REPUBLIC OF TURKEY

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

MACHINE LEARNING ALGORTHIMS FOR URLS CLASSIFICATION

Nagham Amjed ABDULZAHRA

Master's Thesis

Supervisor

Asst. Prof. Dr. Sefer KURNAZ

Istanbul, 2022

MACHINE LEARNING ALGORTHIMS FOR URLS CLASSIFICATION

Nagham Amjed ABDULZAHRA

Electrical and Computer Engineering

Master's Thesis

ALTINBAŞ UNIVERSITY

2022

The thesis titled MACHINE LEARNING ALGORTHIMS FOR URLS CLASSIFICATION prepared by NAGHAM AMJED ABDULZAHRA and submitted on 18/05/2022 has been **accepted unanimouslys** for the degree of Master of Science in Electrical and Computer Engineering.

_____

Asst.Prof.Dr. Sefer KURNAZ

Supervisor

Thesis Defense Committee Members:

Asst. Prof. Sefer KURNAZ	Faculty of Engineering and Natural Sciences, Altınbaş University	_____
Asst. Prof. Dr. Abdullahi Abdu IBRAHIM	Faculty of Engineering and Natural Sciences, Altınbaş University	_____
Asst. Prof. Dr. Serdar KARGIN	Faculty of Engineering and Architecture, Beykent University	_____

I hereby declare that this thesis meets all format and submission requirements of a Master's thesis.

Submission date of the thesis to the Graduate Education Institute: ___/___/___

I hereby declare that all information data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Nagham Amjed ABDULZAHRA

Signature

DEDICATION

First I would like to Allah Almighty for the power of mind , health, strength , guidance knowledge and skills to complete this study .This thesis is wholeheartedly dedicated to my beloved father, who have been my source of inspiration, he tells me that" every success in your life will be best gift for me". To my mother, who have been supporting me with the kind and pure love.

PREFACE

I would like to thank my supervisor Asst. Prof. Dr. Sefer KURNAZ 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 degree.

ABSTRACT

MACHINE LEARNING ALGORTHIMS FOR URLS CLASSIFICATION

Abdulzahra, Nagham Amjed

M.Sc, Electrical and Computer Engineering , Altınbaş University

Supervisor: Asst. Prof. Dr. Sefer KURNAZ

Date: 05/2022

Pages: 50

Phishing is a technique used to collect sensitive data from a user (password or credit card information) for future misuse by posing as a trustworthy source. It often takes advantage of the user's gullibility in ways that the user will not detect at first look and, in the worst-case scenario, the attacker maintains the user's data without the user's awareness.

Typically, the URL is the first and simplest piece of information we know about a website. As a result, it is logical to design algorithms for distinguishing harmful from benign URLs. Additionally, accessing and downloading the website's material may be time-consuming and involves the danger of downloading potentially hazardous information.

Machine Learning techniques are used to train a model on a collection of URLs specified as a set of characteristics and then predict and categorize the URLs as benign or dangerous. This technology enables us to identify and avoid possibly dangerous URLs in the near future.

We concentrated on the challenge of detecting malicious URLs using machine learning approaches in our thesis.

Keywords: URL, Machine Learning Techniques, Web Security , Malicious Content.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

ML	:	Machine Learning
NLP	:	Natural Language Processing
AI	:	Artificial Intelligence
NB	:	Naïve Bays
SVM	:	Support Victor Machine
LSI	:	Latin Semantic Index
GOM	:	Goals of Matrices

1. INTRODUCTION

Phishing is a method of obtaining sensitive data (passwords or credit card numbers) from a user by impersonating a trustworthy institution. Most of the time, it takes advantage of the user's gullibility in a way that the user will not notice at first look, and in the worst-case scenario, the attacker keeps the user's data without their knowledge[1].

As the number of websites and users continues to grow, so does the creation and distribution of new phishing websites that employ sophisticated techniques to deceive consumers. Attackers are continually looking for new ways to fool users with something they haven't seen before, and as a result, people are readily duped. The data in this section is taken from [2].

The URL is typically the most basic and least expensive piece of information we know about a website[5]. As a result, developing algorithms to distinguish between malicious and benign URLs is an obvious choice. Furthermore, accessing and downloading the website's material might be time-consuming, as well as posing the danger of downloading potentially hazardous content.

Machine Learning approaches are using a list of URLs defined as a set of features and, based on them, train a prediction model to classify a URL as benign or malicious. This gives them the ability to detect also new potentially dangerous URLs [17].

As previously stated, this strategy generally consists of two steps: the first is the selection and proper representation of characteristics, and the second is the training of a prediction mechanism utilizing this representation. Depending on whether the algorithm knows or doesn't know if URLs are dangerous or benign, machine learning can be characterized as supervised, unsupervised, or semi-supervised.

We may divide the model into two groups based on how it accepts training data and learns from it: batch learning (learn over finite data) and online learning (accepting and subsequently learning over data flowing in streams).

In this thesis, we focused on the problem of the detection of malicious URLs with machine learning techniques. The goal is to determine if chosen algorithms can efficiently decide if the given URL on input is malicious or not. Based on their

accuracy can be later used their estimated probability of URL being malicious as a relevant precondition for its further analysis.

2. BACKGROUND AND LITERATURE REVIEW

This chapter gives defnition of malicious content that is found on websites. Furthermore it contains a description of methods on how to detect some types of these attacks and malicious content.My thesis is specialized to detect malicious content based on the content of the tested website and on processing URL on given input.

2.1 MALICIOUS CONTENT ON WEBSITES

2.1.1 Phishing

Phishing is a method of obtaining sensitive data (passwords or credit card numbers) from a user by impersonating a trustworthy institution. Most of the time, it takes advantage of the user's gullibility in a way that the user will not notice at first look, and in the worst-case scenario, the attacker keeps the user's data without their awareness.

Sending fake e-mails is one of the Arts phishing strategies. These emails contain links to malicious websites that resemble the user's bank's website, for example. A form for acquiring the user's information or a link to download harmful material might be found on the website.

There are four primary methods for disseminating phishing material to consumers. The first is the email-email method, in which all communication is done over email. The second is email-website, which occurs when a person receives an email with a link to a malicious website. Another method is when a user finds a link on an uninfected website that leads to an infected one. The last method is browser-website, which occurs when a user downloads a malicious browser extension that links them to phishing websites.

Phishing over the phone is another option. When a caller requests the user's personal information, it is a false call from the user's bank. An SMS can also be used to persuade the victim to provide their information via their phone.

Nowadays, as the number of websites and users grows, so does the production and dissemination of more phishing websites that employ sophisticated techniques to deceive the user. Attackers are continually looking for new ways to fool users with something they haven't seen before, and as a

result, people are readily duped. The data in this section is taken from [17].

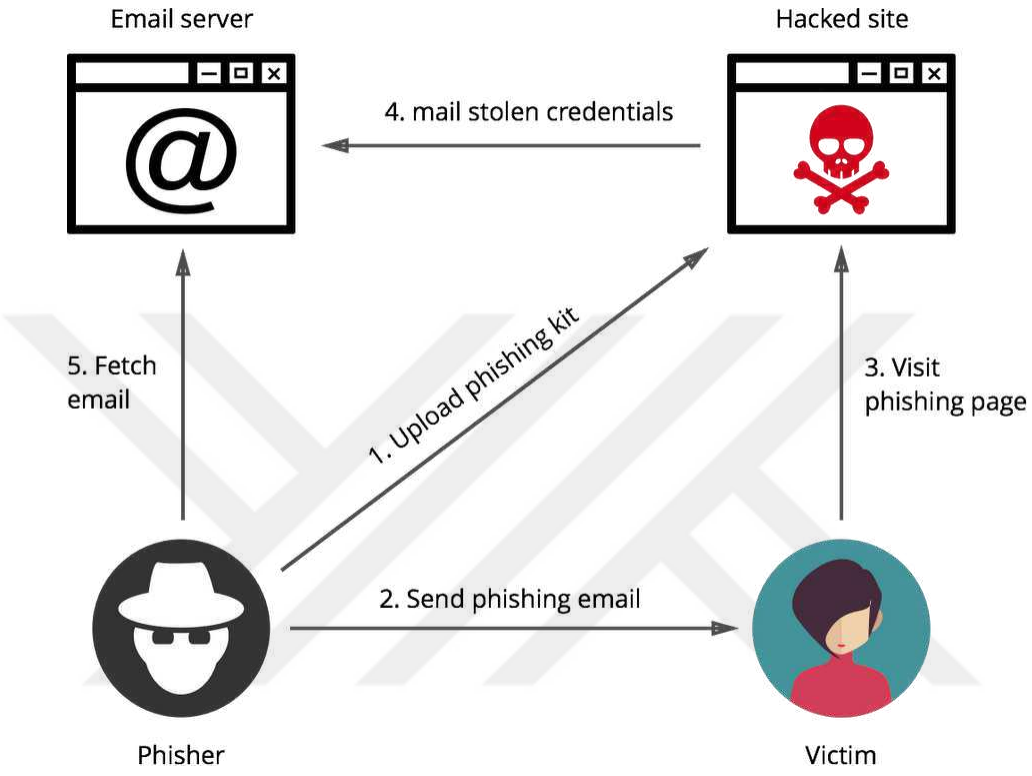

Figure 2.1: How phishing works [18].

Cross-site scripting: Cross-site scripting, sometimes known as XSS, is a technique for causing website disruption by exploiting untreated input flaws in the website's scripts. The attacker can leverage this security flaw to introduce custom Javascript code that can obtain sensitive information, cause website harm, or display different content to the user [19].

This method of phishing involves redirecting the visitor to another website after they visit the first one. Javascript, HTML meta tags, and plugins can all cause redirection. It is not necessary for the redirection to go to just one infected website; it might go to multiple different infected websites.

Link manipulation: The majority of phishing tactics require the use of a link, which may be found in emails or on a website. These connections appear to be legitimate websites, such as the user's bank. The link's name is modified such that the original version has mistakes, or it utilizes many

domains and subdomains that the original version does not. Its purpose is to perplex the victim, who is unaware that he or she is being routed to a malicious website.

Manipulation of links may be done directly in an HTML text or through Javascript as well. In this situation, the link's URL changes as soon as the page loads or as soon as you click on it.

