SOCIAL ENGINEERING

A Project Report

Submitted by

KARTIK KUMAR: 21BCS8403

PRATIK MUKHERJEE: 21BCS8404

KRISHMA: 21BCS8405

In the partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION SECURITY

CHANDIGARH UNIVERSITY

2023

BONAFIDE CERTIFICATE

Certified that this project report of **“SOCIAL ENGINEERING”** is the Bonafide work of **Kartik Kumar, Pratik Mukherjee, Krishma** who carried out the project work undermy/our supervision.

Signature of the HoD	**Signature of SUPERVISOR**
(Mr. Aman Kaushik)	**(Mrs. Priyanka Jammwal)**
HoD of CSE – AIT	**Project Supervisor**

Submitted for the project viva-voce examination held on /11/2023

Signature of Internal Examiner	**Signature of External Examiner**

ACKNOWLEDGEMENT

First, we'd like to thank our supervisor **MS. PRIYANKA JAMMWAL** who was a constant source of alleviation. She encouraged us to think creatively and motivated us to work on this design without giving it an alternate study. She expressed full support and handed us with the different training aids that were needed to complete this design. She believed in us indeed when we couldn't believe that we could do it. We're also thankful to every member of this group. It was each existent's contribution that made this assignment a success. We were always there to lift each other up, and that was what helped us stay together till the end who guided us in this design. The group members continuously delved and tried to find out numerous effects related to the working of design and some other aspects which are helpful in unborn compass of this design. They worked day and night and proposed this system for our society so that they can mileage the benefits of this system. Every group member has his/ her unique part in this design, and we cannot suppose the success without the part of any group members. We thank our parents for always trusting in us and tutoring us to believe in our capacities and strengths and no way give up until the thing is achieved. We're thankful to all our musketeers who extended their moral support, and over all, we're thankful to God for being with us and giving us the wisdom and capability to do this design.

Thank You

TABLE OF CONTENTS

List of Figures

Figures ....Page No.

List of Tables

ABSTRACT

Our daily activities, such as our financial transactions, work-related activities, and other everyday activities, have moved to the internet as of late, so we are now exposed to increased risks because of cybercrime. One of the most common threats to internet users is URL-based phishing attacks. In this type of attack, the attacker exploits human vulnerability rather than software flaws. The malware targets both individuals and organizations, tricks them into clicking on URLs that appear secure, and steals confidential information or installs malware on our computers. Different machine learning algorithms are being applied for the detection of phishing URLs, that is, to classify a URL as phishing or legitimate. Researchers are constantly trying to improve the performance of existing models and increase their accuracy. In this work we aim to review various machine learning methods used for this purpose, along with datasets and URL features used to train the machine learning models. The performance of different machine learning algorithms and the methods used to increase their accuracy measures are discussed and analyzed. The goal is to create a survey resource for researchers to learn about current developments in the field. This will enable them to contribute to phishing detection models that yield more accurate results.

Keywords: —Phishing; URL features; Machine Learning; Phishing Detection, Machine Learning Algorithms, Confidential Information.

Chapter 1
INTRODUCTION

1.1 Background

Artificial intelligence is a new innovative wisdom that reviews and creates suppositions, strategies, procedures, and operations that recreate, grow and broaden mortal knowl-edge. ML is an arm of artificial intelligence and it's like (and constantly lap with) computational measures, that also concentrates on making prognostications with the use of PCs. Machine leaning has solid relationship with scientific enhancement, which tells styles, thesis and application regions to the field. ML is occasionally, in a while combined with data mining, but the data mining subfield focuses more on prepare- tory information disquisition and is called as unsupervised literacy.

ML can likewise be unsupervised and be employed to learn and set up pattern biographies for colourful realities and used to find important anomalies. Cyber security is a set of inventions and procedures intended to secure PCs, networks, systems and information from assaults and unapproved access, revision, or anni- halation. A system security frame comprises of a system assurance frame and likewise a PC protection frame. Every one of these fabrics incorpo- rates firewalls, antivirus programming, and intrusion discovery system (IDS). IDSs help find, decide and distinguish unapproved system conduct, for case, use, replicating, change and obliteration.

There are three important kind of network analysis for intrusion detection system:

- Abuse grounded discovery strategies mean to distinguish realized attacks by utilizing the marks of these attacks.
- Anomaly- grounded styles study the typical system and its conduct and distinguish anomalies as diversions from ordinary gets.
- mongrel discovery conflates anomaly and abuse discovery. It's employed to expand the rate of discovery of accepted intrusions and to drop the rate of false cons of unknown attacks.

The use of machine learning (ML) in cybersecurity is becoming increasingly prevalent, as demonstrated by Figure 1.1. ML is being applied to tasks such as IP business categorization and intrusion detection, making it an effective solution for combating zero-day attacks.

Researchers are exploring the use of ML techniques to develop customized business characteristics that can improve the accuracy of intrusion detection systems.

Phishing, which involves stealing an individual's personal and identity information online, has been a prevalent issue since its introduction in 1987. Skilled developers can create fake websites that closely resemble legitimate ones, making it difficult to recognize them as fraudulent. These phishing websites deceive users into providing their account details, often using HTTPS to give users a false sense of security. Since most financial transactions are conducted online, it is crucial to identify and block these fake websites.

According to the Anti-Phishing Working Group, there were 647,592 distinctive phishing sites recorded until September 2018. Once a hacker gains access to a user's login credentials, they can easily carry out malicious activities. ML can be a valuable tool in identifying and blocking these phishing sites, thus helping in the fight against cyber threats.

The use of ML in cybersecurity is on the rise, and it is being applied to a wide range of tasks to improve security. Phishing remains a prevalent issue, and it is critical to identify and block fraudulent websites to protect users' sensitive information. ML can help in this regard and is a valuable addition to cybersecurity.

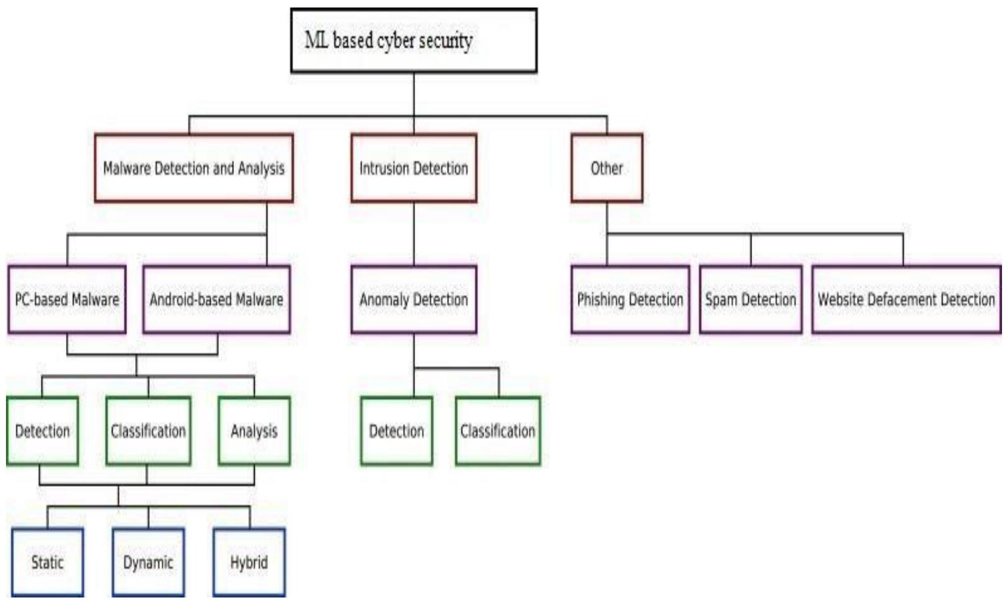


Figure 1.1 "Application of Deep Learning to Cybersecurity: A Survey"

Phishing attacks have been increasing in recent times, and to combat this issue, several approaches have been proposed. Different methods for detecting phishing attacks include blacklists, fuzzy rule-based, white list-based, cantina-based, machine learning-based, heuristic, and image-based approaches. Various studies have been conducted to detect different types of phishing attacks using a variety of techniques. However, phishing sites are designed to look like genuine websites, making it difficult for individuals to recognize them.

Some browsers have built-in anti-phishing techniques, but these may not be sufficient to protect against all types of attacks. Therefore, it is important to develop a framework that provides a robust solution to the problem of phishing attacks. Machine learning-based approaches have shown promising results in detecting and preventing phishing attacks, and researchers continue to explore and develop new methods to improve the effectiveness of these techniques.

1.2 Literature Servey

Phishing attacks remain a serious concern for individuals and organizations alike, and as a result, researchers have proposed a number of different approaches for detecting these attacks. One common theme among many of these approaches is the importance of feature selection in building an effective model.

One such model is the OFS-NN model, which uses an optimal feature selection technique based on a newly generated index called the feature validity value (FVV). This approach helps to overcome the problem of over-fitting of the neural network by selecting only the most relevant features for detecting phishing websites.

Another approach is the Fuzzy Rough Set (FRS) theory, which uses standardized datasets to identify the most appropriate features for detecting phishing. This approach was tested on a dataset of 14,000 website samples to build a generalized detection model.

Feature engineering is crucial for detecting phishing websites, and researchers have proposed several methods for achieving this. The Multidimensional Phishing Detection Feature (MFPD) approach is designed to rapidly detect phishing by using deep learning techniques. The Web Crawler-based Phishing Attack Detector (WC-PAD) takes input features from the web's content, traffic, and URL to accurately detect phishing attacks.

Other approaches focus on using deep learning techniques, such as PhishingNet, which uses deep learning to detect phishing URLs in real-time. Another approach involves using a client-side solution that matches distinctive features from the source code of webpages and URLs.

The Parse Tree Validation method is another innovative approach that intercepts all hyperlinks on a webpage using Google's API to build a parse tree and determine whether the website is phishing or legitimate. The Random Forest classifier is another model used for detecting phishing websites using URL methods.

One online tool combines collection, validation, and detection of phishing websites in real-time by monitoring the PhishTank blacklist. The Fresh-Phish framework generates machine learning data for phishing websites using 30 different features of a website that can be queried using Python.

The Phishing-Detective framework evaluates the authenticity of websites by building a bond between content-based heuristics and the legitimacy of the website. Finally, the C4.5 decision tree classifier is used in combination with certain URL features to detect phishing websites.

While there are many schemes for detecting phishing websites, the visual similarity scheme uses screenshots of websites to identify phishing attacks. However, this approach can be inaccurate if there are multiple similar websites, and therefore, it may not be effective in recognizing the target website.

Overall, researchers have proposed a variety of different approaches to detect phishing attacks, many of which rely on effective feature selection and deep learning techniques. These models can help individuals and organizations to protect themselves from the harmful effects of phishing attacks.

Table 1.1: Literature Survey.

No	Paper Title	Method/Techniques	Publish year	Limitations
1	OFS-NN: ”An Effective Phishing Websites Detection Model Based on Opti- mal Feature Selection and Neural Network”	Proposed method has 3 stages:1. Defines a new in- dex -FVV. 2. Designs an op- timal feature selection algo- rithm.3. Produce the OFS- NN model	2019	The continuous growing of features that are sensitive of phishing attacks need collection of more features for the OFS
2	”Fuzzy Rough Set Feature Selection to Enhance Phishing Attack Detection”	The proposed method uses Fuzzy Rough Set (FRS) theory to identify the fea- tures. The decision bound- ary is decided lower and up- per approximation region. Using the lower and up- per approximation member- ships, a set member is de- cided to which category it belongs	2019	The specific features used in the method is not specified.

3	”Phishing Website Detection based on Multidimensional Features driven by Deep Learning”	The proposed method has the following stages: 1.char- acter succession features of the URL are extricated as well as utilized for fast char- acterization 2. the LSTM (long short-term memory) network is utilized to catch setting semantic and depen- dency features of URL char- acter groupings. 3. softmax classifies the features ex- tracted	2019	It requir es more com- putation an d therefore an expensive method
4	”WC-PAD: We b Crawling base d Phishing Atta ck Detection”	It is a 3-phase detection of phishing attack approach. The 3 phases of WC-PAD are 1) blacklist of DNS 2) Approach based on Heuris- tics and 3) Approach based on Web crawler. Feature ex- traction as well as phishing attack detection both makes use of web crawler.	2019	Time c on- suming as it involves three phases and each website has to go through the three phases.

5	”Phishing URL Detection via CNN and Attention-Based Hier- archical RNN”	CNN module is used to derive representation of spatial feature that is character level of the URLs. Then the representational features are combined by using a CNN of 3 layers to create precise feature repre- sentations of URLs. That is then used for training the classifier of phishing URLs.	2019	false posit ive rate is high
6	”An Adaptive Machine Learning Based Approach for Phish- ing Detection Using Hybrid Features”	A phishing detection system was developed by making use of classifier of Machine learning called XCS. It is an adaptive ML technique that is online. This advances a lot of rules called classi- fiers. This model derives 38 features from source code of webpage and URLs.	2019	

7	”Phishing Detection in Websites using Parse Tree Valida- tion”	If the number of recurrence of root node is: 1. more than half the number of nodes, then probability of authen- ticity is more. 2. quarter the number of nodes, the probability of authenticity is moderate. 3. less than the quarter number of nodes, then probability of authen- ticity is low which means phishing probability is high.	2018	The false neg- ative and false positive rates are high.
8	”A new method for Detection of Phishing Websites: URL Detec- tion”	The three major phases in this work are Parsing, Heuristic Classification of data, Performance Analysis in this model. All of these phases use various and dis- tinctive methods for data processing to get results that are better.	2018	Does not give full information about the tech- niques used.

9	”PhishBox: An approach for phishing validation and detec- tion”	The approach that is proposed makes use of 2 phase detection model to increase its performance. 1. An ensemble model is designed for validating the phishing data and for decreasing the cost of labeling manually,active learning is applied. 2.The model for detection is be- ing trained using these validated data.	2018	The bla ck- list contained invalid data when monitored with an interval set as 12 hours.
10	”Fresh-Phish:A framework for Auto- Detection of Phishing Websites”	This framework was developed considering there are no other open source frame- works which, for a given website, measures the fea- tures. The work also created an updated set of data that could be used by researchers for their work. Analysis of TensorFLow based neural network and linear classifier and SVM with kernels both Gaussian and linear were done against dataset of FreshPhish	2017	Less accuracy and assumption of the dataset considered for legitimate web- site is accurate.

11	”Phishing Website Detection Frame- work Through Web Scraping and Data Mining”	A web crawler that scrapes the constituents of both le- gitimate and phishing web- sites was developed. The constituents were then an- alyzed to get the heuris- tics rate and their commit- ment scale factor towards the wrongness of a site. A data mining tool was used to analyze the data that was derived from the web scraper and patterns were found.	2017	Exact accuracy of the model is not mentioned.
12	”Phishing Sites Detec- tion based on C4.5 Decision Tree Algo- rithm”	The approach proposed makes use of features that were extracted from the URL to make decision about the legitimacy of the URL given as input. To generate the rules, the c4.5 algorithm was used. The rules produced are utilized to order the submitted URL as genuine or phishing with better productivity.	2017	Overall accu- racy is less as the paper con- siders limited URL features.

13	”Visual Similarity-based Phis hing Detection Scheme using Image and CSS with Target Website Finder”	The main focus is on the fact that authentic web-sites are usually linked by many websites and those websites are regarded as le-gitimate, the screenshot and CSS of which are stored in a database. Because CSS is a file which characterizes the sites visual substance, as-saulter regularly take real CSS to imitate the real site. Hence, by finding the site which counterfeits appear-ance or CSS of real site, we identify phishing site and its objective at the same time.	2017	The websites that are linked at least by one website are also recorded in the white list assuming it to be legitimate.

The above discussion highlights the crucial role played by machine learning techniques in various cybersecurity applications, making it a promising area for further research. However, implementing these techniques in practice can be challenging due to several limitations. While numerous approaches have been proposed for detecting phishing website attacks, each of them has its own limitations. Thus, the goal of this project is to develop a novel machine learning technique for the detection of phishing website attacks. This technique will aim to address the existing limitations and provide an effective solution to identify and prevent phishing attacks. By leveraging the latest advances in machine learning, this project aims to contribute to the development of a more robust and reliable system for protecting individuals and organizations from phishing threats.

1.3 Motivation

There are many anti-phishing techniques available that can help us protect ourselves from phishing sites. Some of the popular web browsers like Mozilla Firefox, Safari, and Google Chrome make use of Google Safe Browsing (GSB) service to block phishing websites. There are also many other tools like McFee Site Advisor, Quick Heal, Avast, and Netcraft that are widely used for this purpose.

