

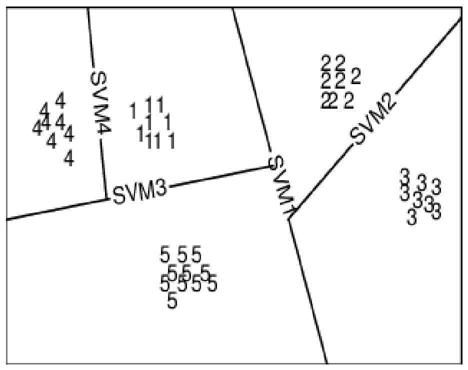
4.1.2. Properties of SVM:

1.SVM algorithms draw straight lines in case of single dimensional space and drawhyperplanes in multi-dimensional space in order to separate different groups.

2.Primarily SVM is a binary linear classifier as well as it can work as multiclass classifier. Following figure (a) shows SVM as a binary classifier and figure (b) shows SVM as a multiclass classifier.



Fig(a):Binary classifier



Fig(b):Multiclass classifier

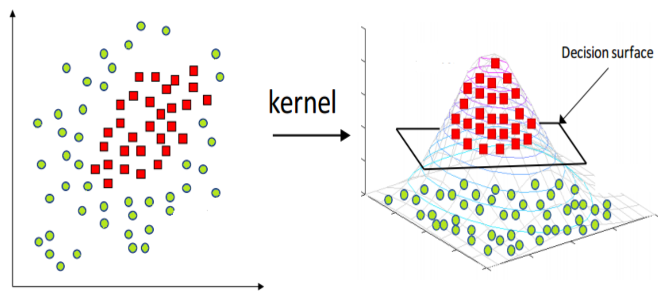
If we look at Fig(a), we can see that we have some blue and some red dots in a single dimensional space which are separated by a dotted straight line which iscalled the Margin and the other two straight lines are called support vectors. The rule to draw a Margin is that it should have maximum distance from the nearestobject in both of its categories.

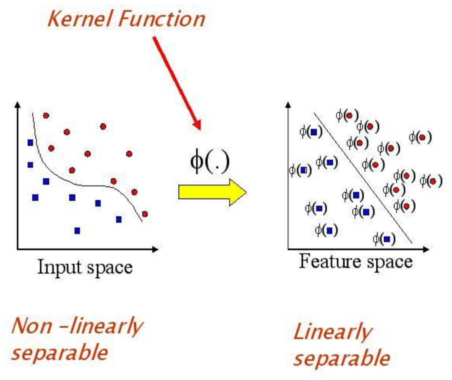
Similarly, if we observe Fig(b) we can see the digits from 1 to 5 and in order to separate them we need more than two dimensions here, i.e. we need multidimensional space.

4.2. Tuning parameters: Kernel, Regularization, Gamma and Margin.

4.2.1.Kernel:

Now these all data shown in the figures are linearly separable as we can see, but what if we have linearly inseparable data. For linearly inseparable data a special function known as **K**ernel function in needed. A kernel function is applied on the linearly inseparable data to make it separable data. Example:





A B

Fig:Kernel functions to transform linearly inseparable data into separable data

In the above figure we can see that we havesome red dots surrounded by some green dots in two dimensional space and we can not draw a straight line in order to separate them. If we apply an appropriate kernel function on this data to imagine it to a three dimensional space as shown in the figure, then we can easily have the red dots just above the green dots and we can draw a separation hyperplane between them (Decision Surface).

Similarly in B we can separate the mixed up of blue and red dots easily after applying a proper kernel function to it.

The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra. This is where the kernel plays role.

4.2.1.1. Linear Kernel:

For **linear kernel** the equation for prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

f(x) = B(0) + sum(ai * (x,xi))

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B0 and ai (for each input) must be estimated from the training data by the learning algorithm.

4.2.1.2. Polynomial Kernel:

It is popular in image processing.
Equation is:

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$$

where d is the degree of the polynomial.

4.2.1.3.Gaussian kernel:

It is a general-purpose kernel; used when there is no prior knowledge about the data.
Equation is:

$$k(x, y) = \exp\left(-\frac{||x - y||^2}{2\sigma^2}\right)$$

4.2.1.4.Gaussian radial basis function (RBF):

It is a general-purpose kernel; used when there is no prior knowledge about the data.
Equation is:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2) \text{for,} \qquad \gamma > 0$$

Polynomial and exponential kernels calculates separation line in higher dimension. This is called **kernel trick**

4.2.2. Regularization

The Regularization parameter (often termed as C parameter in python's sklearn library) tells the SVM optimization how much you want to avoid misclassifying each training example.

For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

The images below are example of two different regularization parameter. Left one has some misclassification due to lower regularization value. Higher value leads to results like correct one.