Imitating trusted entity: When the attacker poses as a trustworthy entity with which the victim may create an account, this is the most advanced kind of phishing. This approach can take the shape of an email exchange between a user and an entity, or the victim may stumble into a website that is nearly identical to the original. The user enters personal information on the infected website, and then is routed to the original page without their knowledge. There are, however, more effective techniques to fool a user. For example, the victim might click on a link that opens an original website with a phony form for inputting confidential information about the victim in a new browser window; this is known as phishing.

2.2 OTHER MALICIOUS CONTENT

Adware, spyware, viruses, trojan horses, ransomware, rootkits, and phishing are all examples of malicious software, or malware. Malware is a type of malicious software that allows an attacker to get access to a victim's computer.

Different varieties of malware may be found on websites, typically in the form of download-able binary files that are difficult to identify without downloading them. However, some virus kinds, which are detailed below, can be found in the website's source code.

Spyware: A keylogger is one sort of malware that can be found on a website. It's a piece of software that scans the user's keyboard action and transmits it back to the attacker's server. This keyboard activity may be customized, and it only scans the input> tag with the parameter type set to password. An attacker may quickly obtain a victim's passwords or credit card details using this approach.

Ransomware: It is a "exorbitant" approach that encrypts all files on the victim's storage and requires the user to pay an untraceable Bitcoin ransom (since Bitcoin transactions are untraceable) in exchange for the storage password. Ransomware is propagated through the use of websites. When an attacker uses this sort of malware to infect a website server [20], the owner is obliged to pay for encryption.

2.3 DETECTION OF PHISHING

A website can get infected in a variety of ways. The identification of dangerous information is examined from the user's perspective in this thesis. This implies that the most important question is what happens to a user when he or she visits an infected website. We may accomplish the detection from the source code and the provided URL from this perspective. It is not identified if there is an infected le. It is also undetectable (from the perspective of an administrator) if the site is broken or malicious files have been posted. This is why the identification of trojan horses, viruses, and other sorts of malware is not included in this thesis.

2.3.1 Cross-site Scripting

There are three types of XSS by which an attacker can inject their Javascript code into the website. It is about inserting their own script into inputs on the site or into the URL.

DOM Based: A local type of XSS in which it is possible to move untreated URL variables into the code. To achieve this attack, it is necessary to manipulate this URL variable in the Javascript code as a document.write(). Then the harmful code can be written directly into the address, for example in the following URL:

```
http://website.com/page.html?variable=<script>doEvilCode()
    ;</script>
```

Figure 2.2: Example of DOM based XSS.

This command will be added by document.write() to DOM and exe- cuted on the website.

Recected: Also known as non-persistent XSS, it is almost the same as the DOM based type. The diference is that this problem can occur on unsecured dynamic websites. If a site is, for example, written in PHP and there is something which can print the variable, it prints the variable into the website code. Therefore, an attacker can also put some unsafe code into the URL.

Persistent: Because of a database vulnerability on the website, this attack permanently alters the webpage. This XSS is caused by a malicious Javascript code being inserted into a website's input element. It might, for example, be part of a form's text input. Because this sort of XSS cannot occur without human involvement, it is not recognized in the main implementation.

Detection of XSS: The script> element isn't the only place where you may put Javascript code. It's also possible that it'll show up in other tags with DOM event handler characteristics (onclick, onmouseover, onerror, etc.). This approach adds encoded URI (Uniform Resource Identifier) schemes to the src element in order to fool. It's also possible to utilize 64-bit encoding. Other methods for detecting XSS may be found here [21].

2.3.2 URL Redirects

Changing the window.location.href variable is the simplest technique to redirect in Javascript. By changing document.location.href, you may get the same result. The window.navigate() and location.assign() methods load and show the new page specified in the URL. There are various techniques to modify the URL in the browser's history so that the infected website can be accessed by returning to the browser. Many Javascript libraries utilize similar techniques but with different syntax. A meta element in HTML code that specifies the URL and time in seconds when the window will be redirected may also be used to redirect the window for example:

```
<meta http-equiv="refresh" content="0;url=http://evil.com/"
    />
<meta http-equiv="location" content="0;url=http://evil.com/"
     />
```

Figure 2.3: Meta redirects.

Iframes can also be used for a redirect. An *iframe*, there is a HTML tag which allows us to display a website within a website. Another way which is used for a redirect is an unpaired *base* tag which adds content from its src attribute.

URL Redirects can be done by plug-ins like Flash, Java, PDF, Quick- Time or XML in ways which are speciĄed in greater detail in [22].

2.3.3 Link Manipulation

We cannot guarantee that a link will lead to an infected website when detecting link manipulation, but we can detect some suspicious activity.

Because most genuine websites do not utilize it, links nowadays do not employ an IP address

instead of a domain name. As a result, if the link is an IP address, it is extremely suspect.

More than 5 dots in a link is likewise a red flag that should be investigated. It might be phishing if there are too many domains or two addresses in a URL.

The "@" sign can be used in the URL. In this case, the browser ignores anything to the left of the "@" symbol and redirects to fake-paypal.com. For example, paypal.com@fake-paypal.com will be forwarded to fake-paypal.com.

Links can also be encoded in hexadecimal format, which the web browser can decode but which the user cannot see. This is used to get around blacklists and other types of filters.

Blacklists (such as phishtank) that contain phishing domains can be used to identify the link of an infected website.

[23] is the source of the information in this section.

Imitating trusted entity

The most mimicked entities on the Internet are the most viewed web pages from which an attacker may profit (like paypal.com, ebay.com or facebook.com, etc.). On alexa.com, you may get a list of the most popular websites.

A comparison of the screenshot of the identified website to other screenshots of the most frequented web pages can detect this phishing approach. This approach of detecting a phishing website is quite accurate [24]. There are issues to be resolved, such as advertisements, animations, and time-dependent multimedia. This may be fixed by taking multiple screenshots at different moments after loading the website.

Another way for detecting the same website content is to compare all of the text from the identified website.

Detection of other malicious content

Other forms of malware, such as viruses and trojan horses, are difficult to detect since they take the shape of a link on a website, but they can be discovered by verifying patterns from reputable

sources or by finding the same malware source code. Machine learning is another way for detecting malware.

Keylogger: This sort of malware may be identified by looking for Javascript code that captures keyboard events and sends them to a server. The keyboard events include keydown, which begins with pressing a key, keypress, which begins with holding a key, and keyup, which begins with releasing a key. When the code additionally reads the input> elements and sends them to the server, this web page activity ceases to be innocent.

2.4 THE POPULARITY OF PHISHING ATTACKS

Phishing attacks have been around since the internet's inception. According to current data, phishers refine their techniques and construct more complex attacks as the Internet evolves. The data below, which represent phishing patterns from the views of two well-known organizations: the APWG and Google, back up this assertion.

2.4.1 The APWG

is a worldwide coalition of over 2200 members that unifies the global response against cybercrime across industries, governments, law enforcement, and non-governmental organizations. They also provide quarterly Phishing Activity Trend Reports, which look at phishing attacks reported by APWG members, among other things. Figure 2.3 depicts annual unique figures.

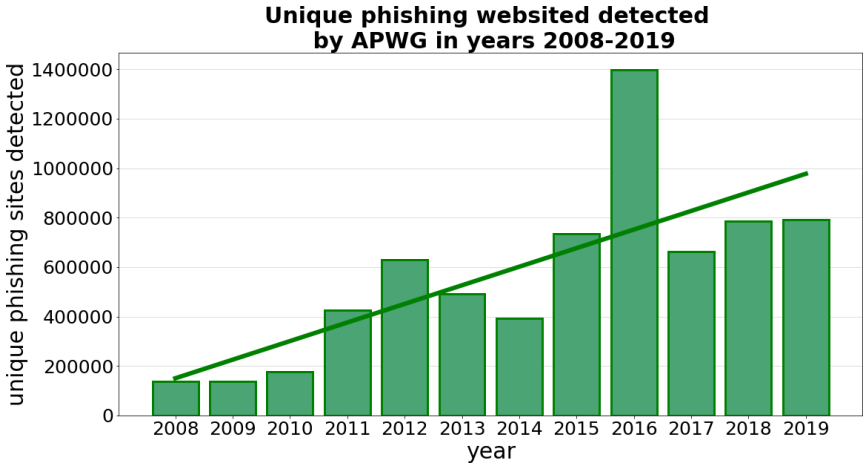

Figure 2.4: Yearly Unique Phishing Sites.

Even yet, it's worth noting that APWG has added a lot of new members in the last 12 years. A clear upward trend can be seen when the regression line is drawn over the numbers. "Continuing to strengthen its monitoring and reporting approach, as well as adding more data sources to its findings," the APWG said in its reports. [2] Another key point to keep in mind is that phishing attempts might involve hundreds of modified URLs all leading to the same target site.Google Safe Browsing Service

Google offers information from its Safe Browsing program in its transparency reports. Google's Safe Browsing program detects questionable websites and alerts users and webmasters to potentially hazardous information. The program can scan billions of online pages and alert users if they visit one of the sites that are considered dangerous. The weekly statistics of phishing and malware websites judged risky by Google's Safe Browsing service are shown in Figure 1.2.

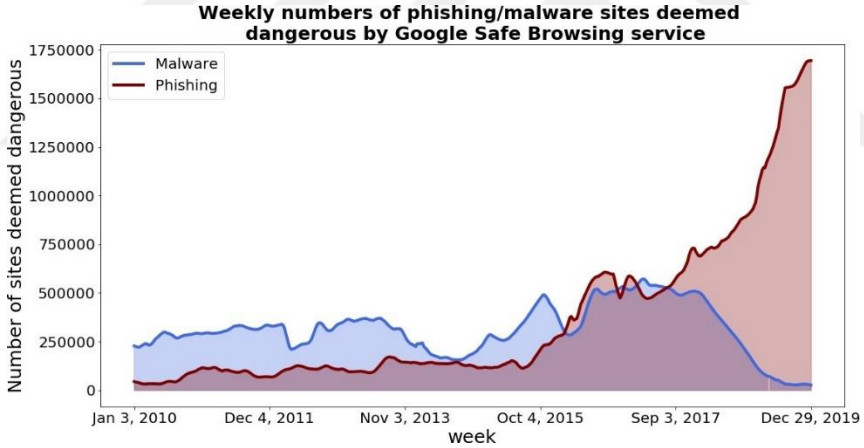

Figure 2.5: Weekly Unique Phishing/Malware in 2010 - 2019.

