DETECTION OF PHISHING WEBSITES

A Project Report

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

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted By

M.SATHIESH	**166R1A0516**
M.SIREESHA	**166R1A0517**
M.BHUVANESWARI DEVI	**166R1A0521**
P.NAGABRAMHANANDAM	**166R1A0528**

Under the esteemed guidance of

Dr.SK.MEERA SHARIF, Ph.D.

Associate Professor& HOD-CSE Dept.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GIET COLLEGE OF ENGINEERING

(Approved by A.I.C.T.E, Affiliated to JNTUK, Kakinada)

NH-16, Velugubanda, Rajamahendravaram, E.G.Dist, A.P, Pin-533296

(2019-2020)

GIET COLLEGE OF ENGINEERING

(Approved by A.I.C.T.E, Affiliated to JNTUK, Kakinada)

NH-16, Velugubanda, Rajamahendravaram, E.G.Dist, A.P, Pin-533296

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**DETECTION OF PHISHING WEBSITES**", is being carried by **M.SATHIESH (166R1A0516), M.SIREESHA (166R1A0517), M.BHUVANESWARI DEVI (166R1A0521), P.NAGABRAMHANANDAM (166R1A0528),**in a partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING,** is a record of bonafide work carried out by them under my guidance and supervision.

The results embodied in this thesis have not been submitted to any other university or institution for the award of any degree or diploma

Internal Guide

Dr.SK. MEERA SHARIF, Ph.D.

Associate Professor& HOD-CSE Dept.

Head of the Department

Dr.SK. MEERA SHARIF, Ph.D.

Associate Professor& HOD-CSE Dept.

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We take immense pleasure in conveying our thanks and deep sense of gratitude to our Guide **Dr.SK.MEERA SHARIF** for his exhilarating supervision, timely suggestions and encouragement during all phases of this work.

We wish to thank with deep sense of acknowledge to the management of GIET College of Engineering and to our principal, **Dr.T.VAMSEE KIRAN** for extending all facilities.

We wish to express our sincere thanks to **Dr.SK.MEERA SHARIF** Head of the Department of Computer Science and Engineering and CSE Department Coordinator also for his excellent encouragement in course of this work.

We wish to express our sincere thanks to **Mr. M. SREENIVASU** Project Coordinator for his excellent encouragement in course of this work.

We would like to thank all the teaching and non-teaching faculty members of our department for advising us whenever in need, co-operating with us and arranging the necessary facilities.

We would like convey gratitude to our **parents** whose blessing was always with us. Last but not the least we would like to thank our Team members, **Friends** and **Others** who helped us in successful completion of this work.

M.SATHIESH **166R1A0516**

M.SIREESHA **166R1A0517**

M.BHUVANESWARI DEVI **166R1A0521**

P.NAGABRAMHANANDAM **166R1A0528**

DECLARATION

We hereby declare that the project entitled **"DETECTION OF PHISHING WEBSITES"** has been undertaken by us and this work has been submitted to **GIET COLLEEGE OF ENGINEERING** affiliated to JNTU Kakinada, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in **COMPUTER SCIENCE & ENGINEERING.**

PLACE: RAJAMAHENDRAVARAM

DATE:

M.SATHIESH - 166R1A0516

M.SIREESHA - 166R1A0517

M.BHUVANESWARI DEVI - 166R1A0521

P.NAGABRAMHANANDAM - 166R1A0528

ABSTRACT

There are number of users who purchase products online and make payment through various websites. There are multiple websites who ask user to provide sensitive data such as username, password, or credit card details etc. often for malicious reasons. This type of websites is known as phishing websites. In order to detect and predict phishing website, we proposed an intelligent, flexible and effective system that is based on using classification data mining algorithm. We implemented classification algorithm and techniques to abstract the phishing datasets criteria to classify their legitimacy. The phishing websites can be detected based on some characteristics like URL and Domain Identity, and security and encryption criteria in the final phishing detection rate. Once user makes transaction through online when he makes payment through the websites our system will use data mining algorithm to detect whether the website is phishing website or not. This application can be used by many e-commerce enterprises in order to make the whole transaction process secure. Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms. With the help of the systems user can also purchase products online without any hesitation. Admin can add phishing websites URL or fake websites URL into systems where system could accesses and scan the phishing websites and by using algorithm, it will add new suspicious keywords to database.

INDEX

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 INTRODUCTION TO DATA MINING

With the amazing progress of both computer hardware and software, a vast amount of data is generated and collected daily. There is no doubt that data are meaningful only when one can extract the hidden information inside them. However, the major barrier for obtaining high quality knowledge from data is due to the limitations of the data itself. These major barriers of collected data come from their growing size and versatile domains. Thus, data mining that is to discover interesting patterns from large amounts of data within limited sources (i.e., computer memory and execution time) has become popular.

Clustering is considered an important tool for data mining. The goal of data clustering is aimed at dividing the data set into several groups such that objects have a high degree of similarity to each other in the same group and have a high degree of dissimilarity to the ones in different groups . Each formed group is called a cluster. Useful patterns may be extracted by analyzing each cluster. For example, grouping customers with similar characteristics based on their purchasing behaviors in transaction data may find their previously unknown patterns. The extracted information is helpful for decision making in marketing.

Clustering is similar to classification except that the groups are not predefined, but rather defined by data alone. Clustering is also known as Unsupervised Learning. Clustering means partitioning or segmenting the data into groups that might not be disjointed. The most similar data is grouped into Clusters.

1.2 INTRODUCTION TO PHISHING WEB SITES

Phishing is a online deception technique in which scam artists uses an e-mail or website to illegitimately obtain confidential information such as user names and passwords. Phishing makes use of spoofed emails that are made to look authentic and purported to be coming from legitimate sources. It is a semantic attack which targets the user rather than the computer. It is a new internet crime. Social networking sites are now a major objective of phishing. The motivation behind this study is to create a resilient and effective method that uses Data Mining algorithm and tools to detect phishing websites.

Class Based Associative classification algorithms can be very useful in predicting Phishing websites. It can give us answers about what are the most important social networking phishing websites characteristics and indicators and how they relate with each other.

1.3 PROJECT OUTLOOK

A phishing scam is an email that seems legitimate but is an attempt to get your personal information or steal your money. Scammers can also use a technique called spoofing to make it appear as if you've received an email from yourself. Here's how to deal with online abuse and phishing and spoofing scams sent to or coming from Outlook.com accounts.

If you conduct this type of phishing attack for another organization, the friendly thing to do is transfer the domain to their IT staff after the conclusion of the test. At the least, it takes a potential phishing domain off a list that a real attacker could draw from.

1.4 SCOPE OF THE PROJECT

Phishing is a considerable problem differs from the other security threats such as intrusions and Malware which are based on the technical security holes of the network systems. The weakness point of any network system is its Users. Phishing attacks are targeting these users depending on the trikes of social engineering. Despite there are several ways to carry out these attacks, unfortunately the current phishing detection techniques cover some attack vectors like email and fake websites. Therefore, building a specific limited scope detection system will not provide complete protection from the wide phishing attack vectors. This paper develops detection system with a wide protection scope using URL features only which is relying on the fact that users directly deal with URLs to surf the internet and provides a good approach to detect malicious URLs as proved by previous studies. Additionally, Anti-phishing solutions can be positioned at different levels of attack flow where most researchers are focusing on client side solutions which turn to add more processing overhead at the client side and lead to losing the trust and satisfaction of the users. Nowadays many organizations make centralized protection of spam filtering. This project proposes a system which can be integrated into such process in order to increase the detection performance in a real time. The simulation results of the proposed system showed a phishing URLs detection accuracy with 93% and provided online process of a single URL in average time of 0.12 second.

1.5 OBJECTIVE OF THE SYSTEM

Stopping general phishing (the mass mail, badly written, generic kind) isn't impossible, and can be achieved without too much sacrifice, through proper employee training. Highly targeted, professional grade phishing though, is incredibly hard to stop, as it may appear to be from (or be from) legitimate email accounts, be well written and highly relevant to the person receiving it. I would say stopping this kind of attack is near impossible for any organization that relies on interaction with the outside world.

PHISHER'S OBJECTIVES

- To steal passwords to access bank accounts and money with the help of banking Trojans
- To subscribe users to a paid SMS-mailing and various content services
- To infect computers in order to turn them into botnet nodes (to steel money from victim bank accounts, permanently monitor user activity, send spam, etc.)
- To steal accounts in a social network and send spam on behalf of victims
- To steal email addresses from contact books to send spam.

In the field of computer security, phishing is the criminally fraudulent process of attempting to acquire sensitive information such as usernames, passwords and credit card details by masquerading as a trustworthy entity in an electronic communication. A phishing website is a broadly launched social engineering attack that attempts to defraud people of their personal information including credit card number, bank account information, social security number and their personal credentials in order to use these details fraudulently against them. Phishing has a huge negative impact on organizations' revenues, customer relationships, marketing efforts and overall corporate image. Communications purporting to be commonly used to lure the unsuspecting public. Phishing is typically carried out by e-mail or instant messaging, and it often directs users to enter details at a fake website whose look and feel are almost identical to the legitimate one. Phishing is an example of social engineering techniques used to fool users, and exploits the poor usability of current web security technologies. Attempts to deal with the growing number of reported phishing incidents include legislation, user training, public awareness, and technical security measures. As per Symantec Message Labs Security report of March 2011, the following statistics were observed.

• Spam – 79.3% in March

• Viruses – One in 208.9 emails in March contained malware (an increase of 0.13 percentage points since February 2011)

• Phishing – One in 252.5 emails comprised a phishing attack

• Malicious websites – 2,973 web sites blocked per day

2. SYSTEM ANALYSIS

2.1 PROBLEM DESCRIPTION

Phishing is a fraudulent attempt, usually made through email, to steal your personal information. The best way to protect yourself from phishing is to learn how to recognize a phish.

Phishing emails usually appear to come from a well-known organization and ask for your personal information such as credit card number, social security number, account number or password. Often times phishing attempts appear to come from sites, services and companies with which you do not even have an account.

In order for Internet criminals to successfully "phish" your personal information, they must get you to go from an email to a website. Phishing emails will almost always tell you to click a link that takes you to a site where your personal information is requested. Legitimate organizations would never request this information of you via email.

