T.C.

ALTINBAS UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

DETECTION OF MALICIOUS URLS USING MACHINE LEARNING.

Raed Hameed GBURI

Master of Science

Supervisor

Prof. Osman Nuri UÇAN

Istanbul, 2021

DETECTION OF MALICIOUS URLS USING MACHINE LEARNING

by

Raed Hameed GBURI

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2021

The thesis titled “DETECTION OF MALICIOUS URLS USING MACHINE LEARNING” prepared and presented by Raed Hameed GBURI was accepted as a Master of Science Thesis in Electrical and Computer Engineering.

Prof. Osman UÇAN

Supervisor

Thesis Defense Jury Members:

Prof. Osman UÇAN	School of Engineering and Architecture, Altınbaş University
Asst. Prof. Sefer KURNAZ	School of Engineering and Architecture, Altınbaş University
Prof. Ahmet RAZBONYALI	School of Engineering and Natural Sciences, Maltepe University

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Raed Hameed GBURI

Signature

Approval Date of Institute of Graduate Studies:
____/____/____

DEDICATION

I extend my thanks and praise to God Almighty, Lord of the worlds, for granting me strength, patience, and health, and then I thank my mother, who did not forget me in her prayers and wish me success and success, and my thanks to my dear wife who supported me and stood with me in all circumstances, and to my children, I dedicate this success to you

ACKNOWLEDGEMENTS

I would like to thank my supervisor Prof. OSMAN UÇAN for all support during my study. It’s great pleasure to express my deepest gratitude to my friends who have shared with me best moments during my study for the master’s degree.

ABSTRACT

DETECTION OF MALICIOUS URLS USING MACHINE LEARNING

GBURI, Raed Hameed

M.Sc., Electrical and Computer Engineering, Altınbaş University,

Supervisor: Prof. Dr. Osman Nuri UÇAN

Date: November/2021

Pages: (74 pages)

Data is a valuable and significant asset in computer systems for businesses, institutions, and governments, and it must be protected from malevolent computer crimes that try to steal and sell it. and be threatened by the most common URLs addresses and this threat is considered a cybersecurity threat and uses malicious URLs link containing unwanted content such as (spam, phishing, download- from the drive) and thus users are victims of fraud and are the loss of money or theft of information and software installation. Previously, malicious URLs were often detected by blacklists, but they currently lack edited URLs addresses that are newly created. The focus of the attackers is to use more effective methods of stealing information, which is phishing technology, which is one of the most used types of social engineering.

Therefore, the research aims to develop a model to detect and classify URLs into legitimate URLs or malicious URLs by using Machine learning techniques, algorithms, and the lexical feature extracting from URLs.

We also verify that a new set of samples that were never trained on as well as a plausible scenario that the predictive model can predict processes can distinguish between legitimate and malicious samples.

Keywords: Malicious, Machine learning, Detection URLs

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

1. INTRODUCTION

1.1 BACKGROUND

Data is a valuable and important asset in the information technology systems for companies institutions and governments and this means more crimes and with the high demand for various sectors of information technology and the rise in technology artificial intelligence and cloud computing and data intelligence known as the term (big data) has become the data used by companies and institutions of great importance For this, cybersecurity has become an important role to preserve data from malware with new technologies developed by attackers and we can classify the most common attacks of URLs malicious link are phishing, spam, and download-from the drive so URLs addresses are the dominant ways for attackers to infiltrate, steal data and spread malware.

According to reports documented by Verizon in 2020 about phishing and attacks on cloud-based data, these attacks were high, 86% of data breaches of financial gains in the 2019 report, and attacks on cloud web applications doubled to 43% and 67% were breaches by stealing credentials and 1-20 breaches exploiting security vulnerabilities. The report analyzes more than 32,000. Financial gains are the main reason for computer crimes and the ratios are almost high because of these breaches caused by third parties and organized crime that caused data theft based on malware.

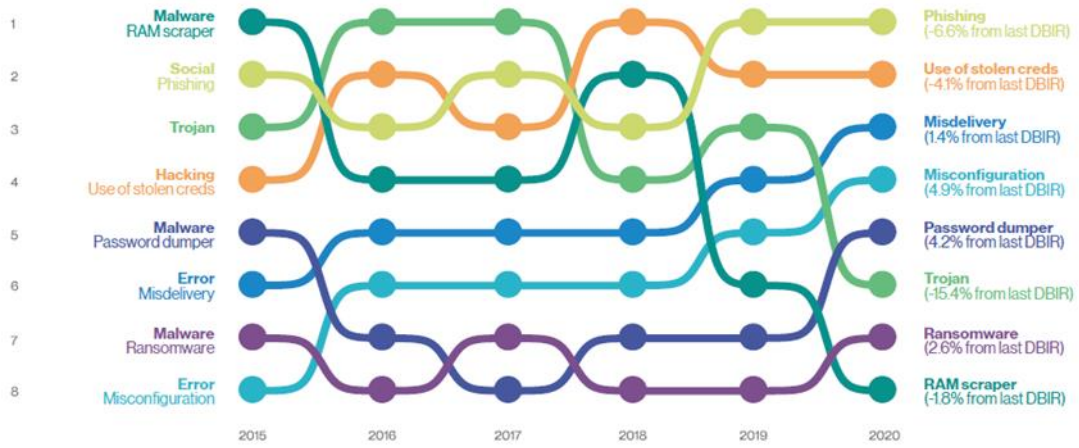

Figure 1.1: Action varieties in breaches

1.2 PROBLEM STATEMENT

Many of the solutions currently applied are based on the detection of malicious URLs addresses using antivirus database information and analysis of certain patterns based on historical patterns of attacks that have occurred and it is critical to identify and react to these risks as soon as possible because of the existence of malicious software that must be addressed using the best techniques to prevent attackers from infiltrating and spreading malware previously was the detection of malicious URLs in a manner Blacklist, a database of malicious URLs in these lists, does not contain malicious URLs because of the lack of new malicious URLs and there are also implementation environments known for their terms (sandbox) that are set up to examine the behavior that occurs when accessing the URL received as a scam and determining whether it is malicious or legitimate content. However, they remain significant countermeasures to the possibility of attackers developing more complex attacks to detect and due to A predictive model will be trained using data sets that have already been classified according to a series of features, including URL. Automated learning will be used as well as the existing automated algorithm.

1.3 OBJECTIVES

GENERAL OBJECTIVE

The main objective of this master's work is to perform the detection of URLs (uniform resource locators) that are used as a social engineering tool for obtaining confidential information fraudulently, using machine learning software techniques.

To meet the general objective, the most relevant specific objectives that I plan to achieve in this research work are the following:

1. Explore how the cybercriminal uses the fraudulent URL to make the victim believe that he is a trusted person and acquire their confidential information.
2. Analyze what are the parameters and attribute that contains a fraudulent URL to determine how they are used in phishing

3. Understand the operation and implementation of a machine learning system.
4. Analyze the different machine learning algorithms existents to determine which ones can be useful for the overall objective to be solved.
5. Learn how to implement a machine learning system by using a programming language.
6. Contrast through the results obtained with the learning algorithms used if the detection performed approximates or improves the results of other authors related to the topic of the work.
7. Learn the different classification techniques used by learning algorithms to better understand how the results they extract are obtained.

1.4 METHODOLOGY

The methodology used to meet the objectives of this project is based on some good practices that are applied in the iterative and incremental software development model. We have chosen some ideas from this model because it allows you to plan in the different iterations, the different requirements to implement and verify together with the milestones of the deliverables of this work. So, at each iteration, the requirements to be met at each stage of the project are planned.

The objective of this methodology is to fulfill the tasks planned at the beginning of the project and be very clear in each iteration what work must be done and the time stipulated for each task and its dependencies. It is important to add that it is necessary to adequately satisfy the tasks assigned in one iteration, because the tasks assigned to the next iteration will depend on the work done later.

Therefore, this project will have the following iterations:

Development of the work plan

In this initial iteration it is intended to elaborate all the planning of the project that is carried out and the list of tasks necessary for its resolution. This phase consists of few days for its elaboration and is one of the most important phases since the problem to be solved is explained and the previous study of the requirements that the type of project entails is carried out.

Analysis and design of the project

This iteration goes into depth in the analysis of the attacks that are currently carried out and what information will be extracted from the data sets of victims who have suffered this type of attacks. To do this, it is also important to know the operation and implementation of machine learning systems and how the algorithms that serve for the elaboration of predictive models act. In addition, the use cases, architecture, and requirements of the computer system to be implemented are defined.

Deployment and testing

This iteration puts into practice all the previous analysis performed on existing technologies and uses the programming language chosen for the implementation of data analysis using machine learning techniques. This allows us to subsequently obtain metrics and make an assessment of the results obtained.

Presentation of results and conclusions

In this final iteration, as the last part of the project, the results obtained with the different algorithms used by the defined predictive models are presented and that so many percentages of success extract based on the set of data processed and how effective in the solution implemented for use in a real scenario. So that it can be used as a detection engine for attacks of this type in a corporate environment.

1.5 MOTIVATION

The motivation for this work is to focus on the application of artificial intelligence and be more challenging and oriented to automated education using algorithms and using this methodology increasingly and developing predictive programs that allow to address problems of different types and large amounts of data and to solve these problems is to analyze a wide range of samples of URL and classify them as malicious and legitimate and thus we have identified information that will be useful in continuing to train the predictive model and in this way we

have a trained system in which we have a new set of data that determines the likelihood of success in classifying legitimate or malicious addresses.

1.6 CONTRIBUTION

This study aims to conduct the detection of the malicious URLs that use social engineering as a tool to detect sensitive information fraudulently and how the cybercriminal uses the fraudulent URL to make the victim believe that the person is trusted but steals his confidential information. For malicious URL titles analyze different and existing machine learning algorithms to identify those that may be useful for the overall goal to be solved and learn the different classification techniques used by learning algorithms to better understand to get the results extracted either in contrast to the results obtained with the learning algorithms used and if the same approach or improves the results of other authors related to the subject of work and if we get good results it will be possible to create a system that detects malicious URLs in real-time without the need for the Blacklist.

1.7 THESIS ARRANGEMENT

The thesis is arranged accordingly: Section 2 is where we look back at some of our prior work. and implementation on smart grids. Section 3 is where we give a background about all the components. Section 4 is where we explain our model in detail and implementation of that model in detail and where we simulate our model and record the results obtained. Section 5 is where we conclude our work and put a future scope into perspective.

2. LITERATURE REVIEW

One of the emerging problems facing academic middles in the discovery of malicious URLs bar and the anti-phishing technique depends on the method of using attackers to malicious URLs which appear to be a malicious URLs to trick users to steal their sensitive information and has found many studies to solve these problems through the application of electronic learning techniques and in general and measure false-positive and false- negatives. With the increase of attacks, the need to secure the URLs has led to many users of internet applications and those users do not know all the pages and URLs are malicious and this exposes them to the risk of fraud and the most dangerous is phishing, which is one of the most common attacks faced by users while browsing. The attackers use websites and URLs to deceive users and obtain sensitive data such as passwords and usernames and recent interest in this topic has focused on machine learning and the availability of algorithms helped to address these problems and achieve a wide and appropriate set of literature in the realm of artificial intelligence and cybersecurity.

There are several ways to detect phishing attacks based that are mainly URLs and there are advantages to URLs detection, including the classification algorithms used and the extraction of features. Many researchers have focused on URLs analysis and external information to determine the parameters of URLs.

The authors [1] used the stacked restricted method Boltzmann machine to select a feature from the vector of URL lexical and host-based features and then feed these features into four classification algorithms: the Deep Belief Network (DBN), the Artificial Neural Network (ANN), the Support Vector Machine (SVM), and the Naive Bayes (NV) to detect malicious URLs and the results were proving that the performance (DBN) exceeds the performance (SVM, NB and ANN) accurately at a false positive rate and a true positive rate. The advantage of these models is their ability to detect every category of URL malicious: phishing spam and malware attacks.

The authors of [2] dissected dubious sites utilizing a technique alluded to as Search and Heuristic Rule & Logistic Regression (SHLR), which comprises of three sections: first, perceiving a benevolent site exclusively by utilizing the title label substance of the site as the fundamental passage to a web search tool; second, if the site couldn't be distinguished as un-phishing,

removing the URLs highlights (word records) and utilizing seven heuristic standards to identify phishing pages; and, at long last, utilizing three classifiers to perceive the remaining sites, to be specific: a strategic relapse, Naive Bayes and SVM, and separating URL highlights from Whois, HTML, DNS, lexical highlights and likeness with phishing jargon. Their outcomes show a high exactness, their methodology relies upon a web search tool and an outsider. Utilizing the web qualities and the highlights of the URL string,

The authors of [3] developed an approach in which the URL text is fed directly into a convolution neural network, which collects and aggregates locally discovered characteristics before classifying dangerous URLs. They compared their results based on a system that automatically extracted features using two manual extraction approaches and found that the method was efficient based on the neural networks of the convolution.

The authors [4] has contributed by collecting more than 213 thousand users who compromised on Koobface, such websites are taken more than four days to blacklisted? and only 27% of URLs are detected as malicious and stated 81% of users clicked on Koobface spam.

The authors [5] has conducted two years of unique research on short URLs, analyzing the security problem and explaining the countermeasure. They said users are rarely exposed to a threat when browsing through a short URL or at least no more.

The authors [6] has Examine how many phishing sites are created by modifying a popular website. He uses content-based methods to examine phishing sites, such as lexical analysis, HTML content, domain name, and Google page rating.

The authors [7] has He did similar work, but he added more particular variables such as host machine features such as IP address, WHOIS, domain name, and regional location, and he categorized them using Bayesian and SVM machine learning classifiers. Exposed through long URLs to the same thread.

Dong, Shang and Yu [8] Reported the use of a malicious URL detection method that showed promising results. Master learning approach This research also examines comparable state-of-the-art approaches that used not only machine learning but also a variety of other techniques. Approaches for detecting malicious URLs can be split into three groups based on the published

literature: (1) machine learning approaches, (2) blacklisting approaches, and (3) heuristic approaches. These methods are described further down.

Sahingoz, SI Baykal and D Bulut [9] has been analyzed is which uses two types of classifiers as a prediction methodology, artificial neural networks and the neural network in depth where the components of a URL are extracted, and the most relevant characteristics are selected for classification. To do this, they extract 16 features that allow the predictive model to conclude fraudulent or legitimate URLs. This study demonstrates a 90% success solution. However, for such detections, algorithm runtime latency is important, so they have limitations in calculating one of the features used in the model, and you need to disable this important parameter for discovery.