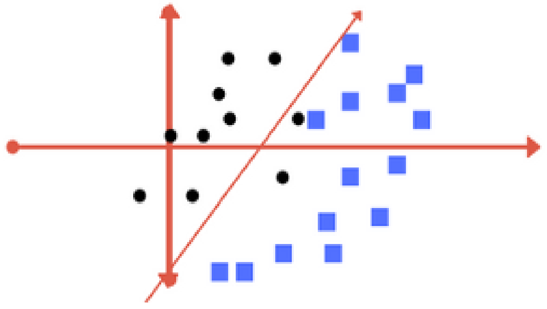


Fig: low regularization value,

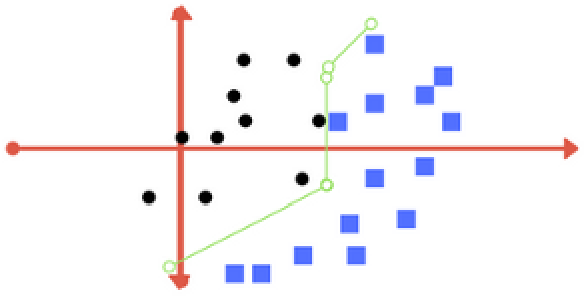
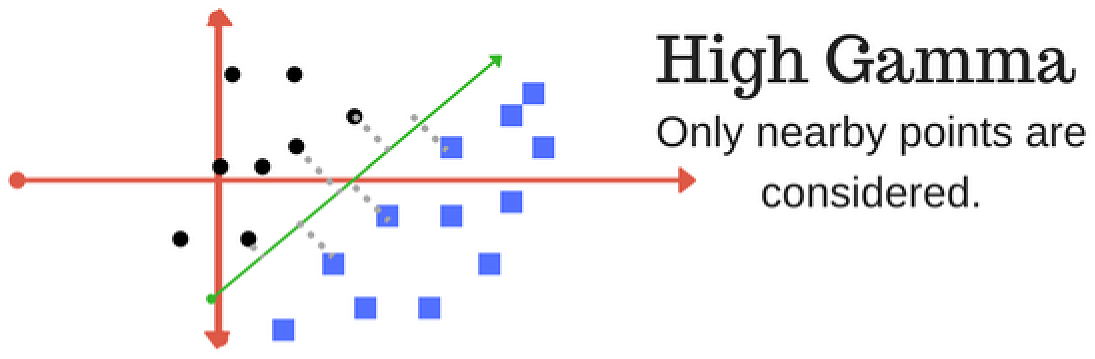
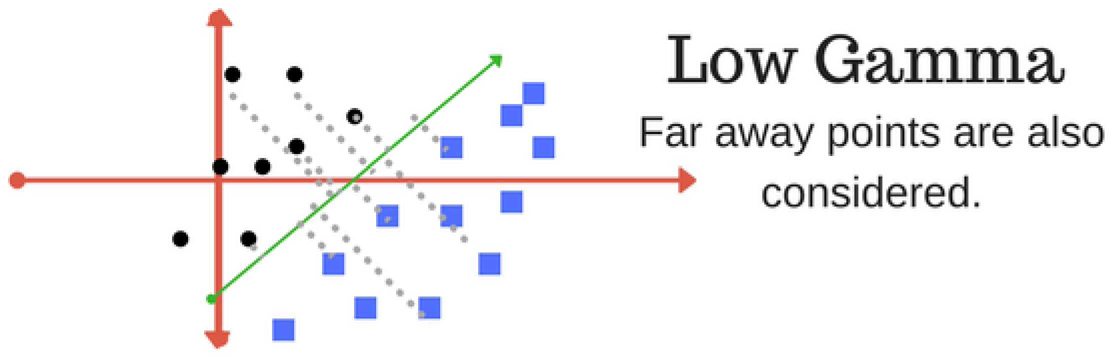


Fig: high regularization value

4.2.3. Gamma

The **gamma parameter** defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. In other words, with low gamma, points far away from plausible separation line are considered in calculation for the separation line. Where as high gamma means the points close to plausible line are considered in calculation.



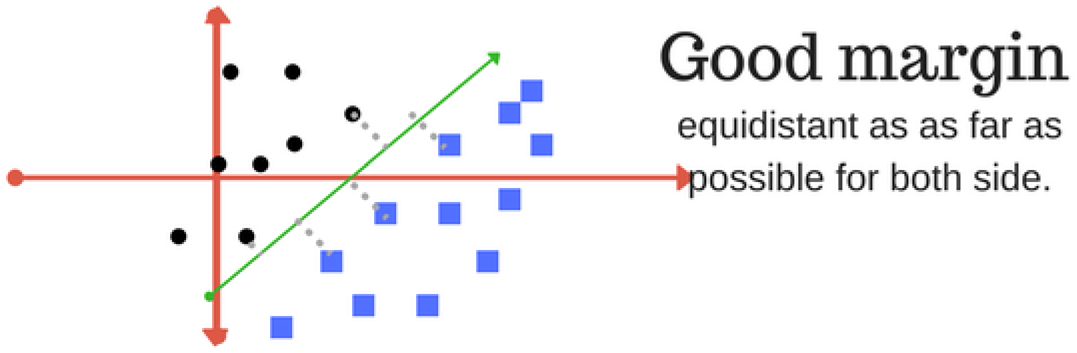


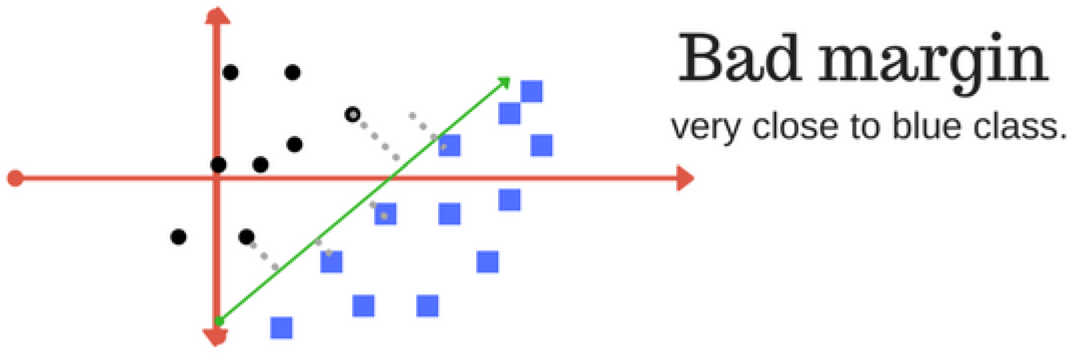
4.2.4. Margin

And finally last but very importrant characteristic of SVM classifier.