There are two theories (or a combination of both) for the behavior witnessed in the previous two years, as Claudio Guarnieri puts out in his article The Year of the Phish [3]. Either phishing detection has improved dramatically in recent years, or cyber criminals' conduct has shifted alarmingly, with phishing being a more effective source of unlawful financial/information gain than previously.

2.5 NUANCES OF PHISHING URLS

Regardless of whether of the strategies listed above is used by a phisher, link manipulation is at the heart of most phishing efforts. An attack method used by attackers to make a URL that a link

directs to appear trustworthy. This section will go through numerous methods used by attackers to hide the malicious nature of a URL by manipulating particular components of it.

URL stands for uniform resource locator, and it is used to refer to a specific resource on a network, such as the Internet. Recognizing a bad URL is a critical component of phishing detection since it serves as a "first-line" of defense before any further security procedures, such as page content assessment, are implemented.

The protocol, hostname, and path components of the URL will be discussed in this section. It's worth noting that the URL can also include optional components such as port, query, user info, and fragment. Their investigation, on the other hand, does not always yield helpful information for phishing detection.

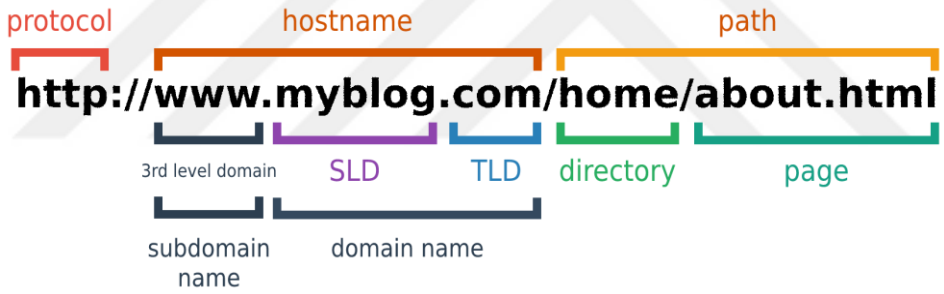

Figure 2.6: Basic URL Structure Diagram.

2.5.1 Protocol

The protocol specified in the scheme portion of the URL indicates how information should be shared between communication parties, such as a client and a server. Although most users are familiar with HTTP and HTTPS, FTP, mailto, file, and other protocols are also supported. This element of URL undergoes a significant alteration in the context of phishing.

Looking for a padlock icon next to the URL, which indicates the usage of a secure connection provided by the HTTPS protocol, is one of the most common safety tips for internet users. Until recently, this guideline was more or less adequate to avoid being phished. The owner or operator of a website must get an SSL/TLS certificate from a trustworthy authority in order to utilize HTTPS. Attackers, on the other hand, appear to be taking advantage of a recent advancement in

the ease of access to certificates via a certifying authority such as Let's Encrypt3. They aim to deceive potential victims by employing "encrypted" communication while using a genuine certificate. The percentage of phishing websites protected by HTTPS encryption is tracked by PhishLabs4, one of The APWG's contributors. As a result of their efforts, The APWG's quarterly Phishing Activity Trends Report showed that TLS/SSL was employed in 74 percent of phishing assaults recorded in Q4 2019. The graph below depicts the rapid growth of HTTPS use in phishing websites. [7].

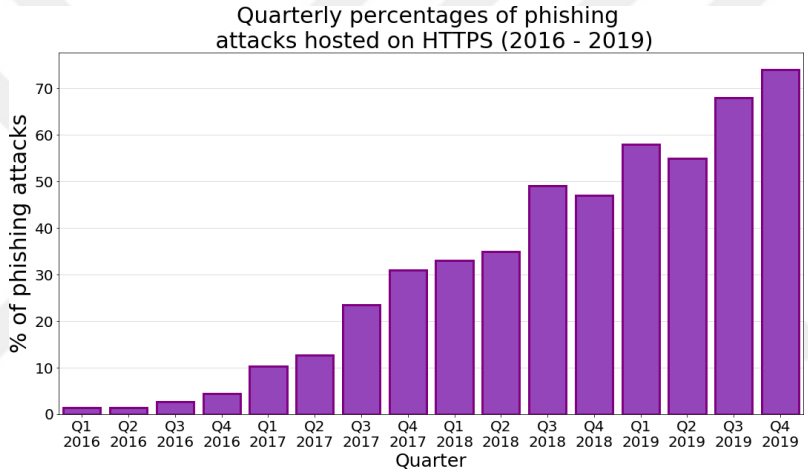

Figure 2.7: Quarterly Percentages of Phishing Sites Using HTTPS.

The conclusion is that although we might still consider the use of non-secure protocol as an indicator of a (maybe less sophisticated) phishing attempt, we surely cannot rely merely on a "padlock" for protection anymore.

2.5.2 Hostname

The hostname is supplied in the authority component of the URL. It comprises of a registered domain name, as well as extra subdomains or an IPv4 address written in the traditional, well-known dot format (IPv6 requires square brackets "[]") [8]. This component identifies a single machine in the network, such as a server. The Domain Name System (DNS) regulations determine the registered domain name (DNS).

The hostname region allows phishers to utilize their "ingenuity" to trick consumers into falling

into their trap. Cybersquatting and typo squatting are two of the most common approaches.

Cybersquatting (also known as domain squatting) is the unauthorized use of a domain name associated with a brand owned by another company in order to make a profit. As an example, an attacker may discover that a successful firm presents itself using the domain "best-company.com." He might attempt registering names such as "bestcompany.biz" or "bestcompany.co.uk." These domains may then be utilized to create a phishing website that masquerades as an official "Bestcompany" website. Typosquatting, also known as URL hijacking, follows a similar approach but takes advantage of user inputs that are incorrect. An attacker might use domain names that are similar to well-known websites or businesses. This sort of phishing website frequently has a user interface that seems identical to the genuine website.
As a result, a user who inputs "google.com" instead of "google.com" may be routed to a phishing page that appears exactly like the genuine thing, prompting him to hand up his login credentials to the phisher. So far, we've looked at strategies that take use of top-level and second-level domains (TLDs and SLDs). A Registrar, an authority that provides domain registration services, regulates these to some extent. The domain owner, on the other hand, has complete authority over subdomains and their names. This suggests that a phishing attack against a URL like "http://secure.paypal. anydomainname.com/payments" is simple to set up. As a result, the victims might not realize that they are visiting the pages of anydomainname.com instead of official PayPal websites.

2.5.3 Path

The route component specifies a resource's location inside the scope of a particular authority, such as a host. It has a file system-like structure, with segments separated by the forwarding slash "/" symbol [8].

The complexity of the route, such the subdomain component of the URL, is up to the phisher. Complicated URLs, as previously discussed, can easily fool a user into not paying attention to their structure. As a result, phishers frequently utilize lengthy, irrelevant routes to inflate the total length of the URL. Techniques comparable to the subdomain manipulation mentioned in the previous paragraph are also available, albeit less popular.

3. MACHINE LEARNING ALGORITHMS

Machine learning [20] is the concept of a system learning, finding patterns, and making decisions without human involvement. It is a branch of artificial intelligence that may be classified into three groups based on the learning approach: supervised, unsupervised, and semi-supervised [21]. The presence of labels (goal values) for input data during learning distinguishes these types.
Several measures are used to assess the performance of machine learning models:

i. Confusion matrix.
ii. Accuracy.
iii. F1-measure.
iv. Precision.
v. Recall

These measures allow all trained models to be compared to one another. As long as the problem is classified as a classification problem, four basic metrics, True vs. False and Positive vs. Negative output from the trained model, must be introduced initially. As an example, consider a binary issue with positive (1) and negative (0) classes. A true positive (TP) model output is one that properly predicts the positive class, whereas a true negative (TN) output is one that correctly predicts the negative class. A false positive (FP) is a model output that predicts the positive class erroneously, whereas a false negative (FN) is a model output that predicts the negative class wrongly [22].
The confusion matrix is the most often used measure, and it requires at least one input.
The first used metric, confusion matrix, takes as input at least two arrays, the predicted values for the data, and actual (original) values. As an output, it shows how many of predicted values by model belongs to categories TP, TN, FP or FN [23]. An example of a confusion matrix with two classes is shown in figure 3.1:

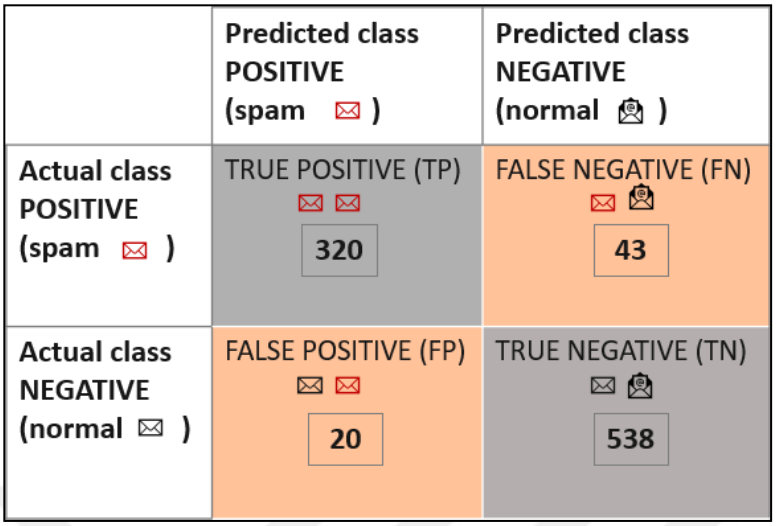

Figure 3.1: An example of the confusion matrix with two classes [24].

