The Academic College of Tel Aviv-Yaffo

School of Computer Science

Detecting Phishing Attacks by Machine Learning

Thesis submitted in partial fulfillment of the requirements for the M.Sc. degree in the School of Computer Science of the Academic College of Tel Aviv University

By

Asaf Rachmani

The research work for the thesis has been carried out under the supervision of

Prof. Adi Shraibman

December 2023

Abstract

In recent years, online services have become an important part of our lives. They make everything more accessible and help organizations and consumers get what they need in a fast and easy way. Most of the online services that exist today are using websites that contain sensitive data such as passwords, credit card details, medical information, etc. This sensitive data is valuable for cybercriminals that can use it in harmful ways. One of the major online security threats faced by the cyber-world today is phishing websites. In a phishing attack, the attackers create a dummy website, by copying the behavior of a legitimate website, send URLs to the target victim via emails, text messages, or social media, and ask the victim to provide sensitive data. One way to prevent such attacks is to learn to identify phishing websites. Of course, one can identify such a website after it has operated and claimed victims. The ultimate goal is to be able to automatically detect such suspicious websites before they cause any harm. Machine learning and URL classification can be used for this purpose and thus help prevent phishing attacks.

This thesis presents methods for detecting phishing attacks using machine learning techniques. The approach presents chosen machine learning models and algorithms such as Support Vector Machines, Logistic Regression, K-Nearest Neighbors, Decision Tree, Gradient Boosting, Light Gradient Boosting Machine, eXtreme Gradient Boosting, Categorical Boosting, Neural Networks, Multilayer Perceptrons, online machine learning, and describes the dataset and its features. The experiments mainly focus on reducing the features and keeping the high accuracy by using feature selection methodology, displaying the problems in the existing dataset, and creating an updated dataset.

Acknowledgments

I would like to express my gratitude to my supervisor Prof. Adi Shraibman for the guidance, direction, and investment. In addition, I want to thank him for his suggestions and support throughout this work. I would also like to extend my gratitude to all of my lecturers in the Academic College Tel-Aviv Yaffo, for their knowledge, skills, and support they imparted to me.

Table of Content

1. Introduction

Phishing is a type of social attack and technical subterfuge often used to steal user-sensitive data such as passwords, login credentials, and credit card details, or to install malware on the victim's device. In such an attack, the attackers send messages that appear to come from a reputable source, through emails, text messages, or social media. These messages may direct the users to click on a link to a scam website, where they are required to provide confidential information. Phishers also use emotions like fear, curiosity, and urgency to cheat the victims and convince them to click on the link in the message.

In May 2017, a massive phishing attack targeting millions of Gmail users hit Google, the attacker sent a message that appeared to come from a trusted source asking to open a Google Document attached. By clicking the link to the attached file, it referred to a page that looks the same as the Google Docs app, where the user was required to type their Google account credentials. This granted the attackers access to email and contact accounts.

In the last years, since March 2020, as the COVID-19 pandemic erupted across the world, lockdowns were imposed in many places, and the use of digital channels was increased, attackers took the opportunity to manipulate people's fears.

In April 2020, Google reported that they blocked 18 million coronavirus scam emails every day. Since then the number of phishing attacks has rapidly grown, and in August 2020 the number of unique phishing websites detected was 201,591 based on the APWG report [1]. With the ever-increasing use and dependency on online services, the risk of losing sensitive data to fraudsters has also been increasing.

1.1 Thesis Goals

The aim of this thesis is to use machine learning techniques in order to identify phishing websites with high accuracy, find the best classification models and tools based on performance and accuracy, and find the most significant features and minimal set of features that still provide good classification accuracy. Moreover, to create a new dataset containing updated URLs for legitimate and phishing URLs websites.

1.2 Thesis Overview

This thesis is focused on identifying phishing websites with the help of machine learning algorithms. Chapters 1 and 2 describe what a phishing attack is, and how it is related to URLs and websites, they also show how phishing attacks have increased in the last couple of years, this will be backed up by reports and graphs, in Chapter 3 I'll describe machine learning algorithms and models. Chapter 4 will describe the dataset and features to be used in the experiment, and Chapter 5 will present the experiments and the results of the experiments. Finally, in chapter 6 I'll present the conclusion and future work.

2. Background

This chapter introduces an overview of phishing attacks, the first section explains what a phishing attack is and how it works. The second section focuses on phishing activities in recent years followed by a description of the tools that are used for detecting phishing attacks.

2.1 Phishing Attack

A Phishing attack is a type of cybersecurity and a social attack with a goal to steal personal sensitive information. The attackers use fake websites and by manipulation, they try to encourage the victim to provide sensitive information. In most cases, the attackers warn the victim that they must provide the data as soon as possible in order to avoid their account being blocked.

Following are the main steps in a phishing attack
(see Figure 2.1 for illustration):

1. The attacker creates a phishing website.
2. Attacker sends a message to the potential victim, the message contains a link to the phishing website.
3. The victim reads the message and by clicking the link is redirected to the phishing website in order to provide some sensitive data.
4. The attacker collects the victim's sensitive data from the phishing website
5. The attacker uses the victim's sensitive data to perform actions on legitimate

websites.

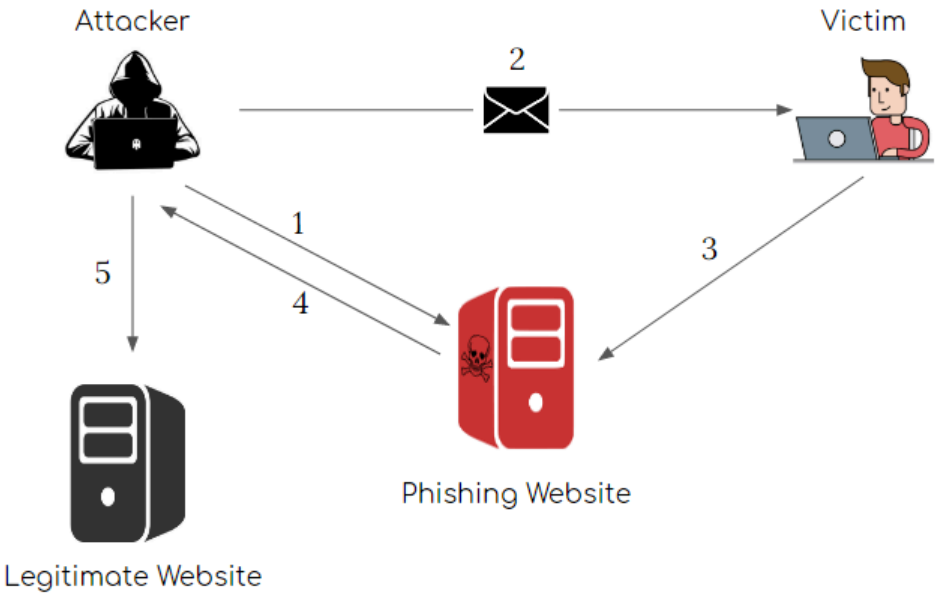


Figure 2.1: Main Steps in a Phishing Attack

2.2 Phishing Activity

According to the Anti-Phishing Working Group (APWG) reports [1], phishing attacks increase every year and become one of the major online attacks across the globe. Figure 2.2 represents the number of unique phishing websites detected as phishing attacks per quarter between 2020 and 2022.