Another researcher R.Kiruthiga, D.Akila [10] focused on this topic is that of which uses machine learning algorithms such as Naive Bayesian, SVM, Decision Tree and Random Forest. In addition, he says they propose a new system called PhishScore and PhishChecker to add accuracy to URL detection. The authors of the paper commented that they have 93% in this type of detection, despite this, they describe that the techniques employed by cybercriminals who use phishing techniques constantly change and believe it necessary to improve the system by adding new features.

Ultimately, we analyze the work done by Ms. Sophiya Shikalgar [11] that in addition to using common algorithms such as (NB, RF and SVM) introduce another called XGBOOST that produces a prediction model in the form of a set of weak prediction models, this means that it builds the model in a staggered way. In this research, they extract about nine features and train models with a dataset of around 2900 URLs. The metrics they get, even though they say they are good, are lower than the other research analyzed and the highest accuracy they get is 86.3% using SVM. It may have been due to the small data set used in the training of the models or features used in the process.

3. MATERIALS AND METHODS

3.1 PHISHING ATTACKS

Before analyzing the different attack methods that can be found when analyzing a URL that is used as phishing, a brief explanation of how a URL (Uniform Resource Locator) is composed is required. The main purpose of a URL and so it is created is to easily direct it to a web page and not have to use an IP address that is difficult to remember. The following figure (extracted from the internet) shows the basic structure and the different parts of a typical URL:

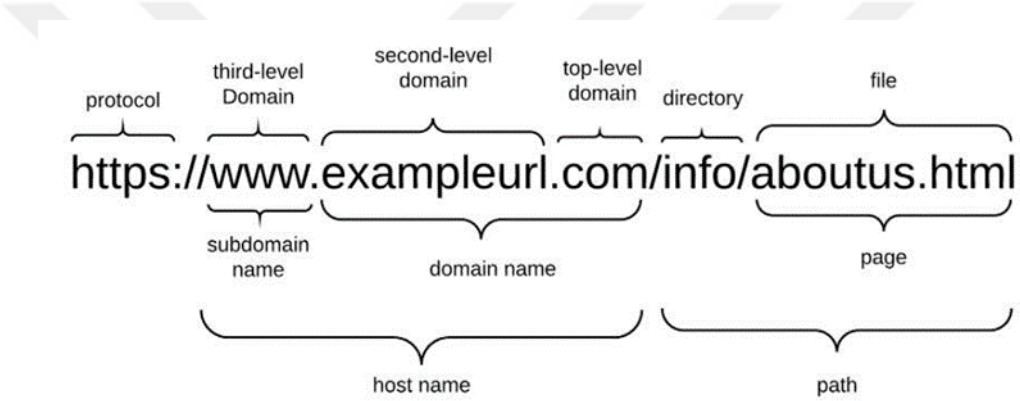

Figure 3.1: Parts of URL

From the image above it is extracted:

- Protocol: protocol used for website access.
- FQDM (Fully Qualified Domain Name): Identify the web page hosting service. Includes the following domain levels:
    - Hostname or subdomains. It's the part that attackers normally take advantage of to trick the victim.
    - Domain. This is the part where you need to pay to purchase from a web service or TLD (Top Level Domain). Indicates the top-level domain.

- Directory: Paths of the resources that the website hosts.
- File: Indicates the page or content file that is intended to be displayed to the user.

For example:

https://sis.altinbas.edu.tr/login.aspx?lang=en-US

From which we extract the following:

- **Protocol:** https//
- **Subdomain:** sis
- **Domain:** altinbas.edu.tr
- **Directory:** login.aspx?lang=en-us

Figure 3.2: parts of a URL

Scheme: The scheme is the first element of the URL, and it specifies the protocol which the browser has to use to request the source, which is commonly HTTPS or HTTP for websites.

Authority: The authority is then followed by the scheme, which is separated from it by the character pattern :/. If present, both the domain (for example www.example.com) and the port (80) are included, separated in this Authority by one colon:

- The domain identifies the Web server that is being accessed. This is usually a domain name, though it could also be an IP address (but this is rare as it is much less convenient).
- The port is the technical "gate" that allows you to access the web server's resources. If the webserver grants access to its resources using conventional HTTP ports (80 for HTTP and 443 for HTTPS), it is frequently removed. Otherwise, it is required.

Path for resource: */path/to/myfile.html* is the URL of the Web server's resource. A path like this used to reflect a physical file location on the Web server in the early days of the Internet. It is primarily an abstraction handled by Web servers nowadays, with no actual substance.

Parameter: *?key1=value1&key2=value2* are additional parameters that the Web server receives. The & symbol separates a list of key/value pairs in those parameters. Before returning the resource, the Web server can use those arguments to do further tasks. Each Web server has its unique set of parameters rules, and the only trustworthy way to find out how a particular Web server handles parameters is to ask the Web server owner.

Anchor: *#SomewhereInTheDocument* is a link to a different part of the material. An anchor acts as a "bookmark" within the resource, instructing the browser to display the content at that "bookmarked" location. The browser will go to the location where the anchor is specified in an HTML page, for example, in a video or audio file, the browser will try to navigate to the time the anchor indicates. It's worth noting that the fragment identification, which comes after the #, is never provided to the server with the request.

3.2 TYPES OF MALICIOUS URLS

We can distinguish different types of malicious URLs, based on unsolicited content they host and the type of cyberattacks they are aimed for. The most common are spam, phishing, and drive-by download, each type can have its distinguishing features, and a different methodology can be successful in detecting them.

Spam

Webspam refers to websites that are trying to cheat search engines to be ranked higher and increase their traffic. URLs leading to such websites can be called spam URLs. Despite the higher rank, users will not find the legitimate content they searched for. Spam content and spam links are the two most popular types of Webspam techniques.

- **Spam content.** involves all techniques that modify the page content to be rank higher. This includes adding popular words to the actual content, modifying the parts that have a higher importance in a ranking, like a page title or anchor text and other identifies the harmless links. of the page.

- **Spam link.** utilizes ranking algorithms that are focused on links. The more highly ranked websites that link to a website, the higher its rating is determined by these algorithms.

Phishing

Phishing websites try to steal confidential information such as card numbers, bank account numbers, or passwords from users by tricking them to believe they are on a legitimate website. URLs and content of phishing websites resemble original content, making it hard for the victim to recognize the difference. Acquired sensitive information is then mostly used for stealing victims' identities or money.

Drive-by download

The drive-by-download refers to a usage device after visited compromised websites that unintentionally download malicious code. To begin the download, the victim does not need to click on anything. Simply visiting an infected website might cause malware to be downloaded and installed in the background without the user's knowledge. The software can then be used to take control of the infected machine, steal passwords and other sensitive information, demand ransom money, and perform other undesired operations.

3.3 METHODS

Methods that include a URL in the attack

1. Mix legitimate URLs (from the official website you want to impersonate) with the phishing URL in the email sent to the victim.

2. It is based on URL redirection. The technique is known as "time-bombing" is based on the creation of a URL redirect from the legitimate page to which phishing is intended. This technique is somewhat more elaborate since it sends the legitimate URL via mail to the victim

and once it is received (which has evaded the filters of the mail), the fraudulent redirect page is created.

3. Embed in the body of the email message an image instead of text, with the content that would send you the legitimate page. In this way, a hyperlink is embedded in the image, which when clicked will redirect you to the fraudulent page.

Methods that rely on URL manipulation

1. **Use of sub-domains:** This method is based on the use of subdomains of the URL to trick the victim into believing that they are visiting the legitimate page. Since a common user is often unaware that the domain hierarchy goes from right to left, the attacker reverses the order and have believed the victim is visiting the legitimate domain, when in fact he is accessing the fraudulent domain.

2. **URL hiding/shortening:** A common way to trick the victim and that does not require much complexity of execution is to hide the URL of the fraudulent page by including an alternative text string such as “Subscribe” or “Click here". This way the victim does not see the strange URL that might make him suspicious and is more susceptible to being accessed. Another way to exploit this technique by attackers is based on the use of online tools such as TinyURL and Bitly that allow shortening the original URL that is used as phishing.

3. **Misspelt URLs:** In this URL manipulation technique (also known as URL hijacking), the attacker acquires a domain similar to the fraudulent URL to which he wants to impersonate, but with spelling errors. There are different variants of exploiting this type of attack, described below (source: Wikipedia):

   Misspelling. Example: ejemple.com.

- typo. ejemlop.com
- Different expressed URL. ejemplos.com

- Use a different TLD (top-level domain). ejemplo.org

4. Use a different ccTLD (TLD country code). IDN (internationalized domain name) homograph attack. This method attempts to take advantage of using visually similar characters. For example, the uppercase i (I) and the lowercase L (l), which are similar.

   Other ways to prevent URL phishing detection are based on maintaining a blacklist (including keyword lists, URLs, and IP addresses), maintaining a whitelist, inspecting page content, or performing a manual analysis of the page's visual content.

3.4 URLS FEATURES

Using the whole URL string as only information for detecting URLs may not be enough. Therefore, the necessary first step is selecting and extracting useful features that sufficiently describe the URL. Working with features and not using URLs as a whole string would allow creating more effective and successful detection algorithms. For each type of malicious URL, a different set of features can bring better results. Several types of features have been offered in papers addressing malicious URL detection. We can categorize these features into Blacklist, Lexical, Host-based, Content-based features, and Rank-based features [12].

3.4.1 Blacklist Features

A blacklist is essentially a listing of URLs that were previously identified as malicious. It is continuously updating over time using a combination of automatic mechanisms and humans. Using a blacklist for the identification of malicious URLs is considered a very fast and easy technique to implement with a very low false-positive1 rate. Unfortunately, this technique fails to detect newly generated URLs and therefore suffering from a high false-negative2 rate [13]. However, instead of using a blacklist as a solo identification tool, it can be a handy feature for more advanced techniques. Detection tools can use their own created blacklists, or because of the

high data storage capacity needed, one of many public blacklisting APIs such as from Google Safe Browsing [14] or Phish Tank [15].

3.4.2 Lexical Features

Lexical features idea is based on the assumption that malicious websites look different and thus have distinguishable patterns in their URL text. In many cases is a user able to recognize suspicious URLs just from the look. This applies mostly to phishing URLs, where the URL tends to look similar to the original page it claims to be. For example, taking the URLs: https://www.paypal.com/ and https://paypal.co.uk.b8wt.icu/n/

The second URL has from first sight too many domain names and looks suspicious.

Under lexical features, we're talking about the URL's entire linguistic properties, not the content of the page it links to. Lexical features are attractive to use because they are fast to process, need low amounts of data storage, and can be obtained without having to call other services.

Common detection techniques usually treat lexical features of hostname and path separately, because of their different impact on classification [16]. According to Sahoo, Liu and Hoi [17] we can divide lexical features into Traditional and Advanced one.

- **Traditional lexical features**

  Traditional lexical features Include all lexical features commonly used. Most of them are a combination of features motivated by McGrath and Gupta [18] and Kolari et al. [19]. The hostnames are covered by their length, URL and frequencies.

of the characters in the URL. Further, the presence or order of tokens in the path (separated by a '.') and in the hostname (delimited by '.', '/', '?', '=', '-', ' ') are used.

- **Advanced lexical features**

  Advanced lexical features need more computation and are more complex than traditional ones. However, they are more successful in the detection of various obfuscation techniques. Typical examples of obfuscation techniques are (a) host replaced with an IP address, (b) targeted domain in the path (c) large hostnames, additional domains (d)

domain unknown or misspelt [20]. Le et al. [21] proposed advanced features such as the presence of an IP address, the number of special characters, or statistical properties about tokens such as their count, average, or maximal length.

A feature that was used by Daeef et al. [22] or Verma et al[23]. is the usage of n-grams, where instead of the presence of single words we identify the presence of n-characters sub-strings of tokens. This way, we can detect sub-words or misspelt words.

Verma and Dyer [24] used as one of the features Euclidean distance, to spot the differences between phishing URLs and standard English.

Kolmogorov Complexity (or called Kolmogorov entropy, algorithmic entropy) used by authors Pao et al. [25] can be considered as a more advanced feature. This method, in the case of detection of malicious URLs, is deciding if a URL is benign or malicious by comparing if it has more similar character patterns to a given benign or malicious URL database.

3.4.3 Host-Based Features

Specify the attributes that the hostname component of the URL identifies [26]. They allow us to make educated guesses about malicious hosts' location, owner, registration date, management style, and other characteristics. These factors improve detection accuracy overall [27]. but are all-time expensive, considering they are not derived from URL itself but gathered from third party services. Typical host-based features used are:

WHOIS properties. date of registration/update/expiration dates, registrars, and registrants of the domain name. According to Ma et al. [28], a malicious URL often has a registration date or update date in the recent past. Also, if a single person registers a set of malicious URLs, it should be considered a malicious feature.

Domain name properties. TTL values, the presence of a pointer (PTR) record, or whether one colour the host's IP addresses are resolved by a PTR record. Furthermore, we can include here features that can be considered also like lexical features such as the presence of words server and client or IP address in the hostname.

Geographic properties. physical geographic information - The continent, country, or city where the IP address is located.

Connection speed – speed of the uplink connection. Reputable websites tend to have faster connection speeds.[29]

3.4.4 Content-Based Features

The features received after opening and downloading the website are known as content-based features. Those features are data-intensive but provide the most accurate information about the content of the website. However, for phishing websites, they may not be effective as a phishing website should have similar content as the original website it pretends to be. We can divide Content-based features into three sub-categories: HTML, JavaScript, Visual features, and Other Content-based features [30].

- **HTML features.** mostly contain the statistical features of the HTML document, such as page length, the average word length, word count, and distinct word count, or white space percentage. Furthermore, statistical features about certain HTML elements such as count the numbers of HTML tags, lines, hyperlinks, or iframes are also used.

- **JavaScript features.** statistical properties of JavaScript code used on websites. Choi et al. [12] counted the number of seven suspicious native JavaScript functions: escape (), eval (), link (), unescaped (), exec (), link (), and search ().

- **Visual features.** There has additionally been attempting victimization pictures of webpages were made to detect the URL's malicious nature. Most of them Wherever protected pages coexist with legitimate websites, they emphasize identifies the calculating visual similarity with protected pages. A significant degree of visual similarity between a suspected malicious address and a legitimate address and another address could indicate a phishing attempt. One of the initial attempts was to use visual alternatives two calculate the distance between two images. [31]. tackled the same issue and created a visual extraction system Text-block choices and image-block features were supported by webpage characteristics (Using data such as block size, colour, and other variables) This work necessitated the use of a variety of modern computer vision

technologies. Options for Contrast Context Histogram (CCH) were given first [32], followed by Scale Invariant Feature Transform (SIFT) options [33]. [34] devised another method of exploiting visual features, the OCR device used to browse that text with the image of the webpage. [35] To measure similarity, blend each matter and visual alternatives. In recent improvements to deep learning, we can recognize the image, extracting a large number of powerful and useful visual features should be possible.