GSB analyzes a URL using the blacklist approach. However, one of the main disadvantages of GSB is that it is unable to detect phishing websites if the blacklist is not updated. On the other hand, Netcraft records a website as phishing only when it is sure 100% that the website is phishing. The warning is given only when the user clicks the right button on the icon to find the risk rating. This means that the risk lies with the individual who doesn't check the rating or makes a decision to use it after checking the rating.

There are some software tools like QuickHeal and Avast that provide security against security attacks online. However, it is important to note that these tools must be installed independently. A lay person might never install these tools if they are not aware of practices like phishing. In such cases, people rely solely on the GSB service. Therefore, awareness regarding such anti-phishing tools and phishing is very important.

It is also important to note that no individual should fully rely on anti-phishing tools as they might lead to misclassification. For instance, the functioning of Avast anti-virus was checked after installing it, and it was found that the Avast browser was not able to successfully find a phishy URL that was successfully determined by Netcraft and GSB.

1.4 Problem Statement

Phishing detection system designed to deal with cyber-attacks caused due to fake web links that steal confidential informa6tion and even harming the information using malware, virus etc.

In this first dataset of phishing and legitimate emails will be collected and processed further for analysis is used either decision tree or neural network to analyse the dataset.

The problem is derived after making a thorough observation and study about the method of classification of phishing websites that makes use of machine learning techniques. We must design a system that should allow us to:

- Accurately and efficiently classify the websites into legitimate or phishing.
- Time consumed for detection should be less and should be cost effective.

1.5 Problem Formulation

- There are many Anti phishing techniques that helps us protect from phishing sites. Mozilla Firefox, Safari and Google chrome makes use of Google Safe Browsing (GSB) [13] service that will block the phishing websites.
- The main disadvantage of GSB was that it was unable to detect the phishing website since updating of blacklist was not done. In case of Net craft, a website that phishing was recorded as phishing although it wasn't blocked. The blocking is done by Net craft only when it is sure 100% that the website is phishing.
- Hence the focus of the project is on overcoming the disadvantage of GSB using machine learning (ML) methods for network analysis of intrusion detection.
- One of the most widely used algorithms in machine learning technology. Decision tree algorithm is easy to understand and also easy to implement. Decision tree begins its work by choosing best splitter from the available attributes for classification which is considered as a root of the tree.
- Random forest algorithm is one of the most powerful algorithms in machine learning technology and it is based on concept of decision tree algorithm. Random forest algorithm creates the forest with number of decision trees. High number of trees gives high detection accuracy.

1.6 Aim and objectives

The Project’s Objectives are:

1. Study automatic phishing detection methods.
2. Identify appropriate machine learning techniques for phishing detection.
3. Define a solution using the selected method.
4. Select a suitable dataset for the problem statement.
5. Apply appropriate algorithms for phishing attack detection.
6. Develop a system to accurately and efficiently classify websites as legitimate or phishing.
7. Optimize the system to reduce time consumed for detection.
8. Ensure the system is cost-effective.
9. Test and evaluate the system's performance.
10. Improve the system based on feedback and evaluation results.

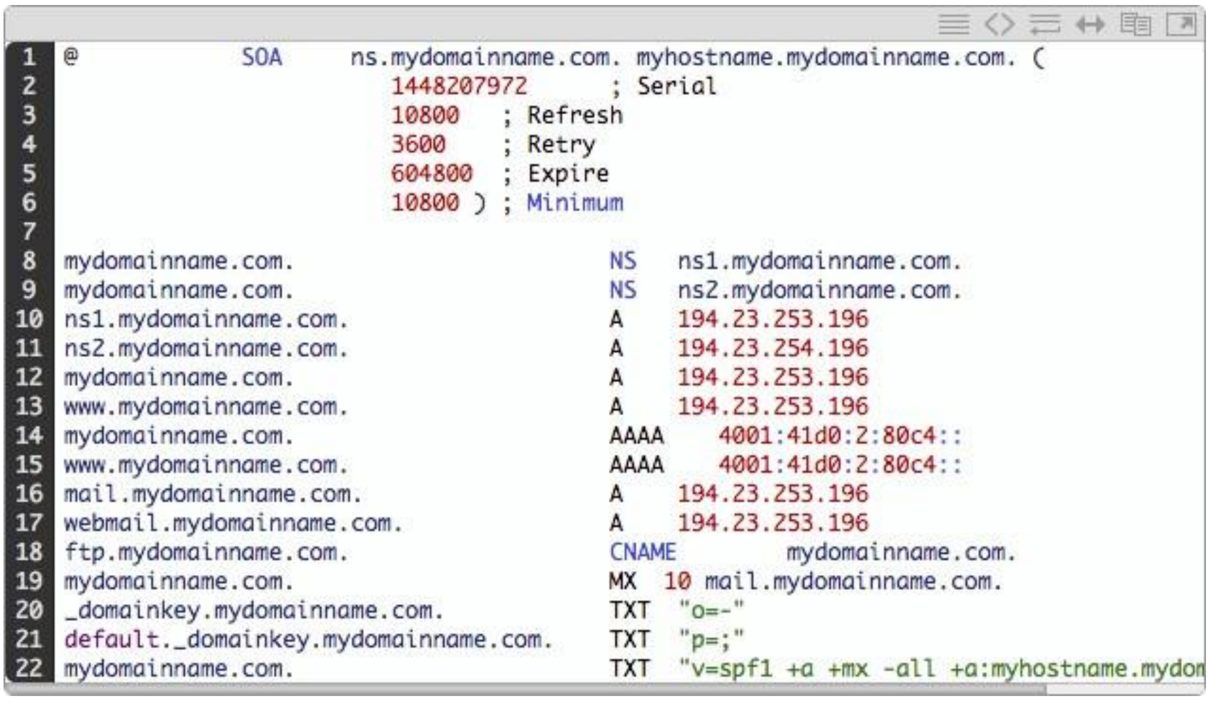
1.7 Scope

Future advancements in technology may allow for the acquisition of structured datasets for phishing, which could lead to faster and more efficient phishing detection compared to current techniques. Combining multiple classifiers may also improve accuracy in detecting phishing attempts. Moreover, researchers plan to explore various phishing techniques that use lexical, network-based, content-based, webpage-based, and HTML and JavaScript features of web pages to enhance the performance of the system. Specifically, features extracted from URLs may be passed through different classifiers for analysis. This approach may help to further enhance the ability to detect phishing attempts and ultimately improve overall cybersecurity**.**

1.8 Features of URL

1.8.1 DNS_Record

A DNS record is a specific type of data stored in the Domain Name System (DNS) database, which maps domain names to IP addresses. It contains information such as the IP address, canonical name, and aliases associated with a domain name. DNS records are used to facilitate communication between computers and to enable web services such as email, websites, and remote access. Common types of DNS records include A records for mapping domain names to IP addresses, MX records for email routing, and CNAME records for aliasing one domain name to another.



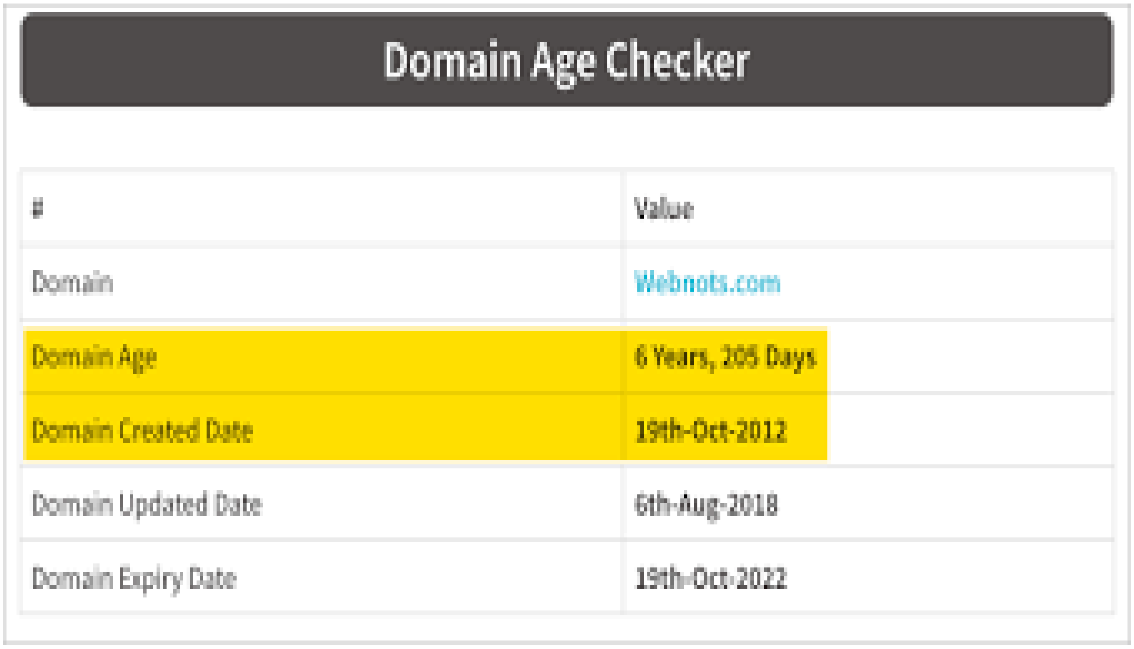
```
@             SOA     ns.mydomainname.com. myhostname.mydomainname.com. (
                         1448207972      ; Serial
                         10800    ; Refresh
                         3600     ; Retry
                         604800   ; Expire
                         10800 ) ; Minimum

mydomainname.com.                         NS    ns1.mydomainname.com.
mydomainname.com.                         NS    ns2.mydomainname.com.
ns1.mydomainname.com.                     A     194.23.253.196
ns2.mydomainname.com.                     A     194.23.254.196
mydomainname.com.                         A     194.23.253.196
www.mydomainname.com.                     A     194.23.253.196
mydomainname.com.                         AAAA     4001:41d0:2:80c4::
www.mydomainname.com.                     AAAA     4001:41d0:2:80c4::
mail.mydomainname.com.                    A     194.23.253.196
webmail.mydomainname.com.                 A     194.23.253.196
ftp.mydomainname.com.                     CNAME        mydomainname.com.
mydomainname.com.                         MX  10 mail.mydomainname.com.
_domainkey.mydomainname.com.              TXT   "o=-"
default._domainkey.mydomainname.com.      TXT   "p=;"
mydomainname.com.                         TXT   "v=spf1 +a +mx -all +a:myhostname.mydon
```

Figure 1.2 DNS_Record

1.8.2 Domain_Age

Domain age refers to the length of time since a domain name was first registered. It is an important factor in search engine optimization (SEO), as search engines tend to give more credibility to older domains. Older domains are assumed to be more established and trustworthy than new ones, and they are often associated with higher-quality content and more robust backlink profiles. Additionally, domains that have been registered for a longer period are less likely to be associated with spam or other malicious activity, which can negatively impact their search engine rankings. Therefore, domain age is a valuable metric for evaluating the credibility and authority of a website.



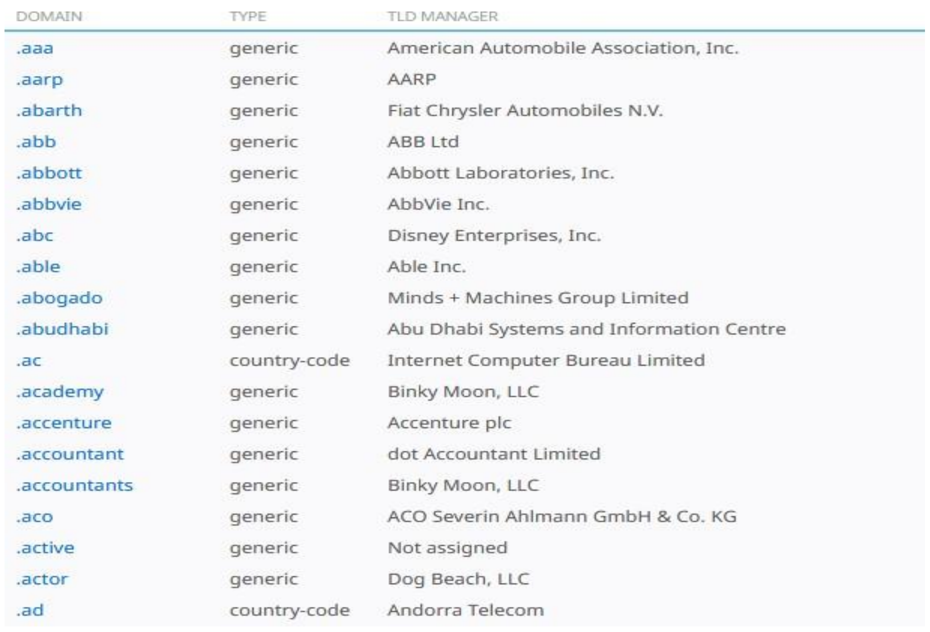
Domain Age Checker

#	Value
Domain	Webnots.com
Domain Age	6 Years, 205 Days
Domain Created Date	19th-Oct-2012
Domain Updated Date	6th-Aug-2018
Domain Expiry Date	19th-Oct-2022

Figure 1.3 Domain_Age

18.3 Domain_End

Domain end refers to the end of a domain name, which is also known as a top-level domain (TLD). The domain end is the final segment of a domain name that comes after the last dot. Examples of popular domain ends include .com, .org, .net, and .edu. The domain end provides information about the type of organization or entity associated with a domain name. For example, .com is typically used for commercial websites, .org for non-profit organizations, .net for networking services, and .edu for educational institutions. The domain end is an important factor to consider when choosing a domain name, as it can impact the perceived credibility and authority of a website.



DOMAIN	TYPE	TLD MANAGER
.aaa	generic	American Automobile Association, Inc.
.aarp	generic	AARP
.abarth	generic	Fiat Chrysler Automobiles N.V.
.abb	generic	ABB Ltd
.abbott	generic	Abbott Laboratories, Inc.
.abbvie	generic	AbbVie Inc.
.abc	generic	Disney Enterprises, Inc.
.able	generic	Able Inc.
.abogado	generic	Minds + Machines Group Limited
.abudhabi	generic	Abu Dhabi Systems and Information Centre
.ac	country-code	Internet Computer Bureau Limited
.academy	generic	Binky Moon, LLC
.accenture	generic	Accenture plc
.accountant	generic	dot Accountant Limited
.accountants	generic	Binky Moon, LLC
.aco	generic	ACO Severin Ahlmann GmbH & Co. KG
.active	generic	Not assigned
.actor	generic	Dog Beach, LLC
.ad	country-code	Andorra Telecom

Figure 1.4 Domain_End

1.8.4 Redirection

Redirection in URLs is the process of directing website visitors from one URL to another. This can be done for a variety of reasons, such as to update an old URL, to merge multiple pages into a single page, or to change a page's location permanently or temporarily. Redirection can be achieved using various HTTP status codes, including 301, 302, and 307. When a user clicks on a redirected URL, their web browser automatically sends a request to the new URL, and the user is directed to the new page. Proper redirection is crucial for maintaining search engine rankings, avoiding broken links, and providing a seamless user experience for website visitors.