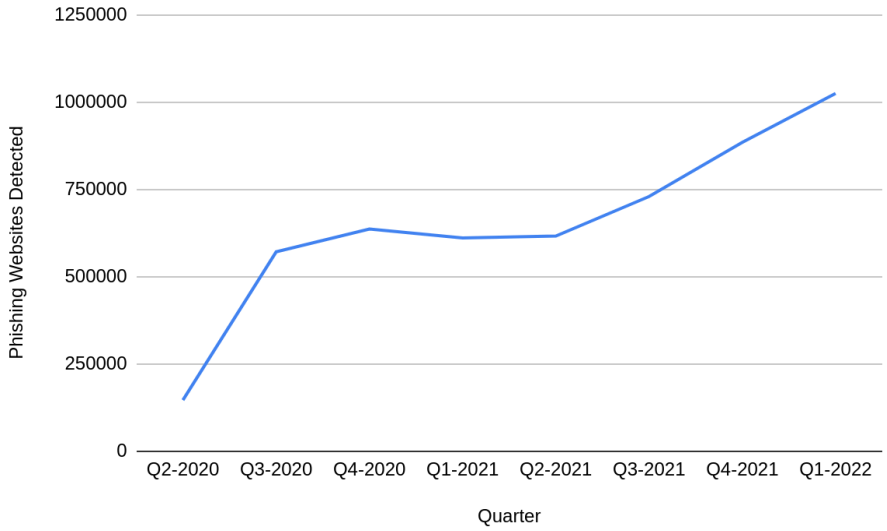


Figure 2.2: Number of unique phishing websites detected per quarter

In the first quarter of 2022, APWG reported 1,025,968 total phishing attacks over the globe (meaning around 340,000 per month), which was the highest number in APWG's reporting history and the first time that the quarterly total has exceeded one million. The number of phishing attacks has more than tripled since early 2020 when APWG was observing between 68,000 and 94,000 attacks per month. Based on Figure 2.2 It is clearly an increasing trend and it probably will continue to grow in the years to follow. In Figure 2.3, APWG represents the phishing attacks divided by industrial sectors for the first quarter of 2022. The largest set of attacks which is 23.6 percent of all the phishing attacks was against the financial sector which includes banks. The second largest sector to be attacked by phishing is software-as-a-service (SAAS) and webmail with 20.5 percent.

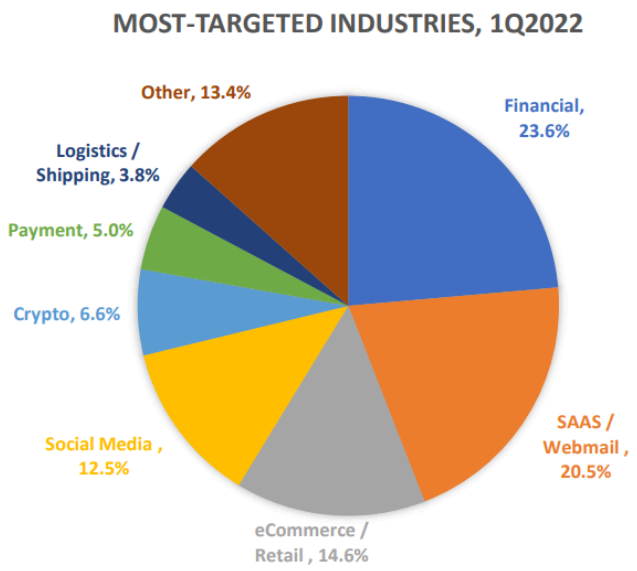


Figure 2.3: Most-Targeted Industry Sectors Q1-2022

2.3 Phishing Detection Tools

Phishing Detection Tools help organizations and consumers identify phishing content in emails, websites, social media, etc. These days most of the familiar browsers use built-in anti-phishing tools, and organizations use third-party software and hardware to identify and block such attacks.

There are a few approaches to detect phishing attacks:

- Browser Toolbars - Security Framework for client-side, using a browser add-on, that shows warnings or security-related information about websites to help users detect phishing attacks.

- Black/White list - A white list is a list of legitimate Domains/URLs/IPs while a black list contains the information of those sites which are fake. Black lists are

built through human feedback and are ineffective in blocking short-lived phishing web pages. Blacklisting has been used by all major Web browsers.

- User Vigilance - One of the most important approaches to preventing phishing attacks is user vigilance and education, in order to increase the awareness of potential attacks. The user should be aware of the risks, think about why he is even receiving this message, verify the message's sender, scan the link/attachment before opening it, etc.

- Machine Learning - Machine learning algorithms can be a powerful tool for detecting phishing websites and designating them as information security threats. This methodology can prove useful to a wide variety of organizations that are seeking solutions to this threat.

Most of the machine learning models are classified as supervised ML, the algorithm tries to find a function that maps an input to an output. It infers a function from training data, consisting of a set of training examples to their classification, either 1 for a phishing website or 0 for a legitimate website.

In principle, building a classification model using machine learning algorithms involves:

- Analyzing the suspect webpage.
- Extracting the features from the webpage.
- Classifying the webpage as legitimate / phishing using the model.

2.4 Related work

A number of techniques have been described in the literature to detect phishing attacks in recent years. A few of them are briefly presented in this section. Jain, A.K et al. [2] presented a phishing detection system based on machine learning using an SVM classifier, where 14 different features were used and produced an accuracy of 91.28%. Tan, C.L. et al. [3] presented the “PhishWHO” system which detects phishing webpages by extracting and identifying keywords from the textual contents of the webpage using a novel weighted URL tokens system based on the N-gram model. Adebowale M. A. et al. [4] use the CNN and LSTM techniques, as a combined classifier in a novel approach called the IPDS, which is able to perform deep analysis of both images and text. The proposed IPDS produced a classification accuracy of 93.28% using one million legitimate and phishing URLs. Sahingoz, O.K. et al. [5] use seven different classification algorithms and NLP-based features using a large dataset with 36,400 legitimate URLs and 37,175 phishing URLs. The proposed approach reached a high accuracy of 97.98% using the Random Forest algorithm.

3. Machine Learning

Phishing detection is a classification problem. Classification is a supervised machine learning (ML) approach that is used to identify and categorize objects into classes. In our research we use binary classification which has two output classes, one class is normal state and the second class is abnormal state. For each class we assign a label, the class for the normal state, which is the phishing one, will be assigned with the class label 0, and the class for the abnormal state, which is not phishing, will be assigned with the class label 1.

There are a various machine learning classification models and tools that can be used to detect phishing websites, I'll present the following ones:

- **Support Vector Machines**
- **Logistic Regression**
- **K-Nearest Neighbors**
- **Decision Tree**
- **Gradient Boosting**
- **Light Gradient Boosting Machine**
- **eXtreme Gradient Boosting**
- **Categorical Boosting**
- **Neural Networks**
- **Multilayer Perceptrons**

3.1 Support Vector Machines

SVM (Support Vector Machines) - The Support Vector learning machine was developed by Vapnik et al. (Scholkopf et al. 1995 [6], Scholkopf 1997 [7]) to constructively implement principles from statistical learning theory (Vapnik, 1998 [8]), The machine conceptually implements the following idea: input vectors are non-linearly mapped to a very high dimensional feature space. In this feature space, a linear decision surface is constructed [9]. In principle, the main goal of SVM is to find an optimal hyperplane(s) that separates the classes; the optimal hyperplane has the maximum distance from both classes (Figure 3.1).

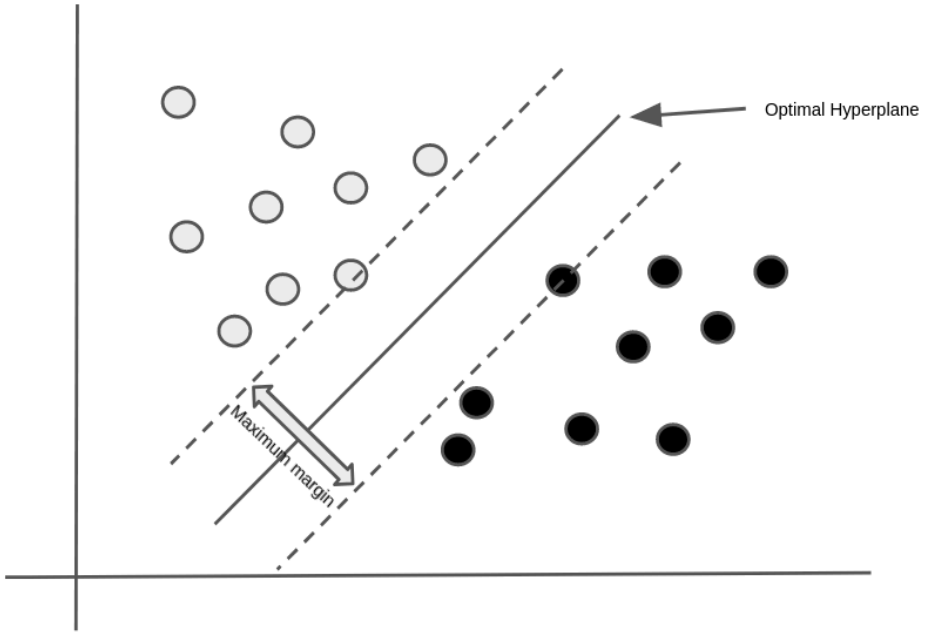


Figure 3.1: Optimal Hyperplane and Maximum Margin

The SVM performs both linear classification and nonlinear classification. The nonlinear classification is performed using the Kernel function.