The accuracy measure, which shows the perfect match between anticipated and original data, is the second most commonly utilized statistic. The accuracy may be described as : if the predicted value of the i-th sample is yi, the matching original value is yi, and the number of samples is n.Accuracy Accurse $(y + y\hat{})^n = \sum_{i=0}^{n-1} 1(\hat{y}i = yi)$

Precision, recall, and F-measure are some of the other measures that are employed. Precision refers to the model's ability to properly categorize only positive classifications. The recall measures the model's ability to correctly categorize all positive classifications. The F-measure is the harmonic mean of the accuracy and recall. Furthermore, if = 1, the value is referred to as the F1-measure, which indicates that both recall and accuracy are equally essential [24]. Equations demonstrate how to compute these values in binary classification.

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (3.1)$$

$$\text{Recall} = \frac{TP}{TP+FN} \qquad (3.2)$$

In a multiclassification job, precision, recall, and F-measure can be applied to each class separately. The multiclass and multilabel classification part of reference [24] explains the complete equations with the appropriate notation.

3.1 SUPERVISED MACHINE LEARNING METHODS

learning from exemplars, also known as supervised machine learning, gives learning on a training data set with accurate replies (targets). the method generalizes such that it may reply appropriately to any input. as a result, all algorithms in the supervised category must first have accurate objectives (labels) for training data, which might be difficult to come by. we could store samples of every potential piece of input data into a large look-up database and eliminate the requirement for machine learning entirely. this isn't the situation in reality [25].

the thesis's purpose, however, is to compare the models' performance. the python package scikit-learn that was chosen provides a large range of algorithms, and it is impossible to discuss them all in the scope of this thesis. instead, the focus was on choosing at least the most important approaches from a different perspective, such as linear models and support vector machines.

3.1.1 Logistic Regression

A machine learning classification approach called logistic regression is used to estimate the likelihood of a categorical dependent variable. Linear regression may be transformed into logistic regression by using the sigmoid function [26]. The sigmoid function is represented by equation (.5), and the linear regression function with vector w = (.6) is represented by equation (.6). (w0, ..., wp). The coefficients of the model are represented by the vector w. The bias or intercept is represented by w0, while the weight of the features is represented by (w1,..., wp) (x1, ..., xp). The sigmoid function returns a number between 0 and 1. Equation (.7) may therefore be used to illustrate logistic regression, together with equation (.7) for mapping (.8). If the output of the sigmoid function is greater than 0.5, logistic regression will predict class 1. Class 0 will be expected otherwise [27].

Logistic regression is one of the best models for anomaly detection categorization. Logistic regression is a dependent variable-based approach. It is now one of the most popular and commonly utilized algorithms. Although the term regression appears in the name, this is a classification algorithm. The Python package scikit-learn implements one of the potential implementations under the class Logistic Regression [28]. When creating an instance, the class has a number of arguments to choose from. Optional regularization 1 is supported by the penalty parameter, which minimizes the cost function seen in equation (.9). Similarly, the equation (.10)

is solved as a minimization of the default regularization 12. Elastic-Net is the last regularization that is supported which is combination of 1_1 and 1_2.

The class LogisticRegression allows you to utilize alternative solvers through the solver parameter. Liblinear, newton-cg, lbfgs, sag, and saga are some of the ones that have been implemented. By setting multiclass to'multinomial,' the solvers may also be utilized for multi classification problems. The Broyden–Fletcher–Goldfarb–Shanno [29] optimization algorithm from the family of quasi-Newton techniques [30] with limited memory is known by the abbreviation lbfgs. Due to speed difficulties, it is only advised for small datasets.

In the actual world, data is seldom separable by linear feature combinations. It is feasible to convert features into polynomial combinations in order to improve model performance by better fitting the data. With the help of a concrete example of two features, feature translation into polynomial can be easily described.

$X = [x_1, x_2]$ and degree 2. The transformed features into polynomial are $X = [x_1, x_2, x_1x_2, x2, x2]$.

3.1.2 Support Vector Classifier

The supervised machine learning approach utilized for classification is the support vector classifier (SVC). It's one of the Support vector machines (SVMs) that the scikit-learn package has developed.

Vapnik created SVMs in 1992, and they quickly gained popularity due to their superior classification performance on datasets of reasonable size. They don't function well on really big datasets as long as they entail data matrix inversion, which is a computationally costly operation. The technique creates a hyper-plane, or a series of hyper-planes, that divides the dataset into classes. The hyper-plane is essentially a line that establishes a decision boundary in two dimensions and two classes. The margin is the distance between the closest point and the decision border.

SVMs are a class of algorithms that may be used for classification, regression, and outlier detection. The memory efficiency of these approaches is caused by a subset of training points in the decision function known as support vectors. Different kernel functions for the decision function

are supported by SVMs. It's also possible to utilize a custom kernel function.

The scikit-learn library's class SVC [31] is a concrete implementation of SVMs. The technique will choose a new decision boundary, for example, a straight line or a polynomial one of the features, using the kernel function, which is changeable by parameter with the name kernel of the class SVC. Provided kernel functions [32] by SVMs in scikit-learn:

i. Linear.

ii. Polynomial.

where degree of polynomial function is specified by parameter with the name degree of class SVC.

The gamma parameter of class SVC must be supplied for the Radial Basis Function. Gamma may be described as the kernel's spread (decision region). When the gamma is higher, the data points must be closer together in order to be influenced. Scikit-Sigmoid is a library that employs the sigmoid function as a kernel, commonly known as Hyperbolic Tangent Kernel [33].

The SVC approach, like the other supervised methods, uses an array X to hold the training samples and an array Y to contain the labels as inputs to the fit method. The array y in our situation just holds the values of the two classes.

3.1.3 Decision Trees, Ensemble Methods and Random Forest Classifier

Decision trees (DTs): For classification and regression, DTs are a non-parametric supervised learning approach. Learning basic decision rules derived from data attributes is used to develop a model that predicts the value of a target variable [23]. DTs are straightforward to comprehend, interpret, and depict. When predicting data with a decision tree, the cost is proportional to the quantity of data records required to train the tree. The substantial variance in the trees, which might be the result of slight alterations in the data, is one of the most major downsides of DTs. Use one of the ensemble approaches that will be discussed later as a solution. Overfitting is another issue that may be addressed by increasing the tree's maximum depth.

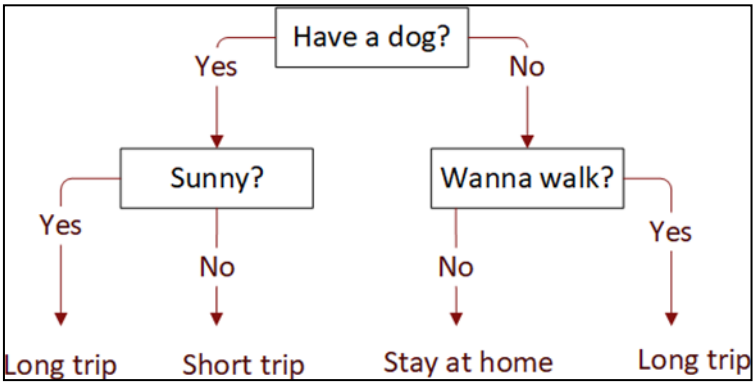

Figure 3.2: An example of the decision tree.

Ensemble Methods: Ensemble methods combine the predictions of various machine learning algorithms to increase the robustness of a single prediction [23].

The majority of the literature divides ensemble techniques into two groups: boosting and bagging. Boosting is a method of creating a powerful learner by combining numerous extremely weak learners (estimators). Bagging is a simpler ensemble approach that combines many forecasts and selects the average of them [24].

The Random forest method is one of the randomized decision tree-based bagging techniques included in the scikit-learn learn toolkit. The Random forest method is a set of perturb-and-combine algorithms [33] tailored to trees. A varied group of classifiers is formed by injecting randomness into the classifier. The average forecast is created by combining the ensemble of individual predictions.

The Random forest classifier, on the other hand, is a concrete supervised implementation of the bagging techniques utilized in this thesis for training.

The Random Forest Classifier [34] class in the Python package scikit-learn may be used to implement the technique. The number of decision trees that will be built in an ensemble from the training set may be specified using the constructor option named n estimators. A sparse or dense array X storing the training samples and an array Y carrying the labels must also be included in the classifier. The optimal split of each node is discovered during the building of each tree, either from all input features or a random selection of size max features from the constructor's argument. The rationale for this is to reduce the forest estimators' variance. Parameter specifies the minimum

number of samples necessary to divide an internal node with the name min samples split. The internal node [35] is recursively partitioned by a decision tree until the maximum permitted depth is achieved. Allow Q to represent the data at node m. A feature j and a threshold tm make up each candidate split = (j, tm).

3.1.4 Naive Bayes-Multinomial Naive Bayes Classifier

The Naive Bayes techniques are a set of supervised probability-based approaches. They assume, in particular, that given the value of the class variable, every pair of characteristics is conditionally independent. The chance of acquiring the string of feature values (equation (.15)) is equal to the product of multiplying all of the individual probabilities (equation (.16)).

This assumption implies that the traits are unrelated to one another, which is not the case. This simplification, on the other hand, improves computing performance, and the algorithm produces results that are comparable to classification techniques in a given area. As a result, the Naive Bayes classifier's classification criterion is to choose class Ci for which the computation from equation is the highest.

Because conditional feature distribution may be computed individually as a one-dimensional distribution, the Naive Bayes classifier is quicker than other, more advanced approaches. When the number of dimensions grows, considerably more data is required, which is where this strategy comes in handy. The different naive Bayes' classifiers differ in their distribution of P (Xk = ak | Ci) according to assumptions they make [23]. Class Multinomial [36] is a concrete implementation for multinomial distributed data from the Python scikit-learn module. Because it is often used in text categorization [23], it was chosen.

3.2 UNSUPERVISED MACHINE LEARNING METHODS

It might be challenging to get labels for data in many cases. It may, for example, entail someone manually labeling the data. Unsupervised machine learning algorithms are an excellent choice in this scenario since they don't require any labels. Due to the lack of proper outputs, the algorithm must detect certain commonalities between input data points. The categorization might also be based on the detection of similarity between data items [24].

Unsupervised techniques, unlike supervised methods, cannot learn based on an external error criteria (the difference between objectives and accurate outputs). They are based on the fact that an aberrant instance generally appears as a remote outlier point from other examples.