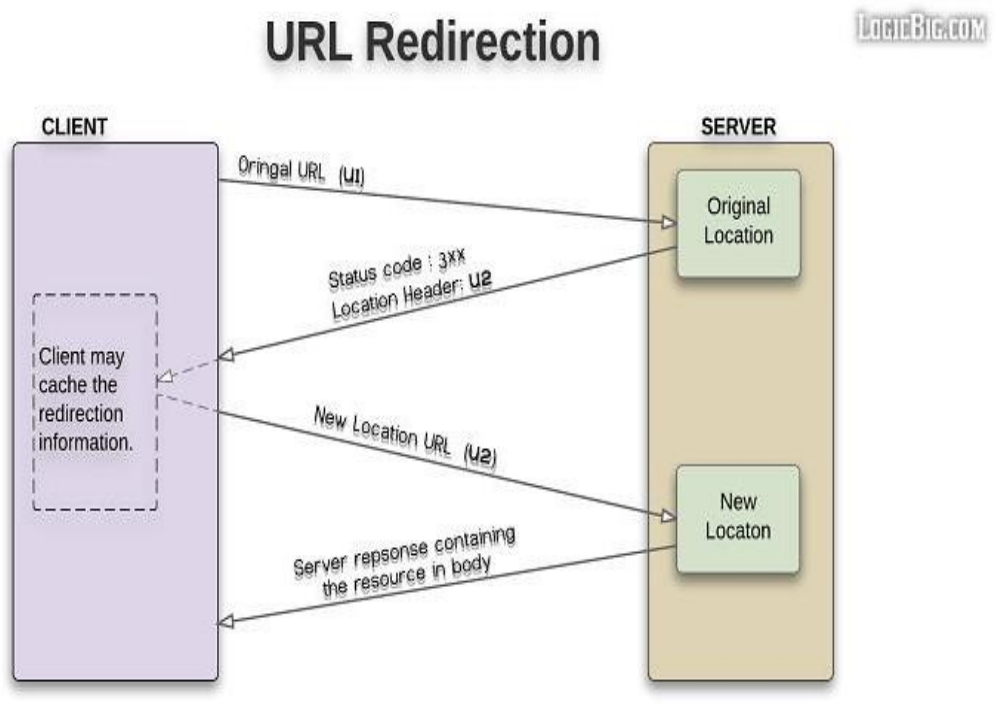
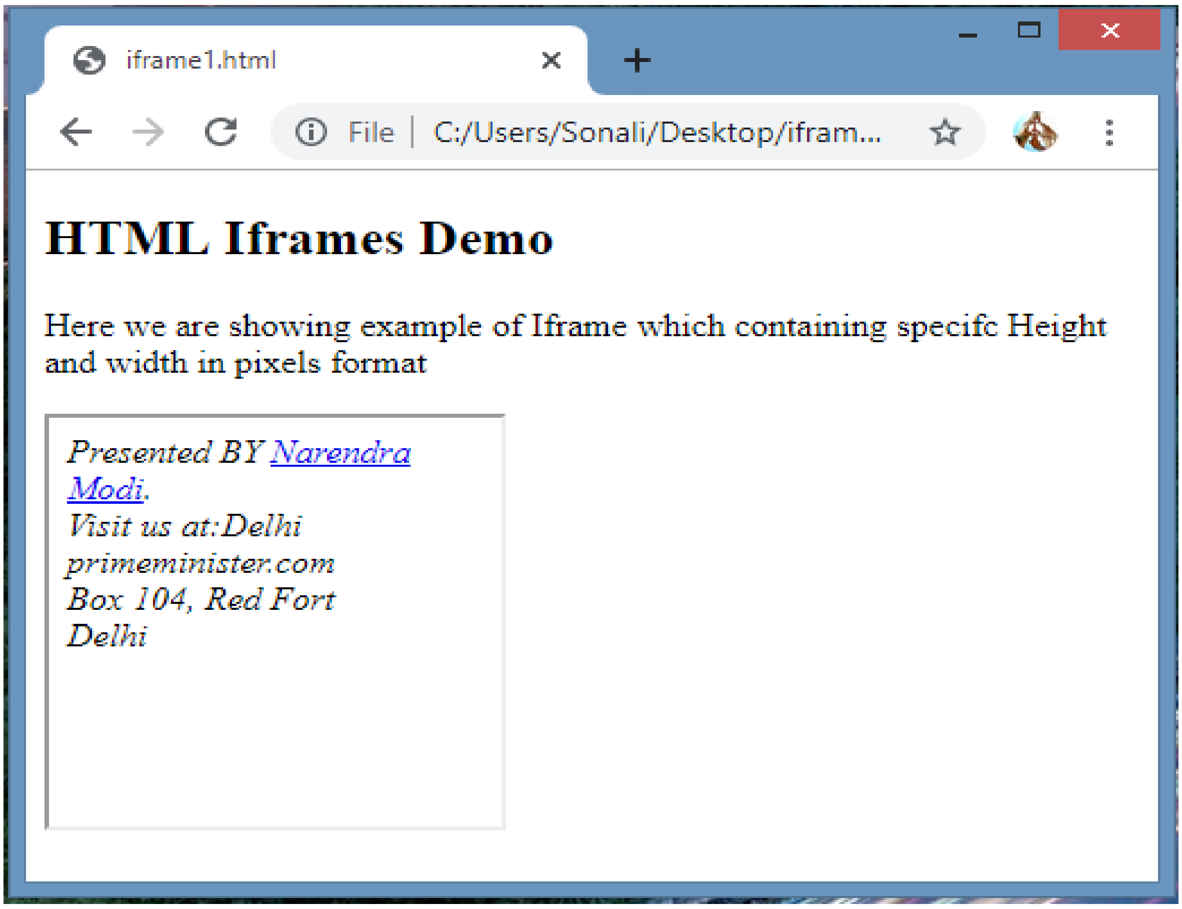


Figure 1.5 Redirection

1.8.5 iframe

iFrame is a tag used in web development that allows embedding one HTML document within another. It is often used to display content from another website or web page within an existing page. The iFrame tag creates a container within a web page where a separate HTML document can be displayed, usually from a different domain. This allows web developers to integrate content from other sources into their own pages, such as embedding videos or displaying social media feeds. However, iFrames can also be used for malicious purposes, such as phishing attacks or to inject malware. Therefore, website owners should use caution when using iFrames and ensure that they are not being used for malicious purposes.



HTML Iframes Demo

Here we are showing example of Iframe which containing specifc Height and width in pixels format

Presented BY Narendra Modi.
Visit us at:Delhi
primeminister.com
Box 104, Red Fort
Delhi

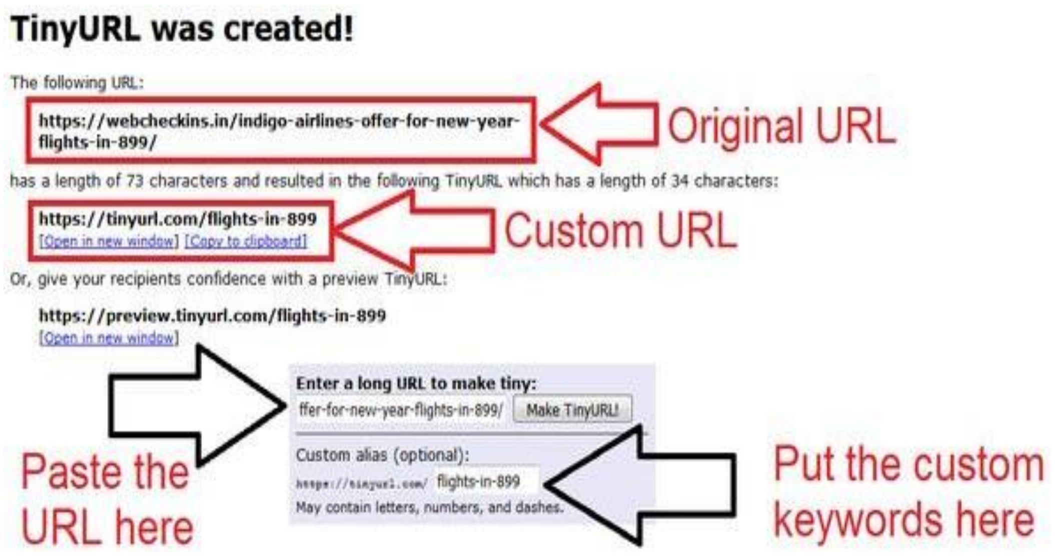
Figure 1.6 iFrame

1.8.6 TinyURL

TinyURL is a web service that allows users to create shorter, more manageable versions of long URLs. The service works by taking a long URL and generating a shortened version that redirects to the original URL when clicked. This can be useful in situations where long URLs are difficult to share, such as on social media platforms or in email messages.

To create a TinyURL, a user simply pastes the long URL into the TinyURL website, and the service generates a unique, shortened version of the URL. The user can then copy and share the shortened URL with others. When someone clicks on the shortened URL, they are redirected to the original, long URL.

One potential downside of TinyURL is that it can be used to hide malicious or spammy links. For this reason, it is important to exercise caution when clicking on TinyURL links, particularly if you don't know and trust the sender. However, overall, TinyURL is a useful tool for making long URLs more manageable and easier to share.



TinyURL was created!

The following URL:

https://webcheckins.in/indigo-airlines-offer-for-new-year-flights-in-899/

has a length of 73 characters and resulted in the following TinyURL which has a length of 34 characters:

https://tinyurl.com/flights-in-899
[Open in new window] [Copy to clipboard]

Or, give your recipients confidence with a preview TinyURL:

https://preview.tinyurl.com/flights-in-899
[Open in new window]

Enter a long URL to make tiny:
ffer-for-new-year-flights-in-899/ Make TinyURL!

Custom alias (optional):
https://tinyurl.com/ flights-in-899
May contain letters, numbers, and dashes.

Figure 1.7 TinyURL

1.8.7 HTTPs Domain

HTTPS (Hypertext Transfer Protocol Secure) is an encrypted version of HTTP, the protocol used for transmitting data over the internet. HTTPS is used to provide a secure connection between a web server and a web browser, preventing unauthorized access or tampering with the data being transmitted.

HTTPS domains are websites that use HTTPS to encrypt data transmitted between the website and the user's web browser. HTTPS domains use SSL/TLS certificates to establish a secure connection between the web server and the user's browser. These certificates are issued by trusted Certificate Authorities (CA) and provide a way for users to verify that they are communicating with the intended website and not an impostor.

HTTPS domains are becoming increasingly important due to the sensitive nature of the information transmitted over the internet, such as login credentials, payment information, and personal data. Web browsers are also beginning to display warnings for websites that do not use HTTPS, making it more important than ever for website owners to ensure that their sites use HTTPS. By using HTTPS, website owners can provide a secure and trustworthy experience for their users, protecting both the users' data and the website's reputation.

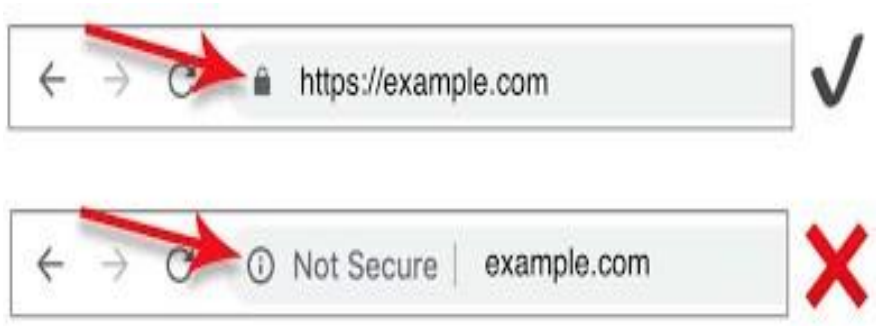


Figure 1.8 HTTPs_Domain

1.8.8 IP Address

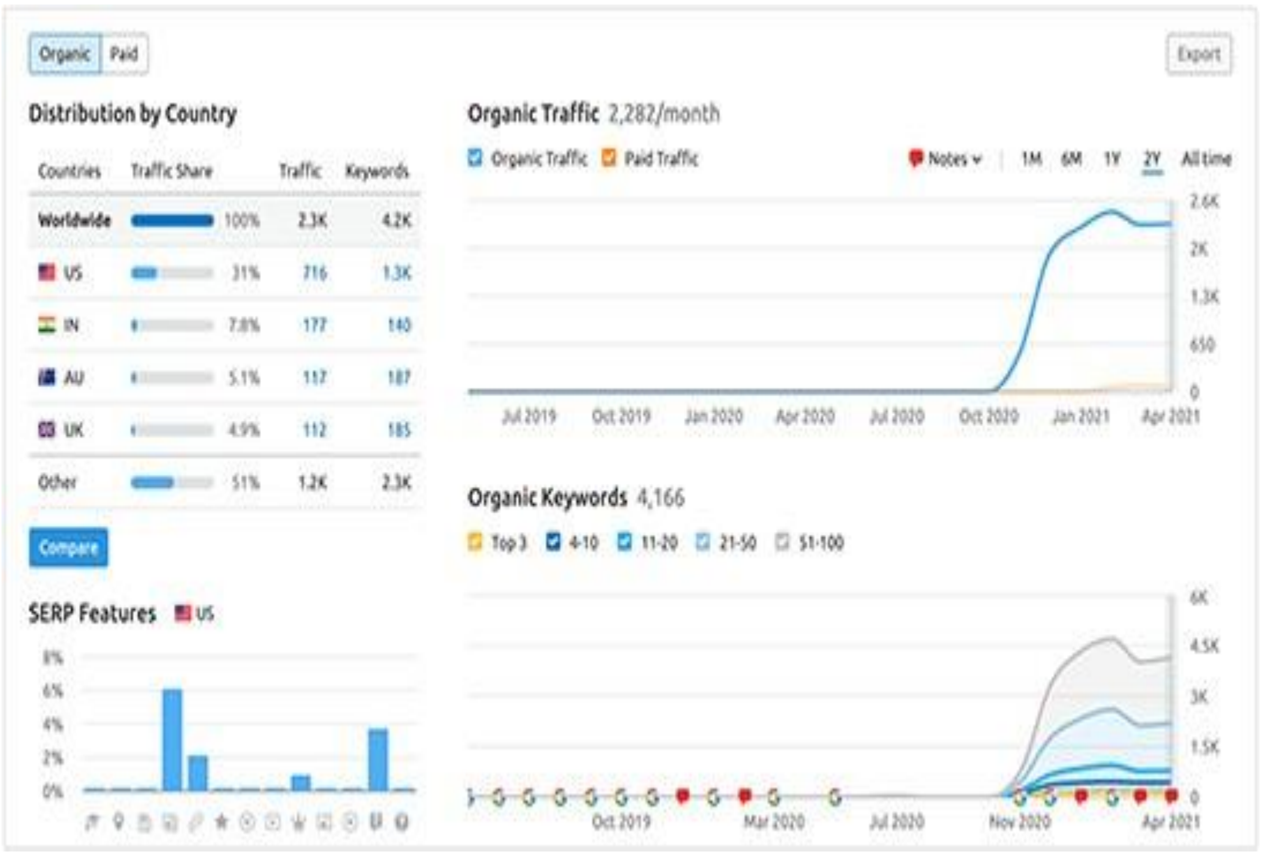
An IP address is a numerical label assigned to every device connected to the internet that uses the Internet Protocol (IP) to communicate. It serves as a unique identifier that allows devices to communicate with each other across the internet. IP addresses consist of four sets of numbers separated by dots, such as 192.168.1.1. There are two types of IP addresses: IPv4 and IPv6. IPv4 is the older standard and consists of 32-bit addresses, while IPv6 is the newer standard and consists of 128-bit addresses. IP addresses are essential for internet communication, allowing devices to send and receive data packets across the internet and enabling users to access websites, send emails, and engage in other online activities.

yourdomain.com
192.x.x.x
URL to IP

Figure 1.9 IP Address

1.8.9 Web_Traffic

Web traffic refers to the amount of data sent and received by visitors to a website. It typically includes the number of page views, unique visitors, and time spent on the site, among other metrics. Web traffic can be measured using various tools, such as Google Analytics, which provides detailed information about how visitors interact with a website. The volume of web traffic a site receives can be influenced by a variety of factors, such as the quality of its content, the effectiveness of its marketing strategies, and its search engine optimization (SEO) efforts. Understanding web traffic patterns is important for website owners, as it can help them make informed decisions about how to optimize their site for better performance, increase their visibility online, and ultimately achieve their business objectives.



Distribution by Country

Countries	Traffic Share	Traffic	Keywords
Worldwide	100%	2.3K	4.2K
US	31%	716	1.3K
IN	7.8%	177	140
AU	5.1%	117	187
UK	4.9%	112	185
Other	51%	1.2K	2.3K

Figure 1.10 Web_Traffic

Chapter 2

FUNDAMENTALS

Classification is a supervised learning approach in ML and statistics, where a computer program learns from input data and uses this knowledge to classify new observations. In the context of detecting phishing URLs, several classification techniques are used, such as Decision Tree, Random Forests. To improve the accuracy of phishing URL detection, researchers analyse different features of URLs, such as lexical, network-based, content-based, webpage-based, and HTML and JavaScript features. By extracting and analysing these features, researchers can classify URLs as legitimate or phishing attempts, enhancing cybersecurity measures. In summary, classification techniques play a crucial role in detecting phishing URLs, and analysing various features can improve the accuracy of detection methods.

2.1 Decision Tree

A decision tree is a popular machine learning algorithm that is used for classification and prediction tasks. The decision tree model consists of nodes, branches, and leaf nodes. The nodes in the decision tree represent specific attributes or features that are used to test the data. The branches represent the different possible outcomes or paths that the algorithm can take based on the values of the features. The leaf nodes are the final output or result that the decision tree predicts.

Let's understand it with example, Consider Fig 2.1, let's say we want to predict whether a person is fit or unfit based on data such as their diet patterns, physical activity, age, and other factors. The decision tree algorithm would use the input data to make a series of decisions, with each decision represented by a node in the tree.