The Support Vector learning machine is used mostly for classification problems. Popular for test classification due to high speed and performance, it can easily handle multiple categories based on the training set provided.

3.2 Logistic Regression

Logistic Regression (LR) - A machine learning technique that is mainly used to solve classification problems. This technique includes dependent variables (response variable) which can be signified in the binary values: '0' or '1', Negative or Positive, and transforms its output using the logistic sigmoid function (Figure 3.2) for returning the probability value which can be mapped to two classes. It is similar to the Linear Regression model except that the Logistic Regression predicts into two classes using the logistic sigmoid function instead of the linear function. The uniqueness of this function is that it maps all real numbers $\mathbb{R}$ to range [0, 1] [10]. Since logistic regression estimates probabilities, it uses Maximum Likelihood Estimation (MLE) to fit the curve.

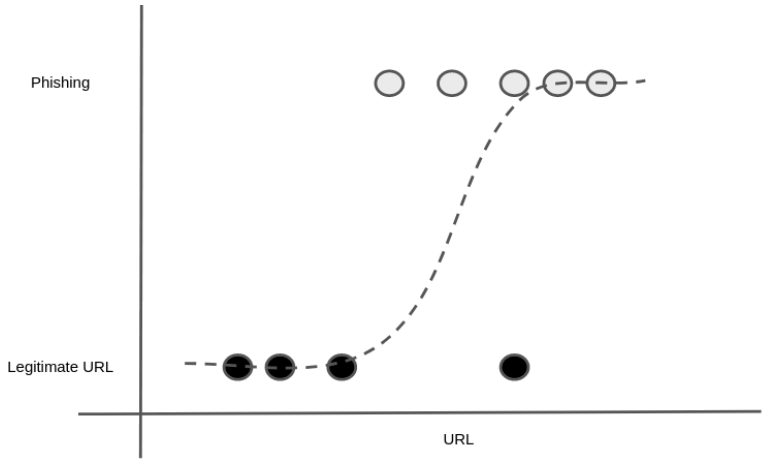


Figure 3.2. The sigmoid function for logistic regression.

3.3 K-nearest neighbors

K-nearest neighbors (KNN) - A supervised machine learning method that was developed in 1951. KNN is one of the most significant classification algorithms and can be used also for regression tasks. It is based on distances and makes predictions by calculating the distance between the test data and the k nearest training points. After calculating the distances, the relevant class is selected by specifying the closest K nearest neighbors to the query (Figure 3.3). KNN can handle large datasets, it's a simple and intuitive algorithm. It's also a non-parametric algorithm which means that it doesn't make assumptions regarding the dataset, and can discover hidden relations in the data. On the other hand, K-NN is a slow algorithm since it has to calculate the distances between the query point and all the other data points in the dataset. As KNN is a non-parametric algorithm, it attaches equal weight for all the features. The "K" value must be set when using the KNN algorithm, but there is no formula/method to find the most optimal value of "K".

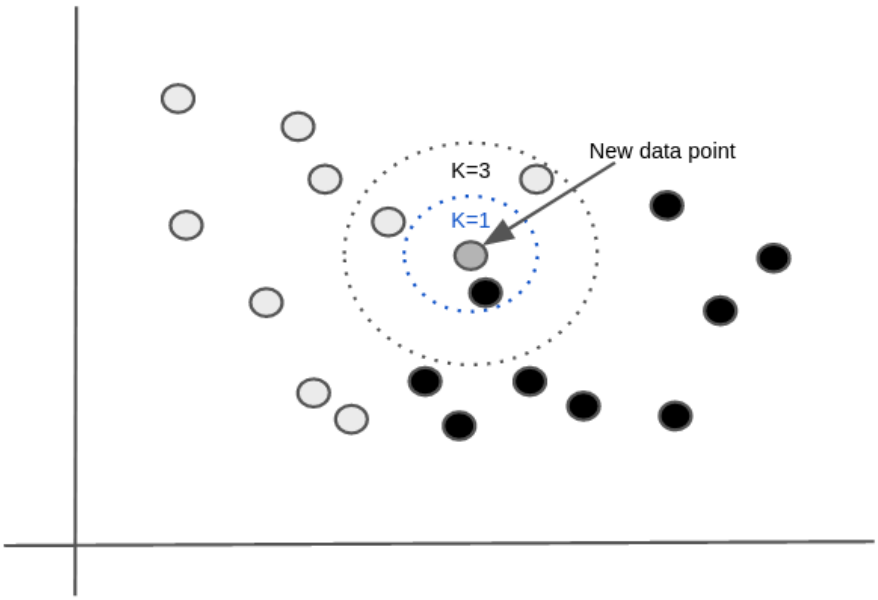


Figure 3.3. K-nearest neighbors - K=1 vs K=3.

3.4 Decision Tree

Decision Tree (DT) - A supervised machine learning method that performs both regression and classification tasks, it is simple and easy to understand. The aim is to predict the target value variable based on the input variables. DT structure contains a root node, internal nodes (none leaf), and leaf nodes. The root node and the internal nodes are assigned to an input feature and selected based on Attribute Selection Measure(ASM), while the leaf nodes are assigned to one class representing the most appropriate target value [11]. Figure 3.3 illustrates the basic DT structure.

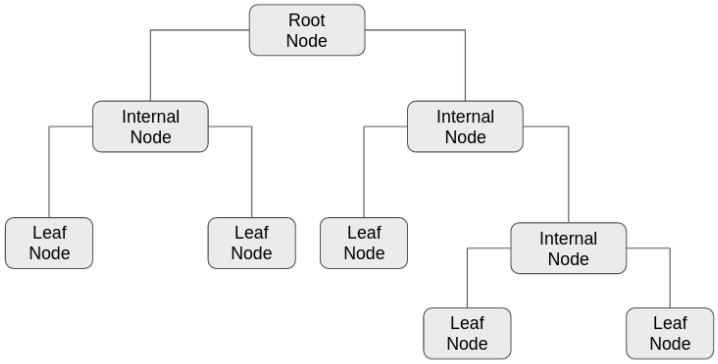


Figure 3.3. Basic Decision Tree Structure.

Figure 3.4 shows an example of a decision tree that trained on the UCI phishing dataset. Each internal node in the decision tree represents a feature. Each sub-tree in the decision tree represents conjunctions of features. The leaves in the decision tree represent the predicted values (Phishing/Legitimate).