So the inputs that are closer together are recognized as similar (they are clustered together to the same class). In contrast, the inputs that are far apart are not recognized as similar (they are put into different classes if possible) [24].

3.2.1 K-means

One of the clustering algorithms, K-means [37], separates input data X into k disjoint clusters C with equal variance, minimizing the inertia, or within-cluster sum of squares, which is defined by equation (.18).

The mean (also known as the centroid) of the samples in the cluster is j, and xi is X. The means aren't input values, although they are from the same space [5].

The package scikit-learn provides a concrete implementation of the technique as a class K-Means [38]. It is possible to modify the number of clusters k using the n-clusters option. There are three steps in the K-means algorithm. The centroids of all clusters must first be initialized. The simplest method is to choose them randomaly , However, the scikit-learn library's K-means library provides a smarter approach to initialize them. The core idea is to arrange centroids' values to be far apart, resulting in demonstrably superior outcomes [39]. The second step involves assigning all data points from X to the cluster's closest centroid. All centroids are updated to the mean value of all data points from X to each preceding centroid to reach a minimum result for equation (18). The difference between the new and old centroids values is calculated. The loop between the second and third stages is repeated until the difference between the new and old centroids values is less than the threshold.

The distance of each data point from X to centroids is often computed as Euclidean distance, but there are other alternatives also [24].

The method will always converge in general, however it may converge to a local minimum rather than the global minimum. It is very dependent on the centroids' starting values. The n-init option

in the K-Means class allows you to change the number of times the algorithm runs with various centroid seeds. The drawback is that it may take a long time for the algorithm to complete. The method K-means from the scikit-learn library may be executed in parallel, making it quicker but at the expense of memory.

3.2.2 Gaussian Mixture

A Gaussian mixture model [23] is a probabilistic model in which all data points are derived from a combination of a finite number of Gaussian distributions with unknown parameters. The Gaussian distribution, often known as the Normal distribution, is a symmetric probability distribution around the mean. Data that are distant from the mean are less common than data that are close to the mean. An example of a probability density function is shown in figure 3.3, where indicates the mean.

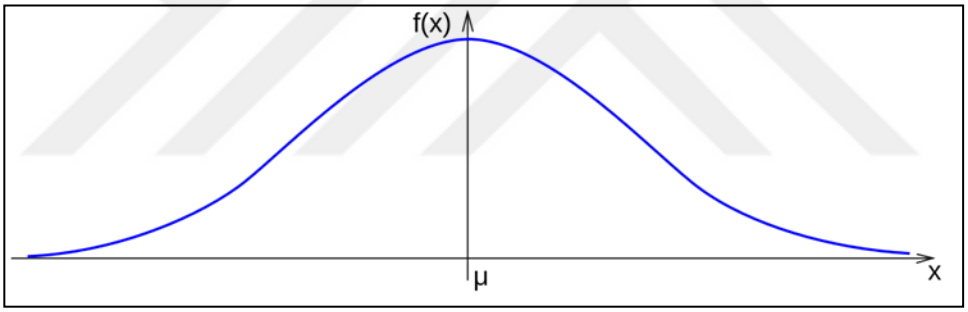

Figure 3.3: An example of a probability density function of Gaussian distribution.

The expectation-maximization (EM) approach is used to fit the Gaussian mixture object built from a class Gaussian Mixture [37] given by the scikit-learn toolkit. The EM method uses an iterative technique to solve the challenge of determining which points came from which hidden component. The EM method assumes random components in the first step, either randomly centered on data points or normally distributed about the origin. The number of components may be changed using the n components parameter of the class Gaussian Mixture. For each point, the chance of being created by each component is calculated. The parameters are tweaked in the second stage to increase the chance of the data given those assignments. The loop between these phases continues, and convergence to a local optimum is always ensured [38]. The class Gaussian Mixture allows you to change the number of times the algorithm runs with various centers of components (means) using the n init parameter.

3.3 SEMI-SUPERVISED MACHINE LEARNING METHODS

Semi-Supervised techniques were developed in response to two major drawbacks of both supervised and unsupervised approaches. The high expense of supervised learning on large amounts of data, as well as the restricted scope of applicability of unsupervised machine learning approaches. Semi-supervised algorithms only require a limited quantity of labeled data, with the remainder remaining unlabeled. So, in most cases, we utilize clustering (an unsupervised approach) to cluster a large portion of the data, and then label the rest using a supervised method [39].

3.4 URL DETECTION WITH MACHINE LEARNING APPROACH

Machine Learning techniques use a collection of URLs specified as a set of characteristics to train a prediction model that can categorize a URL as benign or dangerous based on those attributes. This allows them to detect new possibly harmful URLs as well [17].

As previously stated, this strategy generally consists of two steps: the first is the selection and proper representation of characteristics, and the second is the training of a prediction mechanism utilizing this representation. Depending on whether the algorithm knows or doesn't know if URLs are dangerous or benign, machine learning can be characterized as supervised, unsupervised, or semi-supervised.

We may divide the models into two categories based on how they accept and learn from training data: (learn in batches): *Batch Learning* (learn over finite data group) and *Online Learning* (accepting and subsequently learning over data flowing in streams).

3.4.1 Data Preparation

A process for converting raw data into a clean data collection is known as data preparation. The identification and extraction of useful characteristics that will adequately represent URLs as feature vectors is the initial stage. The characteristics are frequently chosen using heuristic approaches based on past knowledge of dangerous URLs.

Following that, the features vectors must be translated into a format that the algorithm can take as input. Because the learning is based on mathematical equations and computations, they usually

take only numerical vectors if they do not change data themselves. Before feeding the data into the model, some sanitization and normalization is required.

3.4.2 Representing Text Numerically

The conversion of non-numerical (text) information to numerical values is an important aspect of feature processing. Features with a single text value (host country, top domain) or a vector of text values must be transformed (tokens in the path). All of these variables are discrete, categorical data in the context of URLs.

Every encoding approach starts with the creation of a vocabulary of all possible values (words) for specific attributes. We may then perform a variety of things with this terminology.:

3.4.3 Bag-of-Words

In machine learning techniques, a bag of words is a typical method of encoding text data. It is well liked due to its simplicity and ease of usage. The basic existence of words is measured once the vocabulary has been built.

In a nutshell, each text characteristic is converted into a vector of 0 and 1 in the vocabulary size, with 1 denoting present value and 0 denoting absence. If a feature value can include the same word several times, such as tokens along a route, word frequencies can be preserved instead of just their existence.

This method is called "bag" of words because any information about the order or structure of words is discarded, causing lost of words order or their similarity [25]. Moreover, it is very heavy on size, with output as a sparse vector (most of the values are 0).

```
url_one = {'country':'Russia', 'path_tokens': {'be', 'successful'}}
url_two = {'country':'Slovakia', 'path_tokens': {'think','happy', 'be', 'happy'}}

vocabulary_country = {'Russia', 'Slovakia'}
vocabulary_path_tokens = {'be', 'successful', 'think', 'happy'}

url_one_bow = {{1,0},{1,1,0,0}}
url_two_bow = {{0,1},{1,0,1,1}}
url_two_bow_freq = {{0,1},{1,0,1,2}}
```

Figure 3.4: Bag-of-Words example.

3.4.4 One-Hot Encoding

This approach is similar to Bag-of-Words, except it retains the sequence of the words, unlike Bag-of-Words. Each word is converted into a 0-1 vector. A feature with a vector of words will be turned into a vector of vectors. As a result, each word will be encoded as a vector with just one value of 1 and the rest of 0 values.

```
url_one_one_hot = {{1,0},{{1,0,0,0},{0,1,0,0}}}
url_two_one_hot = {{0,1},{{0,0,1,0},{0,0,0,1},{1,0,0,0},{0,0,0,1}}}
```

Figure 3.5: Bag-of-Words example 2.

Because the output is a sparse vector, this method, similar to the previous one, is considered as inefficient.

3.4.5 Encoding To Unique Numbers

With this approach, each unique word is encoded as a unique number. The order of the words is not lost, but as in the previous method, there is no relationship between the similarity of any two words. The output is a dense vector, and therefore this method is more efficient.

```
url_one_unique_num = {1,{1,2}}}
url_two_unique_num = {2,{3,4,1,3}}
```

Figure 3.6: Bag-of-Words example 3.

3.4.6 Word Embeddings

When the encoding of similar words is the same, word embedding is used to represent them. As a consequence, a dense vector of floating-point values is returned. The weights represent the outcomes of the machine learning model, not manually derived values. This format allows us to record a tight link from context (pay – bank) or a word/subword relation (bank – banking). Word2vec [18], invented by Mikolov et al., is the most well-known approach for word embedding representation. The Continuous Bag-of-Words (CBOW) and the Skip-gram model are combined in Word2vec.

The CBOW algorithm predicts a word based on its context, whereas the Skip-gram algorithm predicts a word based on its context. [27]. The construction of a vocabulary is significantly more difficult, as it takes into consideration the distances between each word.

3.4.7 Normalize Data

In machine learning algorithms, varying sizes of numerical features may have a deleterious impact on the learning process. Even though they aren't always more significant, features with higher values can have a greater impact on the outcome. As a result, the properties must be normalized such that they all have the same scale.

There are different types of data normalization:

3.4.8 Standardization (Zero mean normalization)

Standardization is a method of transforming data such that it has the features of a standard normal distribution, such as a mean of 0 and a standard deviation of 1. The following is a typical formula for calculating standardized values:

$$Zi = \frac{\text{Xi} - u}{\acute{o}} \tag{3.3}$$

where x is original value, μ is the mean and σ is the standard deviation from the mean

Min-Max scaling

In this approach, the data is rescaled to a fixed range – usually 0 to 1. Scaled values are achieved by this formula:

$$Zi = \frac{\text{Xi} - Xmin}{Xmax - Xmin} \tag{3.4}$$

where x_{min} is a global minimum and x_{max} maximum

Decimal Normalization

In this method we normalized by moving the decimal point of value

$$\boldsymbol{zi} = \frac{\mathbf{Xi}}{\mathbf{10}\boldsymbol{i}} \tag{3.5}$$

where j is the minimal value such that $|z_{max}| < 1$

3.4.9 Batch Learning

Batch-based learning algorithms take as input and train their models on a set of data. After that, the model is utilized for classification and prediction. After a period of time, the model becomes obsolete and must be retrained on new data.