At each decision node, the algorithm asks a question based on a specific attribute or feature, such as "What is the age?" or "Does he/she work out?". The algorithm then follows the corresponding branch based on the answer to the question. This process continues until the algorithm reaches a leaf node, which provides the final prediction of "fit" or "unfit".

In the example decision tree, the first decision node is based on the age of the person. If the person is under 30, the algorithm proceeds to the next decision node, which asks whether the person works out or not. If the person does not work out, the algorithm predicts that the person is unfit. If the person does work out, the algorithm proceeds to the next decision node, which asks whether the person eats pizza or not. If the person does eat pizza, the algorithm predicts that the person is unfit. If the person does not eat pizza, the algorithm predicts that the person is fit.

Decision trees are popular because they are easy to interpret and understand, and they can handle both categorical and continuous data. However, decision trees can be prone to overfitting, where the model is too complex and fits the training data too well, leading to poor performance on new data. Techniques like pruning and ensemble learning can be used to improve the accuracy and generalization of decision tree models.

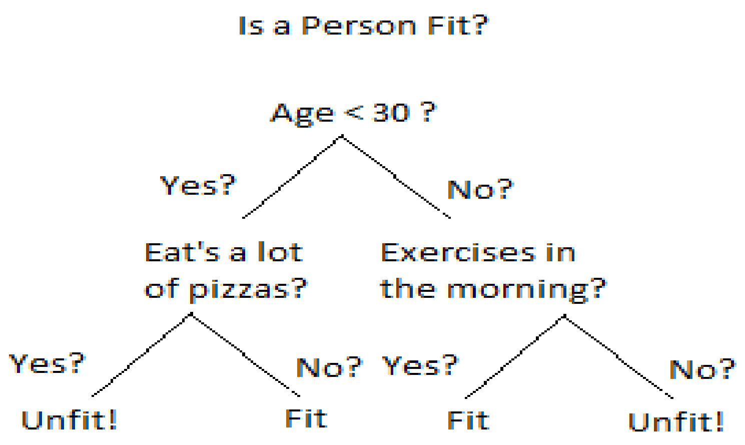


Figure 2.1: Example of a decision tree

Binary recursive portioning is a technique used to create decision trees. It involves recursively partitioning the data into smaller subsets, and then dividing them further on each subsequent branch of the tree. In Decision Tree Classification, a new instance is classified by passing it through a series of tests that determine the appropriate class label for the instance.

These tests are organized into a hierarchical structure called a decision tree, which follows the divide-and-conquer method.

The binary recursive portioning process involves dividing the data into two subsets at each node of the tree. The decision tree starts with a root node that represents the entire dataset. The tree is then split into two child nodes based on a selected feature, with each child node representing a subset of the data. This process is repeated recursively on each child node until a stopping criterion is met, such as a minimum number of instances in a node or a maximum depth of the tree.

In Decision Tree Classification, the decision tree structure is used to classify new instances based on their attributes. The algorithm traverses the decision tree from the root node to a leaf node, following the branch corresponding to the attribute value of the instance being classified. The leaf node represents the class label for the instance.

Decision trees are a powerful and widely used machine learning technique due to their interpretability and ability to handle both categorical and continuous data. However, they can be prone to overfitting and may require pruning or other regularization techniques to improve their generalization performance.

2.2 Random Forest Classifier

Random forest is a machine learning technique that consists of multiple decision trees working together as an ensemble. Each individual decision tree in the random forest produces a class prediction, and the class with the most votes becomes the model's final prediction. This ensemble approach can help to reduce the variance and overfitting issues that can occur with a single decision tree.

Random forest is a popular classification algorithm that is used in a wide range of applications, including image classification, text classification, and bioinformatics. The algorithm is easy to use and can handle large datasets with many features. Additionally, the random forest algorithm provides feature importance scores, which can be used to identify the most important features for classification. Overall, the random forest algorithm is a powerful and flexible machine learning technique that can achieve high accuracy and is widely used in practice.

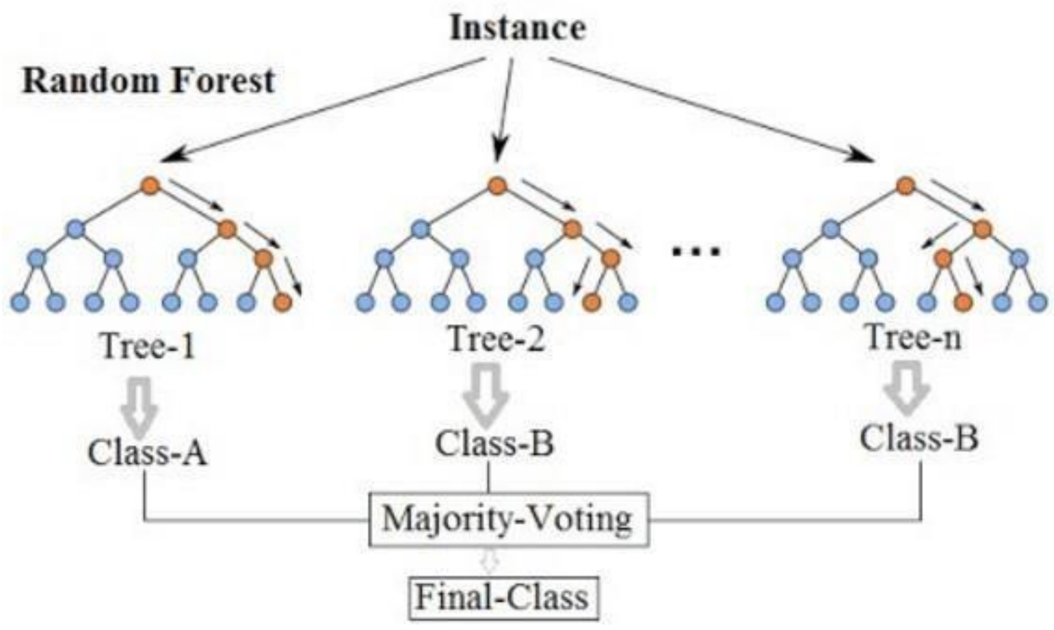


Figure 2.2: Random Forest classification

The driving force behind the success of random forest is the concept of ensemble learning. In simple terms, the random forest model combines the predictions of many individual decision trees to arrive at a final prediction. Each decision tree is built independently and uses a different subset of features and data samples. The key idea behind random forest is that a large number of such independent and uncorrelated trees working together as an ensemble can outperform any individual model.

In the language of data science, the reason why the random forest algorithm is so effective is that it leverages the power of ensemble learning. By combining many different decision trees, each trained on a different subset of the data, the random forest algorithm can achieve better accuracy and generalization performance than any single decision tree on its own. This makes the random forest algorithm a powerful and versatile tool for a wide range of machine learning applications.

2.3 Neural Network

A neural network is a type of machine learning model that is inspired by the structure and function of the human brain. It consists of a large number of simple processing units called neurons, which are organized into layers. The neurons in each layer are connected to neurons in the previous and subsequent layers, forming a network. Each neuron receives input from other neurons, performs a simple computation, and passes its output to other neurons in the next layer.

The structure of a neural network is typically organized into three main types of layers: input layer, hidden layer(s), and output layer. The input layer takes in the input data, and the hidden layer(s) perform a series of computations on the input data to extract relevant features. The output layer then produces the final output, which can be a classification label, a prediction, or a probability distribution.

The computations performed by the neurons are based on weights and biases. The weights determine the strength of the connection between neurons, while the biases determine the threshold at which a neuron is activated. During training, the weights and biases of the neurons are adjusted to minimize the difference between the predicted output and the actual output. This is typically done using an optimization algorithm such as stochastic gradient descent.

One of the key advantages of neural networks is their ability to learn complex patterns in data. This is achieved by adjusting the weights and biases during training to produce more accurate predictions. The process of adjusting the weights and biases is known as backpropagation, and it involves propagating the error from the output layer back through the network to adjust the weights and biases in the hidden layers.

Neural networks have become increasingly popular in recent years due to their ability to make accurate predictions on a wide range of tasks, including image classification, natural language processing, and speech recognition. For example, a neural network can be trained to recognize images of cats and dogs by learning the features that distinguish them, such as the shape of their ears, the color of their fur, and the texture of their skin.

However, neural networks can also be computationally expensive to train and require a large amount of data to avoid overfitting. Overfitting occurs when a model becomes too complex and fits the training data too closely, resulting in poor performance on new data. To avoid overfitting, techniques such as regularization, early stopping, and dropout can be used.

In summary, a neural network is a type of machine learning model that is inspired by the structure and function of the human brain. It consists of neurons organized into layers, and the computations performed by the neurons are based on weights and biases. Neural networks can learn complex patterns in data and make accurate predictions on a wide range of tasks, but they can also be computationally expensive to train and require a large amount of data to avoid overfitting.

2.4 XGBoost Classifier

XGBoost (Extreme Gradient Boosting) is a popular machine learning algorithm that uses an ensemble of decision trees to predict outcomes. It is widely used for structured data prediction tasks such as classification, regression, and ranking problems. XGBoost is based on the gradient boosting framework and uses a combination of decision trees to make accurate predictions.

The XGBoost classifier works by creating a series of decision trees that iteratively correct the errors of the previous tree. In each iteration, the algorithm calculates the gradients and Hessians of the loss function and updates the weights of the decision trees based on the calculated values. This process is repeated for a specified number of iterations until the desired accuracy is achieved.

One of the key features of XGBoost is its ability to handle missing data. It can automatically learn the best imputation values for the missing data during training. XGBoost can also handle both numeric and categorical data, and it can automatically convert categorical data into numeric form using one-hot encoding.

XGBoost is also known for its ability to handle imbalanced datasets. It does this by using weights for each class during training to balance the distribution of the classes in the dataset.

Another important feature of XGBoost is its ability to perform feature selection. It does this by evaluating the importance of each feature in the dataset and removing those that do not contribute significantly to the accuracy of the model.

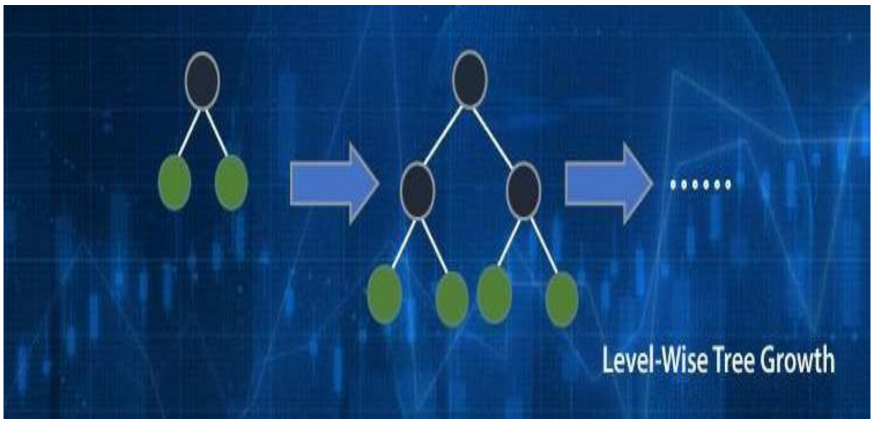


Figure 2.3: Working of XGBoost Classifier

XGBoost also offers several hyperparameters that can be tuned to improve the performance of the model. These hyperparameters include the learning rate, the number of trees, the maximum depth of each tree, the subsample ratio, and the regularization parameters.

In terms of performance, XGBoost is known for its speed and efficiency. It can handle large datasets with millions of instances and thousands of features, and it can train models on distributed systems using Apache Spark.

XGBoost is a powerful and versatile machine learning algorithm that is widely used in industry and academia. Its ability to handle missing data, imbalanced datasets, and perform feature selection make it an attractive option for a wide range of prediction tasks. Its performance and scalability also make it a popular choice for handling large datasets in real-world applications.

2.5 Tensorflow

TensorFlow is an open-source, cross-platform machine learning framework developed by Google Brain Team. It is designed to simplify the process of developing and deploying machine learning models. TensorFlow is widely used for various tasks such as image and speech recognition, natural language processing, and computer vision.

At its core, TensorFlow is based on a computational graph model, where data is represented as a series of interconnected nodes. These nodes are mathematical operations that can be performed on the data. The connections between the nodes represent the flow of data through the graph. The graph can be thought of as a blueprint for a machine learning model.

TensorFlow allows users to define and train machine learning models using Python or C++. It provides a high-level API, called Keras, that simplifies the process of building and training neural networks. Keras allows developers to define models using simple and intuitive building blocks such as layers and activation functions.

TensorFlow also provides a low-level API that allows developers to define custom operations and implement complex algorithms. This API is particularly useful for researchers and developers who need fine-grained control over the model's architecture and training process.

One of the key features of TensorFlow is its ability to perform distributed training across multiple devices, including GPUs and TPUs. This allows users to train models faster and more efficiently by distributing the workload across multiple devices.

Another important feature of TensorFlow is its support for data preprocessing and augmentation. TensorFlow provides a range of tools and functions that allow users to preprocess data, such as image and text data, before training the model. This can include tasks such as data normalization, resizing, and cropping.

TensorFlow also provides a range of tools and libraries that can be used to deploy machine learning models in production. This includes tools for exporting models to various formats, such as TensorFlow Lite for mobile devices and TensorFlow.js for web browsers.

TensorFlow is a powerful and flexible machine learning framework that is widely used in research and industry. Its ease of use and broad range of features make it an excellent choice for developing and deploying machine learning models.

2.6 MXNet Framework

MXNet is an open-source deep learning framework developed by Amazon Web Services (AWS) and Apache Software Foundation. It is designed to provide fast and efficient execution of deep neural networks on a variety of devices, including CPUs, GPUs, and distributed clusters.

At its core, MXNet is based on a computational graph model, where data is represented as a series of interconnected nodes. These nodes are mathematical operations that can be performed on the data. The connections between the nodes represent the flow of data through the graph. The graph can be thought of as a blueprint for a machine learning model.

MXNet supports a range of neural network architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative adversarial networks (GANs). It also provides a range of pre-trained models that can be used for various tasks, such as image recognition and natural language processing.

One of the key features of MXNet is its ability to perform distributed training across multiple devices, including GPUs and distributed clusters. This allows users to train models faster and more efficiently by distributing the workload across multiple devices.

MXNet also provides a range of tools and libraries for data preprocessing and augmentation. This includes tools for data normalization, resizing, and cropping, as well as functions for image and text processing.

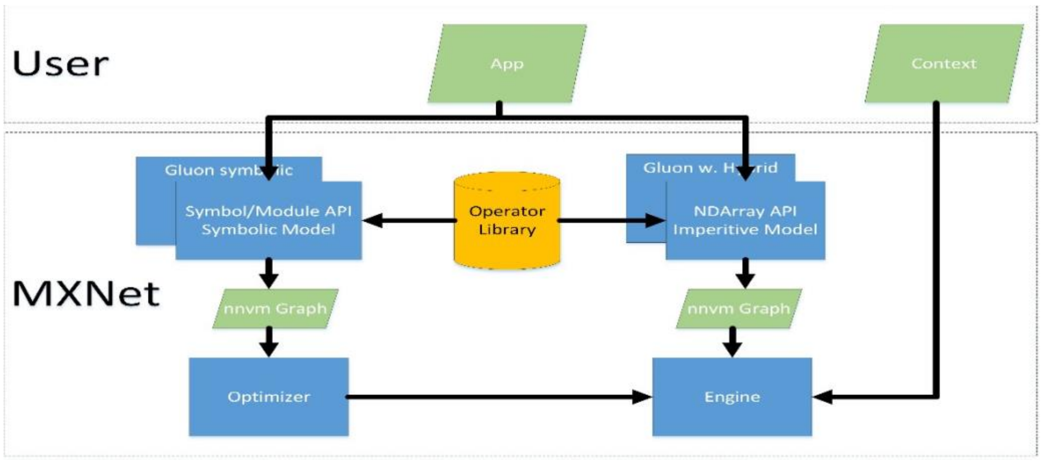


Figure 2.4: Architecture of MXNet Framework

Another important feature of MXNet is its support for multiple programming languages, including Python, R, Julia, and C++. This makes it easy for developers to integrate MXNet into their existing workflows and to use MXNet with their preferred programming language.

MXNet also provides a high-level API, called Gluon, that simplifies the process of building and training neural networks. Gluon allows developers to define models using simple and intuitive building blocks such as layers and activation functions. This makes it easy to experiment with different network architectures and to quickly iterate on model design.
MXNet also provides a range of tools and libraries that can be used to deploy machine learning models in production. This includes tools for exporting models to various formats, such as ONNX and TensorFlow, as well as tools for deploying models on AWS infrastructure, such as AWS Lambda and Amazon SageMaker.
MXNet is a powerful and flexible deep learning framework that is widely used in research and industry. Its ease of use, support for multiple programming languages, and broad range of features make it an excellent choice for developing and deploying machine learning models.