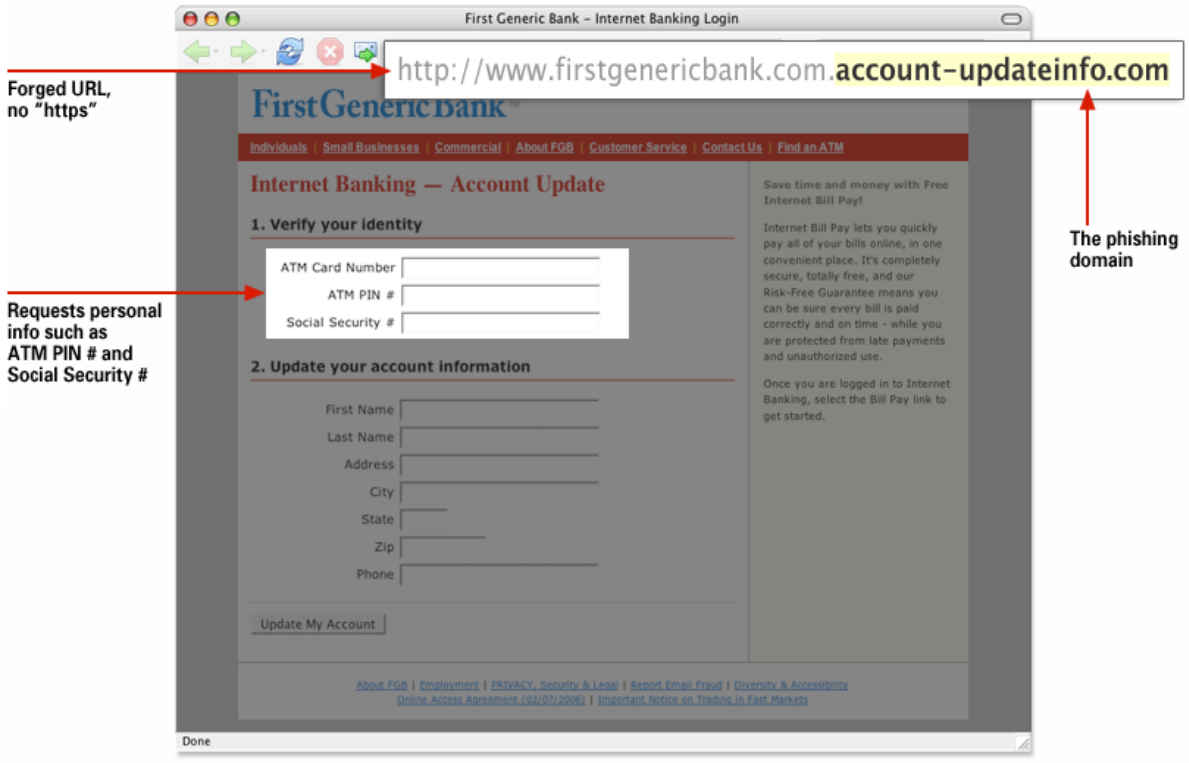

Fig 2.1 Overview Of Phishing Website

What to look for in a phishing website

- **Poor resolution.** Phishing websites are often poor in quality, since they are created with urgency and have a short lifespan. If the resolution on a logo or in text strikes you as poor, be suspicious.
- **Forged URL.** Even if a link has a name you recognize somewhere in it, it doesn't mean it links to the real organization. Read URLs from right to left — the real domain is at the end of the URL. Also, websites where it is safe to enter personal information begin with "https" — the "s" stands for secure. If you don't see "https" do not proceed. Look out for URLs that begin with an IP address, such as: http://12.34.56.78/firstgenericbank/account-update/ — these are likely phishes.

2.2 EXISTING SYSTEM

The existing system uses the Classifiers, Random forest Algorithm, and Bayesian Model to detect the phishing sites. The classifiers can classify the text content and image content. Text classifier is to classify the text content and Image classifier is to classify the image content. Bayesian model estimates the threshold value. Fusion Algorithm combines the both classifier results and decides whether the site is phishing or not. The performance of different classifiers based on correct classification ratio, F-score, Matthews's correlation coefficient, False negative ratio, and False alarm ratio. The threshold value will be decided by the developer only. This leads to the problems like false positive and false negative. False positive means, the probability of being a phishing webpage is greater than the threshold value but that webpage is not a phishing webpage. False negative means, the probability of being a phishing webpage is less than the threshold value but that webpage is a phishing webpage. This results the reduction in security levels. The existing system handles the only one kind of phishing attacks. If that was a phishing site then the existing system only warns the user. The active and passive warnings alone were not enough to control the phishing sites. The active warning gives the user options to close the window or displaying the website. The passive warning displays the popup dialog box.

Online transactions are nowadays become very common and there are various attacks present behind this. In these types of various attacks, phishing is identified as a major security threat and new innovative ideas are arising with this in each second so preventive mechanism should also be so

effective. Thus the security in these cases be very high and should not be easily tractable with implementation easiness.

Today, most applications are only as secure as their underlying system. Since the design and technology of middleware has improved steadily, their detection is a difficult problem. As a result, it is nearly impossible to be sure whether a computer that is connected to the internet can be considered trustworthy and secure or not. Phishing scams are also becoming a problem for online banking and e-commerce users. These all things motivate me to do this work.

In the case of the online banking transactions, security plays a crucial role. According the proposed system provides the secure online banking transactions effectively. It can deal the In-session phishing efficiently. By using username and password, the user can login to his account. Then a session key will be send to the authorized users mobile. Using that session key only the user can perform the further transactions. It was possible for the phisher to get the username and password of a particular user, but without that session key the phisher can't perform any unauthorized transactions.The existing system focuses on finding fake or shadow of websites based on pattern matching principles. It has some set of functions to check the real and bogus websites. Steps involved for inspecting web page.

- Initially checks URL Research Article Volume 7 Issue No. 2 International Journal of Engineering Science and Computing, February 2017 4642 http://ijesc.org/

- If IP address exist its limit then it is bogus web page. If it is not that a real one.

- When the URL length exceeds 54 characters then it is fake web page.

- If the URL contain @ it show that site is false site.

- If the URL contain any suffix or prefixes "-"then is a bogus web page. From the above steps, the URL can be inspected to check whether the web page is real or not

2.3 PROPOSED SYSTEM

Random Forest

Random forest is a classifier that joins numerous tree predictors, where every tree relies on upon the estimations of an irregular vector inspected autonomously. Besides, all trees in the forests have the same appropriation. Keeping in mind the end goal to develop a tree we expect that n is the quantity of preparing perceptions furthermore, p is the quantity of variables (elements) in a preparation set. Keeping in mi nd the end goal to decide the choice hub at a tree we pick K << p as the quantity of variables to be chosen. We select a bootstrap test from the n perceptions in the preparation set furthermore; utilize whatever is left of the perceptions to assess the blunder of the tree in the testing stage. Subsequently, we haphazardly pick k variables as a choice at a specific hub in the tree and calculate the best split in light of the k variables in the preparation set. Trees are constantly developed and never pruned contrasted with other tree calculations. Irregular backwoods can deal with expansive quantities of variables in an information set. Likewise, amid the backwoods building process they generate an inside fair-minded appraisal of the speculation mistake. What's more, they can assess missing information well. A noteworthy downside of arbitrary timberlands is the absence of reproducibility, as the procedure of building the timberland is arbitrary. Further, clarifying the final model and ensuing results is difficult, as it contains numerous free choices trees.

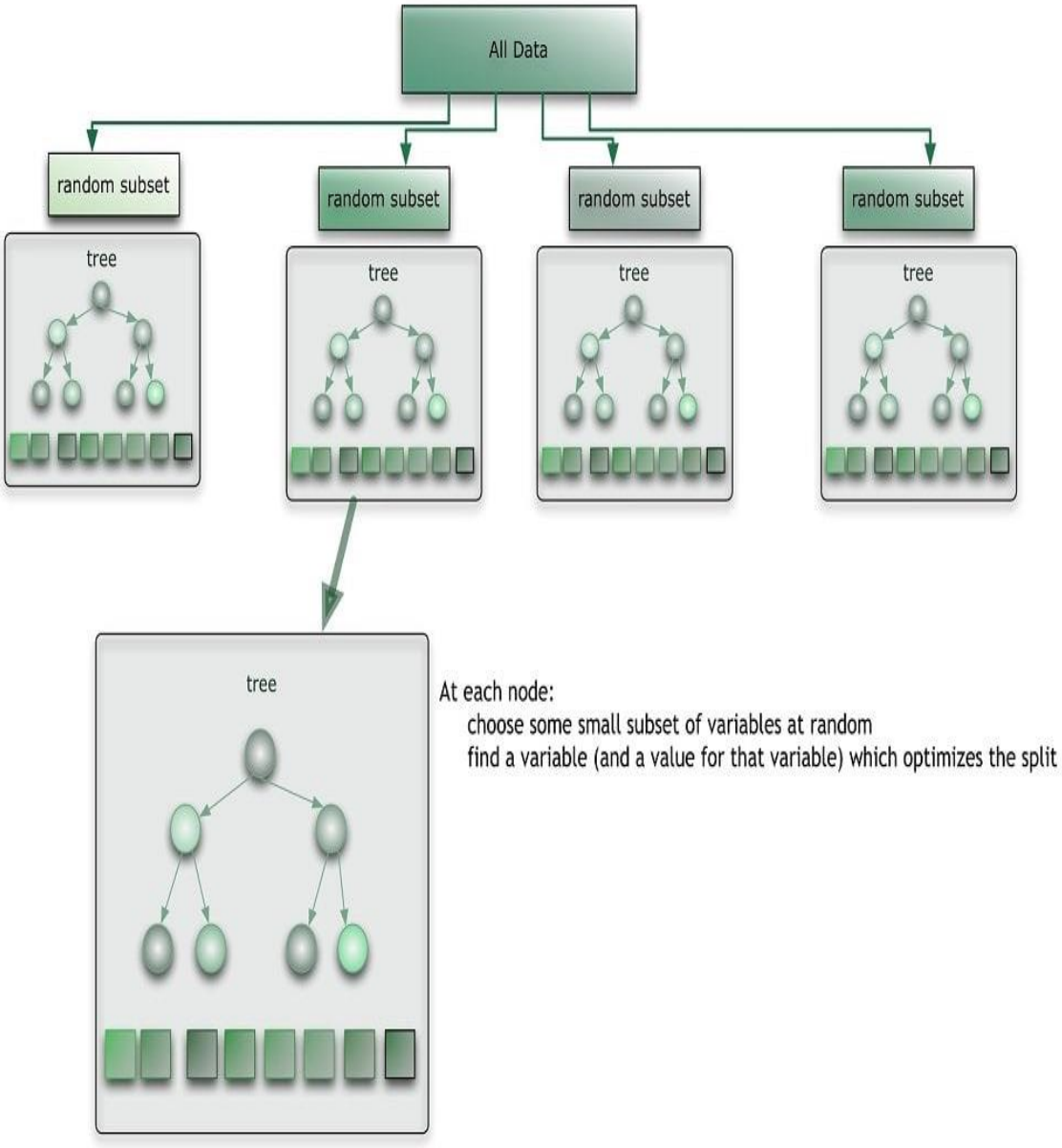

Fig 2.3 Random Forest

2.4 IMPORTANCE OF OUR SYSTEM

Current phishing statistics

Starting from 1996 to the date of writing this paper, there have been a significant number of phishing attacks recorded and analyzed by anti-phishing organizations such as APWG and Phishtank. According to the APWG Phishing Trends Report, 158,544 unique phishing websites were recorded in the fourth quarter of 2015. In most of the cases, financial and online payment companies were targeted, and the majority of these phishing websites are hosted on a server located in the USA. Table I.I describes the current trends (first Quarter 2014 to fourth Quarter of 2015) 21 of phishing attacks in terms of "Number of Phishing Websites," "Number of Phishing Emails," "Top Country Hosting Phishing Sites," "Most Affected Services," and "Most Targeted Top Level Domain (TLD)."

Quarter/parameters	Q1 2014	Q2 2014	Q3 2014	Q4 2014	Q1 2015	Q2 2015	Q3 2015	Q4 2015
Number of phishing websites	126215	128378	92473	47094	136,347	253007	241140	158544
Number of phishing emails	171792	171801	163333	197252	221211	417472	395015	380280
Top country hosting phishing sites	USA	USA	USA	USA	USA	USA	USA	USA
Most affected services	Paym. 46.57%	Paym. NA	Paym. 32.06%	Retail 29.37%	ISP 26.24%	ISP 25.34%	ISP 25.34%	Retail 24.03%
Most targeted TLD	. COM 46%	. COM 51%	.COM 55%	.COM 46%	NA	NA	NA	NA

Table 2.4 Current phishing attack trends.

Note: Paym, payment services; NA, not available; QX, quarter number X.