- **Other Content-based features.** [36] argued that because ActiveX components are so handy, can be used for the harmful creation of DHTML pages. They attempted to determine recurrence for each of the eight ActiveX objects in this manner. "Scripting.FileSystemObject" may be used for document framework I/O activities, "WScript.Shell" can be used to run shell contents on the customer's PC, and "Adodb. Stream" may be used to download documents from the Internet. [37] The iteration is supposed to be aimed at the character and the consistency of the character and the character seen in the attempt to find the character and the words seen in the message, which is discovered by examining. [38] used the catalogue design of the website to gather information.

3.5 OTHER FEATURES

Context feature: What they allow to represent the original URL are service providers, and short URLs, have grown in popularity in recent years on social media platforms where it does not engage with the long-term URL, Unfortunately, this technique has become commonplace to popularize malicious URL, while the providers of these services are trying their best to prevent the creation of a short malicious URL [39].

As a result, the recent search for the characteristics of the URL context has become a trend. It means that basic information can be shared with the URL. Use ongoing context information from conversations that have been shared by URL [40]. The short URL is classified as harmful or not based on click traffic data [41]. To combat forwarding-based harmful URLs, [42] propose forwarding-based features. [43] proposes a different set of features for detecting harmful URLs; They concentrate on URLs shared on social media and attempt to assess whether or not a URL is harmful. by analyzing the behaviour of neighbors, the number of people who shared it and the

number of people who clicked on it. These functions are properly known as "Posting-based" and "Click-based" functions.

Popularity feature: Some of the other settings are provided as heuristics to aid in URL recognition. One of the older strategies for detecting rogue URLs was to employ statistical techniques. [20] is to determine the importance and probability of specific and hand-designed features, including these features used to range the type of features based on (circular types) and word-based features. [44] The registration of the original URL address as well as the use of URL-based features, URL landing address and redirect chain as well as records several renegade windows and delivered extensions that were commonly used by spammers. These features are used in conjunction with the content lexical features and features used to host to make strong manipulation and use of some metrics that check the quality of links is suggested as a measure used for URL detection for spam they use in combination with lexical characteristics spam used to the content and features used to host to make strong manipulation and suggests the use of some metrics that verify the quality of the links and use a measure to detect URL for spam to spam they use these features in [45] tracked URLs' public share counts on Facebook and Twitter to determine their social reputation. Other initiatives used information from redirection chains to create redirection graphs, which helped detect malicious URLs [46]. [47]My query search engine data for word connectivity determination.

3.6 MALWARE

Malware is a type of computer virus. (Mal war) Malicious software is abbreviated Programmable as malware. This includes (code, scripts, active content, and other software) that is designed to interrupt or deny procedure, gather data for exploitation or privacy invasion, gain unauthorized access to the resources, and engage in other abusive behavior Spyware, Trojan horses, viruses, web bugs, and worms are examples of adware, hijackware, slag code (logic bombs), spyware, Trojan horses, infections, and web bugs [48]. are just a few examples. We are going to explain what exactly malware is, a type of threat that has many types, from computer viruses to ransomware. Sometimes it can be a bit confusing to find you so many names to talk about the different types of malwares, and we end up not knowing the differences well. The type of

malware including computer viruses, Trojans, worms, spyware, adware, and ransomware. However, malware is the main term we use to talk about all those computer threats. The way of acting of each one is usually different, but in common they have the objective of damaging the equipment or its user, and that of being installed in the computer or electronic device without you wanting it to be there or without you noticing. For the latter, each type of malware will use different techniques that range from camouflaging itself as a normal application, tricking you into installing it or being installed by a different application. However, all applications that can expose your data or create unwanted operations on a computer are not malware. For example, faulty software is not malware, although it may end up having similar effects in some cases, as it is simply programs that were designed with good intentions, but whose code has bugs that cause it to malfunction.

3.6.1 Malware Types

Virus: Viruses are the oldest malware out there. These pieces of software focus on replacing part of the source code of the system executables so that in the most aggressive cases the files are destroyed and in the less aggressive they are simply a nuisance for the users (showing error messages or generating load processor and useless traffic, for example).

Unlike worms, viruses cannot replicate, so they only focus on infecting the system and damaging it according to the purpose for which they were programmed. Today viruses are practically extinct and represent one of the smallest parts of global malware.

Scareware: is a type of malware that uses social engineering to deceive users into buying unwanted software by inducing shock, anxiety, or a sense of threat. Scareware is a type of malware that includes rogue antivirus, ransomware [49], and other types of scam software that convinces users that their computer is afflicted with a virus and then demands that they download and pay for actual antivirus software to remove it. Typically, the virus isn't real, and the program or malware isn't working. The number of malware threats in circulation jumped from 2,850 to 9,287 in the second half of 2008, according to the Anti-Phishing

Working Group. The APWG found a 585 per cent rise in the number of malware threats in the first half of 2009. Scareware can also refer to any application or virus that deceives users to instil fear or anxiety in them.

Worms: The main characteristic of the worm is the ability to duplicate itself. Worms unlike viruses are capable of infecting multiple computers automatically without the need for human intervention. Worms mainly reside in system memory and their main function is to generate network problems, unlike viruses whose function is based on file corruption [50]. Worms often use multiple protocols to automatically distribute themselves over the network without the need for user intervention.

Trojans: Trojans are currently the most dangerous and common pieces of malware. Trojans are primarily systems remote control tools. When a user gets infected by a Trojan [51], the hacker behind him can gain access to different system resources such as files, webcam, microphone, keyboard, screen, etc.

There are several types of Trojans:

- Back doors: Used to guarantee remote access to a system.
- Keyloggers: They record all keystrokes and send them to a remote server.
- Proxy: Establish a proxy between the victim and the hacker to filter all traffic and redirect certain websites.
- Password Stealer: They focus on password stealing, especially email and banking.
- Banking: They focus on the theft of credentials and bank data such as accounts, cards, etc.
- Downloaders: These are Trojans mainly used to download other pieces of malware to infect users

The main way to get infected with a Trojan is by downloading a malicious application from the internet. In order not to be victims of this malware, we must avoid downloading suspicious files attached to spam emails and illegal Internet content, since hackers often hide these applications in this type of content. Antivirus software will also help us avoid getting infected with this malware.

Spyware: Spyware is a type of malware specifically designed to spy on users. These programs usually focus on stealing different data from users such as browsing histories, cookies, users, passwords and other personal information that is later used by hackers to access different accounts of the victims and impersonate their identity or carry perform other tasks.

Adware: The main function of adware is to show intrusive advertisements to users. This advertising generally generates a profit for the hacker who distributes it and is often especially annoying for users. Similarly, data consumption by adware applications skyrockets, significantly slowing down the system. The adware also often modifies the system's DNS as well as the host file, redirecting users who try to surf the internet to a multitude of advertising web pages and display pop-ups or pop-ups when browsing the internet. Its function is not malicious since it generally does not steal data or damage the system, although they are very annoying.

Bots and Botnets: Bots represent one of the most sophisticated and popular cyberattacks today since they allow hackers to take control of computers, and at the same time, transform them into “zombie” computers, which function as a powerful botnet, which spreads viruses, generates spam, committing other crimes and fraud. A bot is a malicious program, taking control of an infected computer. Generally, bots that are also called web robots are part of a network of infected machines, called botnets, which in turn, is made up of victim machines from all over the world. Because a computer infected by bots carries out the orders that are given to it, many people tend to refer to these computers as zombies, a situation that grows every day. Some of these botnets can encompass hundreds or thousands of computers and others have tens or even hundreds of thousands of zombies at their service.

Ransomware: A new type of malware is Ransomware that is growing exponentially. This malware focuses on "hijacking" user data and demanding a ransom for it from victims. In some cases, it is simply a malicious application that tries to trick users into making a payment to "unlock" their computer, however, in recent months new ransomware models have appeared that encrypt the user's data and the only option to free them is to pay a considerable amount of money to obtain the decryption key to be able to recover the data. Without a doubt, this is one of the most dangerous malwares in recent times and its growth, as well as its arrival on mobile devices, is worthy of concern.

3.7 SOCIAL ENGINEERING

is the utilization of non-technological techniques to fool expected casualties into imparting their data to a hacker. Hackers employ disappointing practices Bank account details and other personal data to obtain their passwords. A hacker can break into a PC or other device associated with the web while never hacking? They can manipulate you into revealing personal information by building trust, and until it's too late, you may not realize it. You risk becoming a victim of social engineering, If, without checking the name of a sender, you open email links, or send your data to email.

3.7.1 Social Engineering Techniques and Types

Social engineering shows itself from various perspectives and structures. Some offline attacks can be carried out, like a well-informed foreigner who goes into your office and obtains the knowledge they require in person. Social engineering assaults can also be over the phone, performed. Vishing (voice phishing) is a type of phishing in which a person poses as a coworker or a reputable source and asks for the information they seek directly.

- **Phishing:** It is considered one of the most important and most harmful and common types of attacks in social engineering. Phishing is in the form of text and email campaigns aimed at treating anxiety and curiosity among victims. This prompts them to disclose sensitive information by opening attachments that contain malware or clicking a malicious URL. An example is an email shipped off clients of online help that cautions them of an approach infringement requiring prompt activity on their part, for example, a necessary secret key change. It incorporates a connection to an ill-conceived site — almost indistinguishable in appearance to its genuine rendition — provoking the clueless client to enter their present accreditations and new secret key. Upon structure accommodation, the data is shipped off the aggressor. Given that indistinguishable, or close indistinguishable, messages are shipped off all clients in phishing efforts,

distinguishing, and hindering them is a lot simpler for mail workers approaching danger sharing stages.

- **Spear-Phishing:** During most phishing efforts, spear phishing targets certain groups of people sending e-mails to the most random addresses possible. Famous as phishers, hackers will use social media to collect information on their objectives, often called spears, to customize their phishing emails and make them appear more believable and functional. Phishers will pose as friends, business associates, or an outside institution that is tied to the victim in some manner to make their attacks seem more realistic. For instance, the phisher could pose as a victim's bank representative and apply that you submit the information they require. To deceive the victim and make him believe that the message is original and not fake, by imitating the official logo of the concerned bank.

- **Pretexting:** This is the use of an enticing pretext to draw the target's attention and entice them to fall. Once the potential victim is engrossed in the story, the hacker will try to dupe them into providing him with valuable information. This type of social engineering is used frequently in so-called Nigerian email scams, in which you are promised a significant sum of money if you disclose your bank account details. If you fall prey to the fraud, you will not only not see a penny, but you may also lose the money in your account.

- **Contact spamming:** The most common form of online social engineering is contact spamming. Attackers employ this approach to transmit spam messages to all contacts between their victims, as the name implies. The emails will be sent from the victim's contact list, giving them a more authentic appearance to the receivers. They also have a far lower chance of ending up in your spam folder. This method is incredibly simple to use. If you receive an email from a buddy with an informal subject (for example, check this out!), delete it. You'll find a text link if you open it. The link is frequently abbreviated, and if it is not clicked, there is no way of knowing what it is. If you click on the link, however, a replica of the email will be forwarded to all of your contacts, continuing the spam chain. Furthermore, the link could direct you to a rogue website, where spyware or other dangerous software could be downloaded into your machine.

- **Quid Pro Quo:** Quid pro quo, which comes from the Latin phrase "a favour for another favour," is a sort of social engineering that entails the hacker and his unwitting victim exchanging favours and services. Hackers will frequently pose as IT service experts and seek your registration information. that they can conduct a crucial ostensibly security test Additionally, you may be prompted to deactivate or to install your antivirus application, to grant access to your computer and allow malware to be installed.

- **Scareware:** Scareware consists of sufferers being bombarded with fake alarms and fictitious threats. Users are deceived to assume their device is inflamed with malware, prompting them to put in a software program that has no actual benefit (aside from for the perpetrator) or is malware itself. Scareware is likewise known as a deception software program rogue scanner software program and fraud were. A not unusual place scareware instance is the legitimate-searching pop-up banners acting for your browser even as browsing the web, showing such textual content as, “Your pc can be inflamed with dangerous adware programs.” It both gives to put in the tool (regularly malware-inflamed) for you or will direct you to a malicious web website online in which your pc will become inflamed. Scareware is likewise allotted thru unsolicited mail email that doles out bogus warnings or makes gives for customers to shop for worthless/dangerous services.

4. MACHINE LEARNING AND LEARNING ALGORITHMS

4.1 INTRODUCTION

Study computer algorithms that automatically improve by experience and data use [52]. It is visible as part of artificial intelligence. Machine learning algorithms construct a version primarily based totally on pattern data, "training data" is known to predict or decide without being programmed explicitly [53]. Machine learning algorithms are utilized in an extensive form of applications, including email filtering and pc vision, where traditional algorithms to perform the tasks necessary are difficult or impossible to develop [54].

A subset of machine learning is carefully associated with computational statistics, which specializes in making predictions with the use of computers; however now no longer all machine learning is statistical gaining knowledge of. The observation of mathematical optimization can provide methods, Machine learning theory and application domains. Data mining is a related area of study, which focuses on the analysis of data by unchecked education [55]. In its utility throughout enterprise problems, machine learning is likewise called predictive analytics.

Machine Learning Approaches: These approaches attempt to examine URLs and related domains or websites. Extracting excellent URLs functionality and training of a prediction model using malicious and benign URL training data. There are different capacities - static capabilities and dynamic capabilities that can be utilized. We analyze a web page using available information without executing the URL in statist analysis (i.e., executing JavaScript, or other code)[56].