2.7 Keras

Keras is a high-level neural network application programming interface (API) written in Python. It was developed with the goal of making it easy to build and experiment with deep learning models. Keras is built on top of lower-level deep learning libraries, such as TensorFlow and Theano, and provides a simplified interface for building and training deep learning models.

One of the key features of Keras is its ease of use. It provides a simple, user-friendly API that abstracts away the complexity of building and training deep learning models. Keras allows developers to define models using simple and intuitive building blocks such as layers and activation functions.

Keras supports a wide range of neural network architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative adversarial networks (GANs). It also provides a range of pre-trained models that can be used for various tasks, such as image recognition and natural language processing.

Keras also supports transfer learning, which allows developers to use pre-trained models as a starting point for their own models. This can save a lot of time and resources, as the pre-trained models have already been trained on large datasets and can be fine-tuned for specific tasks.

Keras provides a range of tools and libraries for data preprocessing and augmentation. This includes tools for data normalization, resizing, and cropping, as well as functions for image and text processing.

Another important feature of Keras is its ability to perform distributed training across multiple devices, including GPUs and TPUs. This allows users to train models faster and more efficiently by distributing the workload across multiple devices.

Keras also provides a range of tools and libraries that can be used to deploy machine learning models in production. This includes tools for exporting models to various formats, such as TensorFlow Lite for mobile devices and TensorFlow.js for web browsers.

Keras is a powerful and flexible deep learning API that is widely used in research and industry. Its ease of use, support for a wide range of neural network architectures, and broad range of features make it an excellent choice for building and training deep learning models.

2.8 Sklearn for Predictive data

Scikit-learn (sklearn) is a Python library for predictive data analysis. It provides a range of machine learning algorithms and tools for building and evaluating predictive models.

One of the key features of sklearn is its ease of use. It provides a simple and consistent API for building and training predictive models. This makes it easy for developers and data scientists to experiment with different algorithms and techniques.

Sklearn provides a range of supervised learning algorithms, including linear regression, logistic regression, decision trees, random forests, and support vector machines (SVMs). These algorithms can be used for a range of tasks, such as regression, classification, and clustering.

Sklearn also provides a range of unsupervised learning algorithms, including k-means clustering, principal component analysis (PCA), and independent component analysis (ICA). These algorithms can be used for tasks such as dimensionality reduction and anomaly detection.

Sklearn also provides a range of tools for data preprocessing and feature engineering. This includes tools for data normalization, scaling, and imputation, as well as functions for feature selection and extraction.

Sklearn provides a range of tools and libraries for model selection and evaluation. This includes functions for cross-validation, hyperparameter tuning, and model evaluation metrics such as accuracy, precision, recall, and F1 score.

Sklearn also provides a range of tools for working with text data, including feature extraction techniques such as bag-of-words and TF-IDF.

Another important feature of sklearn is its ability to work with large datasets. It provides tools for online learning and out-of-core learning, which allow models to be trained on data that is too large to fit in memory.

sklearn is a powerful and flexible library for predictive data analysis. Its ease of use, wide range of algorithms and techniques, and tools for model selection and evaluation make it an excellent choice for building and evaluating predictive models.

Chapter 3
System Requirement Specification

3.1 Hardware Requirement

- Standard Computer with Modern Processor.
- Atleast 8 GB RAM.
- High Performance Computing.
- Cloud Computing Resources.

3.2 Software Requirement

- VS code/ Eclipse/ Anaconda
- Tensorflow/ MXNet frameworks for neural network/ Keras.
- Sklearn for Predictive data.

3.3 Support Python Modules

Python's module system enables developers to define code that can be reused across multiple files or projects. By organizing code into modules, developers can more easily maintain and share code among different components of a larger system. Modules can be imported into other modules or directly into the main program, allowing the code to be accessed and utilized as needed. Table 3.1 provides a list of some of the most commonly used modules in a typical Python project, highlighting the importance of this system in Python development.

Table 3.1 Supporting Python Modules

No	Python Modules	Description
1	Ipaddress	ipaddress gives the capacities to generate, control and work on IPv4 and IPv6 ad- dresses and networks.
2	Re	This module gives regular expression matching activi- ties like those found in Perl.
3	urllib.request	The urllib.request module characterizes functions and classes which help in open-ing URLs (for the most part HTTP) in a complex world.
4	BeautifulSoup	BeautifulSoup is a pack- age in python for parsing HTML and XML records. It makes a parse tree for parsed pages that can be utilized to extricate infor- mation from HTML, which is valuable for web scraping.

5	Socket	The BSD interface of socket is given access by this mod- ule
6	Requests	The HTTP requests are al- lowed to send by this mod- ule make use of Puthon.
7	Whois	WHOIS is an inquiry and response convention that is comprehensively used for addressing databases that store the selected customers or trustees of an Internet re- source. for example, a do- main name, an autonomous framework or an IP address block, also simultaneously used for broad extend of in- formation.

3.4 Other Non-Functional Requirements

- A non-functional requirement is a specification that enhances the functionality and performance of a system.
- Common non-functional requirements include reusability, maintainability, usability, and scalability.
- Reusability enables the same code to be utilized with minimal modifications to detect various phishing attack variants such as vishing and smishing.
- Maintainability implies ease of debugging and maintaining the system's implementation through basic code and print statements.
- Usability requires the software to be user-friendly and open source, supporting various operating systems.
- Scalability refers to the system's capacity to accommodate an increase in data volume and may include detecting additional phishing attack variants such as vishing and smishing.
- Non-functional requirements play a crucial role in designing, developing, and testing robust and efficient systems.

Chapter 4
System Design

4.1 System Architecture

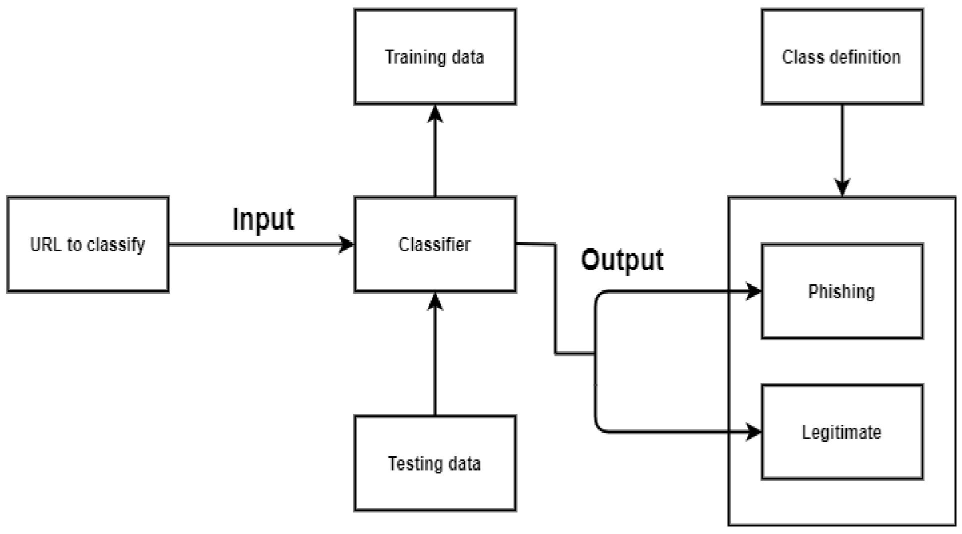


Figure 4.1 System Architecture

The system architecture depicted in Figure 4.1 is designed to classify URLs as either legitimate or phishing based on features extracted from the URL. To accomplish this, the system takes a URL as input and feeds it into a classifier that has been trained on a dataset of URLs labelled as either legitimate or phishing.

The classifiers used in this project include KNN, kernel SVM, Decision tree, and Random Forest classifier. These models are trained to recognize patterns in the input data and use those patterns to classify new URLs as either legitimate or phishing. The performance of each classifier is evaluated, and an accuracy score is generated to determine which model is best suited for the task.

The features extracted from the URL include the IP address, URL length, domain, and whether or not the URL has a favicon, among others. The values of these features are used to generate a list that is fed into the classifiers. The list contains values of 1, 0, and -1 to represent the existence, non-applicability, and non-existence of each feature, respectively.

The project considers 30 features in total, all of which are believed to be relevant in determining whether a URL is legitimate or phishing. By feeding these features into the classifiers, the system can effectively predict whether a given URL is legitimate or phishing with a high degree of accuracy.

The system architecture described above is a powerful tool for classifying URLs as either legitimate or phishing. By using a variety of classifiers and considering numerous relevant features, the system is able to accurately determine the legitimacy of a URL, thereby protecting users from potential phishing attacks.

4.2 Data Flow Diagram

Data Flow Diagrams (DFDs) are a visual tool used to represent the flow of data in a system. They provide a graphical representation of the processes involved in a system from input to report generation. A DFD shows all the possible paths that data can take from one entity to another within the system.

DFDs are typically represented in three different levels, numbered 0, 1, and 2, each level providing a progressively more detailed view of the system. Level 0 provides a high-level overview of the system, showing the major processes and entities involved. Level 1 provides a more detailed view, breaking down the major processes into smaller sub-processes. Level 2 provides a detailed view of the individual processes, showing the specific inputs, outputs, and data flows involved.

There are different types of notations used to draw DFDs, including the Yourdon-Coad and Gane-Sarson methods. The DFDs used in this chapter are based on the Gane-Sarson method, which uses symbols to represent entities, processes, data stores, and data flows. The symbols used in Gane-Sarson DFDs include circles to represent entities, rectangles to represent processes, and double lines to represent data stores. Data flows are represented by arrows that connect the various symbols.

DFDs are a useful tool for representing the flow of data in a system. They can be represented at different levels of detail and can use different notations, such as the Gane-Sarson method, to depict the various entities, processes, data stores, and data flows involved.

4.2.1 Data Flow Diagram Level-0

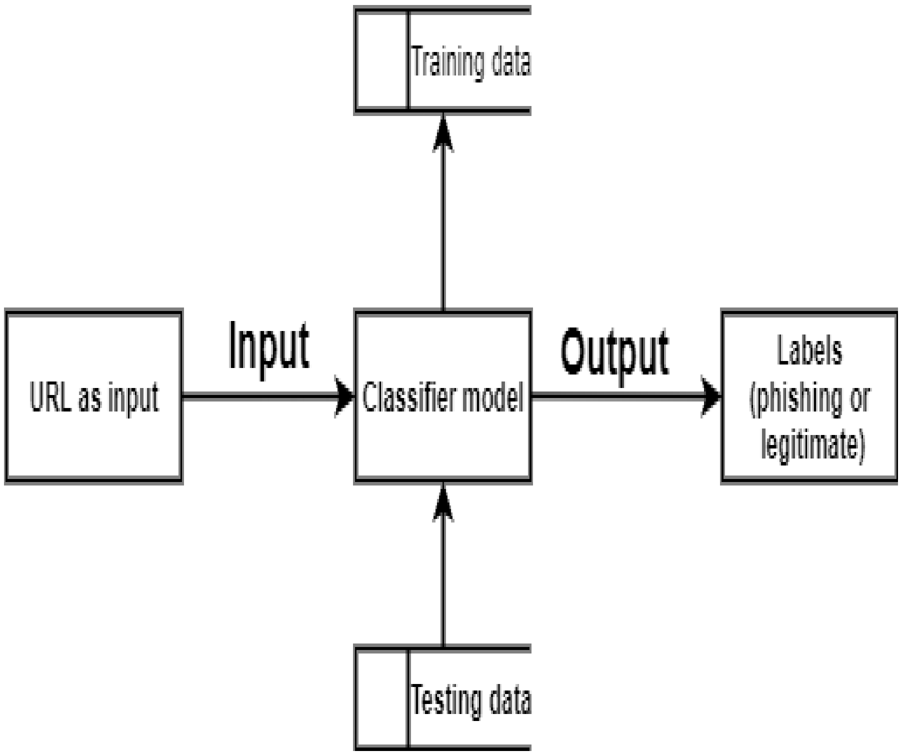


Figure 4.1 DFD Level-0

DFD level 0, also known as the Context Diagram, provides a high-level overview of the system being modelled. It is a simple representation of the entire system and its relationship with external entities. The purpose of the Context Diagram is to provide a clear and easily understood representation of the system to a wide range of stakeholders, including developers, data analysts, and other interested parties. Figure 4.2 shows the DFD level 0 of the system being modelled.

4.2.2 Data flow Diagram Level-1

DFD level 1 provides a more detailed explanation of the high-level process depicted in the Context Diagram. It breaks down the process into its sub-processes. Figure 4.3 shows the DFD level 1 of the system being modelled.

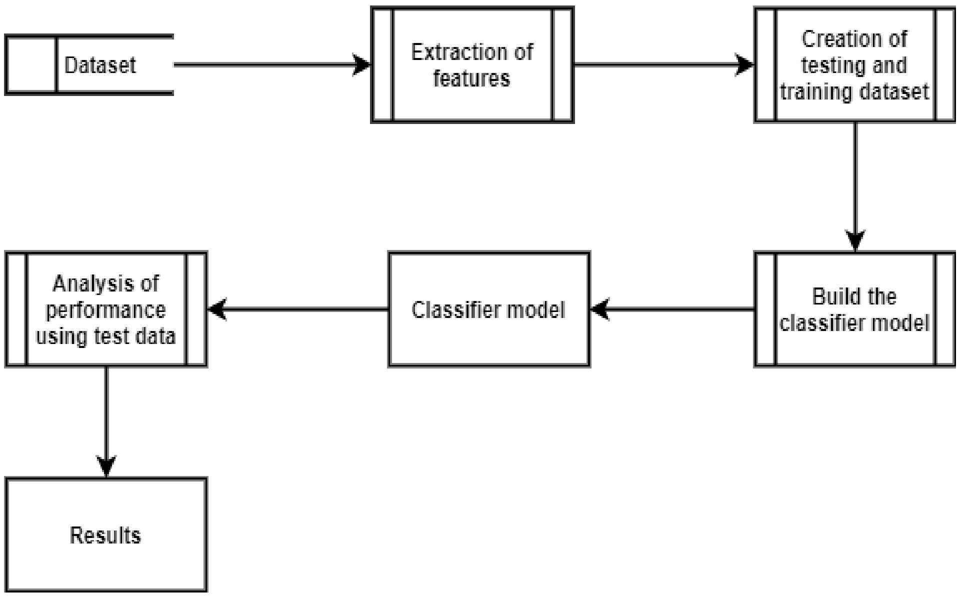

Dataset
Extraction of features
Creation of testing and training dataset
Build the classifier model
Classifier model
Analysis of performance using test data
Results

Figure 4.2 DFD Level-1

DFD level 1 provides a more detailed view of the system by breaking down the high-level process depicted in the Context Diagram into its sub-processes. This level includes the processes involved in the system such as feature extraction, splitting of datasets, building the classifier, and other subprocesses. This level of detail provides a more comprehensive understanding of the system being modeled.

4.2.3 Data Flow Diagram Level-2

DFD level 2 provides an even more detailed view of the system by categorizing the subprocesses of Level 1 into three categories: preprocessing, feature scaling, and classification. This level of detail is depicted graphically in Fig 4.4, and it provides a comprehensive understanding of how the system functions. By breaking down the subprocesses into categories, DFD level 2 offers a more organized and detailed view of the system. The information provided in this level may require additional text to fully comprehend the intricacies of the system being modelled.