As demonstrated by Table1.1, there is no significant decrease in the number of phishing incidents, despite various phishing detection and prevention schemes proposed by researchers. Hence, there is an immense need for research and development on security solutions to prevent or detect phishing attacks over the Internet and to safeguard novice users and online transactions.

Anti-phishing solutions

There has been a great deal of research in the area of phishing prevention and detection. The proposed solutions include training users on phishing-related activities; phishing detection and prevention; use of anti-phishing software; browser extensions and toolbars; DNS and WhoIs information of URLs; new measures of user authentication; filtering phishing emails and websites; real time, proactive detection, monitoring and shutting down of phishing websites; two factor authentication schemes; disabling malicious Java scripts; secure browser developments; and so on. Anti-phishing solutions can be majorly categorized as phishing prevention solutions, user training solutions, and phishing detection solutions.

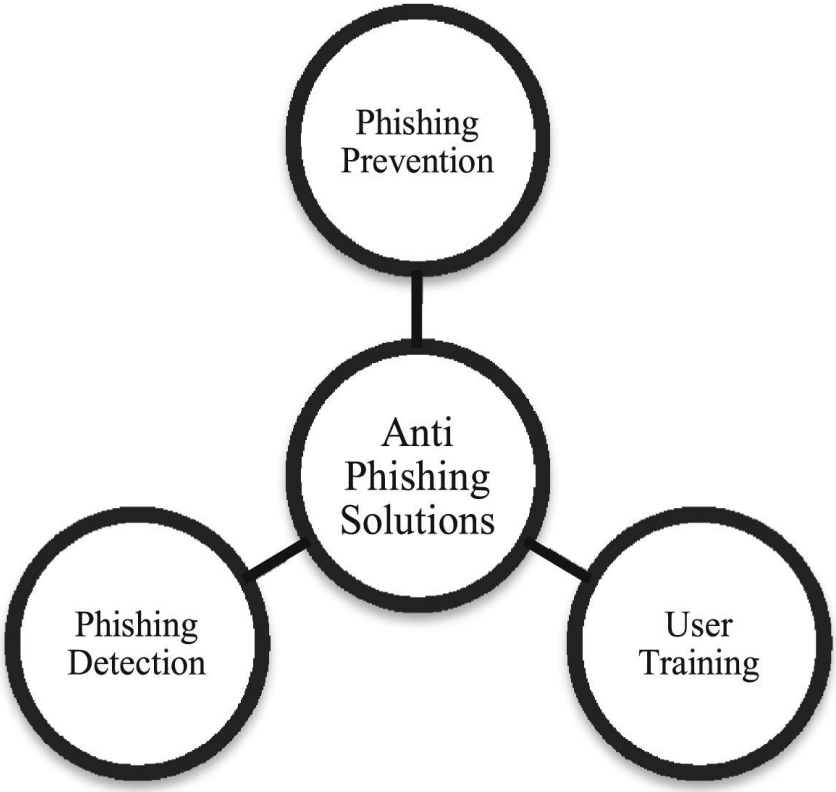

Fig.2.4.1 classification of anti-phishing solutions.

3. REQUIREMENT SPECIFICATIONS

3.1 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

Functional Requirements

The functional requirements describes the interaction between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts.

Non-Functional Requirements

In systems engineering and requirements engineering, a non-functional requirements is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behavior. This should be contrasted with functional requirement that defines specific behavior or functions. In general, functional requirements define when a system is supposed to do whereas non-functional requirements define how a system is supposed to be. Non-functional requirements are often called as qualities of a system. Other terms for non-functional requirements are “constraints” , “quality attributes” ,”quality goals” and “quality of service requirements”, and “non-behavioral requirements”. The interface of this system is GUI. The user interface could be flexible.

3.1.1 Software Requirements

3.1.1.1 java

Java is a general-purpose programming language that is class-based, object-oriented, and designed to have as few implementation dependencies as possible. It is intended to let application developers write once run anywhere(WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to byte code that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but it has fewer low-level facilities than either of them. As of 2019, Java was one of the most popular programming languages in use according to Git Hub, particularly for client-server web applications, with a reported 9 million developers.

3.1.1.2 Eclipse

Eclipse IDE

The Eclipse platform which provides the foundation for the Eclipse IDE is composed of plug-ins and is designed to be extensible using additional plug-ins. Developed using Java, the Eclipse platform can be used to develop rich client applications, integrated development environments and other tools. Eclipse can be used as an IDE for any programming language for which a plug-in is available.

The Java Development Tools (JDT) project provides a plug-in that allows Eclipse to be used as a Java IDE, PyDev is a plugin that allows Eclipse to be used as a Python IDE, C/C++ Development Tools (CDT) is a plug-in that allows Eclipse to be used for developing application using C/C++, the Eclipse Scala plug-in allows Eclipse to be used an IDE to develop Scala applications and PHPeclipse is a plug-in to eclipse that provides complete development tool for PHP.

Eclipse is a Free and Open Source IDE, plus a developer tool framework that can be extended for a particular development need. IBM was behind its development, and it replaced IBM VisualAge tool. The idea was to create a standard look and feel that can be extended via plugins. The extensibility distinguishes Eclipse from other IDEs. Eclipse was also meant to compete with Microsoft Visual Studio tools. Microsoft tools give a standard way of developing code in the Microsoft world. Eclipse gives a similar standard way of developing code in the Java world, with a big success so far. With the online error checking only, coding can be sped up by at least 50% (coding does not include programming).

The goals for Eclipse are twofold:

1.Give a standard IDE for developing code

2.Give a starting point, and the same look and feel for all other more sophisticated tools built on Eclipse IBM's WSAD, and later IBM Rational Software Development Platform, are built on Eclipse.

Standard Eclipse features:

- Standard window management (perspectives, views, browsers, explorers, ...)
- Error checking as you type (immediate error indications, ...)
- Help window as you type (type ., or <ctrl> space, ...)

- Automatic build (changes in source code are automatically compiled, ...)
- Built-in debugger (full featured GUI debugger)
- Source code generation (getters and setters, ...)
- Searches (for implementation, for references, ...)
- Code refactoring (global reference update, ...)
- Plugin-based architecture (ability to build tools that integrate seamlessly with the environment, and some other tools)

3.1.2 Hardware Requirements

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These pre-requisites are known as (computer) system requirements and are often used as guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving updates to existing computer systems than technological advancements.

TYPE	REQUIRED
Processor	INTEL CORE I3
Hard Disk Space	25GB
RAM	8GB

Table 3.1.2 Hardware Requirements

4. TECHNOLOGIES USED

4.1 JAVA

What is a Java IDE?

A Java IDE (for *Integrated Development Environment*) is a software application which enables users to more easily write and debug Java programs. Many IDEs provide features like syntax highlighting and code completion, which help the user to code more easily.

1. Java Technology

   Java technology is both a programming language and a platform.

2. The Java Programming Language

   The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

   - Simple
   - Architecture neutral
   - Object oriented
   - Portable
   - Distributed
   - High performance
   - Interpreted
   - Multithreaded
   - Robust
   - Dynamic
   - Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled

and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

Java compilation

You can think of Java byte codes as the machine code instructions for the Java VirtualMachine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make "write once, run anywhere" possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

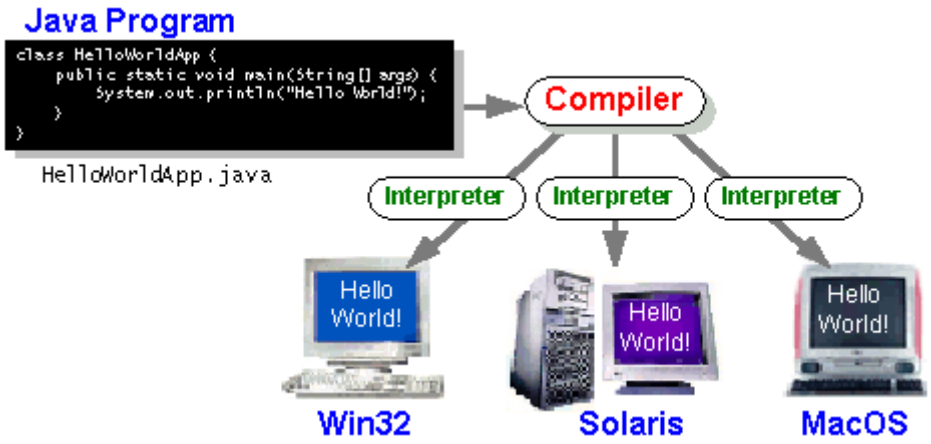

Fig 4.1.1 Java compilation and Execution

Java execution

The Java Platform

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine (Java VM)
- The Java Application Programming Interface (Java API)

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces these libraries are known as packages. The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

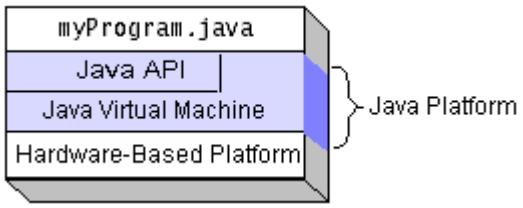

Fig 4.1.2 Java hardware layout

Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in- time byte code compilers can bring performance close to that of native code without threatening portability.

4.2 MACHINE LEARNING

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

Some machine learning methods

Machine learning algorithms are often categorized as supervised or unsupervised.

1. Supervised machine learning algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

2. In contrast, unsupervised machine learning algorithms are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn't figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.

3. Supervised machine learning algorithms fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiring unlabeled data generally doesn't require additional resources.

4. Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal.

5. SYSTEM DESIGN

5.1 INTRODUCTION TO UNIFIED MODELLING LANGUAGE

Unified Modeling Language is the one of the most exciting tools in the world of system Development today. Because UML enables system builders to create blue prints that capture. Their visions in a standard, easy to understand way and communication them to others. The UML is brain child of Grady Brooch, James Rumbaugh and Ivar Jacobson.

The UML is a language for

- Visualizing
- Specifying
- Constructing
- Documenting

These are the artifacts of a software-intensive system. The abbreviation for UML is Unified Modeling Language and is being brought of a designed to make sure that the Existing ER diagram which do not serve the purpose will be replaced by this UML diagram where in these language as its own set of diagrams.

Using the UML diagrams we can show the entire system regarding the working of the System or the flow of control and sequence of flow the state of the system and the Activities involved in the system.

Goals of UML

A picture is worth a thousand words,this absolutely fits while discussing about UML Object Oriented concepts were introduced much earlier than UML. So at that time there were no standard methodologies to organize and consolidate the object oriented development. At that point of time UML came into pictures.There are a number of goals for developing UML but the most important is to define some general purpose modeling language which all modelers can use and also It needs to be make simple to understand and use.