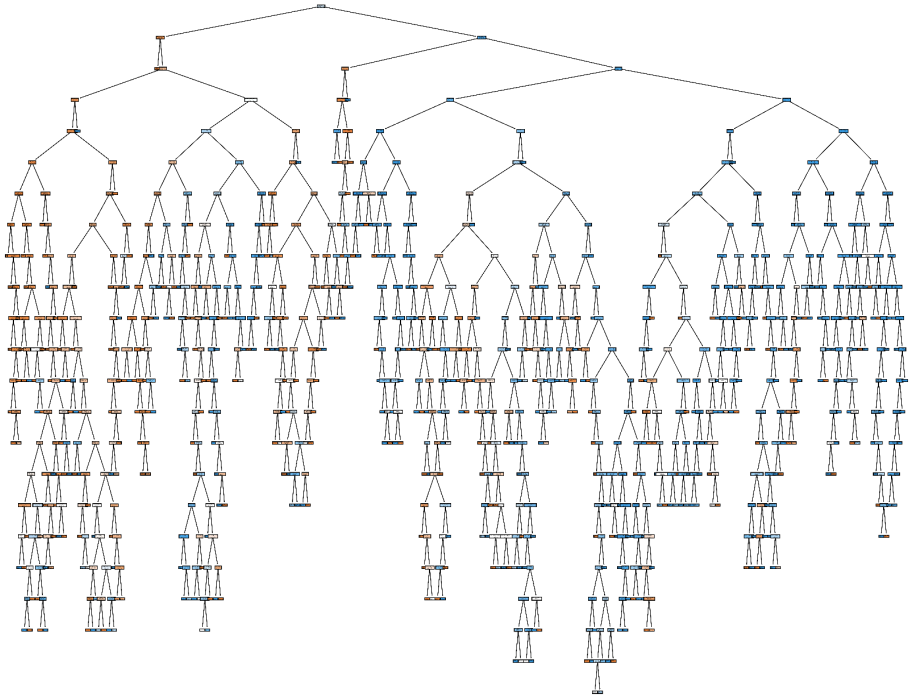


Figure 3.4. DT trained on the phishing dataset features

3.5 Gradient Boosting

Gradient Boosting (GB) - A machine learning method that performs both regression and classification tasks. To improve the model and get a better one, a single weak model is combined with multiple other weak models. Gradient boosting uses a loss function depending on the type of task, and it is based on decision trees as weak models. It is possible to limit the weak models in different ways to ensure that the models remain weak. An additive strategy is used in gradient boosting, which means adding only one new tree at a time in order to correct the errors of the previously fitted one in addition to keeping the existing trees unchanged. Gradient Boosting is very

flexible and can optimize different loss functions and support several hyperparameter tuning options, it doesn't require pre-processing data and works great with categorization, but can be computationally expensive, which makes high memory and time consumption.

3.6 Light Gradient Boosting Machine

LightGBM (LGBM) - LGBM is one of the fastest frameworks developed in Microsoft. It uses tree-based learning algorithms and provides a highly efficient implementation of gradient boosting along with lower memory consumption. LGBM supports parallel, distributed, GPU learning, can handle large-scale data, and provides good accuracy compared to other models. The framework supports many features, including sparse optimization, multiple loss functions, parallel training, regularization, bagging, and early stopping. Unlike most decision tree learning algorithms, LGBM grows trees leaf-wise which chooses the leaf with max delta loss instead of level-wise which chooses the whole level. Leaf-wise algorithms tend to achieve lower loss than level-wise algorithms.

3.7 eXtreme Gradient Boosting

eXtreme Gradient Boosting (XGB) - XGB is an open-source library that provides an implementation of gradient boosting framework and parallel tree boosting (GBM, GBDT) which is used for regression and classification tasks. XGB is an ensemble learning algorithm that combines several models for increasing predictiveness and improving overall performance, in addition, it's a scalable distributed decision tree and has been extensively used in machine learning in recent years. XGB is very fast

compared to other gradient boosting, it supports CPU parallelization, cache optimization, and distributed computing, which is a big advantage for large datasets.

3.8 Categorical Boosting

Categorical Boosting (CatBoost) - A high-performance framework for gradient boosting on decision trees that was developed by Yandex in 2017 [12]. Unlike LightGBM and XGBoost, the main advantages of CatBoost are high performance with little parameter tuning and handling of categorical variables. The framework implements the gradient boosting algorithm on GPUs and the scoring algorithm on CPUs, which are significantly faster than other gradient boosting frameworks [13]. It builds blanched trees which decreases prediction time [14]. CatBoost can work with multiple data types to solve a wide range of tasks.

3.9 Neural Networks

Neural Networks (NNs) - Neural Networks, also known as artificial neural networks (ANNs), is a computational model that is inspired by biological neural networks. The model consists of neurons and connections between them, which are divided into three main types of layers: an input layer, hidden layers, and an output layer. Each neuron in the hidden layers and in the input layer has one or more inputs and based on inputs the neuron yields a single output that is sent to one or more other neurons in the hidden or output layers. The difference between the neurons in the hidden layer and the neurons in the input layer is that the neurons in the input layer received external data while the neurons in the input layer received the data from the previous layer. Each neuron in the output layer is responsible for producing the final result.

Broadly speaking, there are two types of NNs: Feedforward Neural Network (FNN) and Recurrent Neural Network (RNN), FNN is uni-directional and RNN is bi-directional. Uni-directional means that the output data of each neuron in the current layer moves only forward to the next layer, while in bi-directional the output data can move forward to the next layer, backward to the previous layer, or move to the neuron in the same layer which allows cycles or loops. Figure 3.5 illustrates FNN and RNN types.

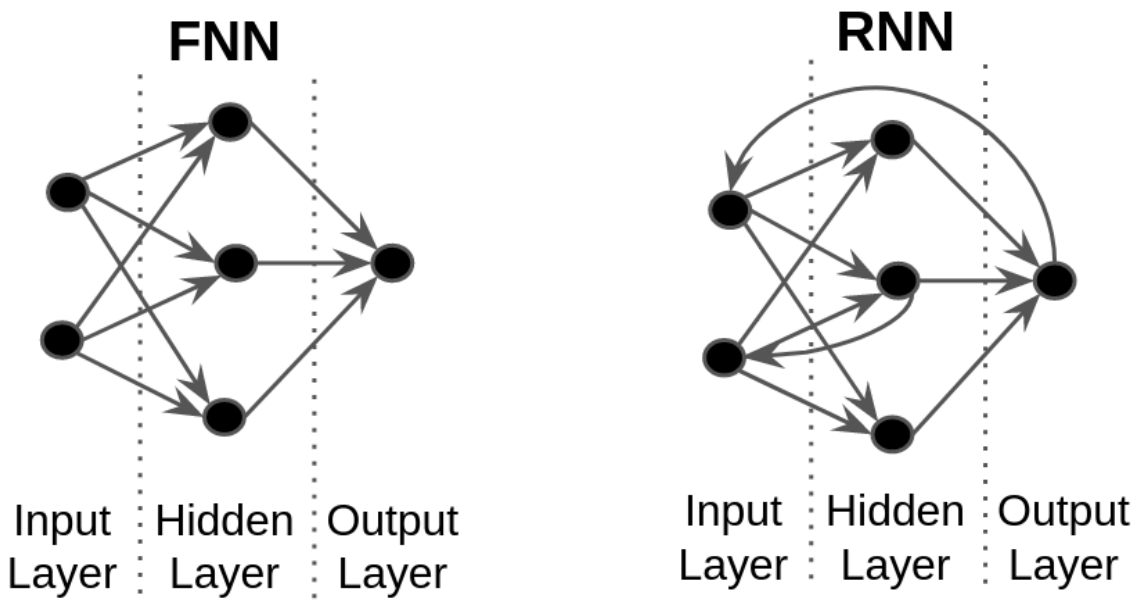


Figure 3.5. FNN vs RNN

3.10 Multilayer Perceptrons

Multilayer Perceptrons (MLP) - A supervised learning algorithm, based on a fully connected Feedforward Neural Network with at least three layers: an input layer, one or more hidden layer, and an output layer. It supports linear and non-linear activation functions. It can handle large datasets and handle complex tasks. It quickly predicts the output after training the model and thanks to the backpropagation algorithm that is used in MLP, it increases accuracy and reduces prediction error.

4. The Dataset

The dataset has been taken from the UCI machine learning repository [15], which is provided in ARFF format and converted to CSV format for easy manipulation. The dataset focuses on some important features that help us to consider if a specific URL is legitimate or phishing. The provided dataset includes 11055 instances which represent URLs with 30 extracted features and are collected mainly from PhishTank archive, MillerSmiles archive, and Google searching operators. The dataset contains 4898 instances of phishing ULRs and 6157 instances of legitimate URLs. Each instance in the dataset has a binary target variable, the “Result” field, when assigned with a “-1” value indicates the instance as a phishing URL, and the “1” value indicates the instance as a legitimate URL. The rest of the dataset features can have {1, 0, -1} as a value, where 1 will be considered a legitimate URL, 0 will be considered a suspicious URL, and -1 will be considered a phishing URL. The feature itself is classified as legitimate, suspicious, or phishing based on the entire dataset analysis as described below.