Because of the frequent requirement to retrain and the quantity of data on which the algorithm can train, batch learning algorithms may not always be the most efficient approach.

3.4.10 Support Vector Machine (SVM)

A frequently used batch learning algorithm is the Support Vector Machine. It has been utilized effectively in the identification of malicious URLs [19],[20],[21],[22].

SVM builds a collection of hyperplanes in a high or infinite-dimensional space for classification. The hyperplane with the largest distance between the nearest training sample and the hyperplane

is then chosen[23]. This may be accomplished by using the following formula:

$$max \sum_{i=1}^{N} \propto i - \frac{1}{2} \sum_{i.j=1}^{N} y_i y_j \ \propto_i \propto_j K(x_i x_j) \quad (3.6)$$

Subject to

$$\sum_{i=1}^{N} y_i \propto i = 0; \propto i \geq 0 \, . i = 1,2 \ldots . . N \quad (3.7)$$

where $K(x_i\, x_j)$ is a Kernel function used, and α_i and α_j are coefficients associated with examples x_i, x_j computed to maximize the margin of correct classification on the training set.

4 RESEARCH METHODOLOGY

4.1 INTRODUCTION

In general, the whole process of data science as shown in figure 4.1 , or in this case, a machine learning experiment follows an all-purpose template, which can be outlined in the following seven steps:

1. Problem Definition
2. Dataset Collection.
3. Dataset Pre-Processing
4. Feature Extraction
5. Model Training
6. Evaluation

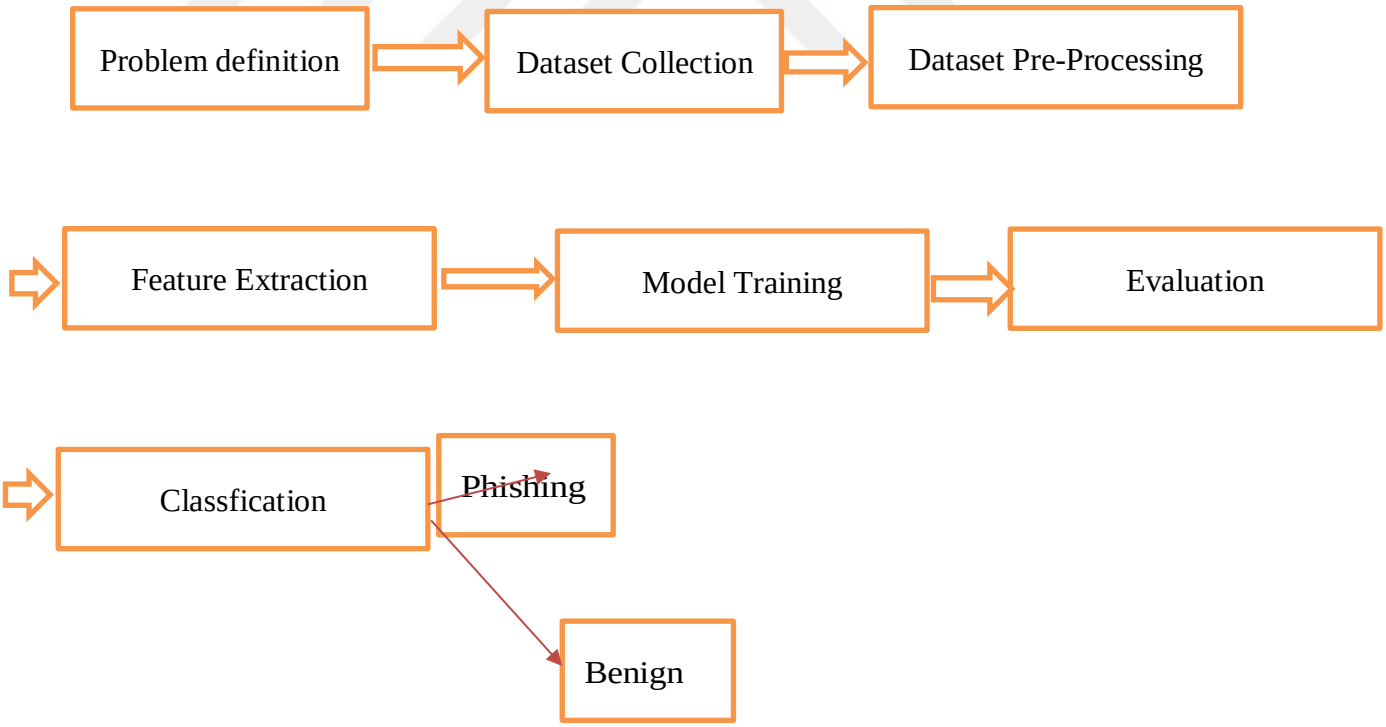

Figure 4.1: Research Methodology.

The first two steps have already been addressed in the previous chapters. Therefore this chapter will describe methodologies used for the remaining five. An important note to make is that although these steps are generally realized in a similar order to the one given above, it is common that individual steps are intertwined to some degree.

4.2 DATASET

All subsequent processes in the data analysis pipeline are built on the foundation of data. No matter how many steps we take throughout the feature selection and model training phases, the system will not be able to generalize successfully to new data unless the underlying training data is reliable. The collecting of data from relevant sources that offer an accurate picture of the prevalence of various types of URLs on the Internet was a top objective for this investigation. To eliminate bias against any of the categories, special attention was paid to establishing a dataset that was as fair as possible. The size of the dataset also matters, because choosing a small dataset is a simple way to speed up the procedure.

4.2.1 Dataset Overview

We used data set that have been used before by the source [thesis], Naturally, there were some duplicate URLs or several URLs that were nearly identical, which would be detrimental to the model training phase. Some of the URLs were also discovered to be incorrect. These were either discarded or, if feasible, their structure was repaired. The structure of the dataset after pre-processing phase described in is summarized in the following table:

Table 4.1: Overview of the dataset structure.

URL Type	No. of Samples
HTTP requests from NFDUMP	28148
PhishTank's archive crawl	25156
Benign SNIs crawl	5427
Openphish Feeds	6961
Phishtank Downloadable Dataset	3587
Phishstats Feeds	2014

Table 4.2: Examples of phishing vs. benign URLs.

URL	Type
http://tax.account-1payment.com/	phishing
http://www.apple.com.ntemus.cn/mim/56h66999v22	phishing
https://grupwhatsapp18terbaru.ezua.com	phishing
https://udalosti.signaly.cz/201411	benign
https://www.flightconnections.com/route-map-ai...	benign
http://download2.epicgames.com/Builds/UnrealEn	benign

This table showcases a few examples of phishing versus benign URLs.

4.3 FEATURES SELECTION

This section will describe the methods used for extracting and engineering features that were used for this experiment.

4.3.1 URL Based Features

The idea behind these characteristics is that phishing URLs tend to stray from the norm in some way. The following sorts of characteristics were used to try to capture these deviations:

i. HTTPS, explicit port specification, or IP address rather than a hostname.

ii. Special characters such as dash, at ('@'), and colon are present (counted).

iii. URL, subdomain, SLD, netloc, and path lengths

iv. The number of digits in the URL, subdomain, SLS, netloc, and path.

v. The number of subdomains and the depth of the route.

4.3.2 Page Based Features

The premise is that phishing websites are typically located on shady domains, thus page-based elements should reflect the site's trustworthiness. OpenPageRank (OPR) 5, an open-source page ranking tool, was utilized to extract these attributes. The OPR is a free alternative to the more well-known Alexa 6. Through its REST API, it gives users access to billions of website evaluations and rankings.

For this experiment, two characteristics were extracted: rating and ranking. The rating is a number between 0 and 10, with 10 signifying the highest level of trustworthiness. The rating is based on the popularity of the website, with 1 signifying the most "popular" website.

4.3.3 Domain Squatting Features

We should try to assess how suspicious the URL is from the standpoint of domain/typosquatting because phishers frequently try to replicate the names of well-known businesses and organisations. We used OPR to extract the top 500 most popular websites for this work. The top 500 list was then compared to each subdomain and route segment. The route, subdomain, and SLD components with the highest obtained similarity were recorded as characteristics. The normalized Levenshtein distance was employed as the similarity metric for string matching (also known as normalized edit distance).

5. EVALUATION

5.1 INTRODUCTION

This chapter provides a summary of the results and demonstrates some of the weaknesses in the presented approach and suggests possible improvements. We used precision, Recall, F-Measure and accuracy to evaluate each algorithm , as shown in Figure 5.1 the equations for precision, Recall, F-Measure and accuracy.

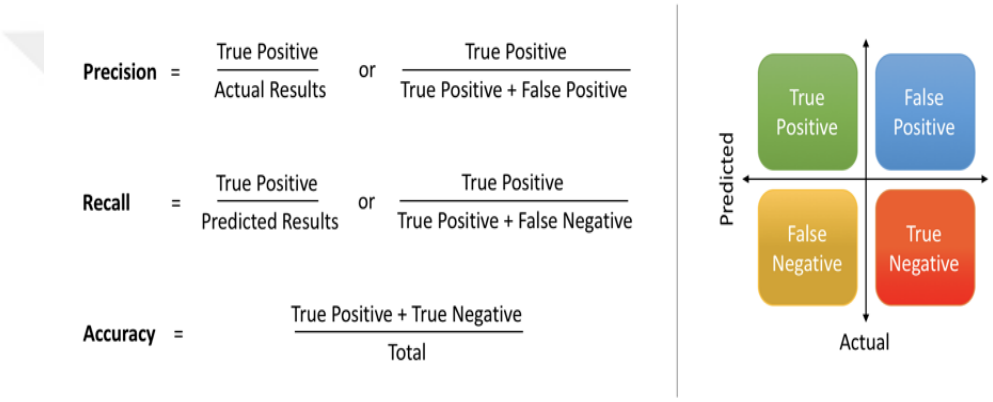

Figure 5.1: Equations for Precision, Recall, F-Measure and Accuracy.

$$f1\ Score = 2 * \frac{precision * Recall}{precision + Recall} \qquad (4.1)$$

And the overall results were as below:

In term of Precision, the analysis results show that random KNN has the heights (Figure 5.2):