5.2 UML Diagrams

5.2.1 Usecase Diagram

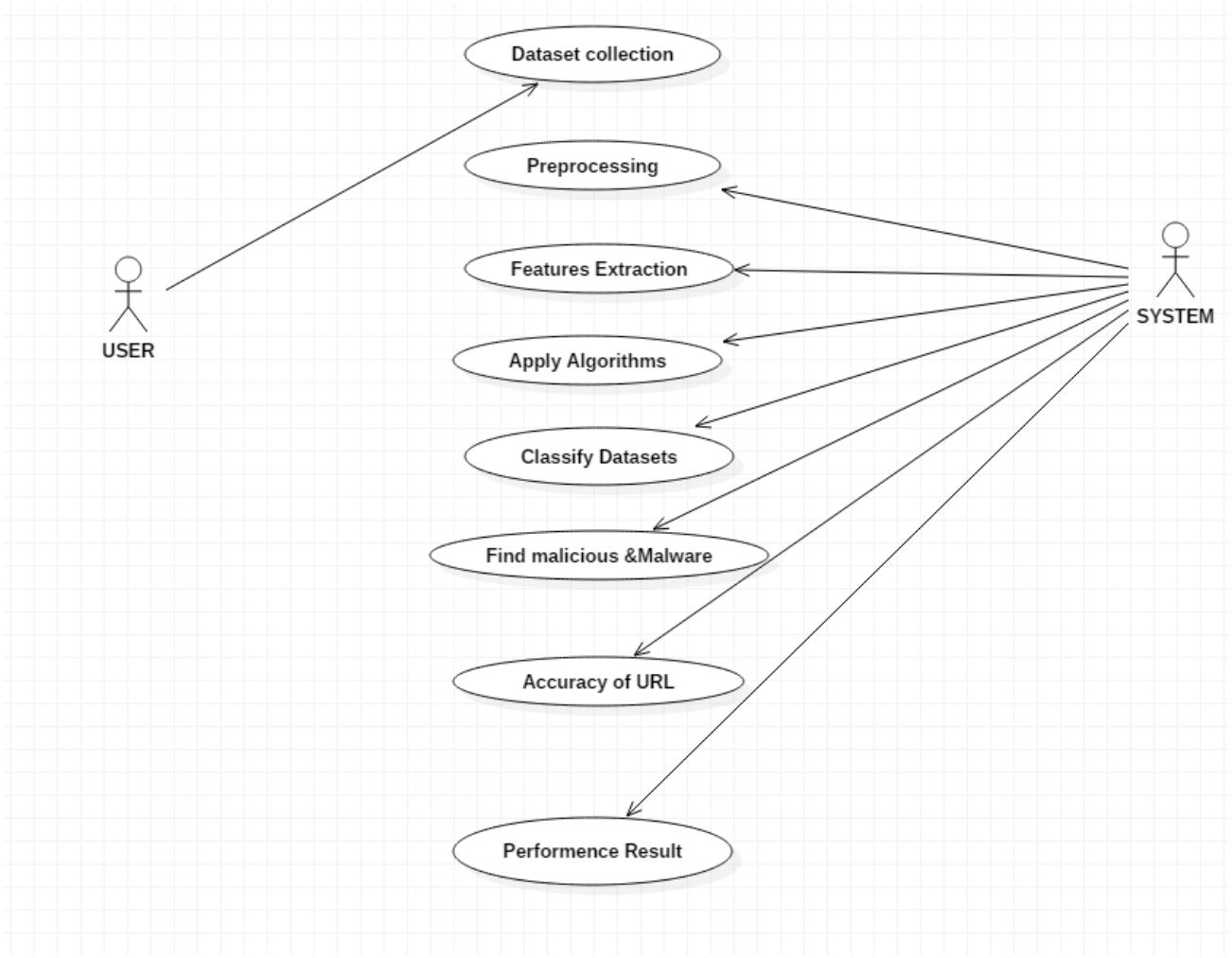

Fig 5.2.1 Usecase diagram

5.2.2 Sequence Diagram

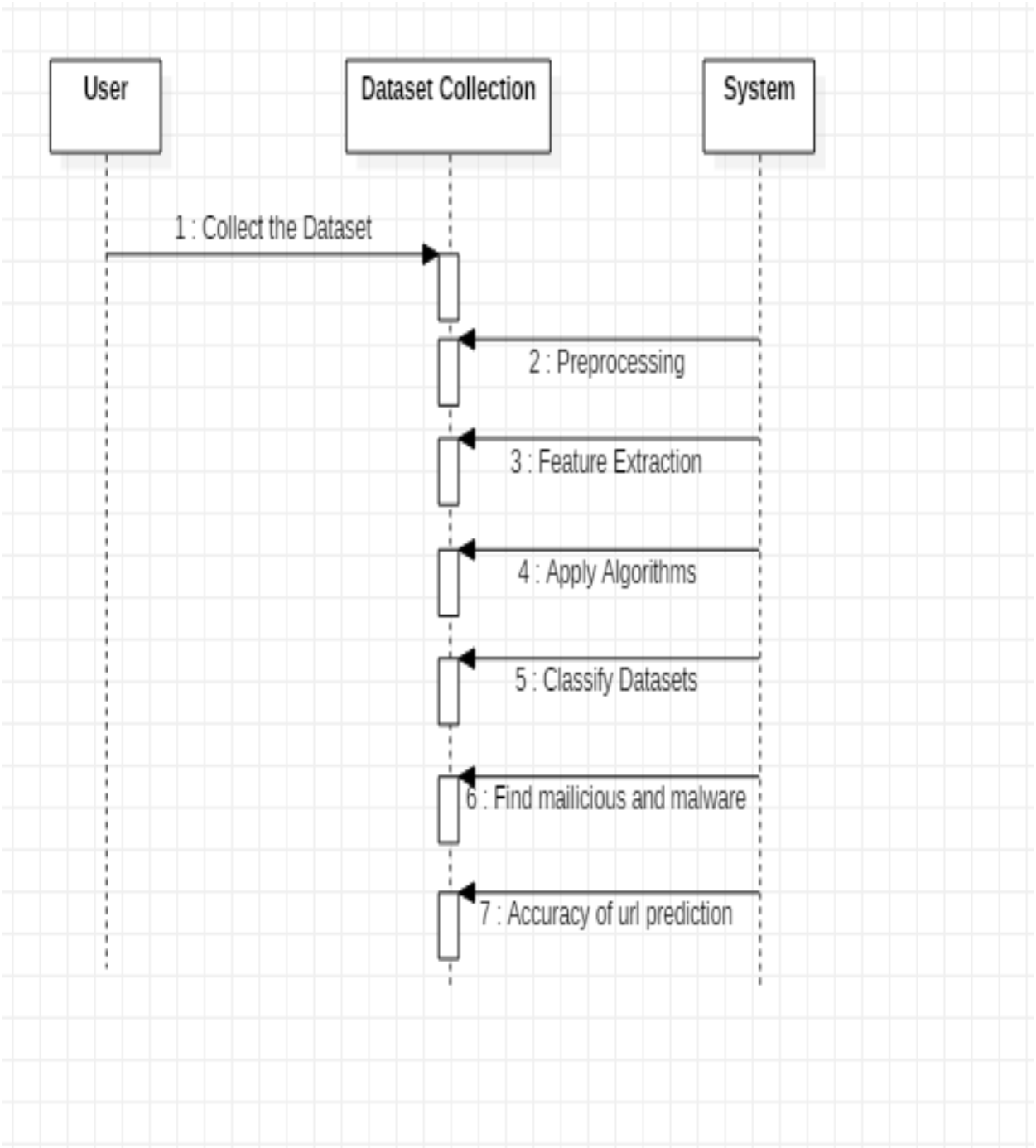

Fig 5.2.2 Sequence diagram

5.2.3 Class diagram

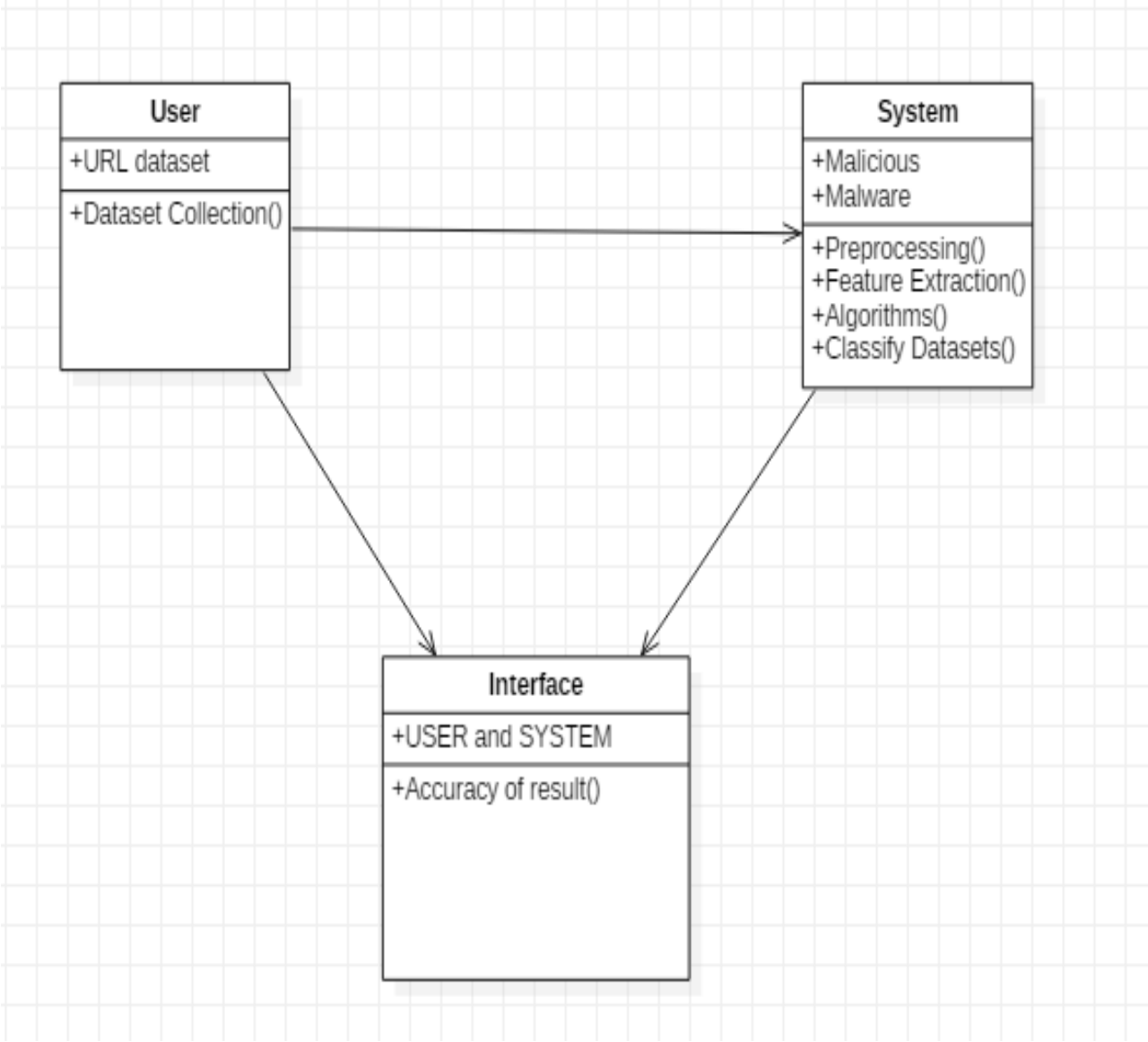

Fig 5.2.3 Class diagram

5.2.4 Activity diagram

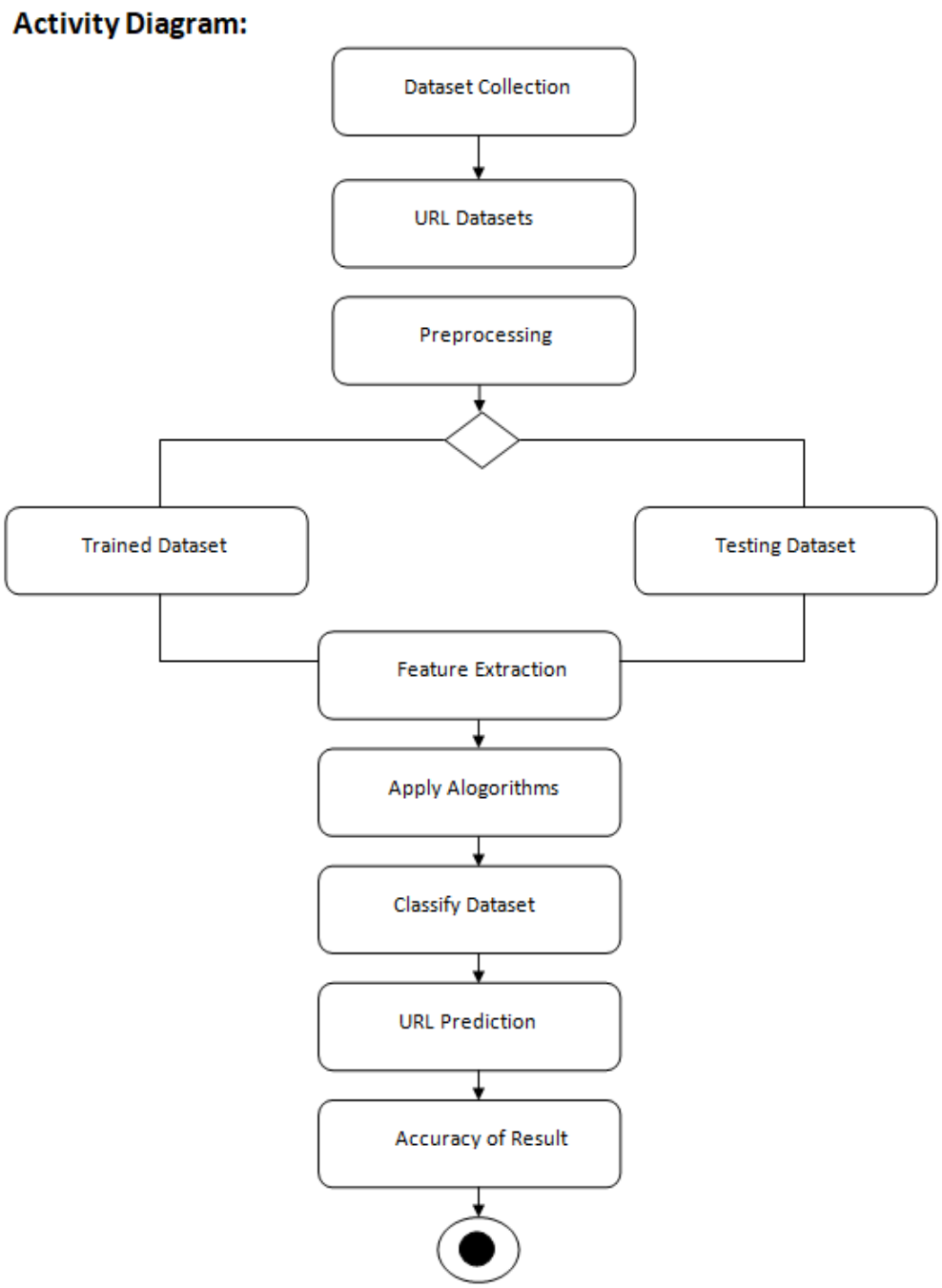

Fig 5.2.4 Activity Diagram

5.3 METHODOLOGY

In our model we have developed Random forest is a classifier that joins numerous tree predictors, where every tree relies on upon the estimations of an irregular vector inspected autonomously. Besides, all trees in the forests have the same appropriation. Keeping in mind the end goal to develop a tree we expect that n is the quantity of preparing perceptions furthermore, p is the quantity of variables (elements) in a preparation set. Keeping in mi nd the end goal to decide the choice hub at a tree we pick K << p as the quantity of variables to be chosen. We select a bootstrap test from the n perceptions in the preparation set furthermore; utilize whatever is left of the perceptions to assess the blunder of the tree in the testing stage. Subsequently, we haphazardly pick k variables as a choice at a specific hub in the tree and calculate the best split in light of the k variables in the preparation set. Trees are constantly developed and never pruned contrasted with other tree calculations. Irregular backwoods can deal with expansive quantities of variables in an information set. Likewise, amid the backwoods building process they generate an inside fair-minded appraisal of the speculation mistake. What's more, they can assess missing information well. A noteworthy downside of arbitrary timberlands is the absence of reproducibility, as the procedure of building the timberland is arbitrary. Further, clarifying the final model and ensuing results is difficult, as it contains numerous free choices trees.

6. CODING AND IMPLEMENTATION

Begin RF Algorithm
Input: N: number of nodes
M: number of features
D: number of trees to be constructed
Output: V: the class with the highest vote
While stopping criteria is false **do**
Randomly draw a bootstrap sample A from the training data D
Use the steps below to construct tree T_i from the drawn bootstrapped sample A:
(I) Randomly select m features from M; where $m \ll M$
(II) For node d, calculate the best split point among the m features
(III) Split the node into two daughter nodes using the best split
(IV) Repeat I, II and III until n number of nodes has been reached
Build your forest by repeating steps I–IV for D number of times
End While
Output all the constructed trees $\{T_i\}_1^D$
Apply a new sample to each of the constructed trees starting from the root node
Assign the sample to the class corresponding to the leaf node.
Combine the decisions (or votes) of all the trees
Output V, that is, the class with the highest vote.
End RF Algorithm

Fig.6.1 RF Algorithm

6.1 Construction of Decision Trees

```
import java.io.BufferedReader;

import java.io.File;

import java.io.FileInputStream;

import java.io.InputStreamReader;

import java.util.StringTokenizer;

import java.util.Vector;

import javax.swing.tree.DefaultMutableTreeNode;

public class RandomForest {

        JTreeFrame screen;
```

```
int numAttributes; // The number of attributes including the output attribute
String[] attributeNames; // The names of all attributes. It is an array of
                                        // dimension numAttributes. The last attribute
                                                                // is the output attribute
/*
 * Possible values for each attribute is stored in a vector. domains is an
 * array of dimension numAttributes. Each element of this array is a vector
 * that contains values for the corresponding attribute domains[0] is a
 * vector containing the values of the 0-th attribute, etc.. The last
 * attribute is the output attribute
 */
Vector[] domains;
/* The root of the decomposition tree */
TreeNode root = new TreeNode();
DefaultMutableTreeNode guiRoot;
public RandomForest(JTreeFrame screen) {
        this.screen = screen;
}
/*
 * This function returns an integer corresponding to the symbolic value of
 * the attribute. If the symbol does not exist in the domain, the symbol is
 * added to the domain of the attribute
 */
public int getSymbolValue(int attribute, String symbol) {
```

```
            int index = domains[attribute].indexOf(symbol);
            if (index < 0) {
                  domains[attribute].addElement(symbol);
                  return domains[attribute].size() - 1;
            }
            return index;
      }
      /* Returns all the values of the specified attribute in the data set */
      public int[] getAllValues(Vector data, int attribute) {
            Vector values = new Vector();
            int num = data.size();
            for (int i = 0; i < num; i++) {
                  DataPoint point = (DataPoint) data.elementAt(i);
            String symbol = (String) domains[attribute].elementAt(point.attributes[attribute]);
                  int index = values.indexOf(symbol);
                  if (index < 0) {
                        values.addElement(symbol);
                  }
            }
            int[] array = new int[values.size()];
            for (int i = 0; i < array.length; i++) {
                  String symbol = (String) values.elementAt(i);
                  array[i] = domains[attribute].indexOf(symbol);
            }
```

```
            values = null;
            return array;
        }
        /*
         * Returns a subset of data, in which the value of the specfied attribute of
         * all data points is the specified value
         */
        public Vector getSubset(Vector data, int attribute, int value) {
            Vector subset = new Vector();
            int num = data.size();
            for (int i = 0; i < num; i++) {
                DataPoint point = (DataPoint) data.elementAt(i);
                if (point.attributes[attribute] == value)
                    subset.addElement(point);
            }
            return subset;
        }
        /*
         * Calculates the entropy of the set of data points. The entropy is
         * calculated using the values of the output attribute which is the last
         * element in the array attribtues
         */
        public double calculateEntropy(Vector data) {
            int numdata = data.size();
```

```
        if (numdata == 0)
                return 0;
        int attribute = numAttributes - 1;
        int numvalues = domains[attribute].size();
        double sum = 0;
        for (int i = 0; i < numvalues; i++) {
                int count = 0;
                for (int j = 0; j < numdata; j++) {
                        DataPoint point = (DataPoint) data.elementAt(j);
                        if (point.attributes[attribute] == i)
                                count++;
                }
                double probability = 1. * count / numdata;
                if (count > 0)
                        sum += -probability * Math.log(probability);
        }
        return sum;

}
/*
 * This function checks if the specified attribute is used to decompose the
 * data set in any of the parents of the specfied node in the decomposition
 * tree. Recursively checks the specified node as well as all parents
 */
```

```
public boolean alreadyUsedToDecompose(TreeNode node, int attribute) {
        if (node.children != null) {
                if (node.decompositionAttribute == attribute)
                        return true;
        }
        if (node.parent == null)
                return false;
        return alreadyUsedToDecompose(node.parent, attribute);
}
/*
 * This function decomposes the specified node according to the J48
 * algorithm. Recursively divides all children nodes until it is not
 * possible to divide any further I have changed this code from my earlier
 * version. I believe that the code in my earlier version prevents useless
 * decomposition and results in a better decision tree! This is a more
 * faithful implementation of the standard J48 algorithm
 */
public void decomposeNode(TreeNode node) {
        double bestEntropy = 0;
        boolean selected = false;
        int selectedAttribute = 0;
        int numdata = node.data.size();
        int numinputattributes = numAttributes - 1;
        node.entropy = calculateEntropy(node.data);
```

```
if (node.entropy == 0)
        return;
/*
 * In the following two loops, the best attribute is located which
 * causes maximum decrease in entropy
 */
for (int i = 0; i < numinputattributes; i++) {
        int numvalues = domains[i].size();