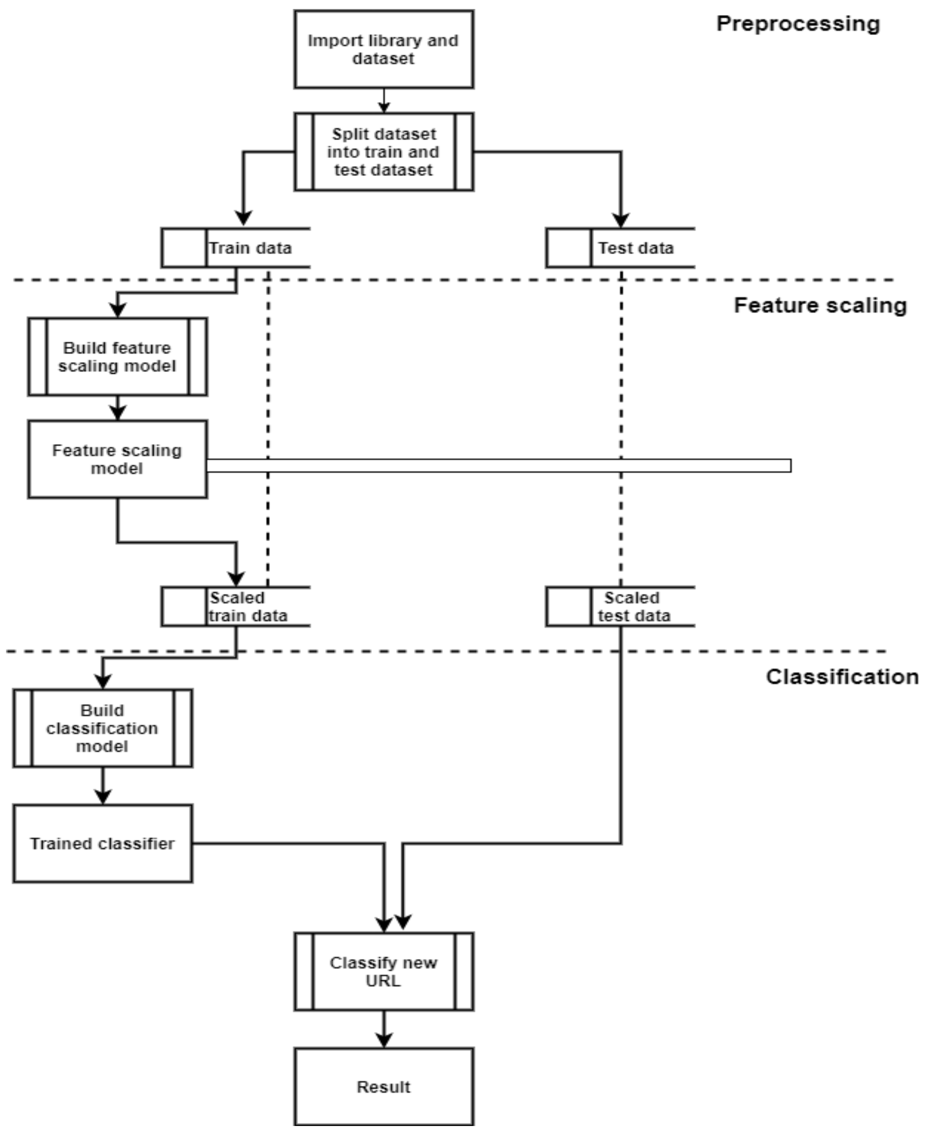


Figure 4.3 DFD Level-2

4.3 UML Activity Diagram

An activity diagram is a type of behavioural diagram that illustrates the flow of control from a start point to an end point, showing the various paths that exist during the execution of the activity. Figure 4.5 shows the activity diagram of the system being modeled, which provides a visual representation of the flow of activities in the system. The activity diagram is an essential tool for modelling the behaviour of a system and can be used to identify potential problems or inefficiencies in the design.

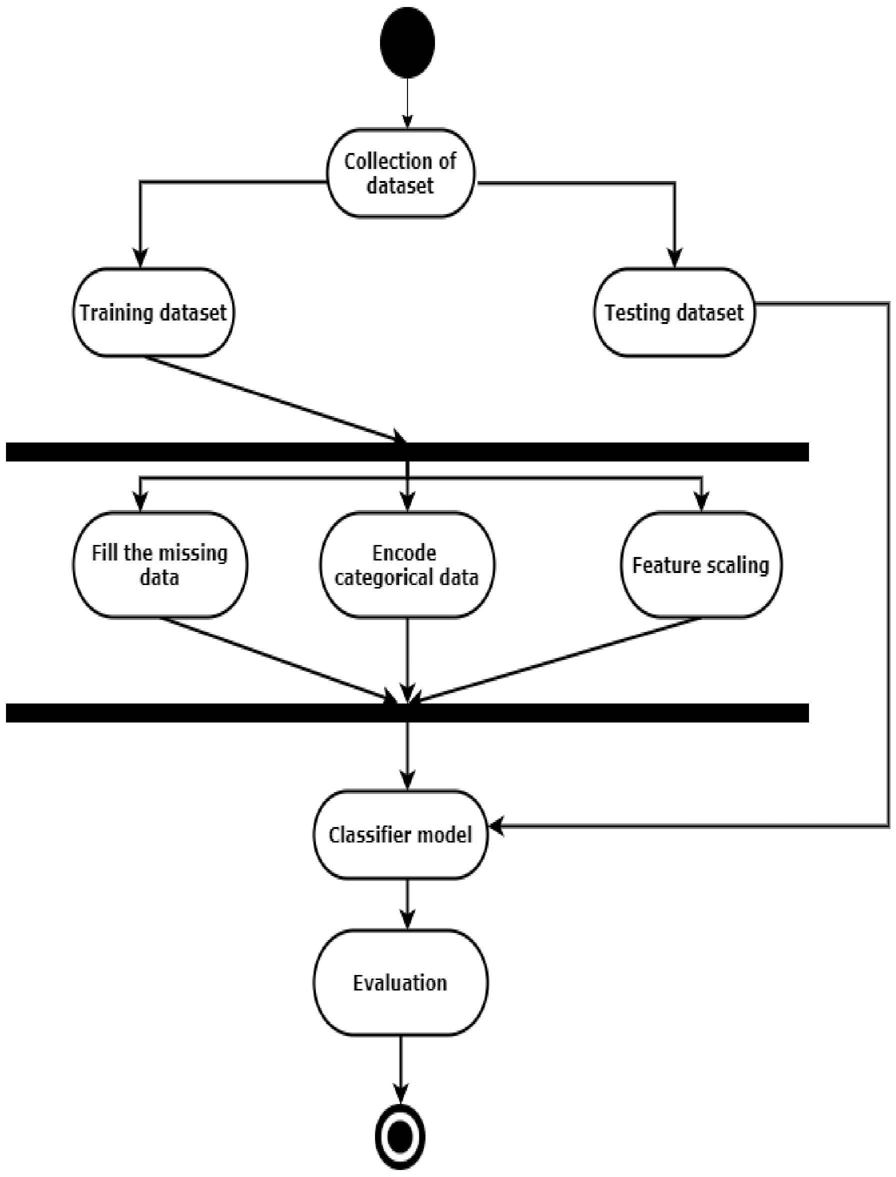


Figure 4.5 UML activity Diagram

4.4 Summary

The System Architecture presents a graphical representation of the system's architecture, depicting the processes involved in transforming input data into output with varying levels of complexity. The data flow diagrams (DFDs) depict the flow of data through the system at different levels of detail, providing a comprehensive understanding of the system's processes. The activity diagram illustrates the system's behaviour, showing the flow of activities from start to end points. These graphical representations facilitate a better understanding of the system and its processes for a wide range of stakeholders, from developers to data analysts.

Chapter 5
Implementation

In this report, the methodology employed to classify URLs as either phishing or legitimate is illustrated. The approach involves building a training set and using it to train a machine learning model, which is referred to as the classifier. The implementation of this approach is diagrammatically represented in Figure 5.1. This chapter provides a comprehensive explanation of the process involved in training the classifier, which is essential for accurately classifying new URLs as either phishing or legitimate.

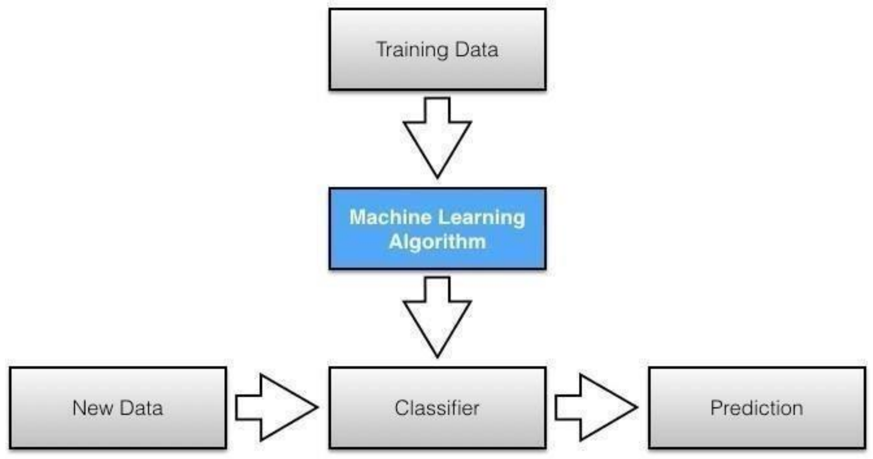


Figure 5.1 Implementation

5.1 Process involved in Implementation

In the initial stage of the research, the appropriate dataset was selected for the task, and the chosen dataset was obtained from Kaggle. The dataset was preferred for its large size, which made working with it interesting. Additionally, the dataset contained 30 features, providing a wide range of attributes to enhance the accuracy of predictions. Figure 5.2 illustrates the considered features. Moreover, the number of URLs in the dataset was distributed quite evenly among the two categories, phishing and legitimate, making it suitable for training a machine learning model to classify URLs.

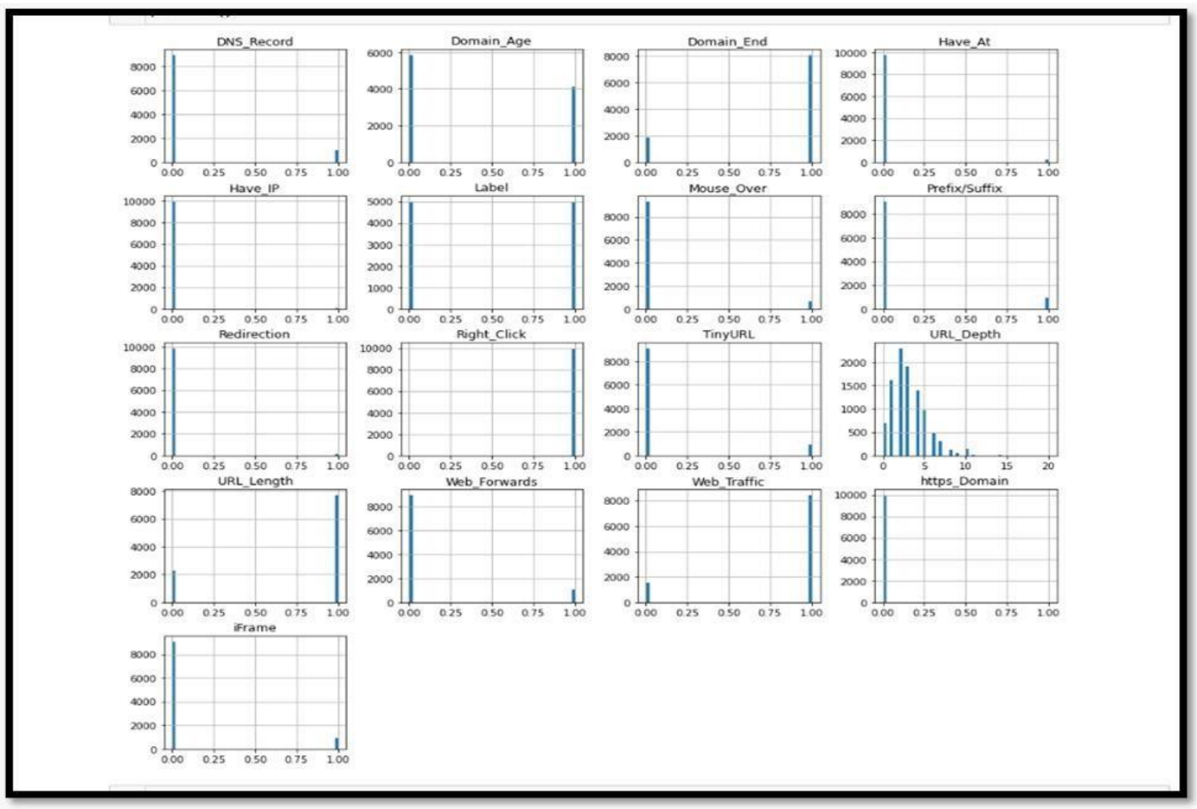


Figure 5.2 The Features in the dataset

- **Splitting:** To prepare the dataset for the machine learning model, it was divided into a training set and a testing set. The dataset was split in such a way that 75% of the data was used for training the model while the remaining 25% was used for testing the accuracy of the trained model. The "train test split" method was used for this purpose. Prior to the splitting, the dataset was organized by assigning dependent and independent variables. This step ensured that the model was trained on the appropriate variables to achieve the desired accuracy.

- **Preprocessing:** In general, preprocessing involves tasks like handling missing data, removing outliers, and cleaning the dataset. However, the dataset selected for this research work was already preprocessed, so there was no need for any further preprocessing. The only step required was feature scaling, which is a common technique used to standardize the range of values of the input variables. This was done to ensure that all the input features had equal importance while training the machine learning model.

- **Future scaling:** Feature Scaling is a technique used to transform and standardize the independent variables in a dataset to a common range. It is a necessary step in data preprocessing, especially when dealing with variables with different magnitudes. There are two methods for feature scaling: normalization and standardization. In this project, standardization was employed. Standardization rescales data to have a mean of zero and a standard deviation of one.

  Standardization helps to ensure that each variable is given equal importance during analysis and that no variable dominates the others. This is because variables with a larger magnitude tend to have a greater impact on the outcome of the model. By standardizing the variables, they can be compared on a common scale, which allows for more accurate analysis and prediction. In this project, feature scaling was performed after splitting the dataset into training and testing sets to ensure that the model was trained and tested on standardized data.

  Standardlization: Standardization is a scaling technique in which the values are normalized based on their mean and standard deviation. This means that the mean of the attribute is shifted to zero and the distribution is transformed to have a unit standard deviation. On the other hand, normalization, also known as Min-Max scaling, rescales the values to be within the range of 0 and 1. The project employs the StandardScaler method, which fits and transforms only the independent variables. In the classification method, there is no need to scale the dependent variables. Whether or not to scale dummy variables derived from categorical data depends on the context. The use of

StandardScaler ensures that the variables are put on the same scale to avoid one variable dominating the others and affecting the result. In contrast to normalization, standardization works better when there are outliers in the data.

- **Feature Extraction:** To extract feature values from URLs, various Python modules such as whois, requests, socket, re, Ip address, and BeautifulSoup are used. These modules provide information about IP addresses, URL length, domain name, subdomains, presence of favicon, and other relevant features. The extracted values are then stored in a list format that corresponds to the format of the dataset. This ensures that the classifier is trained with inputs in the same format as the dataset.

  When a URL is provided as input to the system, it is converted into a Python list consisting of 30 elements, each representing a respective feature. This list is then fed into the trained classifier, which includes KNN, kernel SVM, decision tree, and random forest classifier. These classifiers are used to determine whether a URL is phishing or legitimate. By leveraging these classifiers, the system can accurately predict the category of URLs with high precision and recall.

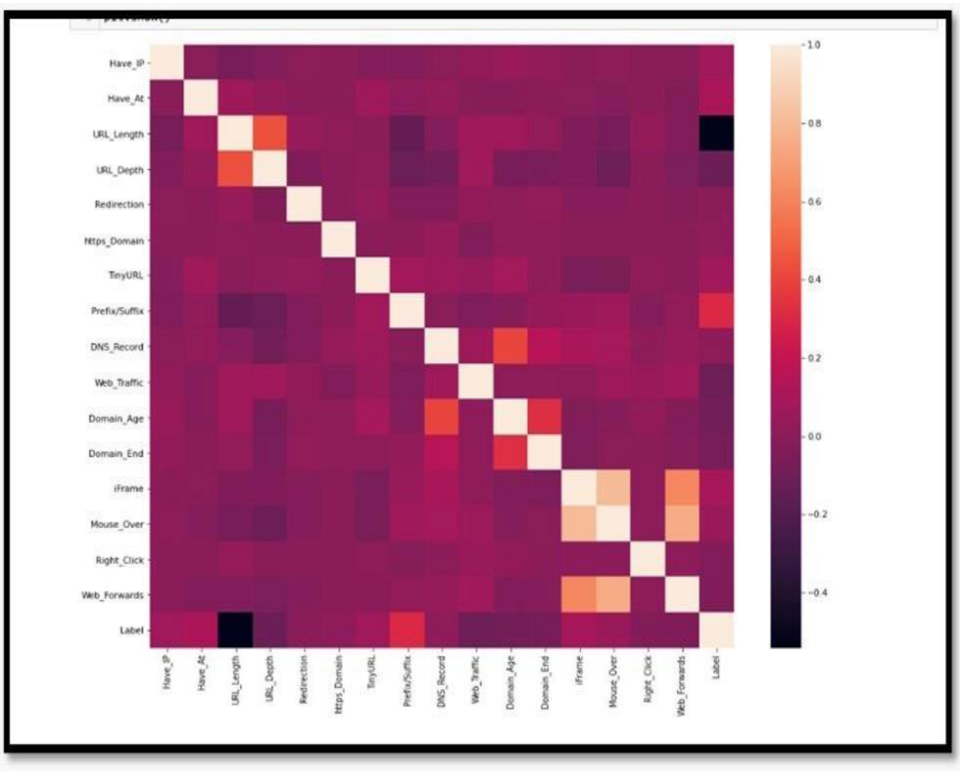


Figure 5.3 Scoring of Features in Dataset

5.2 Classifiers

- **Sklearn.tree.DecisionTreeClassifier**

  `sklearn.tree.DecisionTreeClassifier` is a classification algorithm based on decision trees, where the goal is to create a model that predicts the value of a target variable based on several input features. It works by recursively splitting the dataset into smaller and smaller subsets based on the most significant attributes, resulting in a tree-like structure where each node represents a decision based on an attribute value.