A margin is a separation of line to the closest class points.

A **good margin** is one where this separation is larger for both the classes. Images below gives to visual example of good and bad margin. A good margin allows the points to be in their respective classes without crossing to other class.





4.3. Pros and Cons associated with SVM :

- **Pros:**

    - It works really well with clear margin of separation
    - It is effective in high dimensional spaces.
    - It is effective in cases where number of dimensions is greater than the number of samples.
    - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

- **Cons:**

    - It doesn't perform well, when we have large data set because the required training time is higher
    - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
    - SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is related SVC method of Python scikit-learn library.

Implementation:

5.1.Dataset:

Here, we have used UCI dataset to detect the botnet attack. The details of the dataset is given below:

5.1.1.AboutUCI Data Set:

The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. The archive was created as an ftp archive in 1987 by David Aha and fellow graduate students at UC Irvine. Since that time, it has been widely used by students, educators, and researchers all over the world as a primary source of machine learning data sets. As an indication of the impact of the archive, it has been cited over 1000 times, making it one of the top 100 most cited "papers" in all of computer science. The current version of the web site was designed in 2007 by Arthur Asuncion and David Newman, and this project is in collaboration with Rexa.info at the University of Massachusetts Amherst. Funding support from the National Science Foundation is gratefully acknowledged.

5.1.2.Data Set Information:

(a) Attribute being predicted:

-- Originally we aimed at distinguishing between benign and malicious traffic data by means of anomaly detection techniques.

-- However, as the malicious data can be divided into 10 attacks carried by 2 botnets, the dataset can also be used for multi-class classification: 10 classes of attacks, plus 1 class of 'benign'.

(b) The study's results:

-- For each of the 9 IoT devices we trained and optimized a deep autoencoder on 2/3 of its benign data (i.e., the training set of each device). This was done to capture normal network traffic patterns.

-- The test data of each device comprised of the remaining 1/3 of benign data plus all the malicious data. On each test set we applied the respective trained (deep) autoencoder as an anomaly detector. The detection of anomalies (i.e., the cyberattacks launched from each of the above IoT devices) concluded with 100% TPR.

5.1.3.Attribute Information:

The following describes each of the features headers:

*** Stream aggregation:**
H: Stats summarizing the recent traffic from this packet's host (IP)
HH: Stats summarizing the recent traffic going from this packet's host (IP) to the packet's destination host.
HpHp: Stats summarizing the recent traffic going from this packet's host+port (IP) to the packet's destination host+port. Example 192.168.4.2:1242 -> 192.168.4.12:80
HH_jit: Stats summarizing the jitter of the traffic going from this packet's host (IP) to the packet's destination host.

* **Time-frame** (The decay factor Lambda used in the damped window):
How much recent history of the stream is capture in these statistics
L5, L3, L1, ...

*** The statistics extracted from the packet stream:**
weight: The weight of the stream (can be viewed as the number of items observed in recent history)
mean: ...
std: ...
radius: The root squared sum of the two streams' variances
magnitude: The root squared sum of the two streams' means
cov: an approximated covariance between two streams
pcc: an approximated covariance between two streams

5.2.Explanation:

5.2.1.Importing the libraries:

```
importnumpy as np
importmatplotlib.pyplot as plt
import pandas as pd
fromsklearn.decomposition import PCA as PCA
```

5.2.2.Model Training :

From our dataset, let’s create the target and predictor matrix

- “y” = Is the feature we are trying to predict (Output). In this case we are trying to predict if our “target” is infected (Malignant) or not (Benign). i.e. we are going to use the “target” feature here.
- “X” = The predictors which are the remaining columns (mean radius, mean texture, mean perimeter, mean area, mean smoothness, etc.)

5.2.3.Importing our botnet dataset:

```
dataset = pd.read_csv('combo.csv')
X = dataset.iloc[1:30, 1:114].values
Y = dataset.iloc[1:30, 77].values
dataset.head()
print("Botnet data set dimensions : {}".format(dataset.shape))
dataset.isnull().sum()
dataset.isna().sum()
print("Success 1")
```

5.2.4. Encoding categorical data values:

Categorical data are variables that contain label values rather than numeric values.The number of possible values is often limited to a fixed set.

We will use Label Encoder to label the categorical data. Label Encoder is the part of SciKit Learn library in Python and used to convert categorical data, or text data, into numbers, which our predictive models can better understand.

```
#Encoding categorical data values
fromsklearn.preprocessing import LabelEncoder
labelencoder_Y = LabelEncoder()
Y = labelencoder_Y.fit_transform(Y)
print("Success2")
```

5.2.5.Creation of the training and testing data:

Now that we've assigned values to our “X” and “y”, the next step is to import the python library that will help us split our dataset into training and testing data.
The data we use is usually split into training data and test data. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. We have the test dataset (or subset) in order to test our model's prediction on this subset.

We will do this using SciKit-Learn library in Python using the train_test_split method.

- Training data = the subset of our data used to train our model.
- Testing data = the subset of our data that the model hasn't seen before (We will be using this dataset to test the performance of our model).
- Splitting our data using 75% for training and the remaining 20% for testing.

```
# Splitting the dataset into the Training set and Test set
fromsklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25, random_state = 0)
print("Success3")
```

5.2.6. Feature Scaling:

Most of the times, your dataset will contain features highly varying in magnitudes, units and range. But since, most of the machine learning algorithms use Eucledian distance between two data points in their computations. We need to bring all features to the same level of magnitudes. This can be achieved by scaling. This means that you're transforming your data so that it fits within a specific scale, like 0–100 or 0–1.