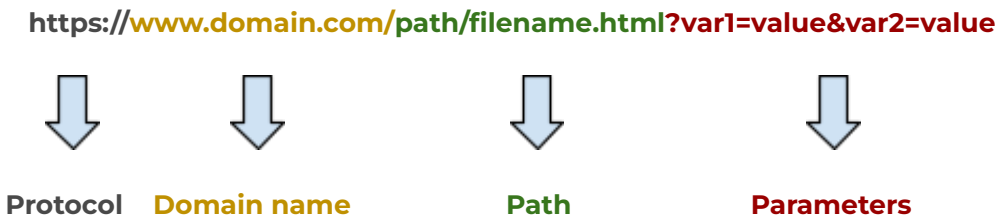
The dataset features are divided into four families:

- **Address Bar Based Features**
- **Abnormal Based Features**
- **HTML and JavaScript Based Features**
- **Domain Based Features**

4.1 Address Bar Based Features

The Address bar is the place in the web browser that shows the current Uniform Resource Locator (URL), and identifies the location of the resources on the remote or local server.

An example of a URL:



The Address Bar based Features has the following 12 features:

4.1.1 having_IP_Address (0)

having_IP_Address attribute indicates that the URL uses an IP address in the domain name, for example, http://125.98.3.123/index.html. The dataset analysis shows that most websites that use IP addresses in the domain name are phishing websites, therefore, this feature will be marked as phishing. The IP address can be IPv4/IPv6 and sometimes even transformed to hexadecimal code as shown in this example: http://0x58.0xCC.0xCA.0x62/2/index.html

This feature can have the following values: { -1, 1}

4.1.2 URL_Length (1)

A *URL_Length* attribute indicates the length of the URL. A long URL is intended to hide a suspicious part in some cases. The dataset analysis shows that most phishing URLs have a length of 76 characters and up while most legitimate URLs have less than 54 characters, therefore, this feature will be marked as phishing if the URL has a length of

76 characters and up. Suspicious when the URL has a length between 54 and 75 characters. Legitimate when the URL has a length of less than 54 characters. The length range was determined after calculating the length of URLs in the dataset and producing an average URL length.

This feature can have the following values { 1, 0, -1 }

4.1.3 Shortining_Service (2)

A *Shortining_Service* attribute indicates that the URL shortening technique "TinyURL" has been used on the source URL. This technique provides short aliases for redirection of long URLs. Using this technique can help phishers hide URLs that use long lengths or URLs that use IP addresses instead of domain names. Based on the dataset analysis, the majority of websites that use this shortening technique are phishing websites. Therefore, this feature will be marked as phishing whenever a shortening service is used on the URL.

For example, the URL "http://125.98.3.123/2/paypal.ca/index.html" can be shortened to "https://tinyurl.com/ybyn57tp" when using https://tinyurl.com/ service.

This feature can have the following values: { 1, -1 }.

4.1.4 having_At_Symbol (3)

having_At_Symbol attribute indicates that the URL contains the "@" symbol. When using the "@" symbol in a URL the browser ignores all characters before this symbol. The dataset analysis shows that most websites that have the "@" symbol in the URL are phishing, therefore this feature will be marked as phishing when the URL contains the "@" symbol.

This feature can have the following values: { 1, -1 }.

4.1.5 double_slash_redirecting (4)

A *double_slash_redirecting* attribute indicates that the URL contains "//" (double slash), which means redirection to another server. Based on the dataset analysis, the majority of websites that use redirection are phishing websites. Therefore, this feature will be marked as phishing when "//" is used in a URL, except after "http:" or "https:".

This feature can have the following values: { 1, -1 }.

4.1.6 Prefix_Suffix (5)

A *Prefix_Suffix* attribute indicates that the URL contains "-" (dash symbol) in the domain name section. The dash symbol is rarely used in legitimate URLs, therefore, this feature will be marked as phishing when the URL has "-" in the domain name.

This feature can have the following values: { 1, -1 }.

4.1.7 having_Sub_Domain (6)

having_Sub_Domain attribute indicates the number of subdomains in the URL. The number of subdomains can be calculated by summing up the number of dots in a URL, excluding the first dot after the "www", and the country-code top-level domain (ccTLD). This feature will be marked as phishing, if the total number of dots is greater than two. In case the total number of dots equals two, this feature will be marked as suspicious, and if the total number of dots is lower, this feature will be marked as legitimate.

This feature can have the following values: { -1, 0, 1 }.

4.1.8 SSLfinal_State (7)

An *SSLfinal_State* attribute indicates that the URL uses "https" (Hypertext Transfer Protocol Secure) protocol and uses Secure Sockets Layer (SSL) encryption. Relying only on having "https" in the URL is not sufficient, checking the certificate age and issuer is

important. This feature will be marked as legitimate if the URL uses “https”, has a trusted issuer, and has a certificate age that is one year old or greater. This feature will be marked as suspicious if the URL uses https but the issuer is not trusted. In all other cases, it will be marked as phishing.

This feature can have the following values: { -1, 1, 0 }.

4.1.9 Domain_registration_length (8)

A *Domain_registration_length* attribute indicates the period of time that the domain lives. Based on the dataset, the domains that were found as phishing URLs have been used for up to one year. Therefore, if the domain expiration date is greater than a year, this feature will be marked as legitimate, otherwise, it will be marked as phishing.

This feature can have the following values: { -1, 1 }.

4.1.10 Favicon (9)

A *Favicon* attribute indicates where the favicon came from, this feature will be marked as phishing if the favicon that is displayed in the URL came from another domain, otherwise, this feature will be marked as legitimate. A favicon (favorite icon) is a small icon used by web browsers to represent a website and helps identify the website in the address bar or the tabs.

This feature can have the following values: { 1, -1 }.

4.1.11 port (10)

A *port* attribute indicates whether the following ports are opened or closed:

21 - FTP, 22 - SSH, 23 - Telnet, 445 - SMB, 1433 - MSSQL, 1521 - ORACLE, 3306 - MySQL, 3389 - RDP. This feature will be marked as phishing If port # is open, otherwise, this feature will be marked as legitimate. Open ports can indicate if a service in a system is

able to communicate. The best practice is to open only the ports that are required for the server/system, the more open ports there are, the easier it will be to penetrate the server/system.

This feature can have the following values: { 1, -1 }.

4.1.12 HTTPS_token (11)

An *HTTPS_token* attribute indicates if the "HTTPS" token has been added to the domain part of the URL, if so, this feature will be marked as phishing.

For example, http://**https**-www-paypal-it-webapps-home.soft-hair.com/

This feature can have the following values: { -1, 1 }.

4.2 Abnormal Based Features

Abnormal Based Features represent abnormal behavior on a particular web page. The Abnormal Based Features have the following 6 features:

4.2.1 Request_URL (12)

A *Request_URL* attribute indicates if images, videos, and sound objects from the requested URL are loaded from another domain. This feature will be marked as legitimate If over 50% of the objects are loaded from within the same domain of the URL. Otherwise, it will be marked as legitimate.

This feature can have the following values: { 1, -1 }.

4.2.2 URL_of_Anchor (13)

A *URL_of_Anchor* attribute indicates whether over 67% of the <a> tag (anchor) defines a hyperlink that starts with '#' or 'javascript' or points to a different domain, in such case, this feature will be marked as phishing. This feature will be marked as suspicious If the number of the <a> tag that defines a hyperlink to a different domain is between 31% and 67%, otherwise, this feature will be marked as legitimate.

This feature can have the following values: { -1, 0, 1 }.

4.2.3 Links_in_tags (14)

Links_in_tags attribute indicates if over 81% of the <Meta>, <Script>, and <Link> tags are linked to a different domain, in this case, this feature will be marked as phishing. This feature will be marked as suspicious when the number of the <Meta>, <Script>, and <Link> tags are linked to a different domain, is between 17% and 81%, otherwise, this feature will be marked as legitimate.

This feature can have the following values: { -1, 0, 1 }.

4.2.4 SFH (15)

An *SFH* attribute indicates if the action attribute in an HTML form is an empty string or "about:blank", in that case, this feature will be marked as phishing, if the action attribute redirects to another domain, this feature will be marked as suspicious, otherwise will be marked as a legitimate. Server Form Handler (SFH) is an HTML form for submitting data to a form-handler. The form-handler is typically a file on the server with a script for processing input data. The user input is most often sent to a server for processing.