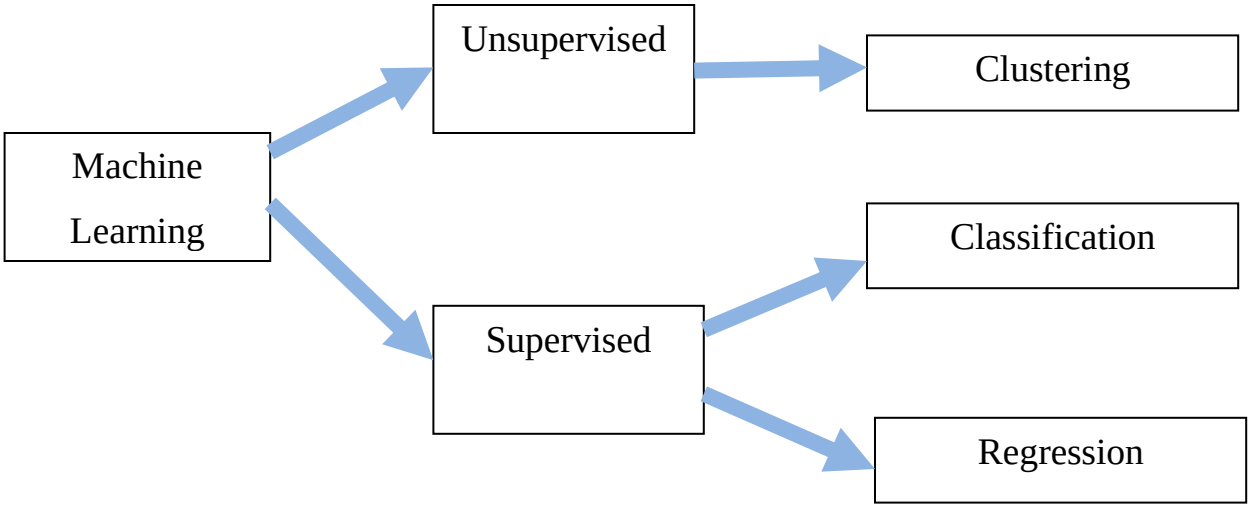

Figure 4.1: Machine Learning Types

4.2 MACHINE LEARNING TYPES

4.2.1 Supervised Learning

Explains a category of downside that includes a model to find a map between input and target variable examples. Supervised learning issues are those Training data consist of input vectors examples and their corresponding destination vectors. [57]. Models match inputs and outputs to coaching information and want to create predictions on check sets wherever solely the model's outputs, as well as the inputs, are matched with the hidden target variables and used for model capacity measurement. Learning is a search for a space that can be successful even in new models past the preparation set. We provide a hypothesis with a test set of models that are not the same as the preparation set to see how accurate it is.[58]. There are two fundamental sorts of supervised learning issues: they are that includes anticipating a class mark and relapse that includes foreseeing a mathematical worth. The training information you feed to the algorithm incorporates the ideal arrangements, called labels (Figure)

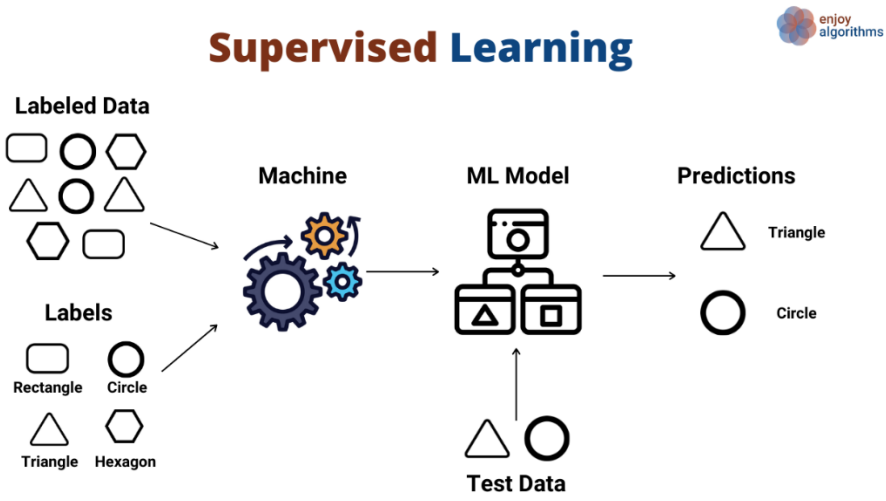

Figure 4.2: Supervised Learning

Classification: Classification is a subclass of supervised learning, in which the objective is to predict new instance category class labels based on past observations. The class labels are discreet, unordered values that can be understood as the group of instances members. A typical example of a binary classification job is the previously mentioned example of e-mail spam detection. Where The learning algorithm of the machine learns several rules to distinguish between spam and non-spam email in two possible classes. The figure below shows the idea of a binary class venture 30 samples of education, 15 education samples are labelled as a negative class (circles) and 15 training samples are labelled as a positive class (plus signs). In this situation, our dataset is two-dimensional, which implies that each example has two qualities related to it:

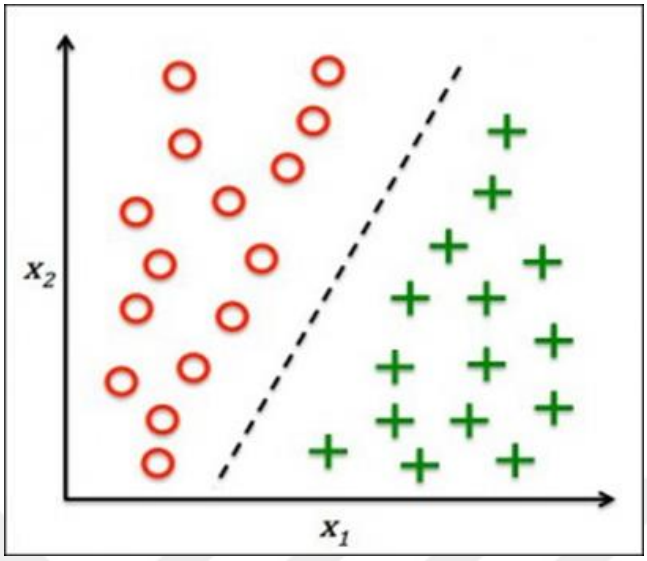

Figure 4.3: Sample of classification

The set of class labels, however, must not be binary Linear teachers tutors class label which was presented in the training data set can be assigned to a new, predictive model by a supervised learning algorithm, unlabelled instance. Handwritten character recognition is a typical example of a multi-class classification task. In this case, one training dataset could be collected consisting of several handwritten examples of each alphabet letter. Now, if a user gives an input device with a new handwritten feature, our prediction model can accurately predict the right letter in the alphabet. However, if they were not part of our training dataset, our machine learning system would be unable to correctly recognize one to nine digits.

Regression: Is a machine learning algorithm primarily based totally on supervised learning? It plays a regression task. Regression models a goal prediction fee primarily based totally on unbiased variables. It is usually used for locating out the connection among variables and forecasting. Different regression models fluctuate primarily based totally on – the sort of dating among based and unbiased variables, they're thinking about, and the range of unbiased variables being used. Galton depicted the natural marvel that the change of stature in a populace doesn't increment over the long haul. He saw that the tallness of guardians isn't given to their youngsters, yet the kids' stature is relapsing towards the populace mean. In this case, our dataset is two-dimensional and means that there are two qualities associated with each example. The figure accompanying the idea of linear regression is represented. In light of an x indicator and a y

reaction variable, this information is contained in a line that limits the distance between the exemplary focuses and the fitted line, most usually the normal squared distance. We would now be able to utilize the catch and incline gained from this information to foresee the result variable of new information.

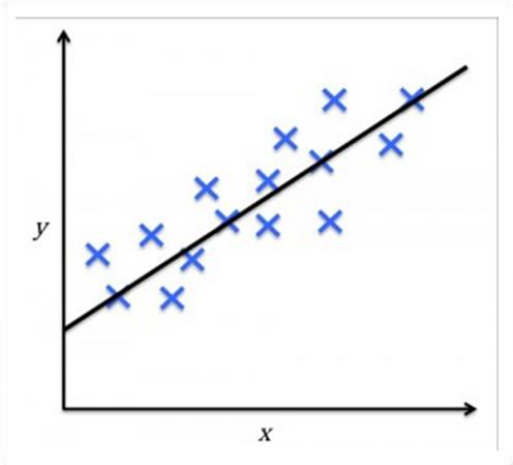

Figure 4.4: Linear Regression

Linear regression proceeds the assignment to are expecting a based variable value (y) primarily based totally on a given impartial variable (x). So, this regression technique reveals a linear courting among x (input) and y(output). Hence, the call is Linear Regression. In the determine above, X (input) is the paintings revel in, and Y (output) is the revenue of a person. The regression line is the high-quality shape line for our model. Among the most popular algorithms used in supervised machine learning for data classification are the following:

- Support Vector Machines (SVM).
- Logistic Regression.
- Naïve Bayes.
- Decision trees
- K-Nearest Neighbors (K-NN).
- Random Forest.

4.2.2 Unsupervised Learning

defines a set of challenges in which a model is used to explain or derive relationships from data. When compared to supervised learning, unsupervised learning is a better option. Uncontrolled learning works only with input data, however, not with yields or target factors. Uncontrolled learning, as in supervised learning, does not have a model correction coach. There are no unattended teachers or tutors, so the algorithm must figure out how to make sense of the data without it [59]. There are numerous sorts of unsupervised learning, although there are two fundamental issues that are regularly experienced by a specialist: they are bunching that includes discovering bunches in the information and thickness assessment that includes summing up the dissemination of information.

In unsupervised learning, as you may figure, the preparation information is unlabeled.

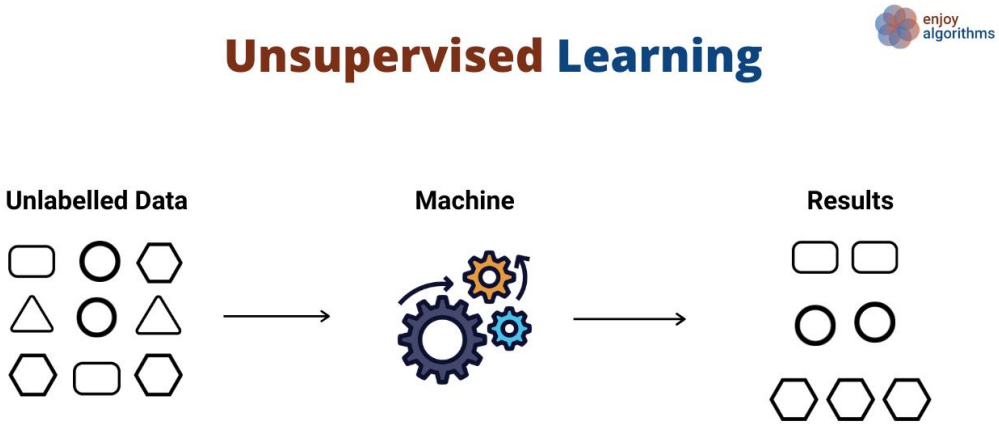

Figure 4.5: Unsupervised Learning

Clustering: Unsupervised learning issue that includes discovering bunches in information
Density Estimation: Unsupervised learning is a problem that entails summarizing information appropriation.

An illustration of a k-means, which refers to the number of bunches to be found in the information, is the algorithm for clustering. An illustration of a thickness assessment calculation

is Kernel Density Estimation that includes utilizing little gatherings of firmly related information tests to gauge the dispersion for new focuses in the issue space. Clustering, or recognizing potentially relevant clusters of input instances, is the most common unsupervised learning activity. For example, a cab driver can build a concept over time of 'good transport days' and 'low transport days' without the teacher ever showing the marked instances of each one [58]. Clustering and density assessment might be performed to find out about the examples in the information. Extra unsupervised methods may likewise be utilized, for example, the perception that includes diagramming or plotting information diversely and projection techniques that include lessening the information's multidimensionality.

Visualization: Unsupervised learning issue that includes making bunches of information.

Projection: Unsupervised learning issue that includes making lower-dimensional portrayals of information.

A dispersed plot framework, which creates one dissipate plot for each pair of components in the dataset, is an example of a representation process. Principal Component Analysis, which sums up a dataset in terms of eigenvalues and eigenvectors, is an example of a projection approach., with direct conditions eliminated. The goal of unsupervised learning problems like these is Clustering is the process of finding groups of similar examples in data, or density estimation is the process of determining the distribution of data within the input space, or projection is the process of projecting data from a high-dimensional space down to two or three dimensions for visualization [55].

4.2.3 Reinforcement Learning

Surely, you already know the 2 great areas of traditional learning of Machine Learning, supervised learning and unsupervised learning. It seems difficult that there was room for other options here; however, there is, and it is Reinforcement Learning. In reinforcement learning we do not have an "exit label", so it is not of the supervised type and although these algorithms learn by themselves, they are not of the unsupervised type either, where we try to classify groups taking into account some distance between samples.

If we think about it, supervised and unsupervised ML problems are specific to a particular business case, be it classification or prediction, they are very delimited, in contrast, in the real world, we have multiple variables that are generally interrelated and depend on other business cases and lead to larger scenarios in which to make decisions[60]. To drive a car intelligence that can detect a red, green, or yellow traffic light is not enough; We will have many factors -all at the same time- to which to pay attention: at what speed are we going, are we facing a curve? Are there pedestrians? Is it night and we must turn on the lights?

A solution would be to have multiple ML machines supervised and interacting with each other - and this would not be bad - or we can change the approach. And their Reinforcement Learning (RL) appears as an alternative, perhaps one of the most ambitious in which tries to integrate Machine Learning in the real world, especially applied to robots and industrial machinery.

In Supervised Learning (or unsupervised) models such as neural networks, trees, knn, etc., an attempt is made to “minimize the cost function”, to reduce the error. On the other hand, the RL tries to “maximize the reward”. And this can be, despite sometimes making mistakes or not being optimal.

Reinforcement Learning is comprised of four essential components:

Agent. The program in which you train to perform a specific task.

Environment. The physical or virtual environment in which the agent operates.

Action. A move made by the agent that results in a change in the environment's status.

Rewards. are the results of a good or negative judgment of an action.