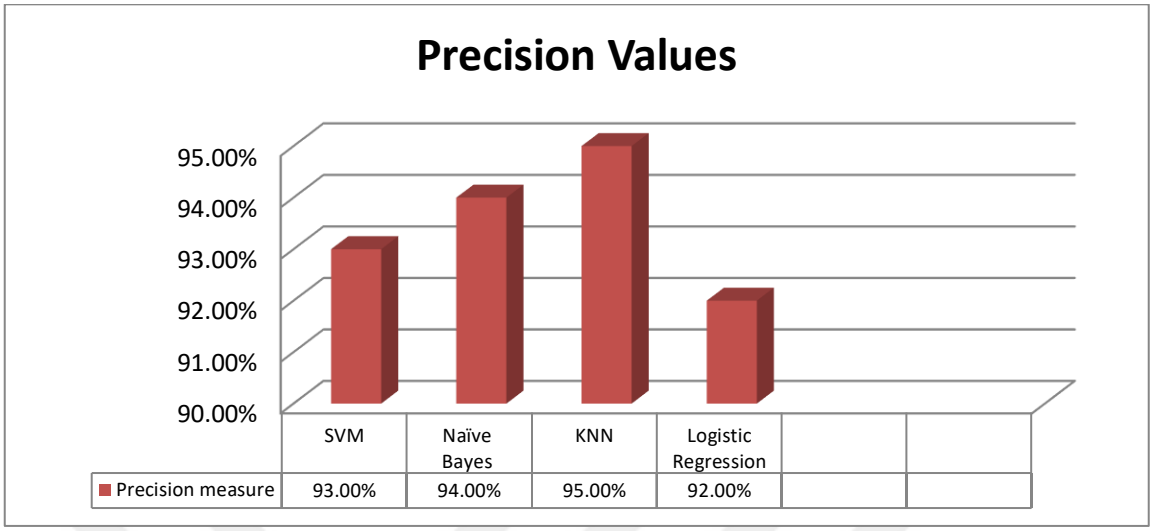

Figure 5.2: Precision Values of All the Classifiers.

In term of Recall, the analysis results show that KNN and Naïve Bayes classifier has the heights value as (Figure 5.3):

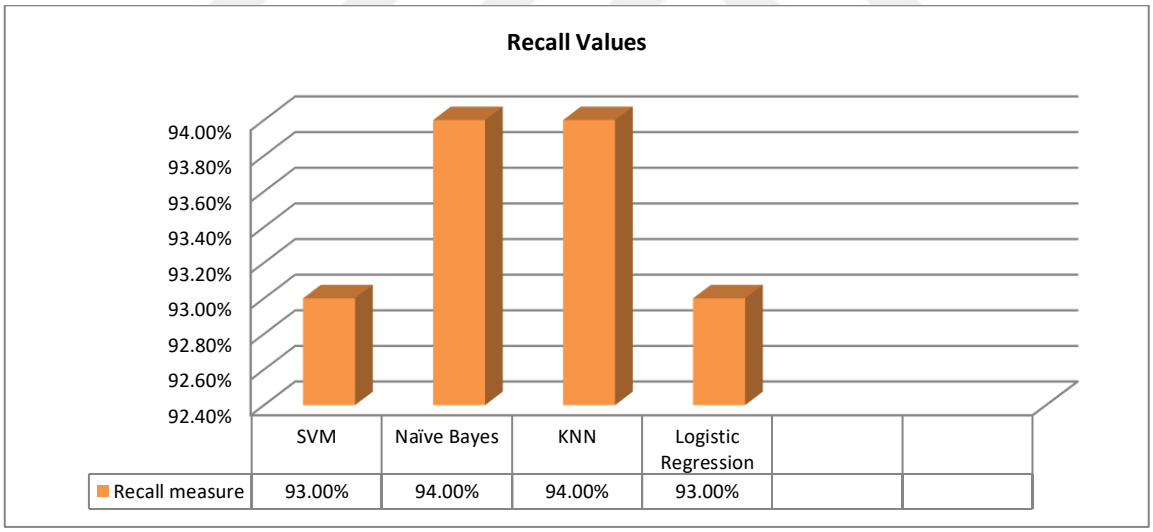

Figure 5.3: Recall Values of All the Classifiers.

In term of F-Measure, the analysis results show that KNN and Naïve Bayes has the heights value (Figure 5.4):

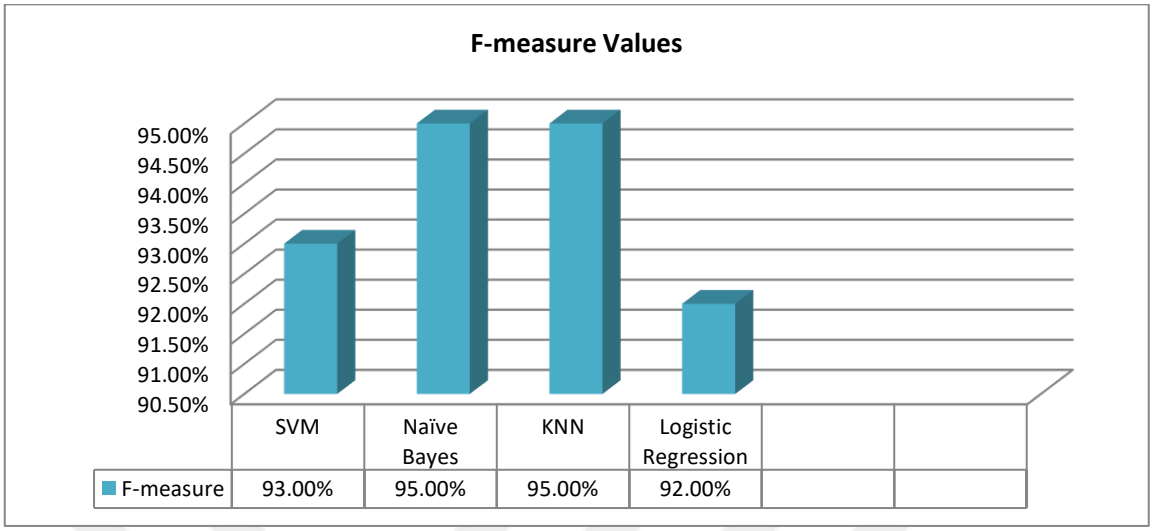

Figure 5.4: F-Measure Values of All the Classifiers.

In term of Accuracy, the analysis results show that KNN classifier has the heights value (Figure 5.5):

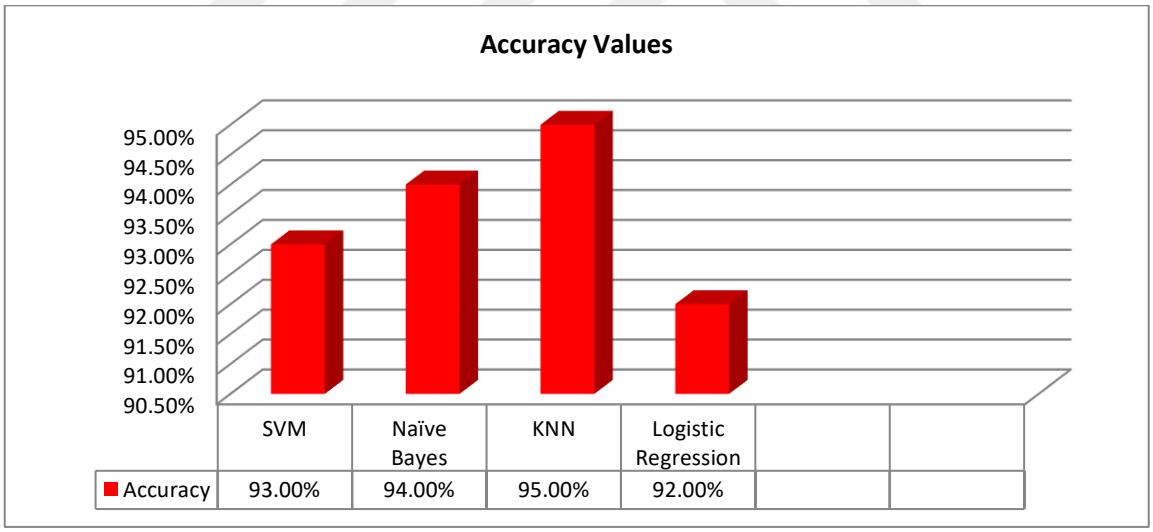

Figure 5.5: Precision Values of All the Classifiers.

The results obtained through the analysis show that KNN classifier is the best classifier as the values of all of the comparison factors were heights then the value of the other algorithms. Analysis results of all the classifiers shown in Table 5.1.

5.2 SUMMARY OF THE RESULTS

Table 5.1: Summary of the experiment results.

Classifier	Accuracy	Precision	Recall	F1 Score
Logistic Regression	92%	0.92	0.93	0.92
Naive Bayes	94%	0.94	0.94	0.95
K-Nearest Neighbors	95%	95	94	95
Support Vector Machines	93%	92	93	93

As we can observe, K-Nearest Neighbors achieved the best results in all of the metrics. Interestingly enough, Naive Bayes was able to achieve almost as good precision as K-Nearest Neighbors, but less impressive recall. If we compare the precision of Logistic Regression versus Support Vector Machines , we can observe how the "wisdom of the crowd" was able to outperform a single Naive Bayes. Naive Bayes, SVM and KNN had almost identical values of AUC metric. support Vector Machines performed the worst during this experiment.

6. CONCLUSION

This thesis looked on the topic of dangerous website content and phishing techniques utilized on then internet. There are a variety of methods for detecting harmful material, and the detection based on the website's datasets was used in this case. The main output of this thesis is a detailed discussion of a method for handling the phishing detection issue without using web scraping. A independent machine learning project is produced as a supplementary output, allowing future replications or alterations of the accomplished experiment. The purpose of this thesis was to see whether machine learning could be used to identify phishing URLs without scraping them. The second part included a literature analysis and background on current phishing trends, as well as a list of common forms of internet-based phishing and an explanation of the major approaches for generating a false URL. The third chapter gave a high-level review of the strategies employed in the area of malicious URL detection. The emphasis was mostly on machine learning approaches, since they are the most effective. The fourth chapter highlighted the fundamental ideas that each of the chosen phases of our technique is based on, as well as the methodology employed in the experiment.