We will use StandardScaler method from SciKit-Learn library

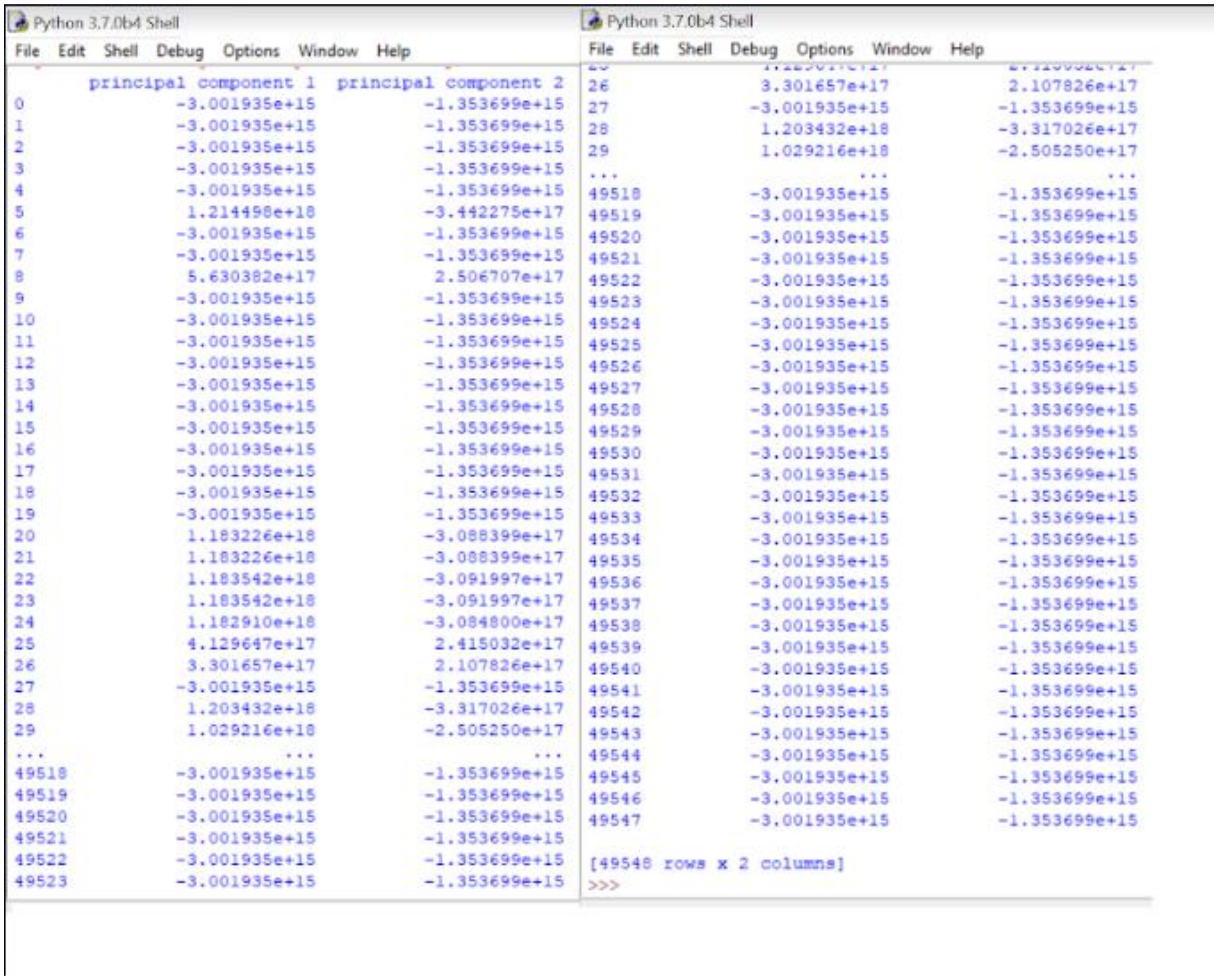
```
#Feature Scaling
fromsklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
print("Success4")
```

5.2.7.Principal Component Analysis:

Principal Component Analysis, or PCA for short, is a method for reducing the dimensionality of data.