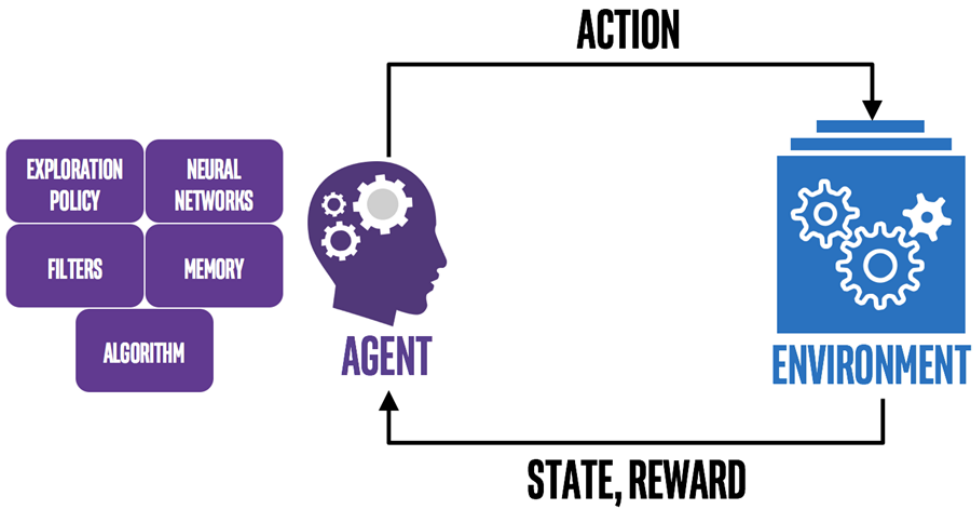

Figure 4.6: Reinforcement Learning

4.3 MACHINE LEARNING ALGORITHM

4.3.1 Support Vector Machines

It is one of the most important types most popular supervised learning and the most powerful machine learning models that can perform regression. Classification is linear or non-linear and even detection is further than that and it is considered the toolbox is one of the most common machine learning models[61], and everyone interested in machine learning should be familiar with it. The model is developed by a supporting vector machine, which takes the preparation inputs, maps them into the multidimensional space and uses relapse to discover a hyperplane, which best partitions two classes of information sources. When the help vector machine has been prepared, it can assess new sections with the isolating hyperplane and characterize them into one of two classes. SVM is an input/output machine[62]. A consumer is capable of input and access and, primarily based totally on the version advanced via training, it's going to go back and exit, The theoretical number of inputs for each supported Vector Machine varies between one and infinite. However, in practical terms, computational capacity limits the number of inputs that Can be used. Can be utilized. If N inputs for a given vector machine are used, for instance (the integer value of N can be from one to infinite), then each input set must be mapped of N dimensions and find a hyperplane of N -1 dimensions that best separate the training data[63]. SVMs, it is well

suited for classifying small and large complex datasets. It exploits the structural danger minimization precept the use The most marginal approach to learning can be considered essentially as a unique example of a regularized loss reduction framework. In particular, SVM may be formulated for subsequent optimization by deciding on hinge loss due to their loss characteristic and maximizing the margin.

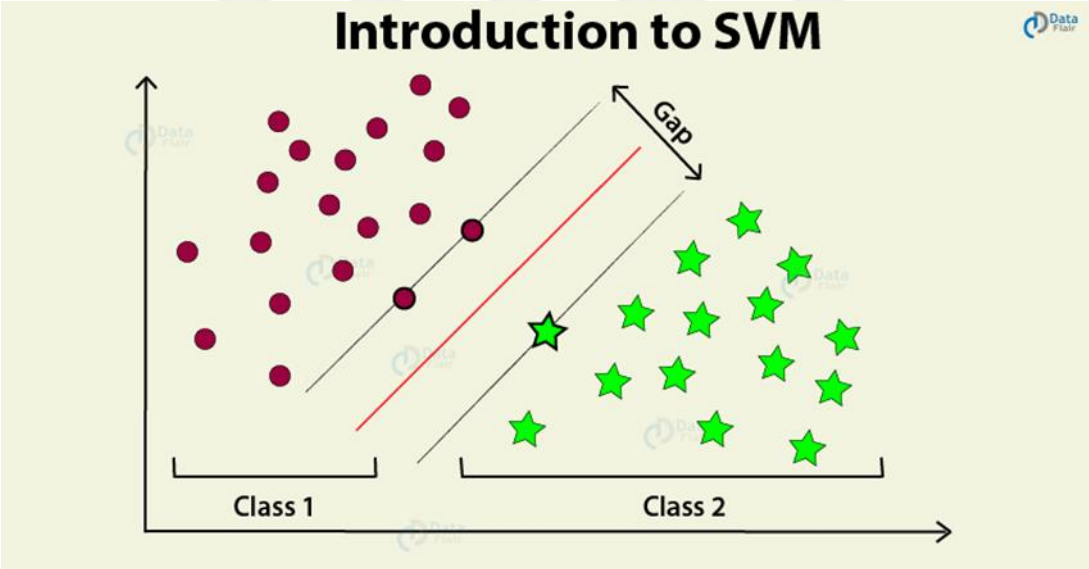

Figure 4.7: Support Vector Machine

SVM also learns to use kernels to learn non-linear classifiers [166]. SVMs are probably one of the most commonly used URL detection classifiers in literature. The following illustration shows the presentation of the data to be classified, where the solid line estimates the boundary between the two classes:

SVM does not always achieve optimal separation in the plane, this is known as overfitting. Therefore, SVM uses a parameter C that tries to control the compensation between training errors and rigid margins, creating a soft margin that is responsible for allowing some errors in the classification but penalizes them.

The illustration shown above would be an ideal case that it is possible to represent in two dimensions, but it is not always possible to apply the SVM algorithm only to this case, so this algorithm introduces rendering utilizing Kernel functions. These functions are responsible for projecting the information in a space of characteristics of greater dimension, which has as a disadvantage the increase of the computational capacity of the machines of linear learning.

4.3.2 Logistic Regression

Logistic regression is an algorithm that deals with classification questions and problems. It is one of the most well-known and used machine learning algorithms in the world[64], being used in different areas, such as cybersecurity and biology. First of all, it is important to keep in mind that logistic regression is not just used to classify things between two categories. For example, a given email is spam or not. Another example: the patient has cancer or does not have it. Yes, logistic regression is widely used for two-class problems. But not just for that. Being a bit greater technical, logistic regression works with the standards of information and probability, Logistic regression measures the connection among the explicit based variable and one or greater unbiased variables, estimating the possibilities the use of a logistic function[20], this means that this type of machine learning algorithm analyses different aspects or variables of an object and then determines a class in which it fits best.

Logistic regression is e famous discriminative version that computes the conditional possibility for a characteristic vector x to be categorized as a classy

$$P(y = 1|x; w, b) = \sigma(w.x + b) = \frac{1}{1+e^{-(w.x+b)}} \qquad (4.1)$$

Eq. (**4.1**) computes the conditional probability

Logistic regression is a completely powerful sort of gadget gaining knowledge of the algorithm. So, it's far one of the favored algorithms of our improvement team. Consequently, it's far one of the maxima used algorithms in our e-mail safety and safety solutions. Here at Gateway, we've got followed the multinomial logistic regression version as one of the most important mechanisms of our synthetic intelligence system[17]. To make it simple, let's create an instance of the usage of your business. Regardless of the way massive your corporation is, it needs to acquire dozens, loads or maybe hundreds of emails daily, Now imagine that your email protection resolution classifies messages into simply 2 groups, exploitation binomial supplying regression: spam (unwanted messages) and ham (desired messages). What would the e-mail inboxes of all company workers seem like today?

They are probably to be somewhat messy, declaring many false positives and false negatives. That is, many desired e-mails would be blocked and plenty of unwanted e-mails would are delivered. The results of this are often that workers would pay tons of your time analyzing these messages and, worse, they might be additional exposure to some variety of fraud that may place the whole company at risk, love phishing attacks, for example. In our e-mail protection solution, Gatefy's synthetic intelligence[65], with the assist of multinomial logistic regression, lets your agency categorize your emails into at least seven extraordinary classes. The result: extra protection and higher e-mail management. That is, your agency has extra visibility and management of the statistics and additionally reduces the threat of being breached or leaking data. In addition, personnel can be cognizant of the agency's middle business. The cool component is that Gatefy's synthetic intelligence device will research out of your agency's e-mail traffic. With every passing day, the answer turns into extra assertive and accurate, enhancing your very own overall performance and overall performance. As a person constantly tells me, generation is getting used for exact and to your favors.

Logistic regression types

- **Binomial logistic regression:** In the binomial logistic regression model, objects are classified into two groups or categories. It is almost a game between what is and what is not. For example, the message is spam or not, the image is colored or not, the cell is cancerous or not.

- **Multinomial logistic regression:** In the multinomial logistic regression model, objects are classified into three or more categories that have no order between them. Let's go to the examples. This animal is a cat, a lion or a tiger. This fruit is an apple, a pear, a mango or a passion fruit.

- **Ordinal logistic regression:** The ordinal logistic regression version is unique as it works with the idea of ordered categories. In this case, gadgets are categorized into three or extra lessons which have an already decided order. For example, the athlete's overall performance is poor, truthful or excellent. Another example: the diploma of affected person pleasure with the remedy is dissatisfied, glad or very glad.

4.3.3 Naive Bayes

For machine learning, naive Bayes features are a special type of classification algorithm. Bayes Theorem-based statistical classification underlies them. The term "naive" is used to describe Bayesian models that are based on simple assumptions[66]. There's a presumption that the predictor variables are independent. So, the presence of one item in the dataset has nothing to do with the presence of another. Despite their simplicity, they provide an easy way to create accurate models. By providing a way to calculate the "backward" probability of a specific event, they accomplish this goal considering a previous event, it is more likely that a new event will occur.

$$P(A|B) = P(B|A) \times P(A)/P(B) \qquad (4.2)$$

Eq. (**4.2**) formula for the Bayes theorem

Where A and B represent events, P the probability, and P(B)

- Q (A | B) is the conditional probability: Defines the probability of an event A occurring when B is true.

- P (B | A) defines the same as the previous point, but in reverse.

- P(A) and P(B) indicate the probability of A and B independently of each other.

These concepts applied to the problem you are trying to solve would represent variable B with the different characteristics that are used in the selected data set and variable A would contain the label that dictates whether B's set of classes is phishing. Furthermore, this theorem assumes that the characteristics are statistically independent, meaning that it expects the input variables to be unrelated[64]. This applies to the domain to which it applies, because in this job the features that are used have no direct relationship to each other to determine whether a URL is fraudulent or not.

The Bayes technique to a category is primarily based totally on a theorem that states that if the distribution densities of every of the training are known, then the desired set of rules may be written out in an express analytical form. Moreover, this set of rules is optimal, that is, it has a minimum opportunity of errors. In practice, the distribution densities of the classes are usually not known. They have to be estimated (restored) from the training sample. As a result, the Bayesian algorithm ceases to be optimal, since the density can be reconstructed from a sample only with a certain margin of error. The shorter the sample, the higher the chances of fitting the distribution to specific data and facing the effect of overfitting. The Bayesian approach to classification is one of the oldest but still retains a strong position in recognition theory. It underlies many fairly successful classification algorithms.

The naive Bayesian classifier (naïve Bayes) is based on the same formula and the additional assumption that objects are described by independent features. The assumption of independence greatly simplifies the problem, since it is much easier to estimate one-dimensional densities than one-dimensional density. Unfortunately, it is extremely rarely performed in practice, hence the name of the method[42]. A naive Bayesian classifier can be either parametric or nonparametric, depending on which method is used to reconstruct one-dimensional densities.

Within Naïve Bayes three types of classifiers are distinguished:

- **Multinominal Naive Bayes:** This type of classifier is usually used in the classification of documents, for example, to determine that a document belongs to a certain category such as economy, health, politics, etc. Here you would use the frequency of the features of the dataset.

- **Bernoulli Naive Bayes:** This type is like the previous type, with the difference that the resulting variable is of Boolean type.

- **Gaussian Naive Bayes:** In this type of the resulting variable is a variable with continuous and non-discrete values. By the name of the type, here the values are expected to be obtained from a Gaussian distribution.

4.3.4 Decision Tree

The decision tree is another algorithm widely used in supervised machine learning to solve regression and classification problems. The basis of this algorithm is simple because it is based on a tree-shaped structure where a root node (the sample of the data) is chosen and from it[67], the other attributes are used to break down the tree into branches that indicate a condition that may be true or false. This forks each node until it reaches the end nodes that do not indicate the solution to the problem we are posing, in the case of this project whether the URL is fraudulent or not. This algorithm is responsible for performing all possible tree combinations according to the input attributes, to dictate which is the best tree for the problem you want to solve[42].

To determine how the tree branches into sunbeds and make predictions it uses various functions, among which are the following among the most popular:

- **Gini Index:** Calculates a value based on the target variable and sample elements of the dataset, to determine which sample gets the most homogeneity and perform the subdivision of nodes according to this criterion.

- **Information Gain:** Uses categorical attributes to try to estimate the information each attribute provides based on entropy (information uncertainty).

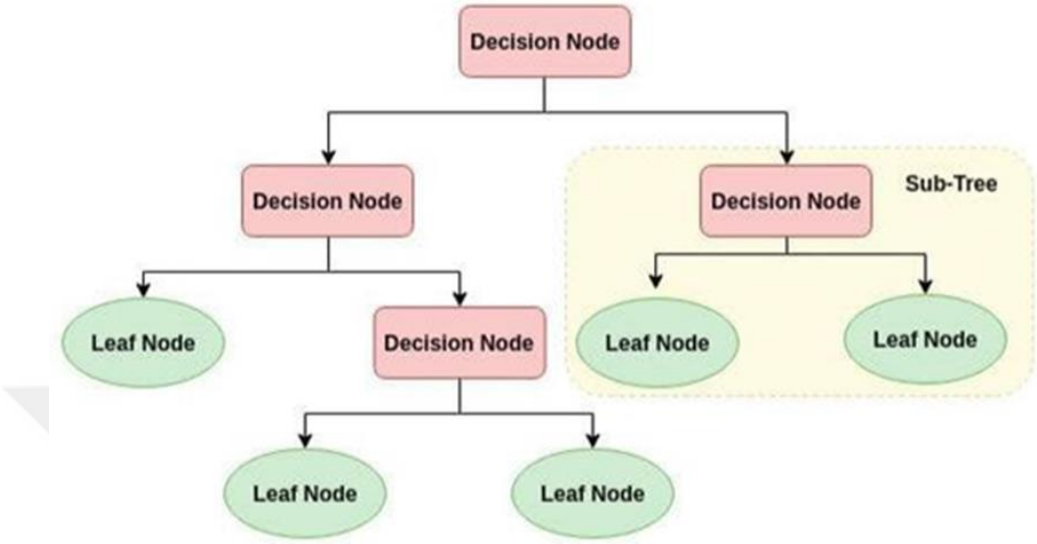

Figure 4.8: Decision Tree

The main disadvantages of this technique are that small variations in the dataset can produce completely different decision trees. Trees may be created that are too complex and not optimal for the problem you are trying to solve. If the dataset that the model is trained with is not balanced, it can cause skewed decision trees to be created if there are some predominant clas

4.3.5 K Nearest Neighbors

within the classification algorithms tested in this project is K nearest neighbors, a supervised machine learning algorithm like the previous ones. This algorithm is based on classifying values using more similar data points by proximity, learned at the training stage of the model[68]. The K of the name means the number of neighboring points that we consider classifying in phishing or non-phishing in the domain of work. Unlike logistic regression or decision trees, this algorithm does not explicitly learn a model but memorizes instances in the training phase as a knowledge base for prediction.

KNN works as follows, first calculates the distance between the item to be sorted and the other items in the dataset used in the training process. The second selects the nearest "K" elements (with less distance using the function you use). Finally, it carries out a majority voting process between the K points and the label that dominates will be the final decision. To measure the closeness between points, the Euclidean distance or the Cosine Similarity that measures the angle between the vectors is often used.