This feature can have the following values: { -1, 0, 1 }.

4.2.5 Submitting_to_email (16)

Submitting_to_email attribute indicates whether the action attribute in an HTML form starts with “mailto:” or mail() function in PHP, is used. If so, this feature will be marked as phishing, otherwise, it will be marked as legitimate. This can help phishers to redirect users' personal information to their email.

This feature can have the following values: { -1, 1 }.

4.2.6 Abnormal_URL (17)

An *Abnormal_URL* attribute indicates whether the hostname is not included in the URL, if so this feature will be marked as phishing, otherwise, it will be marked as legitimate.

This feature can have the following values: { -1, 1 }.

4.3 HTML and JavaScript Based Features

HTML and JavaScript Based Features represent features that can be used in JavaScript code, in the HTML code, or a combination of both. The HTML and JavaScript based Features have the following 5 features:

4.3.1 Redirect (18)

A *Redirect* attribute indicates if the URL has been redirected more than once, if so this feature will be marked as suspicious, otherwise will be marked as legitimate.

This feature can have the following values: { 0, 1 }.

4.3.2 on_mouseover (19)

An *on_mouseover* attribute indicates if there are any changes in the windows status bar especially when using the “onMouseOver'' event. For security reasons window.status has been disabled in most browsers, therefore if window.status is enabled this feature will be marked as phishing, otherwise, will be marked as legitimate. Using this technique can help phishers hide the fake URL links when the mouse is hovering over them.

This feature can have the following values: { 1, -1 }.

4.3.3 RightClick (20)

A *RightClick* attribute indicates if the mouse right-click button is disabled on the website, if so this feature will be marked as phishing, otherwise as legitimate. Using this technique can help phishers to block users from looking at the source code of the webpage.

This feature can have the following values: { 1, -1 }.

4.3.4 popUpWidnow (21)

A *popUpWidnow* attribute indicates if the webpage uses a form in a pop-up window, if so, this feature will be marked as phishing, otherwise as legitimate. Pop-up windows are commonly used, but they are rarely used to collect personal information.

This feature can have the following values: { 1, -1 }.

4.3.5 Iframe (22)

An *Iframe* (inline frame) attribute indicates whether the webpage uses an iframe tag and sets the border to be transparent, if so this feature will be marked as phishing, otherwise it will be marked as legitimate. The iframe HTML element represents a nested browsing context, embedding another HTML page into the current one. Having

a transparent border can mislead users into believing that they are visiting a legitimate website without them realizing that it is not the real website that they are visiting.
This feature can have the following values: { 1, -1 }.

4.4 Domain Based Features

Domain Based Features represent features regarding the domain itself. Domain based Features has the following 7 features:

4.4.1 age_of_domain (23)

An *age_of_domain* attribute indicates if the URL domain name has been created in the last 6 months, if so this feature will be marked as phishing, otherwise it will be marked as legitimate. Most phishing websites live for a short period of time, while the domain age of most legitimate websites is a minimum of 6 months.
This feature can have the following values: { -1, 1 }.

4.4.2 DNSRecord (24)

A *DNSRecord* attribute indicates if the URL has a record in the WHOIS database, which is able to extract data for all the popular TLDs (com, org, net, etc.). If there is a DNS record, this feature will be marked as legitimate, if the DNS record is empty or not found this feature will be marked as phishing.
This feature can have the following values: { 1, -1 }.

4.4.3 web_traffic (25)

A *web_traffic* attribute indicates whether the “Website Rank” in the Alexa database is less than 100,000, which means it is in the top 100,000 most visited websites. If so this feature will be marked as legitimate. If the URL is not recognized or has no traffic, this feature will be marked as phishing, otherwise, it will be marked as suspicious. Since phishing websites live for a short period of time, they may not be recognized by the Alexa database.

This feature can have the following values: { -1, 0, 1 }.

4.4.4 Page_Rank (26)

A *Page_Rank* attribute indicates whether the website is important or not, page rank is a value between “0” and “1”. A greater page rank means an important website that has many links pointing to it from other websites. This feature will be marked as phishing if the page rank is less than 0.2, otherwise, it will be marked as legitimate.

This feature can have the following values: { 1, -1 }.

4.4.5 Google_Index (27)

A *Google_Index* attribute indicates whether the URL has been indexed by Google, which means that Google verified the webpage and it can now be displayed in Google search results. This feature will be marked as legitimate if the URL is indexed by Google, otherwise, it will be marked as phishing.

This feature can have the following values: { 1, -1 }.

4.4.6 Links_pointing_to_page (28)

A *Links_pointing_to_page* attribute indicates the number of links pointing to a URL, irrespective of whether some of the links are within the same domain. This feature will be marked as phishing if no links are pointing to the website. In case there are only one

or two links pointing to the URL, this feature will be marked as suspicious, otherwise will be marked as legitimate.

This feature can have the following values: { 1, 0, -1 }.

4.4.7 Statistical_report (29)

A *Statistical_report* attribute indicates if the particular URL exists in PhishTank or StopBadware and is verified as a phishing URL, based on the “Top 10 Domains” and “Top 10 IPs” of PhishTank and “Top 50” IP addresses of StopBadware, if so, this feature will be marked as phishing, otherwise, will be marked as legitimate.

This feature can have the following values: { -1, 1 }.

5. Experiments and Implementation

This chapter explains and presents the experiments, and tool implementations along with the results of the experiments.

5.1 Running Models on the Dataset

For the experiments I wrote a Python program [16] to load the UCI dataset that is described in Chapter 4, and split it into a train and test where the test size is 30% of the whole dataset. Next, I ran the following models on the dataset in order to compare the accuracy between them:

- Neural Network using Sequential model type
- Support Vector Machine (SVM) Using a polynomial kernel
- Logistic Regression(LR)
- K-Nearest Neighbors (KNN) Using K=3
- LightGBM (LGBM)
- Decision Tree (DT)
- Gradient Boosting (GB)
- eXtreme Gradient Boosting (XGB)
- Catboost
- Multilayer Perceptrons (MLP)

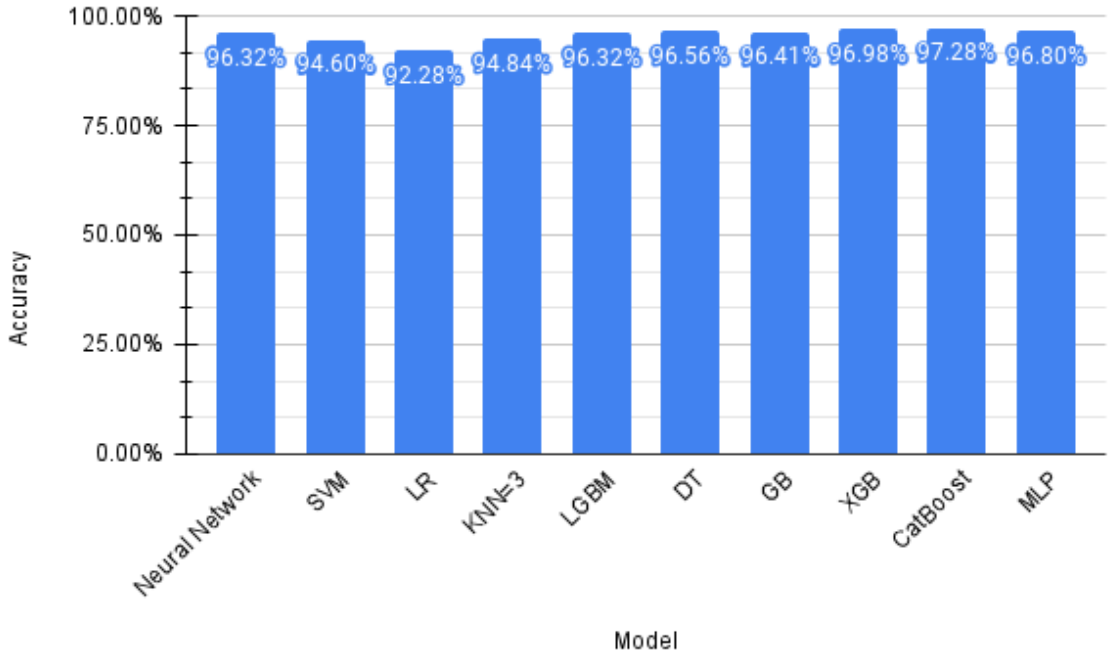


Figure 5.1. Accuracy per model

Figure 5.1 summarizes the accuracy measures for the models mentioned above. As shown in the chart the CatBoost provides the highest accuracy which is 97.28%, while the logistic regression model provides the lowest accuracy with 92.28%. Even though the gap between the highest and lowest is 5%, all the models are quite good and provide accuracy higher than 92%.