- **Sklearn.ensemable.RandomForestClassifier**

  `sklearn.ensemble.RandomForestClassifier` is a machine learning algorithm that fits several decision trees on different sub-samples of the dataset and averages the results. It improves the performance of decision trees and helps to reduce overfitting.

Chapter 6
Testing and Validation

The testing chapter validates the proposed system by comparing the algorithm's results with actual results for both legitimate and phishing URLs. Each algorithm is tested separately, and the testing section provides detailed results for each one. This chapter is essential to ensure the accuracy and reliability of the system in identifying phishing URLs. It is crucial to concentrate on this section to understand the performance of each algorithm and the overall effectiveness of the proposed system.

6.1 Unit Testing

Unit Testing involves testing individual modules to ensure that they are functioning correctly and are ready to be used.

6.1.1 Unit Testing of Decision Tree Algorithm-1

Table 6.1 Unit Testing of Decision Tree Algorithm-1

Test case	05
Test Name	“Testing of Decision tree -1”
Input	paypal.de@secure-server.de/secure-environment
Expected output	Phishing
Actual Output	Phishing
Remark	Success

6.1.2 Unit Testing of Decision Tree Algorithm-2

Table 6.2 Unit Testing of Decision Tree Algorithm-2

Test case	06
Test Name	"Testing of Decision tree -2"
Input	https://www.wikipedia.org/
Expected output	Legitimate
Actual Output	Legitimate
Remark	Success

6.1.3 Unit Testing of RFC Algorithm-1

Table 6.3 Unit Testing of RFC Algorithm-1

Test case	07
Test Name	"Testing of Random forest classifier -1"
Input	http://63.17.167.23/pc/verification.htm=https://www.paypal.com/
Expected output	Phishing
Actual Output	Phishing
Remark	Success

6.1.4 Unit Testing of RFC Algorithm –2

Table 6.4 Unit Testing of RFC Algorithm-2

Test case	08
Test Name	"Testing of Random forest classifier -2"
Input	https://calendar.google.com/calendar/r
Expected output	Legitimate
Actual Output	Legitimate

Remark	Success

6.2 Integration Testing

Integration Testing is a testing approach where the individual modules are combined and then tested to ensure their fitness for use in the system.

6.2.1 Importing Modules

Table 6.5 Import Modules

Test case	09
Test Name	"Importing modules"
Input	Import "module" statements
Expected output	The module to be imported
Actual Output	The module was imported and could be used
Remark	Success

6.2.2 Import Dataset

Table 6.6 Import Dataset

Test case	10
Test Name	"Importing dataset"
Input	Import ”dataset” statement
Expected output	The dataset to be imported
Actual Output	The dataset was imported and could be used
Remark	Success

6.2.3 Importing User Defined Function

Table 6.7 Import function

Test case	11
Test Name	“Importing user defined function”
Input	Import ”extraction” function
Expected output	The function to be imported that re- turns a list
Actual Output	The function was imported and re- turned the list as expected
Remark	Success

6.3 System Testing

System testing validates the completely integrated system.

6.3.1 System Testing

Table 6.8 System Testing

Test case	12
Test Name	“System testing”
Input	Sample URL provided to check whether it is a phishing or legitimate URL
Expected output	All the modules like importing of mod- ules, dataset and functions defined and provide the result
Actual Output	The application reacts as expected
Remark	Success

Chapter 7
Experimental Analysis and Result

This chapter covers the project's implementation and outcomes in detail.

7.1 Experimental Analysis

A confusion matrix (CM) is a useful tool for assessing the effectiveness of a classifier, displaying correct and incorrect predictions in a graphical format. The CM allows for an accurate evaluation of a model's performance. Figure 7.1 shows an abstract representation of the CM, illustrating how true positive, true negative, false positive, and false negative values are organized. By analyzing the information contained in a CM, one can understand a classifier's accuracy, precision, recall, and F1 score.

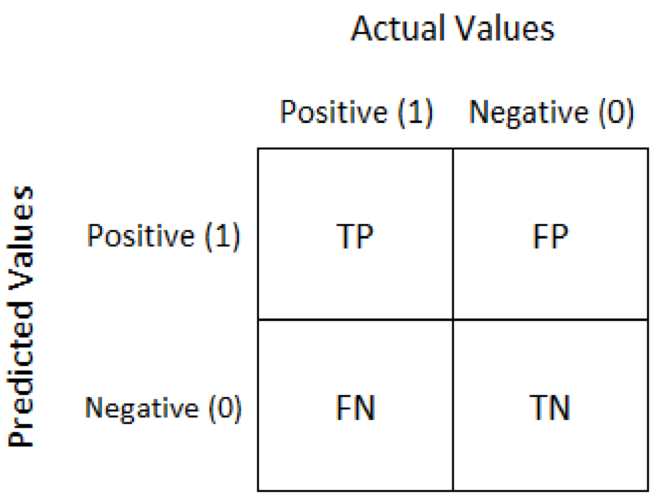


Figure 7.1 Confusion Matrix

The figure above presents the four components of a confusion matrix, including True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). These values are used to evaluate the performance of various algorithms, and their corresponding confusion matrices are displayed in the figure. By examining the values in the confusion matrices, it is possible to assess each algorithm's accuracy, precision, recall, and F1 score.

7.1.1 Decision Tree

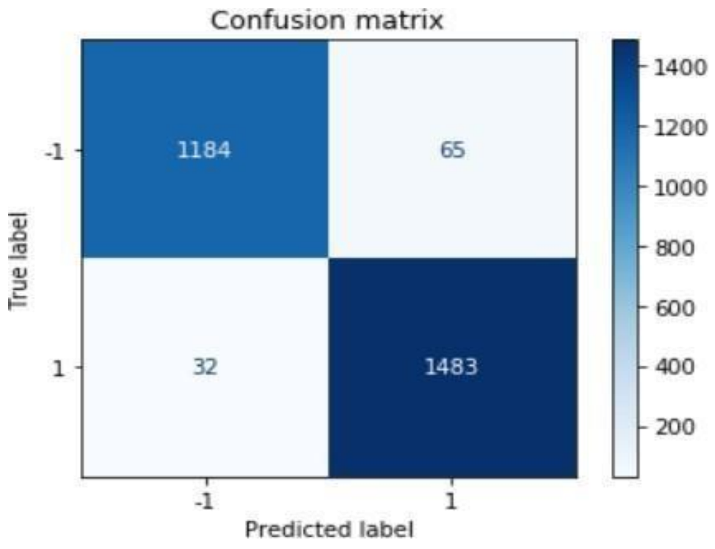


Figure 7.2 Decision Tree – Confusion Matrix

7.1.2 Random Forest Classifier

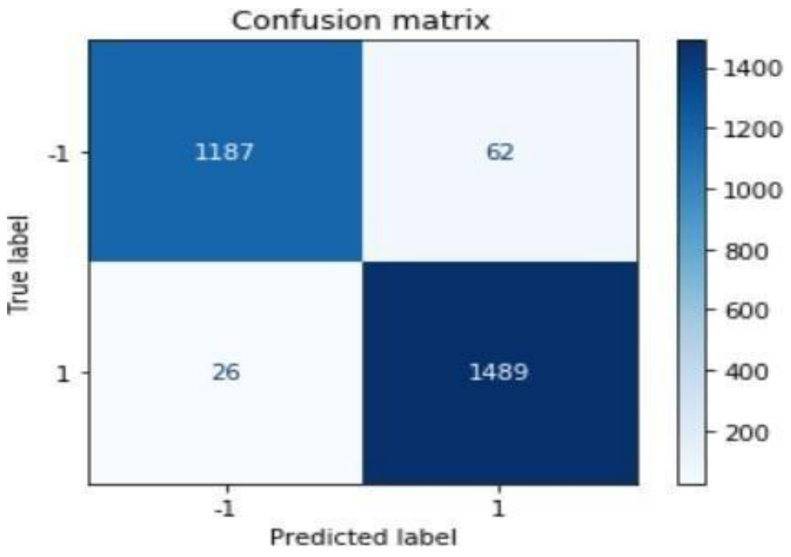


Figure 7.3 Random Forest Classifier – Confusion Matrix

7.1.3 Features of Dataset

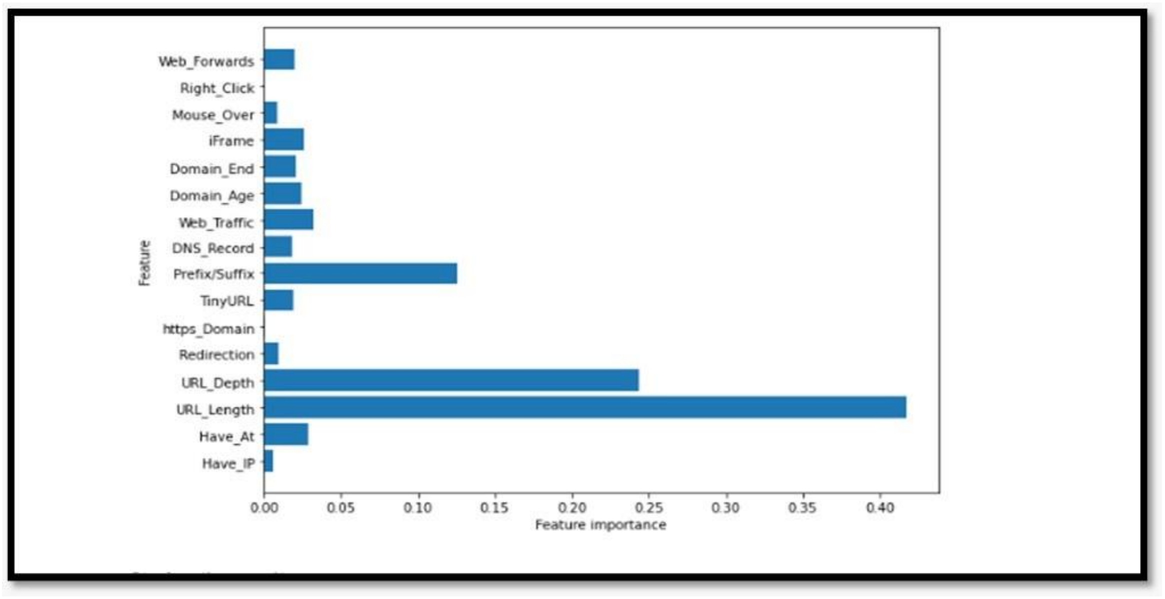


Figure 7.4 Features of Dataset

7.1.4 Importance of Features

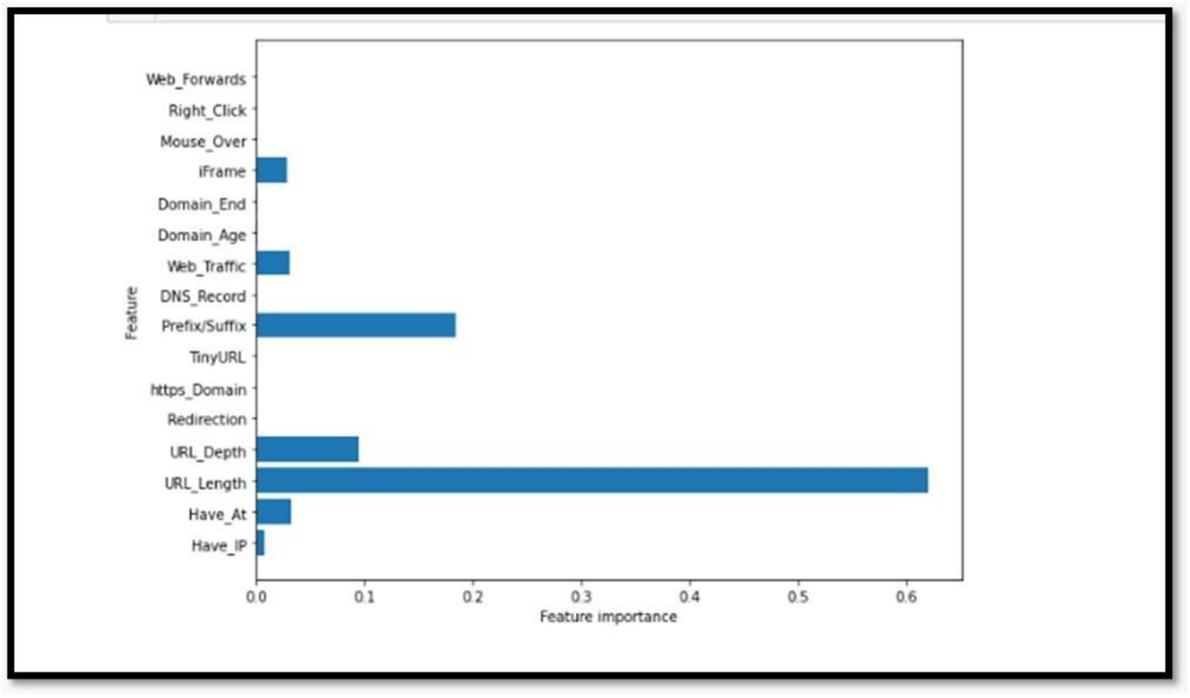


Figure 7.5 Importance of Features

7.2 Comparative Plots evaluating performance

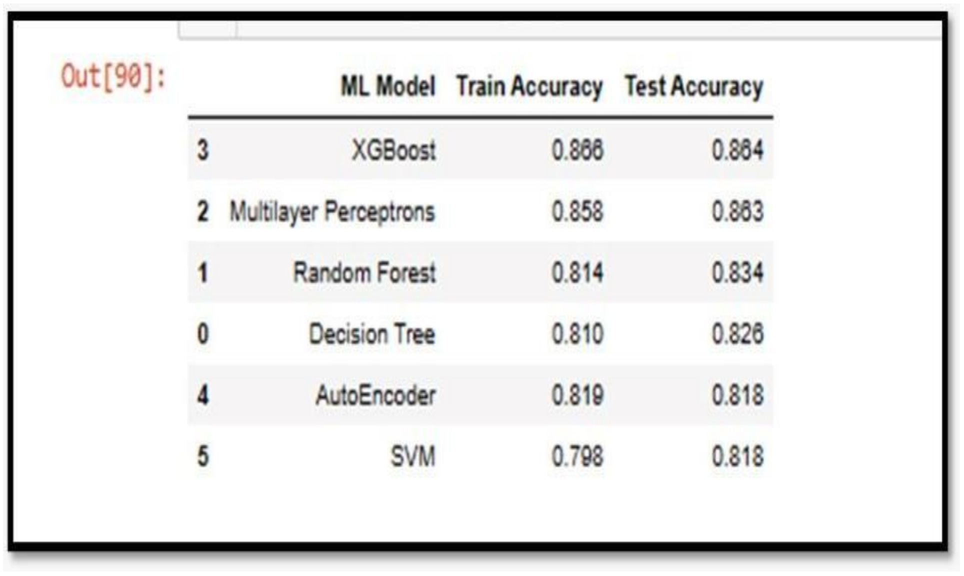
7.2.1 Accuracy Score

Accuracy is the proportion of correctly predicted samples, as shown by the formula in Figure 7.4. Figure 7.5 is a comparative plot that illustrates and compares the accuracy of different models.