        if (alreadyUsedToDecompose(node, i))
                continue;
        // Use the following variable to store the entropy for the test node
        // created with the attribute i
        double averageentropy = 0;
        for (int j = 0; j < numvalues; j++) {
                Vector subset = getSubset(node.data, i, j);
                if (subset.size() == 0)
                        continue;
                double subentropy = calculateEntropy(subset);
                averageentropy += subentropy * subset.size(); // Weighted sum
        }
        averageentropy = averageentropy / numdata; // Taking the weighted
                                                   // average
        if (selected == false) {
                selected = true;
```

```
                    bestEntropy = averageentropy;
                    selectedAttribute = i;
                } else {
                    if (averageentropy < bestEntropy) {
                        selected = true;
                        bestEntropy = averageentropy;
                        selectedAttribute = i;
                    }
                }
            }
            if (selected == false)
                return;
            // Now divide the dataset using the selected attribute
            int numvalues = domains[selectedAttribute].size();
            node.decompositionAttribute = selectedAttribute;
            node.children = new TreeNode[numvalues];
            for (int j = 0; j < numvalues; j++) {
                node.children[j] = new TreeNode();
                node.children[j].parent = node;
                node.children[j].data = getSubset(node.data, selectedAttribute, j);
                node.children[j].decompositionValue = j;
            }

            // Recursively divides children nodes
```

```
            for (int j = 0; j < numvalues; j++) {
                    decomposeNode(node.children[j]);
            }
            // There is no more any need to keep the original vector. Release this
            // memory
            node.data = null; // Let the garbage collector recover this memory
    }
    /**
     * Function to read the data file. The first line of the data file should
     * contain the names of all attributes. The number of attributes is inferred
     * from the number of words in this line. The last word is taken as the name
     * of the output attribute. Each subsequent line contains the values of
     * attributes for a data point. If any line starts with // it is taken as a
     * comment and ignored. Blank lines are also ignored.
     */
    public int readData(String filename) throws Exception {

            FileInputStream in = null;
            try {
                    File inputFile = new File(filename);
                    in = new FileInputStream(inputFile);
            } catch (Exception e) {
            System.err.println("Unable to open data file: " + filename + "\n"+ e);
                    return 0;
```

```
                }
                BufferedReader bin = new BufferedReader(new InputStreamReader(in));
                String input;
                while (true) {
                        input = bin.readLine();
                        if (input == null) {
                                System.err.println("No data found in the data file: "
                                                + filename + "\n");
                                return 0;
                        }
                        if (input.startsWith("//"))
                                continue;
                        if (input.equals(""))
                                continue;
                        break;
                }
                //StringTokenizer tokenizer = new StringTokenizer(input, ",");
                String[] tokens = input.split(",");
                numAttributes = tokens.length;//tokenizer.countTokens();
                if (numAttributes <= 1) {
                        System.err.println("Read line: " + input);
System.err.println("Could not obtain the names of attributes in the line");
                        System.err.println("Expecting at least one input attribute and one output
attribute");
```

```
            return 0;
        }

        domains = new Vector[numAttributes];
        for (int i = 0; i < numAttributes; i++)
            domains[i] = new Vector();
        attributeNames = new String[numAttributes];

        for (int i = 0; i < numAttributes; i++) {
            attributeNames[i] = tokens[i];//tokenizer.nextToken();
        }
        while (true) {
            input = bin.readLine();
            if (input == null)
                break;
            if (input.startsWith("//"))
                continue;
            if (input.equals(""))
                continue;
            //tokenizer = new StringTokenizer(input);
            tokens = input.split(",");
            int numtokens = tokens.length;//tokenizer.countTokens();
            if (numtokens != numAttributes) {
                System.err.println("Read " + root.data.size() + " data");
```

```
                    System.err.println("Last line read: " + input);
                    System.err.println("Expecting " + numAttributes +"attributes");
                    return 0;
                }
                DataPoint point = new DataPoint(numAttributes);
                for (int i = 0; i < numAttributes; i++) {

point.attributes[i]=getSymbolValue(i,tokens[i]/*tokenizer.nextToken()*/);
                }
                root.data.addElement(point);
            }
            bin.close();
            return 1;
        } // End of function readData
        // ---------------------------------------------------------------------
        /*
         * This function prints the decision tree in the form of rules. The action
         * part of the rule is of the form outputAttribute = "symbolicValue" or
         * outputAttribute = { "Value1", "Value2", .. } The second form is printed
         * if the node cannot be decomposed any further into an homogenous set
         */
        public void printTree(TreeNode node, String tab,
                DefaultMutableTreeNode guiNode) {
            int outputattr = numAttributes - 1;
```

```
            if (node.children == null) {
                int[] values = getAllValues(node.data, outputattr);
                if (values.length == 1) {
                    screen.append(tab + "\t" + attributeNames[outputattr]
        + " is \"" +  domains[outputattr].elementAt(values[0])
                                + "\"; ");
                    guiNode.setUserObject(attributeNames[outputattr]);

    guiNode.add(newDefaultMutableTreeNode(domains[outputattr].elementAt(values[0])));
                    return;
                }
                screen.appendNoLine(tab + "\t" + attributeNames[outputattr]+ " is (");
                guiNode.setUserObject(attributeNames[outputattr]);
                for (int i = 0; i < values.length; i++) {
                screen.appendNoLine("\""+ domains[outputattr].elementAt(values[i]) +  "\" ");
                    if (i != values.length - 1)
                        screen.appendNoLine(" , ");
            guiNode.add(newDefaultMutableTreeNode(domains[outputattr].elementAt(values[i])));
                }
                screen.append(")");
                return;
            }
            int numvalues = node.children.length;
            guiNode.setUserObject(attributeNames[node.decompositionAttribute]);
```

```
                for (int i = 0; i < numvalues; i++) {
screen.appendNoLine(tab + "if( "+ attributeNames[node.decompositionAttribute] + " is \""
                                + domains[node.decompositionAttribute].elementAt(i)+ "\") {");
DefaultMutableTreeNode tn = new DefaultMutableTreeNode(
                                        domains[node.decompositionAttribute].elementAt(i));
                        DefaultMutableTreeNode tn2 = new DefaultMutableTreeNode();
                        printTree(node.children[i], tab + " ", tn2);
                        if (i != numvalues - 1)
                                screen.appendNoLine(tab + "} \n\n");
                        else
                                screen.append(tab + "}");
                        tn.add(tn2);
                        guiNode.add(tn);
                }
        }
        /*
         * This function creates the decision tree and prints it in the form of
         * rules on the console
         */
        public DefaultMutableTreeNode createDecisionTree() {
                guiRoot = new DefaultMutableTreeNode("DataSet");
                decomposeNode(root);
                printTree(root, "", guiRoot);
                return guiRoot;
```

```
        }
}
```