5.2 Accuracy Improvement Using Sequential Feature Selection

Feature Selection (FS) - Feature selection is a technique for reducing the number of features from the original data set and using them as inputs in a machine learning model. The main aim is to choose a subset of the relevant features from the original features, the central premise is that the dataset contains some redundant or irrelevant

features, which can lead to wrong use of resources, poor performance, and in some cases loss of accuracy.

As we saw in the previous paragraph (5.1), the accuracy range of the models is between 92% and 98%, using the UCI dataset which contains 30 features. The purpose of this experiment is to use the feature selection technique and reduce the feature sets by removing redundant features, thus improving accuracy (or at least keeping the existing one), along with performance.

For that, I used the Sequential Feature Selector (SFS) of *mlxtend* module [17]. The SFS is a group of greedy algorithms that adds or removes features from the currently selected subset of features; these algorithms are used to reduce an initial d-dimensional feature space to a k-dimensional feature subspace where $k < d$. At each iteration, the selector algorithm chooses the best feature to add or remove based on the cross-validation score of an estimator. The algorithm performs forward selection for adding features or backward selection for removing features, when, in forward selection the initial set is empty, and in backward selection, the initial set is the whole group. The result and the performance of forward and backward are unequal. The backward method is preferred in terms of performance if the size of the subset of features is close to the original, which means that we prefer to remove features and not add them.

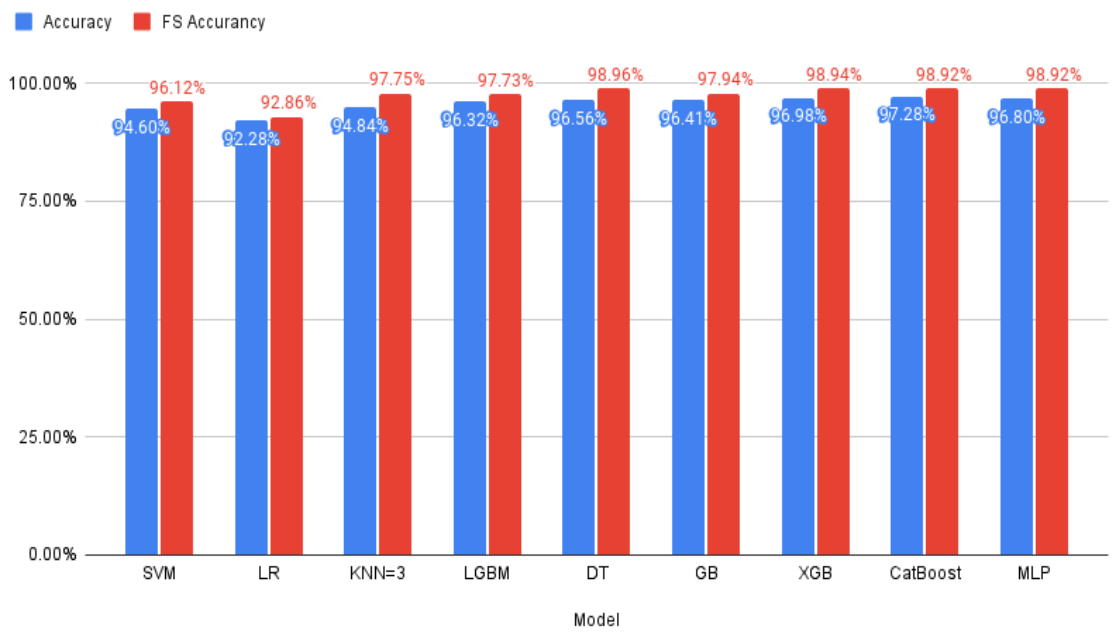


Figure 5.2. Accuracy comparison FS vs. none FS per model

Figure 5.2 shows accuracy comparison per model when feature selection is used vs. not used. The chart shows that the accuracy value has improved between 0.5% and 3%. The highest accuracy value when using feature selection was obtained in the Decision Tree model, where the least amount of features were selected - 23 (out of 30 features) with an accuracy value of 98.96%, while the lowest one remains in the Logistic Regression model which is 92.86%. The highest improvement was obtained in the k-nearest neighbors model, the accuracy value increased by 2.91%. In all the models the accuracy value has increased and the number of features has been reduced.

Model	SFS Accuracy	Number of Features	Feature Number [4]
SVM	96.12%	23	'0', '1', '2', '4', '5', '6', '7', '8', '11', '13', '14', '15', '16', '17', '19', '22', '23', '24', '25', '26', '27', '28', '29'
LR	92.86%	28	'0', '1', '2', '3', '4', '5', '6', '7', '8', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '26', '27', '28', '29'
KNN-3	97.75%	28	'0', '1', '2', '4', '5', '6', '7', '8', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29'
LGBM	97.73%	26	'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '19', '20', '23', '24', '25', '26', '27', '28', '29'
DT	98.96%	23	'0', '1', '2', '3', '5', '6', '7', '8', '11', '12', '13', '14', '15', '16', '18', '21', '23', '24', '25', '26', '27', '28', '29'
GB	97.94%	26	'0', '1', '2', '3', '4', '5', '6', '7', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '23', '24', '25', '26', '27', '28', '29'
XGB	98.94%	25	'0', '1', '2', '3', '4', '5', '6', '7', '8', '10', '11', '12', '13', '14', '15', '16', '17', '18', '23', '24', '25', '26', '27', '28', '29'
CatBoost	98.92%	24	'0', '1', '2', '3', '5', '6', '7', '8', '11', '12', '13', '14', '15', '16', '18', '19', '20', '23', '24', '25', '26', '27', '28', '29'
MLP	98.92%	25	'0', '1', '2', '3', '4', '5', '6', '7', '8', '11', '12', '13', '14', '15', '16', '18', '21', '22', '23', '24', '25', '26', '27', '28', '29'

Table 5.3. Accuracy and number of selected features per model

Table 5.3 displays the number of features used and elaborates which features were required to achieve the highest accuracy value for each model using a Sequential Feature Selector, in a Forward Selection mode. LR model received the lowest accuracy (marked in red), the lowest accuracy obtained in this model was only 92.86% and was

obtained while using 28 features, which means it uses almost all features. DT, on the other hand, is the best model to use (marked in green) as the highest accuracy obtained in this model is 98.96% and obtained while only 23 features were selected.

Feature number	Feature name	Unselected
9	Favicon	89%
22	Iframe	55.5%
10	Port	55.5%
17	Abnormal_URL	44.5%
20	RightClick	44.5%
21	popUpWidnow	44.5%
19	on_mouseover	33.3%

Table 5.4. Top unselected features

Table 5.4 shows the top features, their corresponding name, and the percentages of the run models that did not select this feature. The percentage value is established according to Table 5.3, which tested on 9 different models based on the following classifiers: SVM, LR, KNN, LGBM, DT, GB, XGB, CatBoost, and MLP.

Feature number 9, *Favicon*, wasn't selected in 8 models out of 9, which means that this feature can be dropped as its importance is very low. Feature 22, *Iframe*, was selected only by 4 models: SVM, LR, KNN, and MLP, and on most of these models its effect is very low, same to feature 22 is feature 10, *port*, which was also selected only by 4 models: LR, KNN, LGBM and XGB.