Finally, the fifth chapter evaluated the performance of the models, summarized the experiment's findings, and analyzed the approach's shortcomings and potential improvements. The findings of this experiment indicate that using the KNN algorithm, we can identify phishing URLs with 95% precision (for the positive class) and categorize URLs as benign or phishing with 95% accuracy. Except for Logistic Regression, which exhibited somewhat lower results, all of the other utilized methods behaved similarly.Other broad enhancements to the technique given might possibly improve the performance of the models. Longer hyperparameter tweaking stages would have improved the experiment. The larger the searched hyperparameter subspace and the more iterations of random search there are, the better. The dimensionality was limited to maintain the training duration to a tolerable level, as mentioned in the feature engineering procedure. To detect domain squatting, a bag-of-words approach with a broader vocabulary or the inclusion of additional brands would undoubtedly provide better results. Of course, these recommendations entail lengthier training hours and more expensive technology.

REFERENCES

[1] N. Aydin, M., Butun, I., Bicakci, K., & Baykal, "Using Attribute-based Feature Selection Approaches and Machine Learning Algorithms for Detecting Fraudulent Website URLs," in *10th Annual Computing and Communication Workshop and Conference (CCWC), ,* pp. 0774–0779.2020.

[2] Chen, W., Zeng, Y., & Qiu, M, "Using Adversarial Examples to Bypass Deep Learning Based URL Detection System". *IEEE International Conference on Smart Cloud (SmartCloud)* , ,pp. 128-130.2020.

[3] W. INC., "2019 Webroot Threat Report,". [Online]. Available: www.cdn.webroot.com / 9315 / 5113 / 6179 / Webroot_Threat_Report_US_Online.pdf.2019.

[4] Liang, Y., & Yan, X. , "A comparison of machine learning attributes for detecting malicious websites," in *11th International Conference on Communication Systems & Networks (COMSNETS), ,* pp. 487–492.2019.

[5] Singh, A. K., & Goyal, N. "A comparison of machine learning attributes for detecting malicious websites.," *11th International Conference on Communication Systems & Networks (COMSNETS)*, pp. 352–358.2019.

[6] W3C. "*XPath,"*. 2017. [Online].Available: https://www.w3.org/TR/xpath/.

[7] WATIR.COM,"*Watir documenatation"*.. [Online]. Available: https://watir.com/documentation/.2016.

[8] MALWAREURL,"*Phishing blacklist operated by OpenDNS"*. Availablehttps://www.malwareurl.com/.

[9] URLHAUS,"*Malware blacklist operated by abuse.ch*" Available: https://urlhaus.abuse.ch/.

[10] CHRONICLE." *VirusTotal is an online service that analyzes files and URLs"*. Available https://www.virustotal.com.

[11] GOOGLE, " *Google Safe Browsing - Blacklist service provided by Google*", Available: https :// safebrowsing . google .com/.

[12] OPENPHISH, "*Automated self-contained phishing blacklist"*, Available: https://www.openphish.com/.

[13] L. Lee, Jin-Lee and Kim, Dong-Hyun and Chang-Hoon, "Heuristic-based approach for phishing site detection using url features," in *the Third Intl. Conf. on Advances in Computing, Electronics and Electrical Technology-CEET,* , pp. 131--135. 2015.

[14] M. PRAKASH, P.; KUMAR, M.; KOMPELLA, R. R.; GUPTA, "PhishNet: Predictive Blacklisting to Detect Phishing Attacks," *Proceedings IEEE INFOCOM.,* , pp. 1–5.2010.

[15] R. G. SOLANKI, J.; VAISHNAV, "Website phishing detection using heuristic based approach," in *Proceedings of the third international conference on advances in computing, electronics and electrical technology.*, 2015.

[16] SAHOO, Doyen; LIU, Chenghao; HOI, Steven CH. Malicious URL detection using machine learning: A survey. *arXiv preprint arXiv:1701.07179*, 2017.

[17] MIKOLOV, Tomas, et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[18] G. MA, Justin; SAUL, Lawrence; SAVAGE, Stefan; VOELKER, "Beyond blacklists: learning to detect malicious Web sites from suspicious URLs," *AASRI Procedia*, pp. 1245–1254, 2009.

[19] A. Kolari, P., Finin, T., & Joshi, "SVMs for the blogosphere: Blog identification and splog detection.," *AAAI spring Symp. Comput. approaches to Anal. weblogs*, pp. 92–99, 2006.

[20] Y.-J. L. Pao, Hsing-Kuo, Yan-Lin Chou, "Malicious URL detection based on kolmogorov complexity estimation ," in *International Conferences on Web Intelligence and Intelligent Agent Technology*, 2012.

[21] G. M. MA, Justin; SAUL, Lawrence K.; SAVAGE, Stefan; VOELKER, "Identifying Suspicious URLs: An Application of Large-scale Online Learning.," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

[22] H. CHOI, Hyunsang; ZHU, Bin B.; LEE, "Detecting Malicious Web Links and Identifying Their Attack Types," in *Proceedings of the 2Nd USENIX Conference on Web Application Development*, 2011.

[23] GERÓN, Aurelién. Hands-On Machine Learning with Scikit- Learn, Keras, and TensorFlow. In: O'Reilly Media, *Inc.*, chap. 6. ISBN 978-1-49-203264-9.2017.

[24] A. GERÓN, *Hands-On Machine Learning with Scikit- Learn, Keras, and TensorFlow*. 2017.

[25] *"Decision Trees"*,scikit-learn. [online]. .Available: https://scikit-learn.org/stable/modules/tree. html.

[26] L. R. F. BREIMAN, *Machine Learning*. 2001.

[27] P. COVER, T. HART, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, pp. 21–27, 1967.

[28] V. Cortes, Corinna and Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[29] *"Support Vector Classification"*. scikit-learn.[online]. Available: https://scikit-learn.org/stable/ modules/svm.html#svm-classification.

[30] *"Naive Bayes"* .scikit-learn. [online]. Available : https : / / scikit - learn . org / stable / modules / naive_bayes.html.

[31] T. Mitchell, "Decision Tree Learning", in T. Mitchell, Machine Learning, The McGraw-Hill Companies, Inc., , pp. 52-78.1997.

[32] P. Winston, "Learning by Building Identification Trees", in P. Winston, Artificial Intelligence, *Addison-Wesley Publishing Company*, pp. 423-442.1992.

[33] M. Goldstein, Kn -nearest Neighbor Classification, Mathematics, Computer Science, *IEEE Trans. Inf. Theory* DOI:10.1109/TIT.1972.1054888, Corpus ID: 30697197.1972.

[34] Sumeet Dua, U. Rajendra Acharya, Prerna Dua, Machine Learning in Healthcare Informatics, DOI https://doi.org/10.1007/978-3-642-40017-9, Springer-Verlag Berlin Heidelberg 2014, Print ISBN978-3-642-40016-2.2014.

[35] M. Bichler, C. Kiss, A Comparison of Logistic Regression, k-Nearest Neighbor, and Decision Tree Induction for Campaign Management, Published in AMCIS 2004

[36] Lichman, M.. UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.2013.

[37] Ankerst, M., Keim, D.A., Kriegel, H.-P.: Circle segments: A technique for visually exploring large multidimensional data sets. In: Visualization (1996)

[38] Andriy Burkov, The Hundred-Page Machine Learning Book, Publisher: Andriy Burkov

(2019)

[39] "Building Decision Trees with the ID3 Algorithm", by: Andrew Colin, Dr. Dobbs Journal, June 1996

[40] "C4.5 Programs for Machine Learning", by J. Ross Quinlan, Morgan Kaufmann, 1993

[41] Breiman, Leo, et al. Classification and regression trees. CRC press, 1984.

[42] Kataria, A., & Singh, M. D.. A Review of data classification Using K-nearest neighbour Algorithm. *International Journal of Emerging Technology and Advanced Engineering*, 3 (6), 354–360.2013.

[43] Chomboon, K., Pasapichi, C., Pongsakorn, T., Kerdprasop, K., & Kerdprasop, N. (2015). An empirical study of distance metrics for k-nearest neighbor algorithm. In The 3rd International Conference on Industrial Application Engineering (pp. 280–285).2015.

[44] Ting K.M. Confusion Matrix. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning and Data Mining. Springer, Boston, *MA. https://doi.org/*10.1007/978-1-4899-7687- 1_50.2017.

[45] Goutte C., Gaussier E. A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation. In: Losada D.E., Fernández-Luna J.M. (eds) Advances in Information Retrieval. ECIR 2005. Lecture Notes in Computer Science, vol 3408. Springer, Berlin, Heidelber*g. https://doi.org/10.1007/978-3-540-31865-1_25*. .2005.

[46] E. Byvatov and G. Schneider, "Support vector machine applications in bioinformatics." *Applied bioinformatics*, vol. 2, no. 2, pp. 67–77, 2002.

[47] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Machine Learning: ECML-98,* ser. Lecture Notes in Computer Science, C. N´edellec and C. Rouveirol, Eds. Springer Berlin Heidelberg, vol. 1398, pp. 137–142. [Online]. Available: *http://dx.doi.org/*10.1007/ BFb0026683.1998.

[48] A. Ben-Hur and J. Weston, "A user's guide to support vector machines," in Data Mining Techniques for the Life Sciences, ser. *Methods in Molecular Biology, O. Carugo and F. Eisenhaber, Eds. Humana Press*, , vol. 609, pp. 223–239. [Online]. Available:

http://dx.doi.org/10.1007/978-1-60327-241-4 13.2010.

[49] S. Scott and S. Matwin, “Feature engineering for text classification,” in Proceedings of the Sixteenth International Conference on Machine Learning, ser. ICML ’99. San Francisco, CA, *USA: Morgan Kaufmann Publishers Inc.*, pp. 379–388. [Online]. Available: http://dl.acm.org/citation.cfm?id= 645528.657484.1999.

[50] “Tf-idf :: A Single-Page Tutorial - Information Retrieval and Text Mining.” [Online]. Available: *http://www.tfidf.com/*