This algorithm highlights its ease of use, its calculation time is fast, [63]and no assumptions have to be made about the data set available. As a counter you have that the accuracy of the classification depends on the quality of the data in the training, find an optimal value of "k" (number of nearest neighbors) and can perform a poor classification of the data points at a limit where they cannot be classified in one way or another.

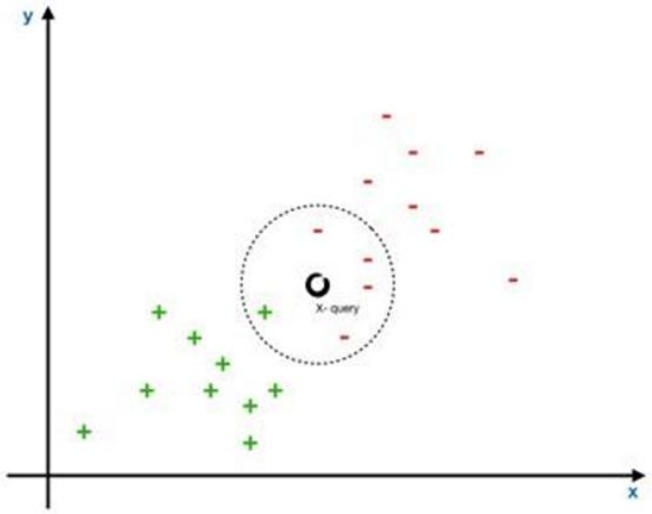

Figure 4.9: K Nearest Neighbors

4.3.6 Random Forest

This technique is based on the above as it consists of using a lot of decision trees that work together. Each tree within the random forest is in charge of predicting a class, and the class with the most votes within the set of random trees becomes the model prediction[45]. Trees are not correlated with each other, so if a tree generates a bad decision tree, the sum of the results of the other trees can dictate a good result. The main difference with the decision tree is that in a

random forest the process of finding the root node(root) and splitting the feature nodes is done randomly. The following figure depicts how random forest chooses model prediction.

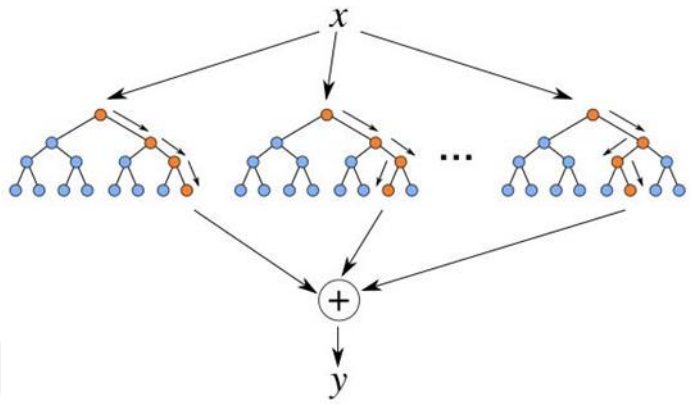

Figure 4.10: Random Forest

To ensure that each tree individually is not closely correlated with the behavior of any of the other trees in the model, the random forest uses a couple of methods to do so.

- Bootstrap Aggregation: A sample size training set is introduced to each tree but replacing sample elements.
- Feature Randomness: Each tree can only choose a random subset of features. This forces greater variation between model trees and less correlation between them with greater diversification.

The main disadvantage of this technique is the complexity of developing this model since the construction of random trees consumes much more time and computational resources than if only the decision tree technique were used.

4.4 CHARACTERISTICS OF THE DATASET

For the analysis of the data and the application of machine learning algorithms for the elaboration of the models, and already sanitized data set is used to avoid having to search multiple resources for phishing addresses and standardize the data, this allows to focus efforts on analyzing the characteristics and algorithms that best suit the type of problem you are trying to solve, this data was collected from available data sources such as Kaggle and GitHub. Below is a table with the characteristics they use and a brief description of what they provide:

Table 4.1: Dataset

NO	Feature	Description
1	Domain	Indicates the URL.
2	Ranking	A metric that measures the importance of the web. Measure the number and the quality of hyperlinks on the page.
3	isIp	Checks whether the URL contains an IP address.
4	valid	This information is obtained from Google's Whois API to know the status of the URL record.
5	Active Duration	It gets the same as the previous feature and indicates the time elapsed since the domain was registered.
6	URL Len	Indicates the number of characters in the URL.
7	is@	Checks to see if the URL includes a sign.
8	Is redirect	Checks whether the URL performs page redirects.
9	Have Dash	Checks if the URL includes "-".
10	Domain Len	Indicates the number of characters in the domain name.
11	No Of Subdomain	The number of subdomains that the URL includes.

12	Label	Is the tag that indicates whether it is a fraudulent URL or legitimate. 0 indicates legitimate and 1 indicates Phishing.

4.5 TOOLS AND PROGRAMMING LANGUAGE

For the elaboration of this project, the tools that are most oriented to solve this type of task and that provide the most ease of implementation have been selected.

Python (version 3): This programming language has been chosen because it provides many libraries dedicated to machine learning and its ease of learning and use.

Jupyter Notebook: It is a web application that allows you to work with the Python language and has been chosen since it allows you to write, document and execute code, visualize data, perform calculations, and see results all from the same interface of the application. This allows the creation of predictive models and pre-dataset analysis and subsequent analysis of metrics and results. It is widely used by the community for machine learning which is why it has been selected.

Scikit-learn: It is a library for machine learning for the language that python uses. Includes classification algorithms or support vector machines or random forests.

Numpy/Pandas: Both are extensions to the Python language and allow data manipulation and analysis. They are useful when dealing with the dataset for model training.

Seaborn: Like the previous ones, Seaborn is a library for Python dedicated to generating graphs from data contained in lists or vectors. This library facilitates the graphical representation of the metrics obtained in the prediction of the models and is used to generate the heat map from the confusion matrix.

Flask: It is a framework written in Python that allows you to create web applications quickly and with few lines of code. It is used to create the microservices part and allows users to make requests to the webserver and perform URL queries. This framework has been chosen for its ease of use and how light it is for what it is intended to use.

Visual Studio Code: This IDE (integrated development environment) has been chosen together with Jupyter Notebook to develop the API (application programming interface) provided to users. This tool makes use of a flask to create the microservice that offers the ability to perform URL queries.

4.6 IMPLEMENTATION AND TESTING

In the elaboration of the models and the obtaining of metrics has been used mainly the sklearn library, it allows to easily import the different classes of the different models that it is intended to use. As the elaboration of the different models is quite similar, we have proceeded to document in depth the process in the logistic regression algorithm carried out using Jupyter Notebooks. However, each model includes the parameters that are accepted by the Sklearn class that implements it and its explanation and the utility it provides. Different models accept the same parameters, therefore the redundant explanation of those same parameters that have been discussed above in other models has been omitted. It will only be taken into more detail if in the model in question it is necessary to modify this parameter so that it influences the generation of the model that is created.in this system we implement the models predictive are (Logistic regression, nave bayes, SVM, tree of decision, random forest and KNN.)

4.6.1 Confusion Matrix

A confusion matrix is an instrument used to monitor the effectiveness of a supervised learning algorithm in the field of artificial intelligence and machine learning. Each matrix column indicates the number of predictions for each class and each row represents the occurrences in the real class, that is, in practice, we can observe what kind of successes and errors our model has when it comes to data learning.

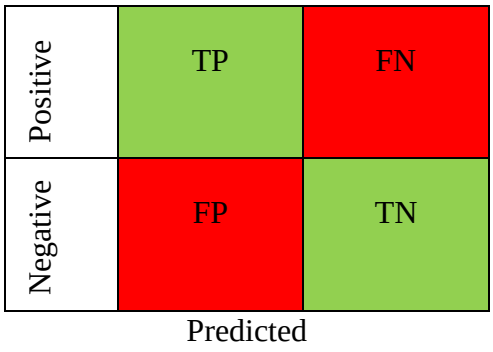

Figure 4.11: Confusion Matrix

Accuracy rate: is calculated as the number of all correct predictions divided by the total number of the dataset. The best accuracy is 1.0, whereas the worst is 0.0. It can also be calculated by 1 – ERR.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP} \quad (4.3)$$

Eq. (4.3) Accuracy low

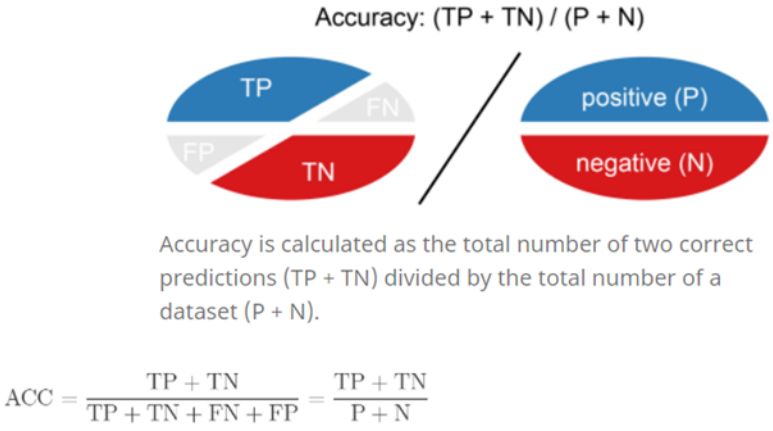

Figure 4.12: Accuracy

4.6.2 Modelling

In all models generated in Jupyter Notebooks, it is necessary at the beginning to import the necessary libraries to be able to generate the models. **pandas** are used to load the dataset, then **Sklearn** to be able to import the class that will allow us to create the classification algorithms model and the class that allows us to divide the training and test set, **Pyplot** use to create a figure, finally **Seaborn** that allows us to visualize a heat map of the confusion matrix generated by the predictive model.

Then, the set of data that we use to train the model is loaded and in the variable X the data corresponding to the characteristics that will be used to train the model is stored, in this variable, the characteristics of the domain and the target label of the training are excluded. The reason for excluding the domain that represents the URL that you are trying to determine whether it is phishing or not is because it is a variable that does not represent any significant value at the time of modelling.

Once you have the dataset loaded into memory, the next step is to divide this dataset into the representative quantity for model training and the number of samples to verify this model, it is observed that the set has been divided into 75% training and 25% for testing, later in this document comparison is made for a division of the set different from the one shown.

Logistic Regression: the process that continues is the instantiation of the class that allows us to define a logistic regression model in this case. The class is **LogisticRegression** and accepts the following parameters:

- **Random state=0:** This allows you to choose random samples from the dataset. Not specified as it is used at the time of splitting the dataset.

When the model is loaded, we proceed to train and prediction by dividing the dataset.

When the model has been made and the test set is passed to it, the metrics obtained by the model are stored in the variable **y_ pred**. Utilizing this variable, we proceed to elaborate a matrix of confusion that will allow us to obtain the following values:

- True Positive (TP): correct positive prediction
- False Positive (FP): incorrect positive prediction
- True Negative (TN): correct negative prediction
- False Negative (FN): incorrect negative prediction

And the Seaborn library it is possible to visually generate the following heat map with the values corresponding to the prediction mode scoring with the test set and the confusion matrix. The values represented are those listed above from top to bottom and from left to right.

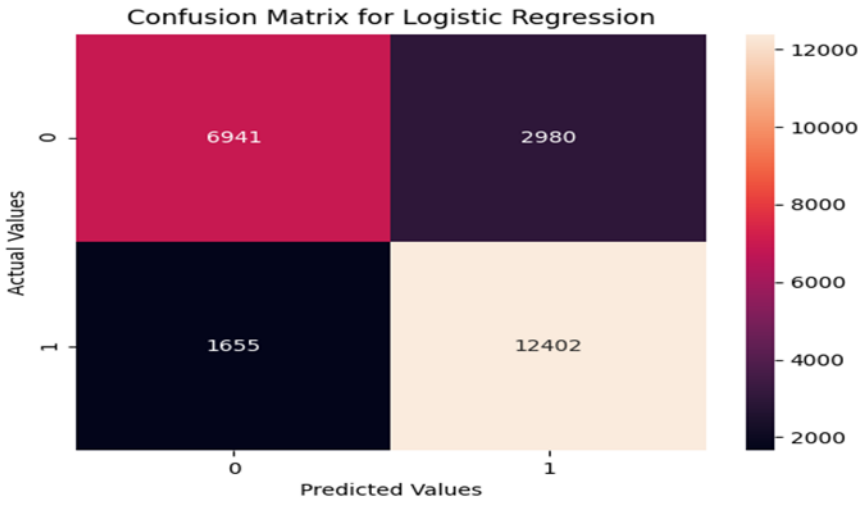

Figure 4.13: Logistic Regression heat map

Finally, the accuracy of the model is calculated using the score method that includes the class Obtaining an accuracy for this model of **80.66%.**

Naïve Bayes: In the elaboration of the model for Naïve Bayes a process like the previous one has been carried out; the difference is in the class that is instantiated that on this occasion is **GaussianNB**

Once the model has been trained and the prediction is made with the test data combination, we obtain the following values from the generated heat map:

- Number of true positive: 8548
- Number of false positives: 2829
- Number of true negatives: 11228
- Number of false negatives: 1373

Obtaining an accuracy for this model of **82.47%.**

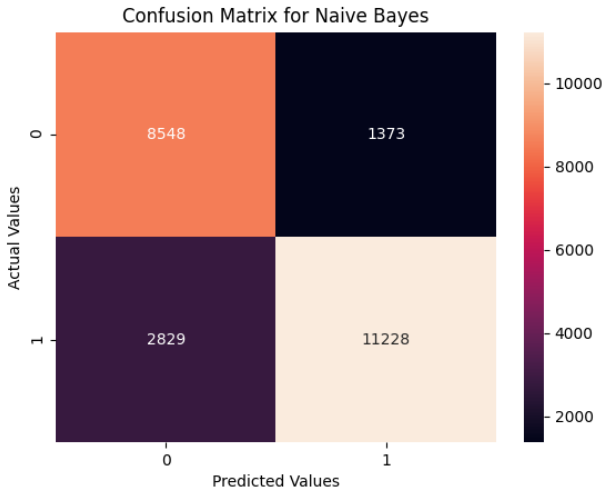

Figure 4.14: Naive Bayes heat map

SVM: For the generation of the SVM model has been instantiated the **LinearSVCclass** that allows you to configure the following are parameters (in general the parameters it includes are quite like the logistic regression model).