It can be thought of as a projection method where data with m-columns (features) is projected into a subspace with m or fewer columns, whilst retaining the essence of the original data.
The PCA method can be described and implemented using the tools of linear algebra.

```
pca = PCA(n_components=2)
pca.fit(X_train)
X_t_train = pca.transform(X_train)
print("Training set size:",X_t_train.shape)
X_t_test = pca.transform(X_test)
print("Testing set size:",X_t_test.shape)
```



```
       principal component 1  principal component 2
0              -3.001935e+15          -1.353699e+15
1              -3.001935e+15          -1.353699e+15
2              -3.001935e+15          -1.353699e+15
3              -3.001935e+15          -1.353699e+15
4              -3.001935e+15          -1.353699e+15
5               1.214498e+18          -3.442275e+17
6              -3.001935e+15          -1.353699e+15
7              -3.001935e+15          -1.353699e+15
8               5.630382e+17           2.506707e+17
9              -3.001935e+15          -1.353699e+15
10             -3.001935e+15          -1.353699e+15
11             -3.001935e+15          -1.353699e+15
12             -3.001935e+15          -1.353699e+15
13             -3.001935e+15          -1.353699e+15
14             -3.001935e+15          -1.353699e+15
15             -3.001935e+15          -1.353699e+15
16             -3.001935e+15          -1.353699e+15
17             -3.001935e+15          -1.353699e+15
18             -3.001935e+15          -1.353699e+15
19             -3.001935e+15          -1.353699e+15
20              1.183226e+18          -3.088399e+17
21              1.183226e+18          -3.088399e+17
22              1.183542e+18          -3.091997e+17
23              1.183542e+18          -3.091997e+17
24              1.182910e+18          -3.084800e+17
25              4.129647e+17           2.415032e+17
26              3.301657e+17           2.107826e+17
27             -3.001935e+15          -1.353699e+15
28              1.203432e+18          -3.317026e+17
29              1.029216e+18          -2.505250e+17
...                      ...                    ...
49518          -3.001935e+15          -1.353699e+15
49519          -3.001935e+15          -1.353699e+15
49520          -3.001935e+15          -1.353699e+15
49521          -3.001935e+15          -1.353699e+15
49522          -3.001935e+15          -1.353699e+15
49523          -3.001935e+15          -1.353699e+15
```

```
26              3.301657e+17           2.107826e+17
27             -3.001935e+15          -1.353699e+15
28              1.203432e+18          -3.317026e+17
29              1.029216e+18          -2.505250e+17
...                      ...                    ...
49518          -3.001935e+15          -1.353699e+15
49519          -3.001935e+15          -1.353699e+15
49520          -3.001935e+15          -1.353699e+15
49521          -3.001935e+15          -1.353699e+15
49522          -3.001935e+15          -1.353699e+15
49523          -3.001935e+15          -1.353699e+15
49524          -3.001935e+15          -1.353699e+15
49525          -3.001935e+15          -1.353699e+15
49526          -3.001935e+15          -1.353699e+15
49527          -3.001935e+15          -1.353699e+15
49528          -3.001935e+15          -1.353699e+15
49529          -3.001935e+15          -1.353699e+15
49530          -3.001935e+15          -1.353699e+15
49531          -3.001935e+15          -1.353699e+15
49532          -3.001935e+15          -1.353699e+15
49533          -3.001935e+15          -1.353699e+15
49534          -3.001935e+15          -1.353699e+15
49535          -3.001935e+15          -1.353699e+15
49536          -3.001935e+15          -1.353699e+15
49537          -3.001935e+15          -1.353699e+15
49538          -3.001935e+15          -1.353699e+15
49539          -3.001935e+15          -1.353699e+15
49540          -3.001935e+15          -1.353699e+15
49541          -3.001935e+15          -1.353699e+15
49542          -3.001935e+15          -1.353699e+15
49543          -3.001935e+15          -1.353699e+15
49544          -3.001935e+15          -1.353699e+15
49545          -3.001935e+15          -1.353699e+15
49546          -3.001935e+15          -1.353699e+15
49547          -3.001935e+15          -1.353699e+15

[49548 rows x 2 columns]
>>>
```

Fig: Dataset after PCA transformation

5.2.8.Import Support Vector Machine (SVM) Model:

Model Selection:

This is the most exciting phase in Applying Machine Learning to any Dataset. It is also known as Algorithm selection for Predicting the best results.Usually Data Scientists use different kinds of Machine Learning algorithms to the large data sets. But, at high level all those different algorithms can be classified in two groups : supervised learning and unsupervised learning.
Without wasting much time, I would just give a brief overview about these two types of learnings.
Supervised learning : Supervised learning is a type of system in which both input and desired output data are provided. Input and output data are labelled for classification to provide a learning basis for future data processing. Supervised learning problems can be further grouped into **Regression** and **Classification** problems.

A **regression** problem is when the output variable is a real or continuous value, such as “salary” or “weight”.

A **classification** problem is when the output variable is a category like filtering emails “spam” or “not spam”

Unsupervised Learning :Unsupervised learning is the algorithmusing information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance.
In our dataset we have the outcome variable or Dependent variable i.e Y having only two set of values, either M (Malign) or B(Benign). So we will use Classification algorithm of supervised learning.

Lets start applying the algorithms :
We will use sklearn library to import all the methods of classification algorithms.

```
#Using SVC method of svm class to use Kernel SVM Algorithm
fromsklearn.svm import SVC

classifier = SVC(kernel = 'rbf', random_state = 0)
classifier.fit(X_t_train, Y_train)
print("Training done..")
```

5.2.9. Confusion Matrix:

To check the accuracy we need to import confusion_matrix method of metrics class. The confusion matrix is a way of tabulating the number of mis-classifications, i.e., the number of predicted classes which ended up in a wrong classification bin based on the true classes.

```
fromsklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test, Y_pred)
```

We will use Classification Accuracy method to find the accuracy of our models. Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

To check the correct prediction we have to check confusion matrix object and add the predicted results diagonally which will be number of correct prediction and then divide by total number of predictions.

5.2.10.Visualization of results:

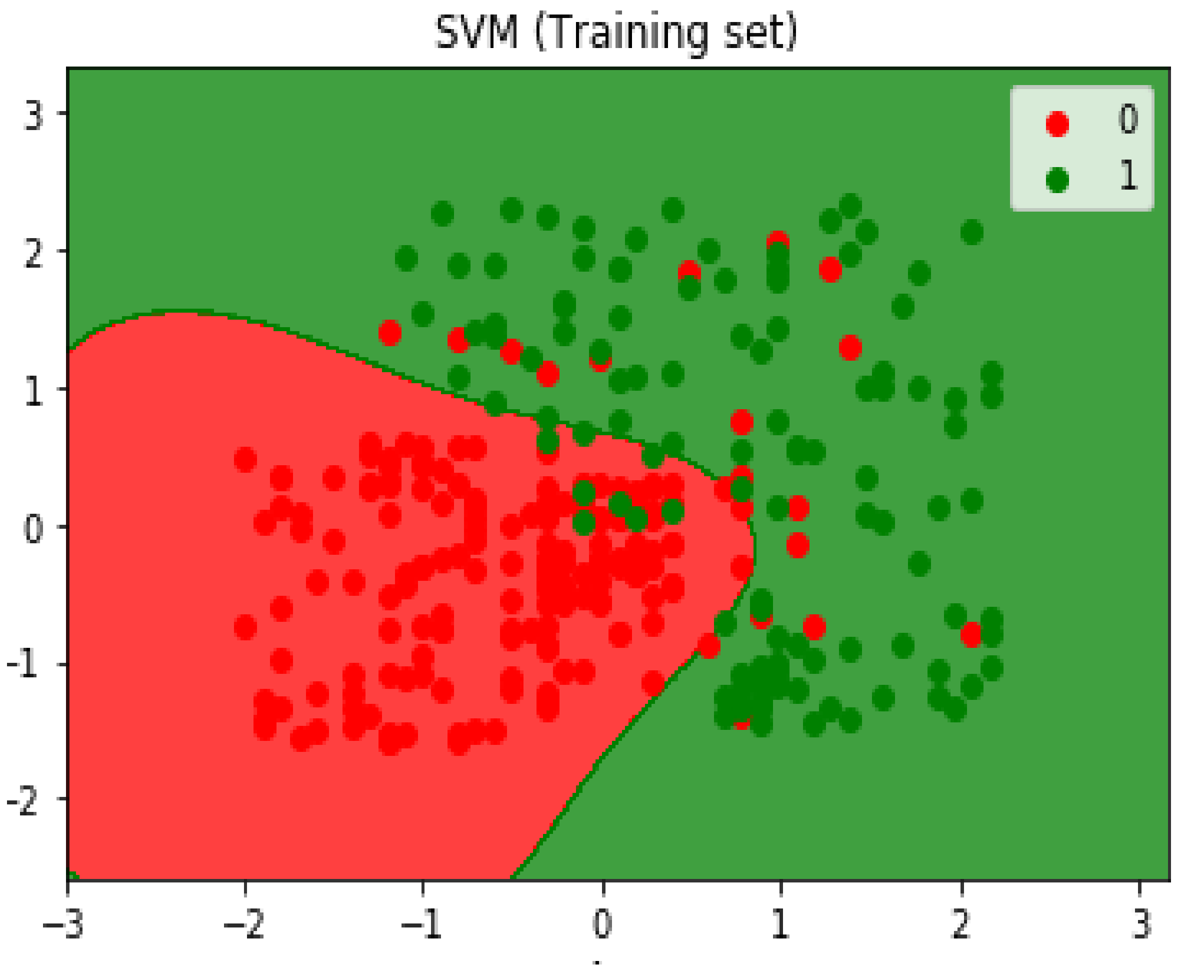
# Visualising the Training set results for 'rbf' kernel

```
frommatplotlib.colors import ListedColormap
X_set, y_set = X_t_train, Y_train
print("copied")

X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step
= 0.01),
np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
print("predicting")

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
print("prediction done...")
```

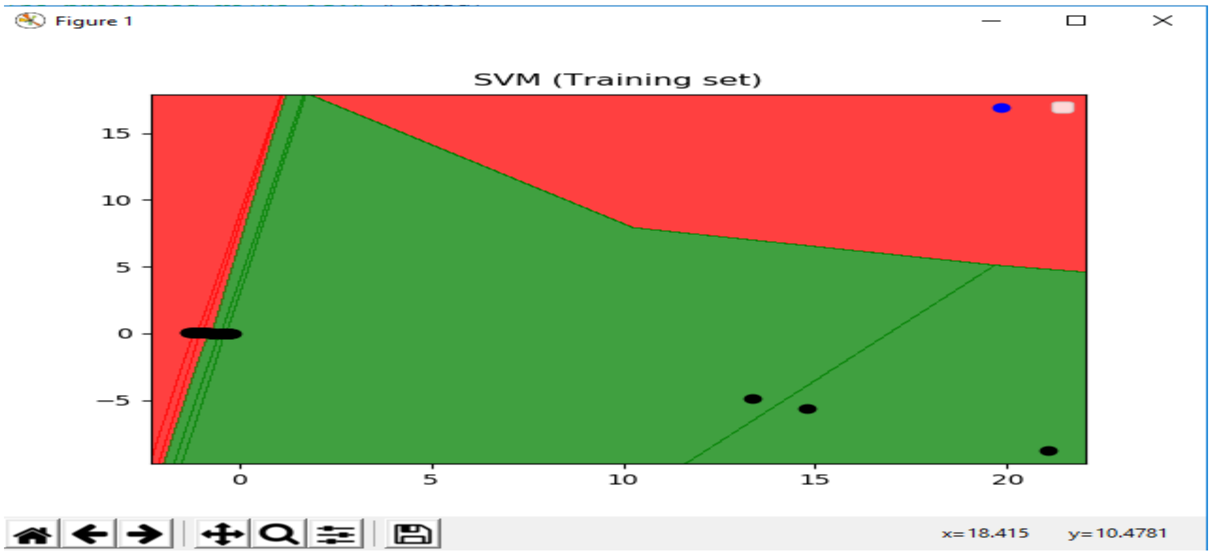
```
for i, j in enumerate(np.unique(y_set)):
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('blue', 'black'))(i))
plt.title('SVM (Training set)')
plt.legend()
plt.show()

frommatplotlib.colors import ListedColormap
X_set, y_set = X_t_test, Y_test
print("copied")

X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step
= 0.01),
np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
print("predicting")
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
alpha = 0.75, cmap = ListedColormap(('red', 'green')))

print("prediction done...")

plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())

print("limit set")
for i, j in enumerate(np.unique(y_set)):
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('blue', 'black'))(i))
plt.title('SVM (Test set)')
plt.legend()
plt.show()
```