6.2 Generating Rules

```
import javax.swing.*;
 public class JTreeFrame extends JFrame {
  JTextArea ta = new JTextArea();
public JTreeFrame(String fnm) {
    JTree jt = null;
   try {
    RandomForest me = new RandomForest(this);
    long startTime = System.currentTimeMillis();    //  To print the time taken to process the data
    int status = me.readData(fnm);
    if (status <= 0) return;
    jt = new JTree(me.createDecisionTree());
    long endTime = System.currentTimeMillis();
    long totalTime = (endTime-startTime)/1000;
    append("\n" + totalTime + " Seconds");
    }
    catch(Exception e) {
     append("Exception:\n\n" + e);
    }
    JTabbedPane tbp = new JTabbedPane();
    tbp.add("Tree",new JScrollPane(jt));
    tbp.add("Rules",new JScrollPane(ta));
```

```
    add(tbp);
    setTitle("Finla Decision Tree: ");
    setBounds(20,20,500,500);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setVisible(true);
   }
   public void append(String line) {
    ta.append("\n" + line);
   }
   public void appendNoLine(String line) {
    ta.append(line);
   }
 }
```

6.3 Analyse Test Results

```
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.Random;
import javax.swing.JButton;
```

```
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import javax.swing.tree.DefaultMutableTreeNode;
public class TestData extends JPanel implements ActionListener {
        JTable jtData = new JTable();
        JButton bNext = new JButton("Load Test Data");
        JButton bA = new JButton("Analyse");
        JPanel jpanel = new JPanel();
        DefaultTableModel tm;
        int numAttributes = 0;

        JButton b = new JButton("Close");
        DefaultMutableTreeNode tree;
        public TestData(DefaultMutableTreeNode tree) {
                setLayout(new BorderLayout());
                this.tree = tree;
                jpanel.add(bNext);
                jpanel.add(bA);
                jpanel.add(b);
                add(jpanel, "South");
```

```
            bNext.addActionListener(this);
            bA.addActionListener(this);
            b.addActionListener(this);
            add(new JScrollPane(jtData));
            setVisible(true);
            setBounds(20, 20, 700, 500);
        }
        public void actionPerformed(ActionEvent ae) {
            if (ae.getSource() == b) {
                System.exit(0);
            } else if (ae.getSource() == bA) {
                for (int i = 0; i < tm.getRowCount(); i++) {
                    String result = analyse(i, tree);
                    tm.setValueAt(result, i, tm.getColumnCount() - 1);
                }
            } else {
                try {
                    JFileChooser fch = new JFileChooser("dataset/");
                if(fch.showOpenDialog(this)!=JFileChooser.APPROVE_OPTION)
                        return;
                    File f = fch.getSelectedFile();
BufferedReader buf = new BufferedReader(new InputStreamReader(new FileInputStream(f)));
                    String line = buf.readLine();
                    String[] s = line.split(",");
```

```
                    numAttributes = s.length;
                    tm = new DefaultTableModel();
                    for (int i = 0; i < numAttributes; i++) {
                        String st = s[i];
                        tm.addColumn(st);
                        System.out.println(st);
                    }
                    tm.addColumn("Result");
                    while (true) {
                        line = buf.readLine();
                        if (line == null)
                            break;
                        s = line.split(",");
                        tm.addRow(s);
                    }
                    jtData.setModel(tm);
                } catch (Exception e) {
                    System.out.println(e);
                    System.exit(0);
                }
            }

        }
        public String analyse(int rowid, DefaultMutableTreeNode root) {
```

```
String result = "NA";
if (root.getChildCount() == 1
        && ((DefaultMutableTreeNode) root.getChildAt(0))
                .getChildCount() == 0)
    result = ((DefaultMutableTreeNode) root.getChildAt(0))
            .getUserObject().toString();
else {
    String attrib = root.getUserObject().toString();
    String value = "";
    for (int i = 0; i < tm.getColumnCount() - 1; i++) {
        System.out.println(tm.getColumnName(i));
        System.out.println(attrib);
        if (tm.getColumnName(i).equals(attrib)) {
            value = (String) tm.getValueAt(rowid, i);
            break;
        }
    }
    if (value.equals("")) {
        System.out.println("leased");
        Random r = new Random(tm.getRowCount());
        return r.nextInt(100) < 50 ? "1" : "0";
    } else {
for (int i = 0; i < root.getChildCount(); i++) {
DefaultMutableTreeNodechild=((DefaultMutableTreeNode)root.getChildAt(i));
```

```
        if (child.getUserObject().equals(value)) {
                                                try {
        result = analyse(rowid,(DefaultMutableTreeNode) child.getChildAt(0));
                                                } catch (Exception ex) {
        Random r = new Random(tm.getRowCount());
                        result= r.nextInt(100) < 50 ? "1" : "0";
                                                }
                                                break;
                                        }
                                }
                        }
                }
                return result;
        }
}
```

7. TESTING AND VALIDATION

7.1 INTRODUCTION

Testing is the process of evaluating a system or application, to check whether the application meets all requirements of the client and to detect the errors. Generally testing can be classified into static testing and dynamic testing. Again Dynamic Testing is classified into two types: Structural Testing (or) white box , Functional Testing (or) Black Box testing.

A test strategy is an outline that describes the **testing** approach of the software development cycle. It is created to inform project managers, testers, and developers about some key issues of the **testing** process

- Static Testing

  Verification activities fall into the category of Static Testing. Static testing refers to testing something that's not running. It is examining and reviewing it. i.e., to check whether the work done meets the standards of the organization. Reviews, Inspections and Walk-throughs are static testing methodologies.

- Dynamic Testing

  Dynamic Testing involves working with the software, giving input values and checking if the output is as expected. These are the Validation activities. Unit Tests, Integration Tests, System Tests and Acceptance Tests are few of the Dynamic Testing methodologies.

Techniques used are determined by type of testing that must be conducted.

Software testing is a process used for verifying the correctness, completeness and quality of the developed software. Software is built out of sub-systems that are composed of modules, which in turn are composed of procedures and functions. The sequence of testing activities performed for the tracking system is as below:

7.2 TESTING AND VALIDATION

7.2.1 Unit Testing

Unit testing there exist a number of components in every sub-system. Every component is

tested using respective test procedures. Each component is tested individually based on their needs. Unit test focuses verification effort on the smallest unit of the software design component. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit test perform basic tests at component level and test a specific business process and system configuration.

7.2.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by unit testing, the combination of component is correct and consistent.

7.2.3 Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : Identified classes of valid input must be accepted

Invalid Input : Identified classes of invalid input must be rejected.

Functions: Identified functions must be exercised.

Output: Identified classes of application outputs must be exercise

Systems/Procedures interfacing systems or procedures mustbe invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data field predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.2.4 System Testing

System testing ensures that the entire integrated software system meets requirements I tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points. with the intent of finding an error.