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

Fraction predicted correctly

Figure 7.6 Accuracy Formula



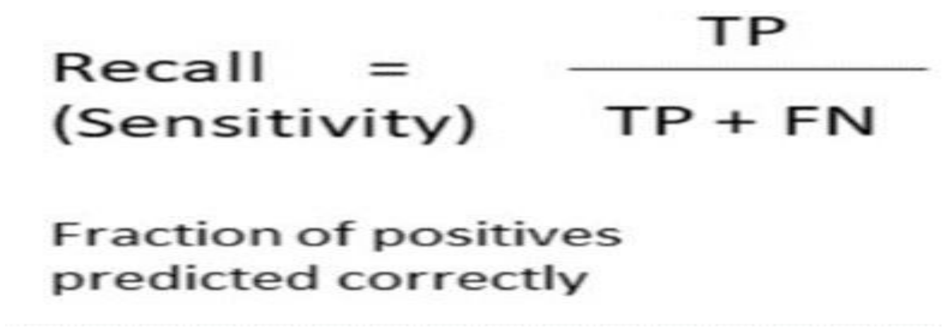
Out[90]:

	ML Model	Train Accuracy	Test Accuracy
3	XGBoost	0.866	0.864
2	Multilayer Perceptrons	0.858	0.863
1	Random Forest	0.814	0.834
0	Decision Tree	0.810	0.826
4	AutoEncoder	0.819	0.818
5	SVM	0.798	0.818

Figure 7.7 Comparative plots of Accuracy Score

7.2.2 Recall Score

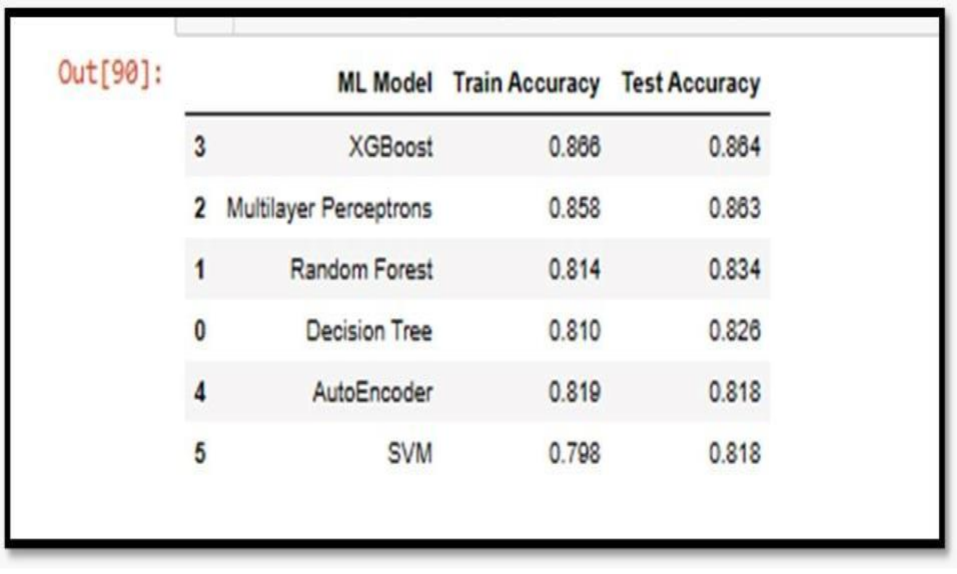
The recall score represents the proportion of correctly predicted positive events, as indicated by the formula shown in Figure 7.8.



$$\text{Recall (Sensitivity)} = \frac{TP}{TP + FN}$$

Fraction of positives predicted correctly

Figure 7.8 Recall Score



Out[90]:

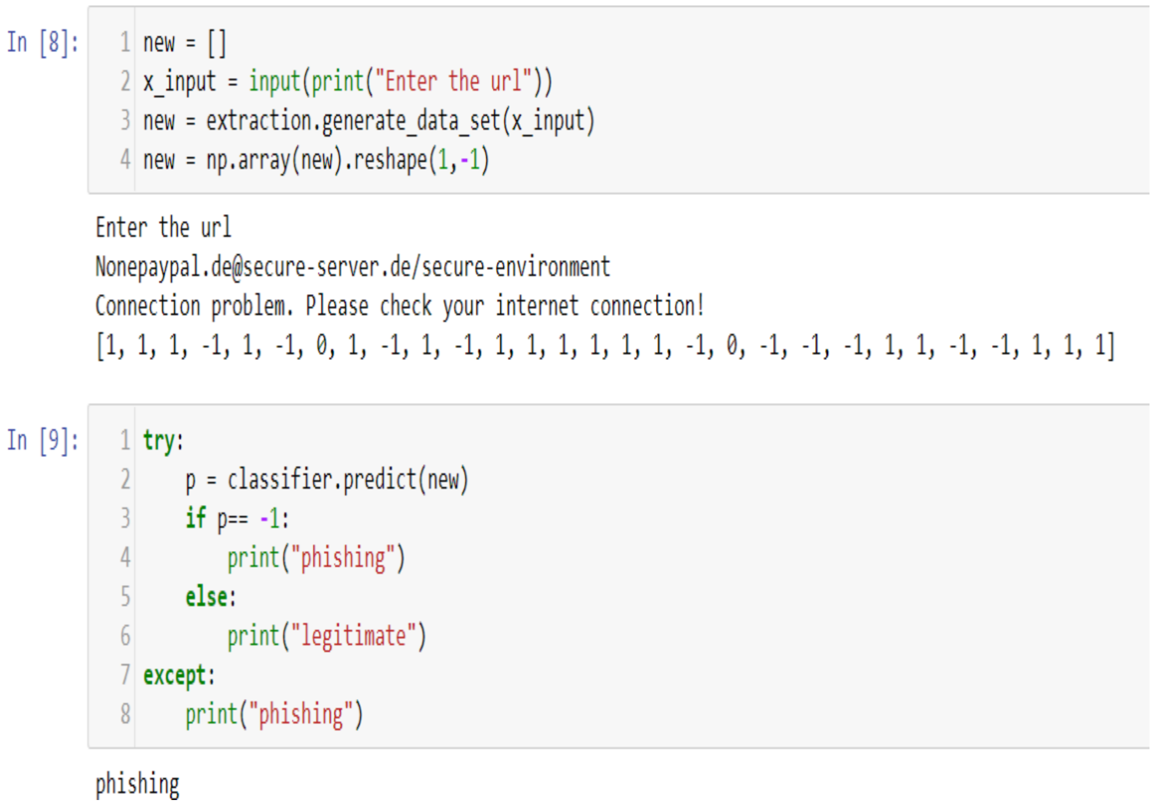
	ML Model	Train Accuracy	Test Accuracy
3	XGBoost	0.866	0.864
2	Multilayer Perceptrons	0.858	0.863
1	Random Forest	0.814	0.834
0	Decision Tree	0.810	0.826
4	AutoEncoder	0.819	0.818
5	SVM	0.798	0.818

Figure 7.9 Comparative Plots of Recall Score

7.3 Results

7.3.1 Decision Tree

The input URL "paypal.de@secure-server.de/secure-environment" was analyzed using the decision tree algorithm. The expected outcome was phishing, and the obtained result was also phishing.



```
In [8]:
new = []
x_input = input(print("Enter the url"))
new = extraction.generate_data_set(x_input)
new = np.array(new).reshape(1,-1)
```

```
Enter the url
Nonepaypal.de@secure-server.de/secure-environment
Connection problem. Please check your internet connection!
[1, 1, 1, -1, 1, -1, 0, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, -1, 0, -1, -1, -1, 1, 1, -1, -1, 1, 1, 1]
```

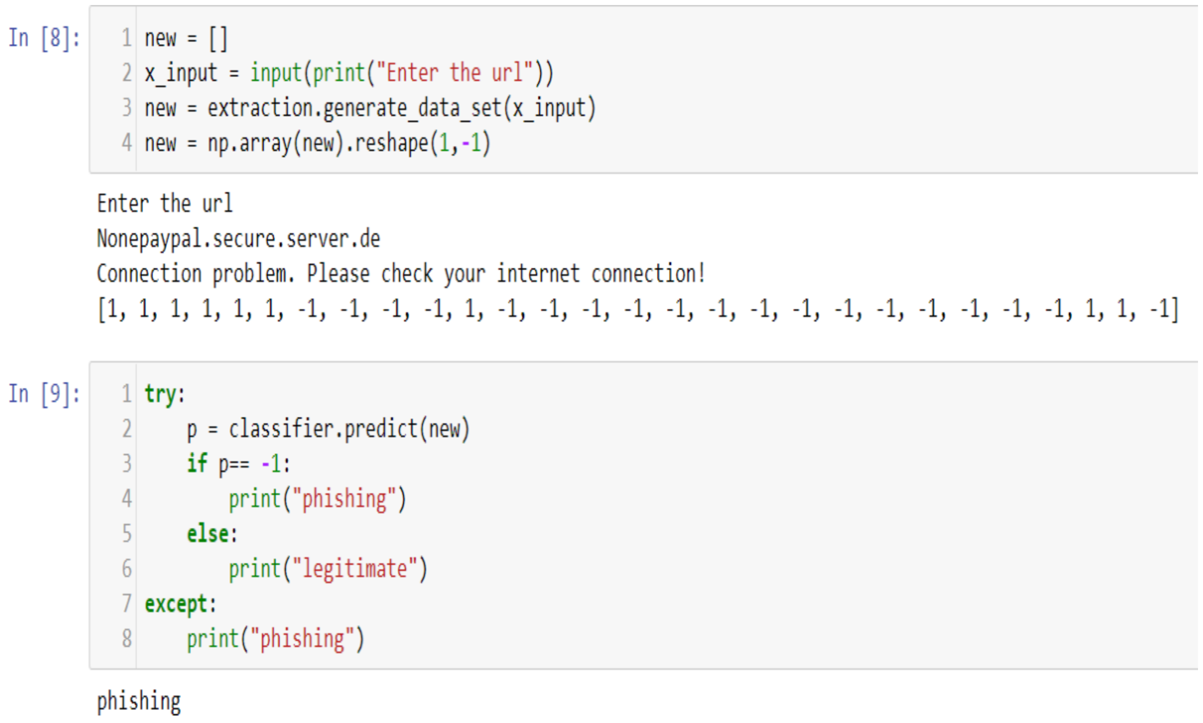
```
In [9]:
try:
    p = classifier.predict(new)
    if p== -1:
        print("phishing")
    else:
        print("legitimate")
except:
    print("phishing")
```

```
phishing
```

Figure 7.10 Prediction by Decision Tree

7.3.2 Random Forest Classifier

Using the Random Forest classifier algorithm, the input URL "paypal.secure.server.de" was analyzed for phishing. The expected outcome was phishing, and the obtained result was also phishing.



```
In [8]:
new = []
x_input = input(print("Enter the url"))
new = extraction.generate_data_set(x_input)
new = np.array(new).reshape(1,-1)
```

```
Enter the url
Nonepaypal.secure.server.de
Connection problem. Please check your internet connection!
[1, 1, 1, 1, 1, 1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1]
```

```
In [9]:
try:
    p = classifier.predict(new)
    if p== -1:
        print("phishing")
    else:
        print("legitimate")
except:
    print("phishing")
```

```
phishing
```

Figure 7.10 Prediction by Random Forest CLassifier

Chapter 8
Conclusion And Future Works

8.1 Conclusion

Phishing has become a growing threat in this rapidly developing world of technology. With more and more countries adopting cashless transactions, online business, paperless tickets, and other advancements to keep pace with the world, phishing is becoming a major obstacle to this progress. People are becoming increasingly skeptical of the internet's reliability. AI can be used to gather data and create unique data products. A layperson who is completely unaware of how to recognize a security threat will never risk making financial transactions online. Phishers are targeting payment industries and cloud services the most.

The purpose of this project was to investigate this area by demonstrating the use of machine learning (ML) to identify phishing sites. The project aimed to build an efficient, accurate, and cost-effective phishing detection mechanism using machine learning tools and techniques. The project was conducted in the Anaconda IDE and written in Python.

To achieve this, the proposed method used four machine learning classifiers, and a comparative study of the four algorithms was conducted. A high accuracy score was also achieved. The four algorithms used were K-Nearest Neighbor, Kernel Support Vector Machine, Decision Tree, and Random Forest Classifier. All four classifiers yielded promising results, with Random Forest Classifier providing the best results with an accuracy score of 96.82%. The accuracy score may vary when using other datasets, and other algorithms may provide better accuracy than the random forest classifier. Random forest classifier is an ensemble classifier, which contributes to its high accuracy. This model can be deployed in real-time to detect URLs as phishing or legitimate.

This project demonstrates the potential of using machine learning to detect phishing sites. With the ever-growing threat of phishing, this technology can help to improve online security and ensure that people can safely engage in online transactions without the fear of being scammed.

8.2 Future Enhancement

To enhance the performance of the current model, ensemble methods can be implemented by combining multiple base models to generate an optimal predictive model. This could potentially improve the accuracy score of the current model and make it more robust in identifying phishing URLs. Moreover, combining multiple classifiers trained on different aspects of the same training set could lead to a more reliable prediction as compared to any of the single classifiers used on their own.

Future work could also include incorporating other variants of phishing such as smishing and vishing to complete the system. This would help in creating a comprehensive and robust phishing detection system that could detect various types of phishing attacks.

In addition, the methodology used in the project would need to be evaluated on how it can handle the growth of the dataset. As the dataset grows over time, the classifier would need to be applied incrementally to new data and potentially receive feedback that could modify it over time. This would ensure that the model remains up-to-date and continues to provide accurate predictions even as the dataset evolves.

The project provides a solid foundation for the development of an efficient and accurate phishing detection system using machine learning techniques. With further improvements and additions, this system can prove to be an effective solution in mitigating the threat of phishing in the rapidly growing world of technology.

References

1. What is Phishing? : https://www.phishing.org/what-is-phishing.

2. Sarker. I.H deep Cybersecurity: A Comprehensive Overview from neural network and deep learning perspective. SN Comput. Sci.2021,2,154.

3. Kim.D.,Kim, Y.-H., Shin,D.: Fast attack detection system using log analysis and attack tree generation clust. Comput.22(1), 1827-1835(2019).

4. Phishing Detection using Machine Learning-based URL Analysis: A Survey- https://www.ijert.org/research/phishing-detection-using-machine-learning-based-url-analysis-a-survey-IJERTCONV9IS13033.pdf.

5. An intelligent cyber security phishing detection system using deep learning techniques.

6. https://link.springer.com/article/10.1007/s10586-022-03604-4.

7. https://github.com/chamanthmvs/Phishing-Website-Detection.

8. https://www.researchgate.net/publication/328541785_Phishing_Website_Detection_using_Machine_Learning_Algorithms.

9. https://bmsit.ac.in/public/assets/PG_NBA_Data/Program_Level_Documents/5%20PROJECT%20REPORTS/2018-20/Bhagya_report_final.pdf.

10. https://www.academia.edu/43230944/A_REPORT_on_DETECTION_OF_PHISHING_WEBSITE_USING_MACHINE_LEARNING.

11. Panos Louridas and Christof Ebert. Machine learning. IEEE Software, 33:110–115, 09 2016.

12. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8504731/

13. Ariyadasa, S.; Fernando, S.; Fernando, S. Detecting phishing attacks using a combinedmodel of LSTM and CNN. Int. J. Adv. Appl. Sci. 2020, 7, 56–67.

14. Malicious Url Recognition and Detection Using Attention-Based CNN-LSTM-KSII Transactions on Internet and Information Systems (TIIS)|Korea Science. Available online: https://www.koreascience.or.kr/article/JAKO201905959996575.page

15. Tang, L.; Mahmoud, Q.H. A deep learning-based framework for phishing website detection. IEEE Access 2022, 10, 1509–1521.

16. URL 2016|Datasets|Research|Canadian Institute for Cybersecurity |UNB. Unb.ca. 2022. Available online: https://www.unb.ca/cic/datasets/url-2016.html

17. Do, N.Q.; Selamat, A.; Krejcar, O.; Herrera-Viedma, E.; Fujita, H. Deep Learning for Phishing Detection: Taxonomy, current challenges and Future Directions. IEEE Access 2022, 10, 36429–36463.

18. Su, Y. Research on Website Phishing Detection Based on LSTM RNN. In Proceedings of the 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, 12–14 June 2020; pp. 284–288.

19. What Is Deep Learning and How Does It Work? Search Enterprise AI. 2022. Available online: https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neural-network.

20. Saurabh Saoji. Phishing detection system using visual cryptography, 03 2015.

21. S. Patil and S. Dhage. A methodical overview on phishing detection along with an organized way to construct an anti-phishing framework. In 2019 5th International Conference on Advanced Computing Communication Systems (ICACCS), pages 588– 593, 2019.

22. Phishing| Phishing Techniques. Phishing.org. 2022. Available online: https://www.phishing.org/phishing-techniques

23. https://www.sciencedirect.com/science/article/pii/S2665917422001106

24. Grover, R. Deep Learning-Overview, Practical Examples, Popular Algorithms|Analytics Steps. Analyticssteps.com. 2022. Available online: https://www.analyticssteps.com/blogs/deep-learning-overview-practical-examples-popular-algorithms.

25. Phishing detection system using machine learning https://youtu.be/pjewBSmWTlU.