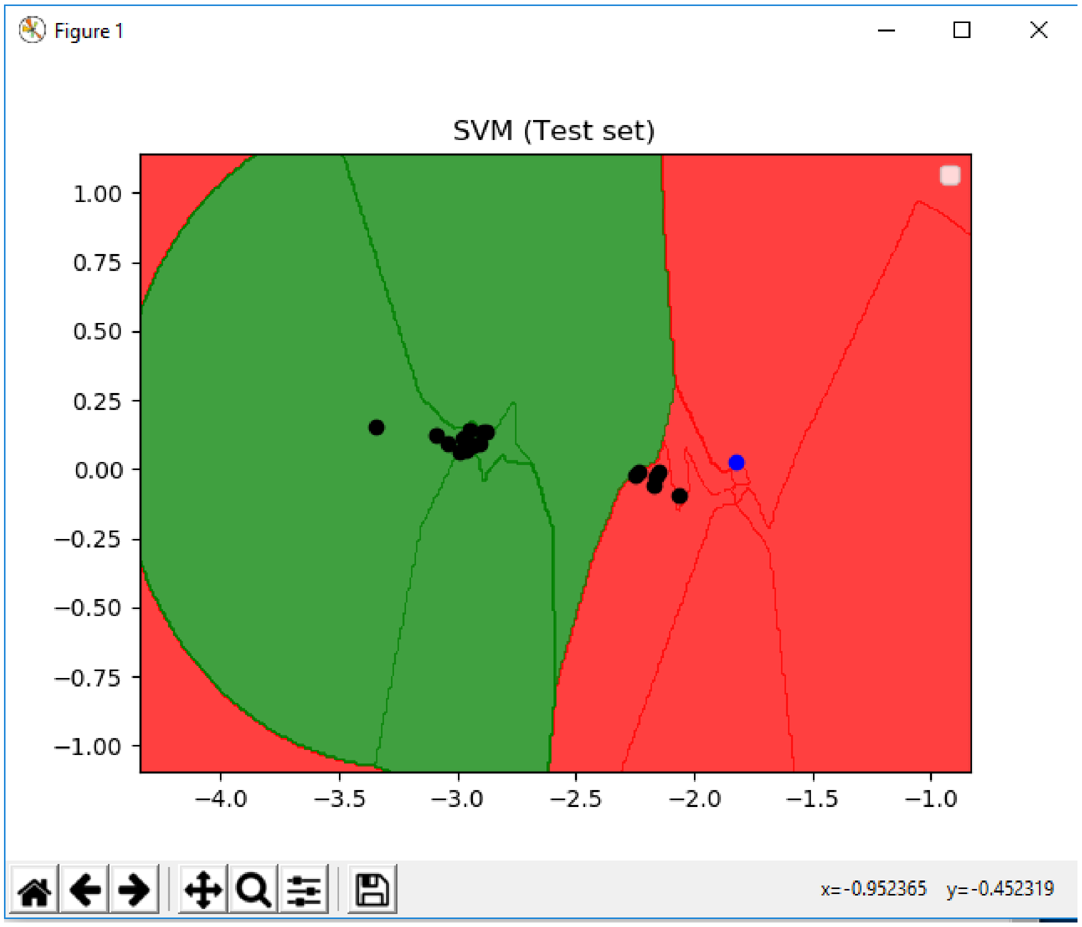
5.3.Result:

5.3.1.Predicted Output:



5.3.2.Experimental Results:





5.3.3.Discussions on Results:

- Blue dots are expected in Red region, and Black dots are expected in Green region.
- Any blue dot in Green region or black dot in Red region is Botnet
- UCI Dataset contains 59717 rows and 115columns, but due to hardware limitation we can take maximum 500 rows.
- Less data yields less accuracy.
- PCA function has been used for visualization purpose.

Scope for Future Research:

6.1.Inferences:

Currently, Support vector machines (SVM) have been successfully applied in many areas such as face detection, hand-written digit recognition and so on. However, although on several real-world problems, SVM ensembles are reported to give improvements over single SVM, but few works showed also negative experimental results about SVM ensembles. In particular, Evgeniou et al experimentally found that leave-one-out error bounds for kernel machines ensembles are higher than the equivalent ones for single machines, but they showed that with accurate parameters tuning single SVM and ensembles of SVM perform similarly. A set of experiments were conducted to prove the efficiency and effectively of the proposed approaches. A number of observations are drawn from the results reported.

• **The proposed technique**, Comparative cross validation has shown smaller improvement in the run time whereas the estimated accuracy is found to be relatively low.(may be due to high bias and variance)

• **The Bagged ensembles** outperform the single classifiers in the important aspect of error rate and accuracy for all the three application domains.

• **Pruned bagged classifiers** even outperform the bagged classifiers in achieving better generalization performance and faster execution.

• Within **text mining approaches**, SVM provide better results in terms of runtime, error rate and accuracy followed by MLP, RBF and k-NN for intrusion detection, direct marketing and signature verification.

6.2.Scope for Future Research :

Traditional classification algorithms assume that the whole training data can fit into the main memory. As automatic data collection becomes a daily practice in many businesses, large volumes of data that exceed the memory capacity become available to the learning systems. Scalable classification algorithms become essential.