7.2.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. it is purpose. It is used to test areas that cannot be reached from a black box level.

7.2.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings structure or language of the module being tested. Black box tests, as most other kinds of tests must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

7.2.7 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meet the functional requirement.

Results All the test cases mentioned above passed successfully. No defects encountered

7.3 TEST CASES

Test cases	Input	Description	Expected Result	Pass/Fail
TC1	URL consists IP address in domain name	URL contain domain name with only numeric values	-1 (url is phishing)	Pass
TC2	URL Consists of https protocol	URL contain secure socket layer	1 (url is benign)	Pass
TC3	URL length Greater than 54	URL contain length greater than 54	-1 (url is suspicious)	Pass
TC4	Entered URL is empty	If your input URL is null	Invalid URL	Fail
TC5	URL has @ symbol	If URL contain @ symbol it will skips the other part of the url	-1 (url is phishing)	Pass
TC6	URL don't have domain name	If URL doesn't contain domain name	-1 (url is phishing)	Pass
TC7	URL contain http in domain name	If http exists in domain name	-1 (url is phishing)	Pass
TC8	If "//" position violates in URL	Occurance position of "//" greaterthan 7 in URL	-1 (url is phished)	Pass
TC9	URL has more dots	If url contains more than three dots	-1 (url is phished)	Pass

TC10	URL contains mail or mailto()	If phisher uses the mail or mailto() in url	-1 (url is phished)	Pass