- **Loss**='squared hinge': Specifies the loss function. The two values it allows are 'hinge' and 'squared hinge' which do not influence the accuracy of the model.
- **Class weight**=None
- **Fit intercept**=True
- **Intercept_scaling**=1
- **Random_state**=None
- **Max_iter**=1000

The heat map is generated, and the following values are obtained:

- Number of true positive: 8716
- Number of false positives: 2330
- Number of true negatives: 11727
- Number of false negatives: 1205

Obtaining an accuracy for this model of **85.25%**

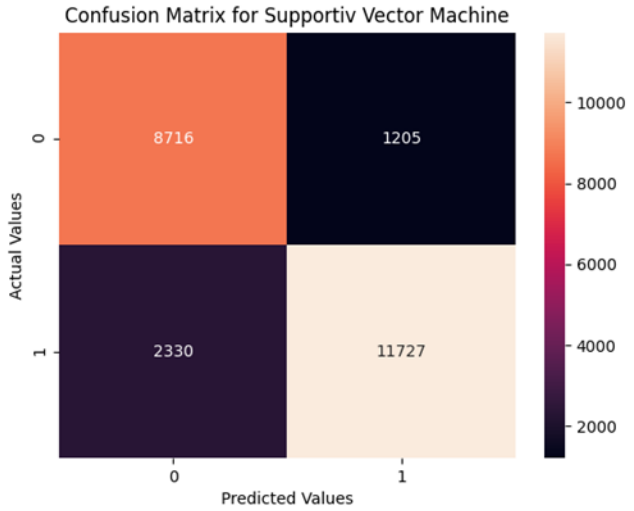

Figure 4.15: SVM heat map

Decision tree: In the elaboration of the decision tree model has made use of the **DecisionTreeClassifier** class that includes some arguments quite varied to those defined to the previous models since this model is based on the generation of a decision tree (as the name of the model indicates).

From the heat map generated we obtain the following values:

- Number of true positive: 9317
- Number of false positives: 657
- Number of true negatives: 13400
- Number of false negatives: 604

Obtaining an accuracy for this model of **94.74%.**

In the elaboration of this model, it was intended to show an image of the tree that is generated, since it generates a tree of a high dimension and due to the levels of depth that are created to reach the prediction, it is not possible to show an image of the tree. For this reason, the image it generates does not allow you to clearly see the different levels it contains. The test was performed to add a value to the argument Max_depth to generate a visual image of the tree, which compromised the model with an accuracy of around15% lower.

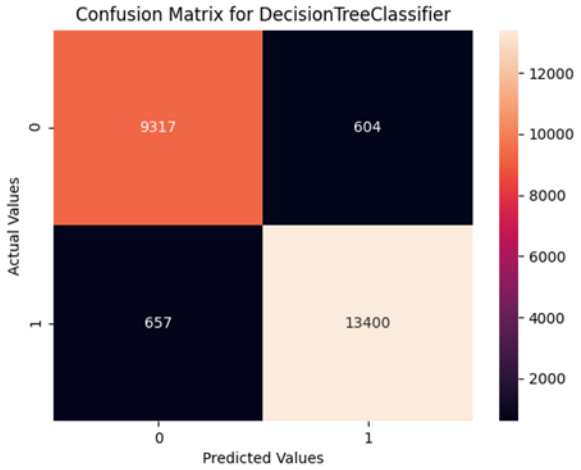

Figure 4.16: Decision Tree heat map

K Nearest neighbors: In the elaboration of this model even though the process followed is like the previously implemented models, it includes a slight difference in the process of obtaining the accuracy of the model. Since this model is based on the elaboration of points clusters on the plane and the distance between these points, the use of the class of this model allows defining a parameter called n_neighbors that allows defining the relationship between the points to determine a point to what class it belongs to. The class that has been instantiated for the elaboration of this model is **KNeighborsClassifier** () that includes the following parameters that differ slightly from those defined in the other models:

- **N_neighbors=x:** This parameter is important in this algorithm since a prediction is given based on this variable, which determines how many neighbors it is necessary to be around to specify that it belongs to one class or another. To check the effectiveness of the model, a prediction has been created for a range of integers.

- **Weights='distance':** Determines the function to be used in the prediction. In this case, distance is used, which determines that the nearest neighbors of a point will have greater influence than the distant ones, providing a somewhat higher precision.

From the heat map generated we obtain the following values:

- Number of true positive: 9369
- Number of false positives: 567
- Number of true negatives: 13490
- Number of false negatives: 552

They obtain an accuracy for this model of **95.33%** depending on the K chosen.

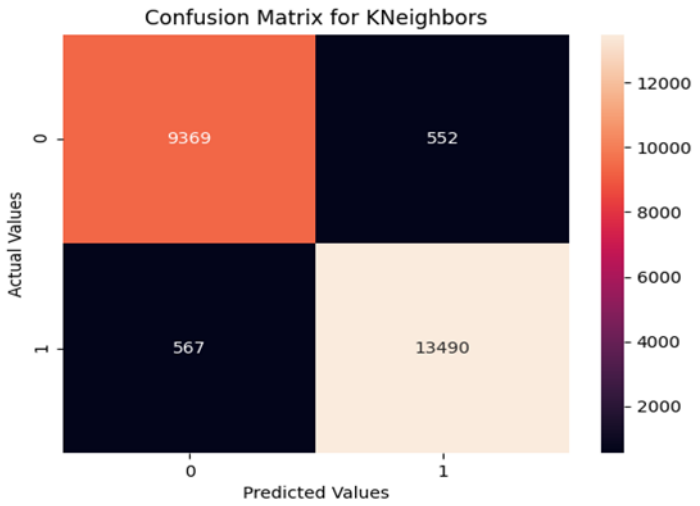

Figure 4.17: KNN heat map

Random forest: In the elaboration of the random forest model, the **RandomForestClassifier** class is used which allows parameters quite like those allowed by the tree decision model, this is

because this model is still an extension of the previous model when making the prediction based on a set of decision trees. The following arguments are found:

- **N_estimators=50**: Indicates the number of trees in the forest. This value has been chosen because it provides a result of precision quite similar and optimal to the number of trees by default that is 100.
- **Max_depth**=None
- **Min_samples_split**=2
- **Random_state**=None

In the same way as the previous cases, the following values are obtained from the generated heat map:

- Number of true positive: 9392
- Number of false positives: 507
- Number of true negatives: 13550
- Number of false negatives: 529

Obtaining an accuracy for this model of **95.67%.** Of all the models generated, this is the one that has produced the highest percentage of model accuracy.

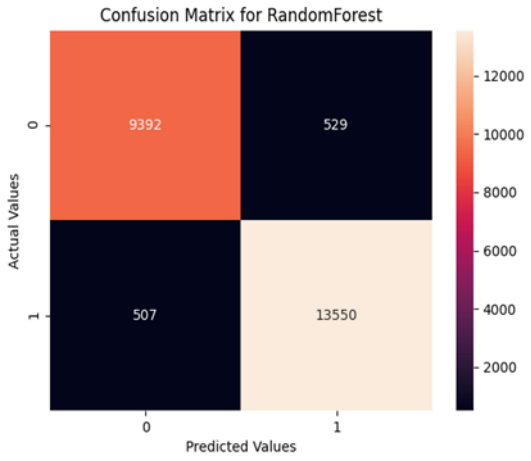

Figure 4.18: Random Forest heat map

4.7 RESULTS

As a summary and as a comparison of the results of the different models, the prediction of the different models has been made for a division of the data set of 90% of training and 10% of the test together with the one that had been previously elaborated of 75% - 25%. These values correspond to the following number of entries in the dataset that contains a total of 95910 samples it was divided by 75% to 25% and again divided by 90% to 10%.

Table 4.2: Results

Split percentage of the dataset	Algorithm classification	Matrix of confusion	Accuracy of the model (%)
Division 75% - 25% (Training and testing)	Logistic Regression	6941 2980 1655 12402	80.66
	Naïve Bayes	8548 1373 2829 11228	82.47
	SVM	8716 1205 2330 11727	85.25
	Decision tree	9317 604 657 13400	94.74
	KNN	9369 552 567 13490	95.33
	Random Forest	9392 529 507 13550	**95.67**
Division 90% - 10% (Training and testing)	Logistic Regression	2231 1739 822 4799	73.29

	Naïve Bayes	3433 537 1094 4527	82.99
	SVM	3431 539 1363 4258	80.16
	Decision tree	3775 195 237 5384	95.49
	KNN	3776 194 199 5422	95.90
	Random Forest	3784 186 184 5437	**96.14**

As can be seen in the table (and as mentioned above) the model that gives the highest accuracy in both divisions is Random Forest and its prediction is based on a set of trees. It is also possible to see a loss of accuracy in the logistic regression of approximately 7% in the division 90% - 10%, concerning the rest of the models there is an increase of around 1% to 4% that is not so significant to the loss of the logistic regression. It is possible to conclude that, in general, the predictions of the different models are high.

COMPARISON

Table 4.3: Comparison Results

Authors	Algorithm	Accuracy
(L. Machhado and J. Godge 2017)	Decision Tree	89.40 %
(R. Kiruthiga, and D. Akila 2018)	NB, SVM, RF, DT	93 %
(Shakan lliyas 2018)	KNN	93.60 %
(S. Parekh, D. Parikh, and P.S. Sankhe 2018)	Random Forest	95 %
(Ms. Sophiya shikalyar, Dr. S. D. Sawarkar 2019)	SVM, NB	86.3 %
(Ashutosh Tiwari, Prof. Vipul Dalal 2020)	(CNN+SVM+KNN)	92.68 %

5. CONCLUSION AND FUTURE

In this work, a solution has been defined that allows, without carrying out a manual analysis of the URL, to determine how likely it is to be legitimate or not. For this, because of this project, service has been implemented that allows sending a URL and returning a score according to the prediction of the automated learning models that have been developed. This approach is very similar to what the well-known VIRUSTOTAL website offers, which uses a list of antivirus engines allow to determine how likely it is that the file/address is malicious. This proposal has been chosen since apart from wanting to offer a solution for the detection of fraudulent URLs, it was believed convenient (based on other studies that have been analyzed) to be able to offer a solution that can be easily implemented in a real environment with the minor impact. Allowing to obtain which models have predicted that the website being sent is fraudulent and how likely it is to be true.

To conclude this project, an analysis of the planning that had been stipulated at the beginning of the work is carried out, then an evaluation of the objectives is carried out, which has been extracted from them and what result has been obtained. Finally, a future review is carried out, where some points are reviewed where it could be improved, innovated, and added so that this project can provide a solution that could be included in real environments where it is necessary to cover this problem so common today.

As future lines it is believed advisable to improve the existing system with some ideas that would be very interesting to try and in what way they would impact on the performance of the system and on the predictability of the service. While it is true that in the conclusions it has been transmitted that the objectives have been met and that the system implemented allows to predict whether a URL is legitimate or fraudulent, there are cases such as those that have been demonstrated above in which the response issued by the system does not present a good estimate and according to which vectors can violate the trained predictive models. One of the first proposals and that would be interesting to add to the system, is the possibility of having a database engine in which the URLs that have been consulted are stored and to be able to develop an intelligent system that can be improved based on the set of URLs that have been processed. A

non-relational database might fit into this type of problem because it allows for great performance over large volumes of data.

REFERENCES

[1] S. G. Selvaganapathy, M. Nivaashini, and H. P. Natarajan, "Deep belief network based detection and categorization of malicious URLs," *Inf. Secur. J.*, vol. 27, no. 3, pp. 145–161, 2018, doi: 10.1080/19393555.2018.1456577.

[2] Y. Ding, N. Luktarhan, K. Li, and W. Slamu, "A keyword-based combination approach for detecting phishing webpages," *Comput. Secur.*, vol. 84, pp. 256–275, 2019, doi: https://doi.org/10.1016/j.cose.2019.03.018.

[3] J. Saxe and K. Berlin, "eXpose: A character-level convolutional neural network with embeddings for detecting malicious URLs, file paths and registry keys," *arXiv Prepr. arXiv1702.08568*, 2017.

[4] K. Thomas and D. M. Nicol, "The Koobface botnet and the rise of social malware," in *2010 5th International Conference on Malicious and Unwanted Software*, 2010, pp. 63–70.

[5] F. Maggi *et al.*, "Two years of short URLs internet measurement: Security threats and countermeasures," *WWW 2013 - Proc. 22nd Int. Conf. World Wide Web*, pp. 861–871, 2013.

[6] L. Singh and J. Zhan, "Measuring topological anonymity in social networks," in *2007 IEEE International Conference on Granular Computing (GRC 2007)*, 2007, p. 770.

[7] D. Acemoglu, M. A. Dahleh, I. Lobel, and A. Ozdaglar, "Bayesian learning in social networks," *Rev. Econ. Stud.*, vol. 78, no. 4, pp. 1201–1236, 2011.

[8] H. Dong, J. Shang, D. Yu, and L. C. Lu, "Beyond the blacklists: Detecting malicious url through machine learning," *Proc. BlackHat Asia*, 2017.

[9] O. K. Sahingoz, I. Saide, and D. Bulut, "(37) (PDF) PHISHING DETECTION FROM URLS BY USING NEURAL NETWORKS | Computer Science & Information Technology (CS & IT) Computer Science Conference Proceedings (CSCP) - Academia.edu," pp. 41–54, 2018, [Online]. Available: https://www.academia.edu/38008213/PHISHING_DETECTION_FROM_URLS_BY_USING_NEURAL_NETWORKS?auto=download.

[10] R. Kiruthiga and D. Akila, "Phishing websites detection using machine learning," *Int. J. Recent Technol. Eng.*, vol. 8, no. 2 Special Issue 11, pp. 111–114, 2019, doi: 10.35940/ijrte.B1018.0982S1119.

[11] S. Shikalgar, S. . Sawarkar, and S. Narwane, "Detection and Classification of Phishing Attacks Using Machine Learning," vol. 8, no. 11, pp. 537–544, 2019, [Online]. Available: http://dspace.dbit.in/jspui/bitstream/123456789/5646/1/Final Blackbook.pdf.

[12] H. Choi, B. B. Zhu, and H. Lee, "Detecting malicious web links and identifying their attack types," *WebApps*, p. 11, 2011, [Online]. Available: http://dl.acm.org/citation.cfm?id=2002168.2002179.

[13] S. Sinha, M. Bailey, and F. Jahanian, “Shades of Grey: On the effectiveness of reputation-based blacklists,” *3rd Int. Conf. Malicious Unwanted Software, MALWARE 2008*, pp. 57–64, 2008, doi: 10.1109/MALWARE.2008.4690858.

[14] “https://safebrowsing.google.com/.” https://safebrowsing.google.com/.

[15] S. Shivdas, “Information Security Reading Room Protecting Home Devices from Malicious or Blacklisted,” 2021.

[16] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Identifying suspicious URLs: An application of large-scale online learning,” *ACM Int. Conf. Proceeding Ser.*, vol. 382, 2009, doi: 10.1145/1553374.1553462.