Previously, the study of classification techniques focused on exploring various learning mechanisms to improve the accuracy on unseen examples. However, recent study on imbalanced data sets has shown that accuracy is not an appropriate measure to evaluate the classification performance when the data set is extremely unbalanced, in which almost all the examples belong to one or more, larger classes and far fewer examples belong to a smaller, usually more interesting class. Since many real world data sets are imbalanced, there has been a trend toward adjusting existing classification algorithms to better identify examples in the rare class. Notwithstanding the development of new techniques and advances in the theory of ensembles, the following future trends of research are identified.

• **Selective combination of ensemble members:** As noted, an effective ensemble requires that each member should be both diverse and 112 accurate. One approach to enable this is to use hill climbing techniques to selectively adjust the members of the ensemble to combine by calculating metrics based on accuracy or diversity (Opitz&Shavlik, 1996b, Carney & Cunningham, 2000) or more simply to selectively reduce the set of ensemble members (Zhou et al., 2002) Potentially this can create a more competent ensemble but also a less complex one. Further work in this area is expected particularly in trying to produce small ensembles of transparent and interpretable models such as regression trees.

• **Computationally Efficient methods**: If ensemble techniques are going to be applied to large-scale data mining methods, they need to be computationally efficient. A potential direction of research is the parallelization of ensemble methods-intrinsically such methods as Bagging are easily parallelizable.

• **Heterogeneous ensemble learning**: Most ensemble method use the same learning algorithm to generate base models-there is scope for techniques that effectively combine more than one base learning algorithm.

•**Prevention better than Detection** :If, somehow we can manage to detect the attack at the time of execution, then may be it can be prevented.While the attack is taking place ,the data will start to change,by seeing the changing nature of the data we need to detect the botnet attack.After detection, we can use different methods to prevent it from complete execution.

References:

- Chityala, R. and Pudipeddi, S., 2014. *Image processing and acquisition using Python*. Chapman and Hall/CRC.
- Jeong, O.R., Kim, C., Kim, W. and So, J., 2011. Botnets: threats and responses. *International Journal of Web Information Systems*, *7*(1), pp.6-17.
- Schaathun, H.G., 2012. *Machine learning in image steganalysis*. Wiley-IEEE Press.
- Hoang, X. and Nguyen, Q., 2018. Botnet detection based on machine learning techniques using DNS query data. *Future Internet*, *10*(5), p.43.
- Karim, A., Salleh, R.B., Shiraz, M., Shah, S.A.A., Awan, I. and Anuar, N.B., 2014. Botnet detection techniques: review, future trends, and issues. *Journal of Zhejiang University SCIENCE C*, *15*(11), pp.943-983
- Miller, S. and Busby-Earle, C., 2016, December. The role of machine learning in botnet detection. In *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)* (pp. 359-364). IEEE.
- https://www.normshield.com/machine-learning-in-cyber-security-domain-9-botnet-detection/
- https://www.kaggle.com/farhanmd29/svm-model-for-social-network-ads
- https://en.wikipedia.org/wiki/Botnet
- https://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT (dataset purpose)
- https://data-flair.training/blogs/svm-kernel-functions/

Appendix :

Program code:

```
#importing the libraries
importnumpy as np
importmatplotlib.pyplot as plt
import pandas as pd
fromsklearn.decomposition import PCA as PCA

#importing our botnet dataset
dataset = pd.read_csv('combo.csv')
X = dataset.iloc[1:30, 1:30].values
Y = dataset.iloc[1:30, 10].values
dataset.head()
print("Botnet data set dimensions : {}".format(dataset.shape))


dataset.isnull().sum()
dataset.isna().sum()
print("Success 1")

#Encoding categorical data values
fromsklearn.preprocessing import LabelEncoder
labelencoder_Y = LabelEncoder()
Y = labelencoder_Y.fit_transform(Y)
print("Success2")

# Splitting the dataset into the Training set and Test set
fromsklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25, random_state =
0)
print("Success3")


#Feature Scaling
fromsklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
print("Success4")

pca = PCA(n_components=2)# adjust yourself
pca.fit(X_train)
```

```
X_t_train = pca.transform(X_train)
print("Training set size:",X_t_train.shape)
X_t_test = pca.transform(X_test)
print("Testing set size:",X_t_test.shape)
```

#Using SVC method of svm class to use Kernel SVM Algorithm

```
fromsklearn.svm import SVC
classifier = SVC(kernel = 'rbf', random_state = 0)
classifier.fit(X_t_train, Y_train)
print("Training done..")
Y_pred = classifier.predict(X_t_test)
#print("The predicted values are:",Y_pred)
```

# Visualising the Training set results for 'rbf' kernel

```
frommatplotlib.colors import ListedColormap
X_set, y_set = X_t_train, Y_train
print("copied")

X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() +
1, step = 0.01),
np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
print("predicting")

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),
X2.ravel()]).T).reshape(X1.shape),
alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
print("prediction done...")

for i, j in enumerate(np.unique(y_set)):
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('blue', 'black'))(i))
plt.title('SVM (Training set)')
plt.legend()
plt.show()

frommatplotlib.colors import ListedColormap
X_set, y_set = X_t_test, Y_test
print("copied")



X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() +
1, step = 0.01),
np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
print("predicting")
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),
X2.ravel()]).T).reshape(X1.shape),
alpha = 0.75, cmap = ListedColormap(('red', 'green')))

print("prediction done...")

plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())

print("limit set")
for i, j in enumerate(np.unique(y_set)):
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('blue', 'black'))(i))
plt.title('SVM (Test set)')
plt.legend()
plt.show()
```