Table.7.4 Testcases

8. SCREENSHOTS

STEP 1: Loading Training dataset

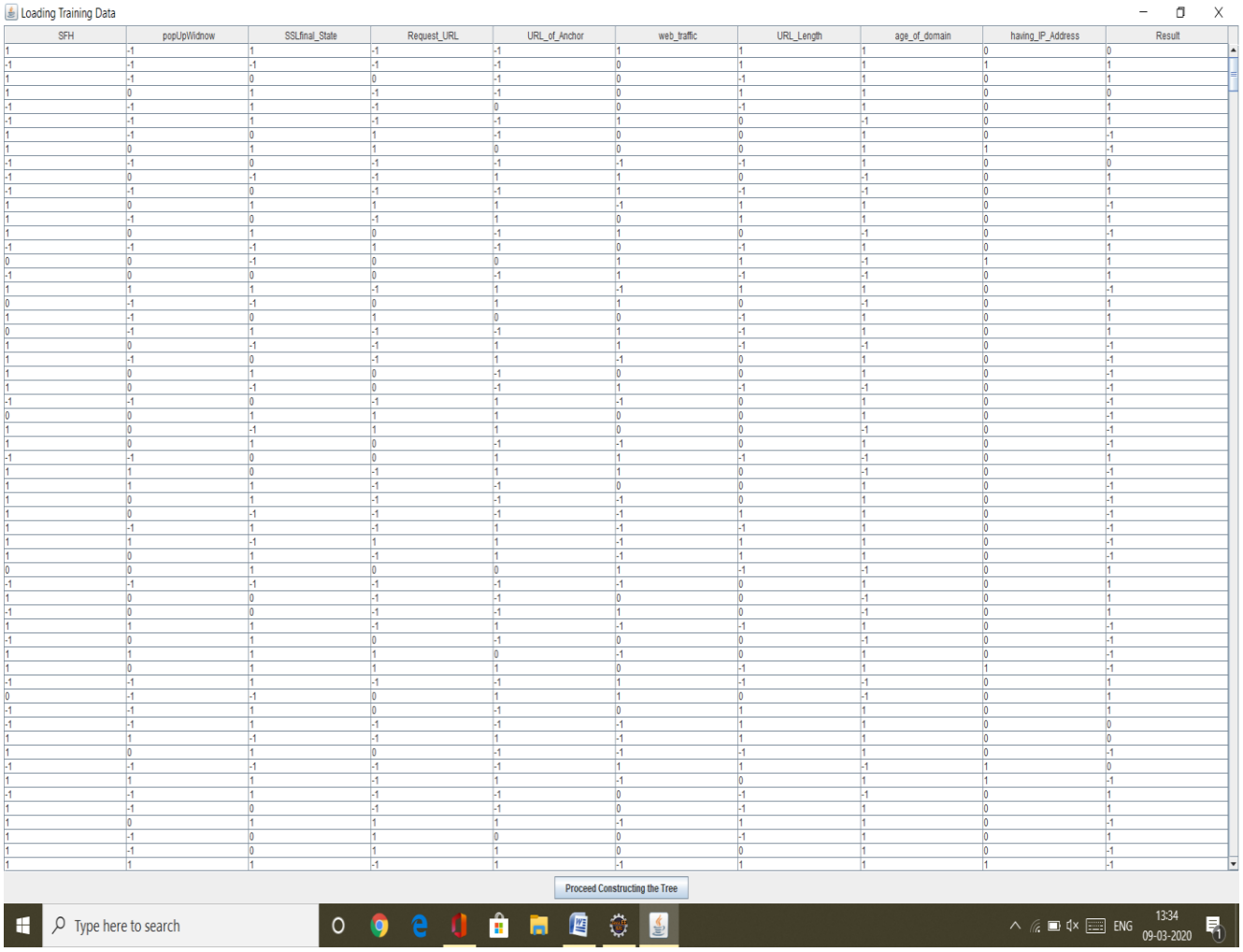

Fig.8.1 Training dataset

STEP 2: Proceed for generation of Decision Trees

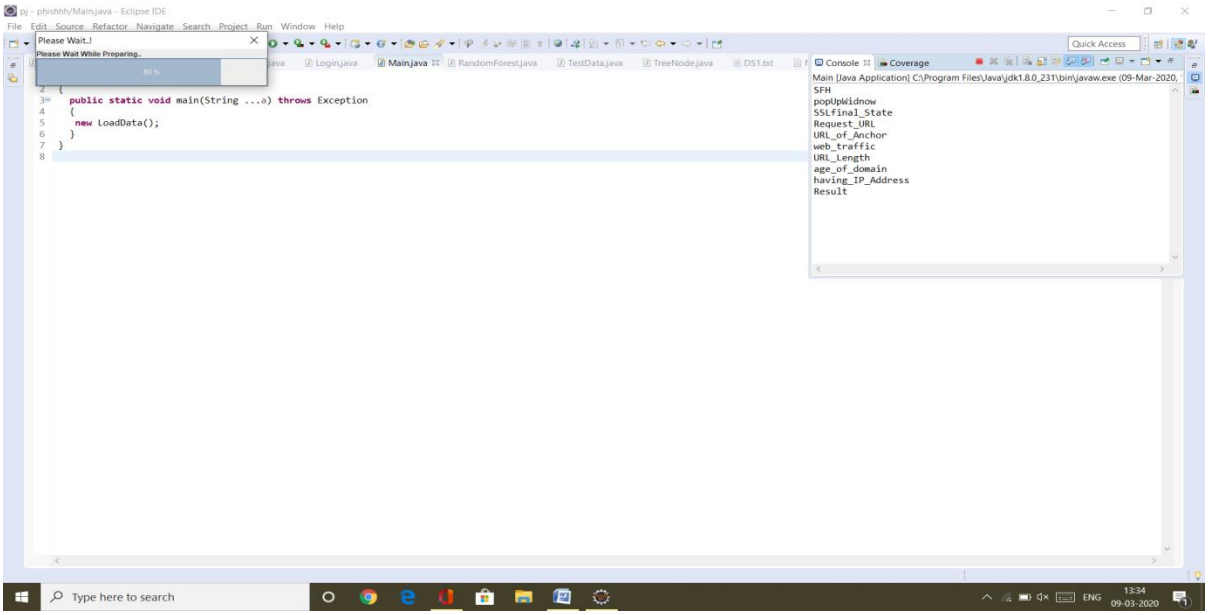

Fig.8.2 Processing

STEP 3: Construction Of Decision Trees

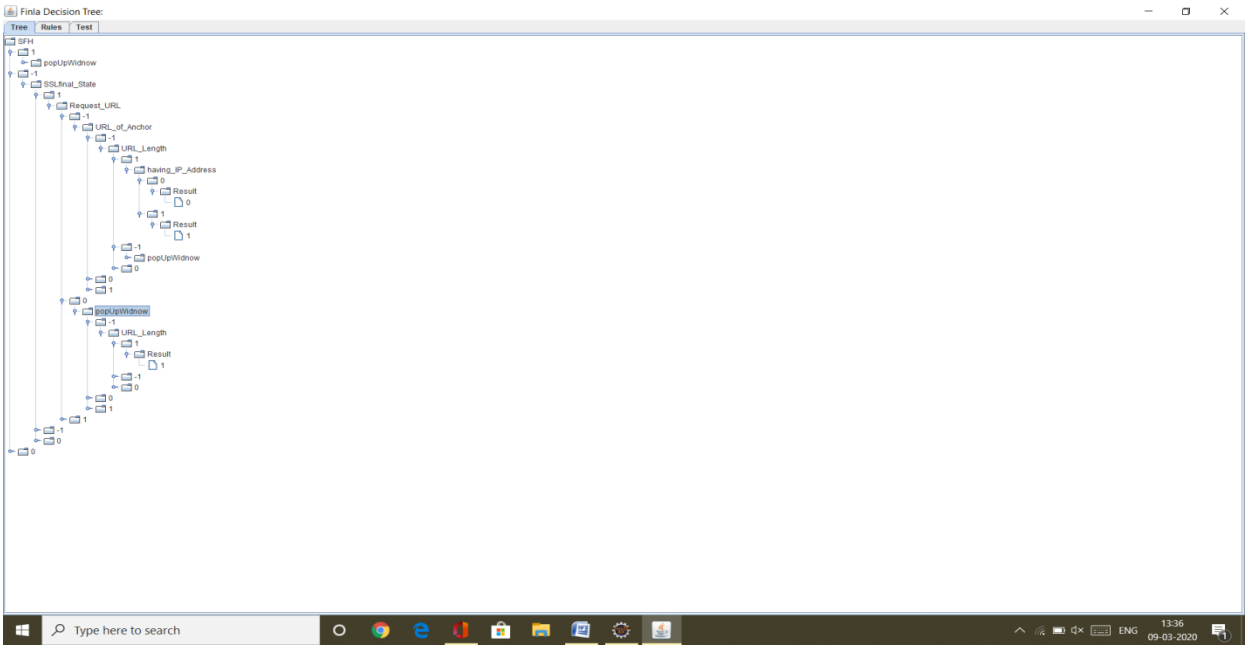

Fig.8.3 Decision Tree

STEP 4: Generation Of Rules

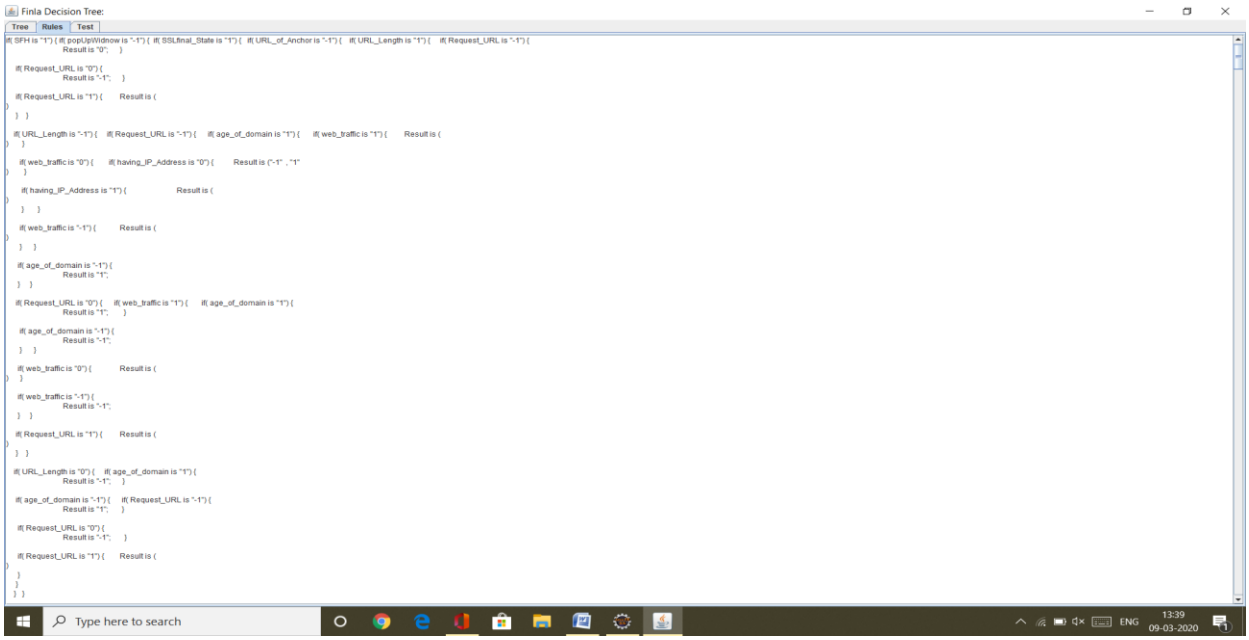

Fig.8.4 Generating Rules

STEP 5: Loading Test Dataset

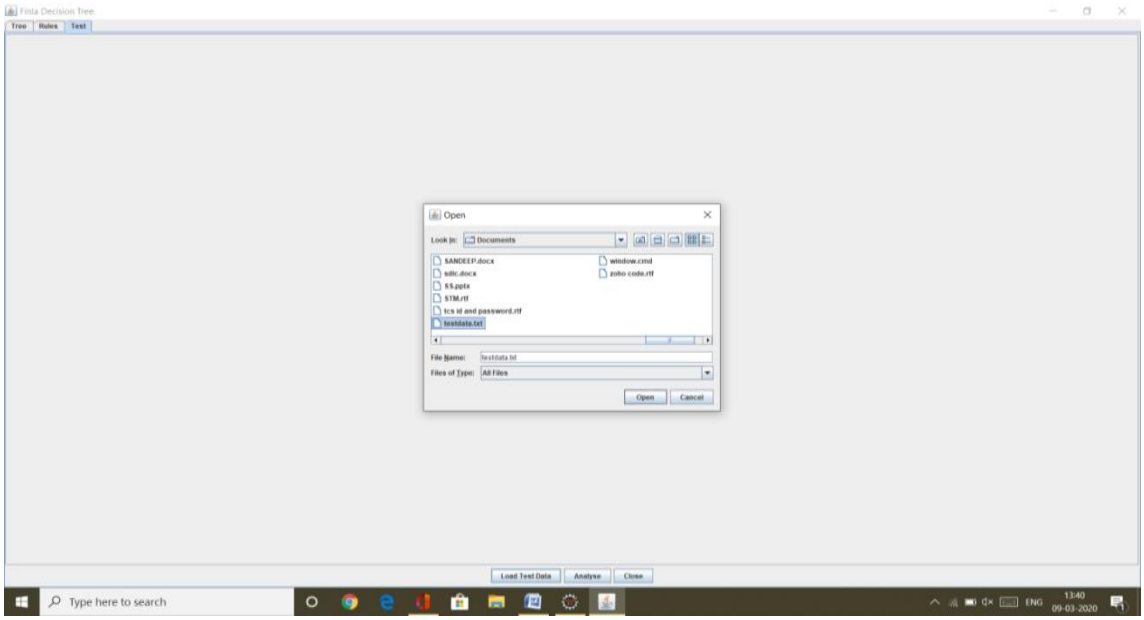

Fig.8.5 Testing Dataset

STEP 6: Analysing Results

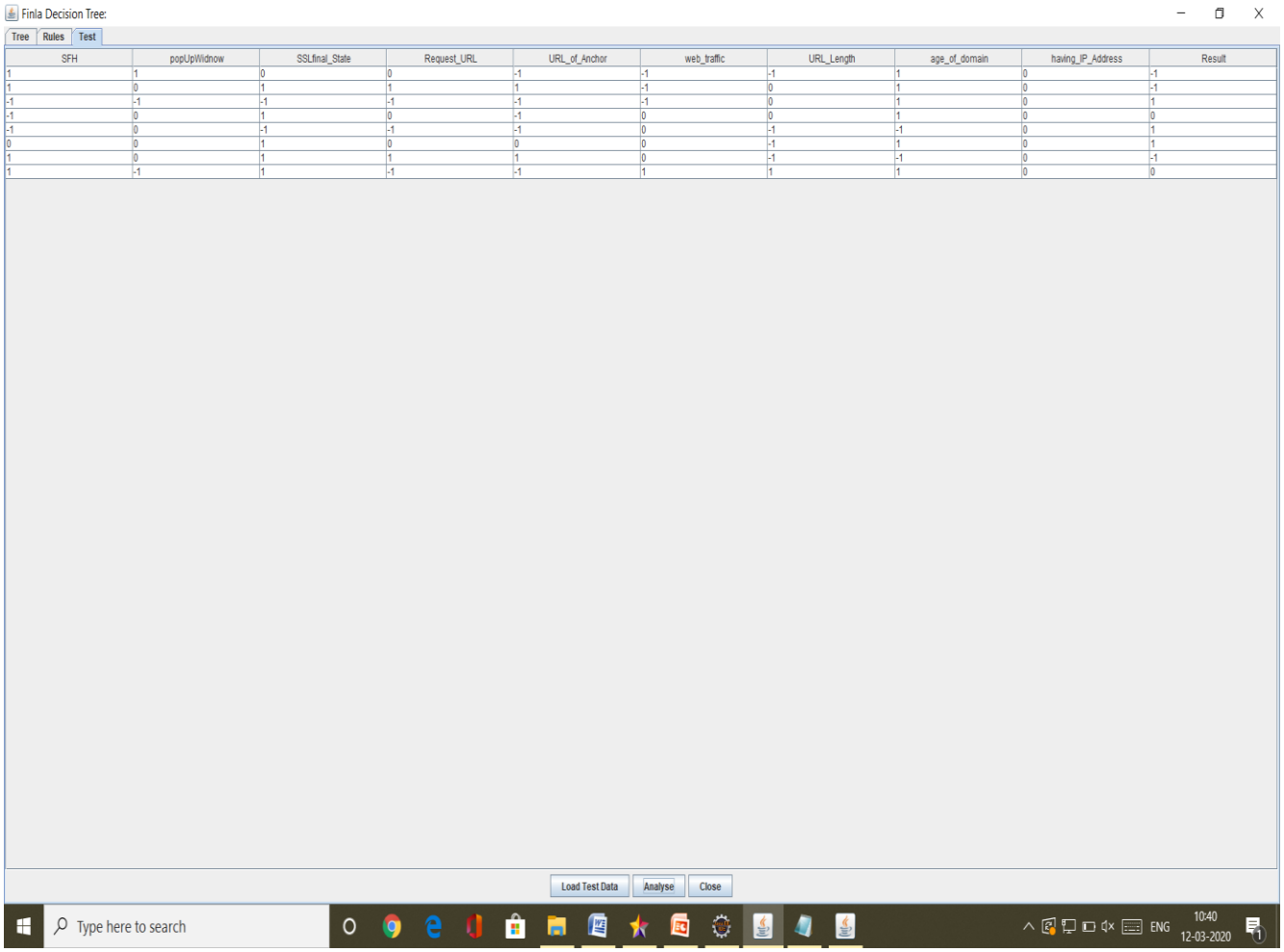

Fig.8.6 Analyse results

9. CONCLUSION AND FUTURE WORK

9.1 CONCLUSION

Phishing detection system is used in online banking & E-business. In the banking applications, the online transaction is very useful for all the customers of the corresponding bank. The client request the website for transaction process then this phishing detection system validate the website with four features such as WHOIS registration, IP address, domain and inter domain values of the corresponding requested website. The website related information's are stored in a WHOIS database for verification process. If the website is phishing website then this system will alert the user. If unfortunately the client enter into the phishing website then the phisher will theft all the information's entered by the user and apply those information into the original website. In the original website the administrator will create a new account and if already created account means user login into that and precede the transaction process. The client enter the account number and password for login into that website that password stored securely in database by using a MD5 algorithm. After login into the website this system generates the session key and send through the user's mobile. The phisher login into the original website means they can't get the session key so they enter a wrong session key then this system deny the access. The original user means the transaction is performed successfully.

9.2 FUTURE WORK

In the future, we plan on improving this work by combining this approach with a nature inspired (NI) technique.NI techniques (such as PSO or ACO) can be used to automatically and dynamically identify the best phishing features (from a feature space) that can be used to build a robust phishing email liter with very high classification accuracy. Using this technique will with no doubt enhance the predictive accuracy of a classier since electives classification of emails depends on the phishing features identified during the learning stage of the classification. Due to the rapid change in phishing attack patterns, current phishing detection techniques need to be greatly enhanced to electively combat emerging phishing attacks.

10. REFERENCES

REFERENCES

[1]. Anti-Phishing Working Group (APWG).APWG Homepage.http://www.antiphishing.org/,2007.

[2]. Blake Ross, Collin Jackson, Nicholas Miyake, Dan Boneh, and John C.Mitchell.Stronger Password Authentication Using Browser Extensions.In 14th Usenix Security Symposium, 2005.

[3]. Engin Kirda and Christopher Kruegel. Protecting Users against Phishing AttacksThe Computer Journal, 2006.

[4]. Fritz Schneider, Niels Provos, Raphael Moll, Monica Chew, and Brian Rakowski. Phishing Protection Design Documentation. http://wiki.mozilla.org/Phishing Protection: Design Documentation, 2007.

[5]. J. Ma, L. Saul, S. Savage, and G. Voel, Beyond blacklists: Learning to detect malicious web sites from suspicious urls, In The 15th ACM SIGKDD ker Conference On Knowledge Discovery and Date Mining, 2009.

[6]. Liu Wenyin, Guanglin Huang, Liu Xiaoyue, Zhang Min, and Xiaotie Deng. Detection of phishing webpages based on visual similarity. In 14th International on World Wide Web (WWW): Special Interest Tracks and Posters,2005.

[7]. Lobato DH,Lobato JM (2008).Bayes Machines for binary classification. Pattern Recognition Letters. Elsevier, 29: 1466- 1473

[8]. S Widodo 2017 Classification of Phishing Sites by Using Neural network perceptron and K-Nearest Neighbor *Information Management for Educators and Professionals* Vol 1 No 2 p145-154 E-ISSN: 2548-3331

[9]. K R Sahu and J Dubey 2014 A Survey on Phishing Attack *International Journal of Computer Applications* Vol 88 No 10

[10]. Hartatik, F W Wibowo, O Antoro 2018 Implementation of Recognizing Batik Motif Pattern Based-on Wavelet Transforms Comparison and Neural Network *Journal of Advanced Manufacturing Technology*

[11]. https://archive.ics.uci.edu/ml/machine-learning databases/00327/Training%20Dataset.arff, accessed on July 1, 2018 at 7:39 o'clock pm.

[12]. “Stopbadware-ip address report top 50 by number of reported urls.” http://stopbadware.org/reports/ip