[17] D. Sahoo, C. Liu, and S. C. H. Hoi, “Malicious URL Detection using Machine Learning: A survey,” *arXiv*, pp. 1–20, 2017.

[18] D. K. McGrath and M. Gupta, “Behind Phishing: An Examination of Phisher Modi Operandi.,” *LEET*, vol. 8, p. 4, 2008.

[19] P. Kolari, T. Finin, and A. Joshi, “SVMs for the Blogosphere: Blog identification and splog detection,” *AAAI Spring Symp. - Tech. Rep.*, vol. SS-06-03, no. January, pp. 92–99, 2006.

[20] S. Garera, N. Provos, M. Chew, and A. D. Rubin, “A framework for detection and measurement of phishing attacks,” in *Proceedings of the 2007 ACM workshop on Recurring malcode*, 2007, pp. 1–8.

[21] A. Le, A. Markopoulou, and M. Faloutsos, “PhishDef: URL names say it all,” *Proc. - IEEE INFOCOM*, pp. 191–195, 2011, doi: 10.1109/INFCOM.2011.5934995.

[22] A. Y. Daeef, R. B. Ahmad, Y. Yacob, and M. N. Bin Mohd, “Lightweight phishing URLs detection using N-gram features,” *Int. J. Eng. Technol.*, vol. 8, no. 3, pp. 1563–1570, 2016.

[23] R. Verma and A. Das, “What’s in a url: Fast feature extraction and malicious url detection,” in *Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics*, 2017, pp. 55–63.

[24] R. Verma and K. Dyer, “On the character of phishing URLs: Accurate and robust statistical learning classifiers,” in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, 2015, pp. 111–122.

[25] H.-K. Pao, Y.-L. Chou, and Y.-J. Lee, “Malicious URL detection based on kolmogorov complexity estimation,” in *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, 2012, vol. 1, pp. 380–387.

[26] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Identifying suspicious URLs: an application of large-scale online learning,” in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 681–688.

[27] K. Althobaiti, G. Rummani, and K. Vaniea, “A review of human-and computer-facing url phishing features,” in *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, 2019, pp. 182–191.

[28] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Beyond blacklists: Learning to detect malicious web sites from suspicious URLs,” *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 1245–1253, 2009, doi: 10.1145/1557019.1557153.

[29] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Beyond blacklists: learning to detect malicious web sites from suspicious URLs,” in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 1245–1254.

[30] W. Nurulsafawati, W. Manan, A. Ghani, and A. Ahmed, *Characterizing Current Features of Malicious Threats on Websites Characterizing Current Features of Malicious Threats on Websites*, no. February. Springer International Publishing, 2019.

[31] A. Y. Fu, W. Liu, and X. Deng, “Detecting phishing web pages with visual similarity assessment based on Earth Mover’s Distance (EMD),” *IEEE Trans. Dependable Secur. Comput.*, vol. 3, no. 4, pp. 301–311, 2006, doi: 10.1109/TDSC.2006.50.

[32] K. T. Chen, J. Y. Chen, C. R. Huang, and C. S. Chen, “Fighting phishing with discriminative keypoint features,” *IEEE Internet Comput.*, vol. 13, no. 3, pp. 56–63, 2009, doi: 10.1109/MIC.2009.59.

[33] S. Afroz and R. Greenstadt, “PhishZoo: Detecting phishing websites by looking at them,” *Proc. - 5th IEEE Int. Conf. Semant. Comput. ICSC 2011*, pp. 368–375, 2011, doi: 10.1109/ICSC.2011.52.

[34] M. Dunlop, S. Groat, and D. Shelly, “GoldPhish: Using images for content-based phishing analysis,” *5th Int. Conf. Internet Monit. Prot. ICIMP 2010*, pp. 123–128, 2010, doi: 10.1109/ICIMP.2010.24.

[35] H. Zhang, G. Liu, T. W. S. Chow, and W. Liu, “Textual and visual content-based anti-phishing: a Bayesian approach,” *IEEE Trans. neural networks*, vol. 22, no. 10, pp. 1532–1546, 2011.

[36] Y.-T. Hou, Y. Chang, T. Chen, C.-S. Laih, and C.-M. Chen, “Malicious web content detection by machine learning,” *Expert Syst. Appl.*, vol. 37, no. 1, pp. 55–60, 2010.

[37] G. Xiang and J. I. Hong, “A hybrid phish detection approach by identity discovery and keywords retrieval,” *WWW’09 - Proc. 18th Int. World Wide Web Conf.*, pp. 571–580, 2009, doi: 10.1145/1526709.1526786.

[38] K. Soska and N. Christin, “Automatically detecting vulnerable websites before they turn malicious,” in *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, 2014, pp. 625–640.

[39] N. Gupta, A. Aggarwal, and P. Kumaraguru, “Bit.ly/malicious: Deep dive into short URL based e-crime detection,” *eCrime Res. Summit, eCrime*, vol. 2014-Janua, pp. 14–24, 2014, doi: 10.1109/ECRIME.2014.6963161.

[40] S. Lee and J. Kim, “Warningbird: A near real-time detection system for suspicious urls in twitter stream,” *IEEE Trans. dependable Secur. Comput.*, vol. 10, no. 3, pp. 183–195, 2013.

[41] D. Wang, S. B. Navathe, L. Liu, D. Irani, A. Tamersoy, and C. Pu, “Click traffic analysis of short URL spam on Twitter,” *Proc. 9th IEEE Int. Conf. Collab. Comput. Networking, Appl. Work. Collab. 2013*, pp. 250–259, 2013, doi: 10.4108/icst.collaboratecom.2013.254084.

[42] J. Cao, Q. Li, Y. Ji, Y. He, and D. Guo, “Detection of forwarding-based malicious URLs in online social networks,” *Int. J. Parallel Program.*, vol. 44, no. 1, pp. 163–180, 2016.

[43] C. Cao and J. Caverlee, “Detecting spam URLs in social media via behavioral analysis,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9022, pp. 703–714, 2015, doi: 10.1007/978-3-319-16354-3_77.

[44] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song, “Design and evaluation of a real-time URL spam filtering service,” *Proc. - IEEE Symp. Secur. Priv.*, pp. 447–462, 2011, doi: 10.1109/SP.2011.25.

[45] B. Eshete, A. Villafiorita, and K. Weldemariam, “Binspect: Holistic analysis and detection of malicious web pages,” in *International conference on security and privacy in communication systems*, 2012, pp. 113–115.

[46] H. Kwon, M. B. Baig, and L. Akoglu, “A Domain-Agnostic approach to Spam-URL detection via redirects,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2017, vol. 10235 LNAI, pp. 220–232, doi: 10.1007/978-3-319-57529-2_18.

[47] Y. Ding, N. Luktarhan, K. Li, and W. Slamu, “A keyword-based combination approach for detecting phishing webpages,” *Comput. Secur.*, vol. 84, pp. 256–275, 2019.

[48] T. Nash, “An Undirected Attack Against Critical Infrastructure,” vol. 1.2, no. September, pp. 1–10, 2005, [Online]. Available: http://ics-cert.us-cert.gov/pdf/undirected_attack0905.pdf.

[49] R. K. Shahzad and N. Lavesson, *Detecting scareware by mining variable length instruction sequences*. IEEE, 2011.

[50] P. K. Kerr, J. Rollins, and C. A. Theohary, *The stuxnet computer worm: Harbinger of an emerging warfare capability*. Congressional Research Service Washington, DC, 2010.

[51] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou, “Designing and Implementing Malicious Hardware.,” *Leet*, vol. 8, pp. 1–8, 2008.

[52] S. Amershi *et al.*, “Software engineering for machine learning: A case study,” in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2019, pp. 291–300.

[53] J. R. Koza, F. H. B. Iii, and D. Andre, “Artificial Intelligence in Design ’96,” *Artif. Intell.*

Des. '96, no. November 2012, 1996, doi: 10.1007/978-94-009-0279-4.

[54] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin, "Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14413–14423, 2020, doi: 10.1109/TVT.2020.3034800.

[55] C. M. Bishop, *Pattern Recognition and Machine Learning*. 2006.

[56] B. Eshete, A. Villafiorita, and K. Weldemariam, "BINSPECT: Holistic analysis and detection of malicious web pages," *Lect. Notes Inst. Comput. Sci. Soc. Telecommun. Eng.*, vol. 106 LNICS, pp. 149–166, 2013, doi: 10.1007/978-3-642-36883-7_10.

[57] Bharadwaj, K. B. Prakash, and G. R. Kanagachidambaresan, *Pattern Recognition and Machine Learning*. 2021.

[58] P. N. S Russell, *Artificial Intelligence: A Modern Approach, 3rd edition*, 3rd editio. Pearson, 2015.

[59] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[60] R. S. A. G. B. Sutton, *Reinforcement Learning: An Introduction, 2nd edition*. 2018.

[61] Y. Alshboul, R. Nepali, and Y. Wang, "Detecting malicious short URLs on Twitter," 2015.

[62] S. N. Bannur, L. K. Saul, and S. Savage, "Judging a site by its content: learning the textual, structural, and visual features of malicious web pages," in *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, 2011, pp. 1–10.

[63] H. Choi, B. B. Zhu, and H. Lee, "Detecting Malicious Web Links and Identifying Their Attack Types.," *WebApps*, vol. 11, no. 11, p. 218, 2011.

[64] D. Canali, M. Cova, G. Vigna, and C. Kruegel, "Prophiler: a fast filter for the large-scale detection of malicious web pages," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 197–206.

[65] M. M. Malik and J. Pfeffer, "A macroscopic analysis of news content in Twitter," *Digit. Journal.*, vol. 4, no. 8, pp. 955–979, 2016.

[66] A. Aggarwal, A. Rajadesingan, and P. Kumaraguru, "PhishAri: Automatic realtime phishing detection on twitter," in *2012 eCrime Researchers Summit*, 2012, pp. 1–12.

[67] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis.," in *Ndss*, 2011, pp. 1–17.

[68] A. Astorino, A. Chiarello, M. Gaudioso, and A. Piccolo, "Malicious url detection via spherical classification," *Neural Comput. Appl.*, vol. 28, no. 1, pp. 699–705, 2017.

APPENDIX A

[Python code]

```
#Pandas is a software library written for the Python programming language for data manipulation and analysis.

import pandas as pd

# Library for the Python programming language, adding support for large

from sklearn. model_selection import train_test_split

from sklearn. linear_model import LogisticRegression

#Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

import matplotlib. pyplot as plt

#Seaborn that allows us to visualize a heat map of the confusion matrix generated by the predictive model.

import seaborn as sn

# Read in the Demo CSV file

data = pd. read_csv ('url_dataset.csv')

# Variables that will not be part of the training are removed.

X = data. drop (['domain', 'label'], axis = 1)

# The prediction dependent variable is stored.

y = data. label

# The data set for training and testing is divided (25%).

X_train, X_test, y_train, y_test = train_test_split (X, y, test_size=0.25, random_state=0)

# The model to be used is loaded.

logistic_regression = LogisticRegression()

# Model training.

logistic_regression.fit (X_train, y_train)

# Test phase with the test set.

y_pred = logistic_regression. predict(X_test)
```

```
# Obtaining the confusion matrix to know the TP FP TN FN
confusion_matrix = pd. crosstab (y_test, y_pred, rownames=['Actual'], colnames=['Prediction'])
# The heat map is represented by the data in the confusion matrix.
sn. heatmap (confusion_matrix, annot=True, fmt="d")
print("logistic_regression\n")
# Obtaining the percentage of accuracy of the model elaborated.
print ('Accuracy: ‘, logistic_regression. Score (X_test, y_test) * 100)
plt.title('Confusion Matrix for Logistic Regression')
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()
print("\n")
from sklearn.neighbors import KNeighborsClassifier
print("KNeighbors\n")
knn = KNeighborsClassifier (n_neighbors=13, weights="distance")
knn.fit(X_train,y_train)
y_pred = knn.predict(X_test)
confusion_matrix = pd. crosstab (y_test, y_pred, rownames=['Actual'], colnames=['Prediction'])
sn. heatmap (confusion_matrix, annot=True, fmt="d")
print ('Accuracy: ', knn.score(X_test, y_test)  *  100)
plt.title('Confusion Matrix for KNeighbors')
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()
print("\n")
from sklearn.ensemble import RandomForestClassifier
print("RandomForest\n")
```

```
rfc = RandomForestClassifier (n_estimators=50, max_depth=None,min_samples_split=2, random_state=None)
 rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)
confusion_matrix = pd. crosstab (y_test, y_pred, rownames=['Actual'], colnames=['Prediction'])
sn. heatmap (confusion_matrix, annot=True, fmt="d")
print ('Accuracy: ', rfc.score(X_test, y_test)  *  100)
plt.title('Confusion Matrix for RandomForest')
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()
print("\n")
from sklearn.naive_bayes import GaussianNB
print ("Naive Bayes\n")
nav=GaussianNB ()
nav.fit(X_train,y_train) #train the model
y_pred = nav.predict(X_test)
confusion_matrix = pd. crosstab (y_test, y_pred, rownames=['Actual'], colnames=['Prediction'])
sn. heatmap (confusion_matrix, annot=True, fmt="d")
print ('Accuracy: ', nav.score(X_test, y_test)  *  100)
# Display the Confusion Matrix of Naive Bayes
plt.title('Confusion Matrix for Naive Bayes')
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()
print("\n")
from sklearn.svm import LinearSVC
from sklearn.pipeline import make_pipeline
```

```
from sklearn.preprocessing import StandardScaler
print ("Support Vector Machine\n")
svm= make_pipeline(StandardScaler(),LinearSVC(random_state=None, tol=500))
svm.fit(X_train,y_train)
y_pred = svm.predict(X_test)
confusion_matrix = pd. crosstab (y_test, y_pred, rownames=['Actual'], colnames=['Prediction'])
sn. heatmap (confusion_matrix, annot=True, fmt="d")
print ('Accuracy: ', svm.score(X_test, y_test)  *  100)
plt.title('Confusion Matrix for Support Vector Machine')
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()
print("\n")
from sklearn.tree import DecisionTreeClassifier
print("DecisionTreeClassifier\n")
dtc= DecisionTreeClassifier(random_state=None)
dtc.fit(X_train,y_train)
y_pred = dtc.predict(X_test)
confusion_matrix = pd. crosstab (y_test, y_pred, rownames=['Actual'], colnames=['Prediction'])
sn. heatmap (confusion_matrix, annot=True, fmt="d")
print ('Accuracy: ', dtc.score(X_test, y_test)  *  100)
plt.title('Confusion Matrix for DecisionTreeClassifier')
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.show()
print("\n")
```