5.4 Creating a New Database and Experimenting With It

The experiments were run on the UCI dataset which was created in 2012 by Mohammad, McCluskey, & Thabtah, the dataset contains 30 features that are described in Chapter 4. The dataset does not contain the URL itself, and the implementation of the tool that has been used to collect the features, which makes it difficult to analyze and Interpret. In order to understand the features and their interpretation I created a Python tool [18] that collects URLs from Google search. Each URL that was collected from Google search was marked as a legitimate URL, I also downloaded phishing URLs from phishtank.org website and marked them as phishing URLs. As explained in Chapter 4, the Python tool determines whether each feature on the webpage is legitimate, suspicious, or phishing, and then creates a new dataset based on those findings. Creating and Implementing a new dataset showed me that some of the features that were relevant in 2012 when the database was created are still relevant today, while others should be dropped as they are not relevant anymore. For illustration see the following examples:

Feature 4.1.8 (7) *SSLfinal_State* - This feature has a list of trusted issuers: "GeoTrust, GoDaddy, Network Solutions, Thawte, Comodo, Doster, and VeriSign", although there are many others that considered as trusted and not in the list, for example, Google uses google certification tools, GeoTrust that appears in the list, merged with DigiCert Inc, and https://www.GeoTrust.com itself uses DigiCert Inc, which means that DigiCert Inc is also a trusted issuer. The certification expiration dates of google.com, cisco.com (Figure 5.6), github.com, and GeoTrust.com (Figure 5.7) were also randomly checked, all of them are legitimate URLs and the expiration date is in

less than one year, which is considered as not legitimate URL. It seems like certification renewal should be done every year.

```
{'subject': ((('jurisdictionCountryName', 'US'),),
(('jurisdictionStateOrProvinceName', 'Utah'),), (('businessCategory',
'Private Organization'),), (('serialNumber', '5299537-0142'),),
(('countryName', 'US'),), (('stateOrProvinceName', 'Utah'),),
(('localityName', 'Lehi'),), (('organizationName', 'DigiCert,
Inc.'),), (('commonName', 'digicert.com'),)), 'issuer':
((('countryName', 'US'),), (('organizationName', 'DigiCert Inc'),),
(('organizationalUnitName', 'www.digicert.com'),), (('commonName',
'DigiCert SHA2 Extended Validation Server CA'),)), 'version': 3,
'serialNumber': '0533363CD858644F5CC31F9ABAD5D542', 'notBefore': 'Dec
11 00:00:00 2022 GMT', 'notAfter': 'Dec  9 23:59:59 2023 GMT',
'subjectAltName': (('DNS', 'digicert.com'), ('DNS',
'www.digicert.com'), ('DNS', 'content.digicert.com'), ('DNS',
'websecurity.digicert.com'), ('DNS', 'www.websecurity.digicert.com'),
('DNS', 'knowledgehub.digicert.com'), ('DNS', 'kh.digicert.com'),
('DNS', 'knowledge.digicert.com'), ('DNS',
'knowledgebase.digicert.com'), ('DNS', 'kb-internal.digicert.com'),
('DNS', 'thawte.com'), ('DNS', 'www.thawte.com'), ('DNS',
'thawte.de'), ('DNS', 'www.thawte.de'), ('DNS', 'thawte.fr'), ('DNS',
'www.thawte.fr'), ('DNS', 'geotrust.com'), ('DNS',
'www.geotrust.com'), ('DNS', 'rapidssl.com'), ('DNS',
'www.rapidssl.com'), ('DNS', 'freessl.com'), ('DNS',
'www.freessl.com')), 'OCSP': ('http://ocsp.digicert.com',),
'caIssuers':
('http://cacerts.digicert.com/DigiCertSHA2ExtendedValidationServerCA.c
rt',), 'crlDistributionPoints':
('http://crl3.digicert.com/sha2-ev-server-g3.crl',
'http://crl4.digicert.com/sha2-ev-server-g3.crl')}
```

Figure 5.6. Certification details of https://www.GeoTrust.com

```
{'subject': ((('commonName', 'www.cisco.com'),), (('organizationName',
'Cisco Systems Inc.'),), (('localityName', 'San Jose'),),
(('stateOrProvinceName', 'California'),), (('countryName', 'US'),)),
'issuer': ((('countryName', 'US'),), (('organizationName',
'IdenTrust'),), (('organizationalUnitName', 'HydrantID Trusted
Certificate Service'),), (('commonName', 'HydrantID Server CA O1'),)),
'version': 3, 'serialNumber': '4001869EECC535ABD516D1156F788630',
'notBefore': 'Mar  1 20:46:02 2023 GMT', 'notAfter': 'Feb 29 20:45:02
```

```
2024 GMT', 'subjectAltName': (('DNS', 'www.cisco.com'), ('DNS',
'www.static-cisco.com'), ('DNS', 'www-cloud-cdn.cisco.com'), ('DNS',
'www.mediafiles-cisco.com'), ('DNS', 'www-dev-cloud-cdn.cisco.com'),
('DNS', 'www-stage-cloud-cdn.cisco.com')), 'OCSP':
('http://commercial.ocsp.identrust.com',), 'caIssuers':
('http://validation.identrust.com/certs/hydrantidcaO1.p7c',),
'crlDistributionPoints':
('http://validation.identrust.com/crl/hydrantidcao1.crl',)}
13-Sep-23 11:43:02 -  - Issue IdenTrust is not in trusted list:
['GeoTrust', 'DigiCert Inc', 'GoDaddy', 'Network Solutions', 'Thawte',
'Comodo', 'Doster', 'VeriSign']
```

Figure 5.7. Certification details of https://www.cisco.com

Feature 4.1.10 (9) *Favicon* - Websites can have a few favicons, in some of them the domain of the favicon is different from the domain URLs

```
>>> print(favicon.get('https://www.github.com'))
[Icon(url='https://github.githubassets.com/favicons/favicon.png',
width=0, height=0, format='png'),
Icon(url='https://github.githubassets.com/favicons/favicon.svg',
width=0, height=0, format='svg'),
Icon(url='https://github.com/favicon.ico', width=0, height=0,
format='ico'),
Icon(url='https://github.githubassets.com/images/modules/site/social-c
ards/campaign-social.png', width=0, height=0, format='png')]
```

Figure 5.8. Favicon details of www.github.com

As we can see in Figure 5.5, www.github.com is not equal to github.githubassets.com and hence will be considered as phishing.

Feature 4.2.2 (13) *URL_of_Anchor* - Domain can be different and uses redirect, this can be seen in walla.com 99% of the links (<a> tag) in walla.com are pointing to walla.co.il. This is a different domain (“com” vs “co.il”) but in reality, it’s the same website. In most cases, pages in the walla.com domain will be considered as phishing in the *URL_of_Anchor* feature when they should be legitimate.

Feature 4.3.5 (22) Iframe - This feature uses the “frameBorder” attribute to display an additional webpage into one that is currently shown. Phishers use the IFrame tag to make the nested webpage invisible, but the “frameBorder” attribute is obsolete and deprecated in HTML5 and the best practice is to use CSS instead. It means that all the new webpages that use HTML5 will be considered as legitimate URL.

Feature 4.4.3 (25) web_traffic - This feature checks the website rank based on the Alexa database, while implementing and collecting the data for the new dataset, I noticed that the data.alexa.com which was the source of this feature has shut down and does not exist anymore [19], it means that this feature is not relevant anymore.

Feature 4.4.4 (26) Page_Rank - Page Rank was a numerical value that indicated the importance of a webpage. Since Google no longer provides this metric, only the owner of the webpage can determine his Google page rank via the Google Search console.

Feature 4.4.6 (28) Links_pointing_to_page - This feature detects how many external links pointing to the websites. In order to find this value it is required to own the website or use none free tools.

Feature 4.4.7 (29) Statistical_report - Statistical reports were collected from PhishTank and StopBadware between 2010 and 2012. Around 2021, StopBadware stopped working because of copyright restrictions, and PhishTank does not provide statistical reports of the last few years.

The new dataset [20] contains 26 features, with 18,173 records when 12,184 records marked as phishing and 6,025 records marked as legitimate.

Figure 5.9 summarizes the accuracy measures for the models when using the new dataset. As shown in the chart and similar to the test on the UCI dataset, the highest accuracy is obtained when XGB and CatBoost models are used, while the logistic regression model provides the lowest accuracy. All the models except the LR model provide accuracy between 90% and 92.5%.

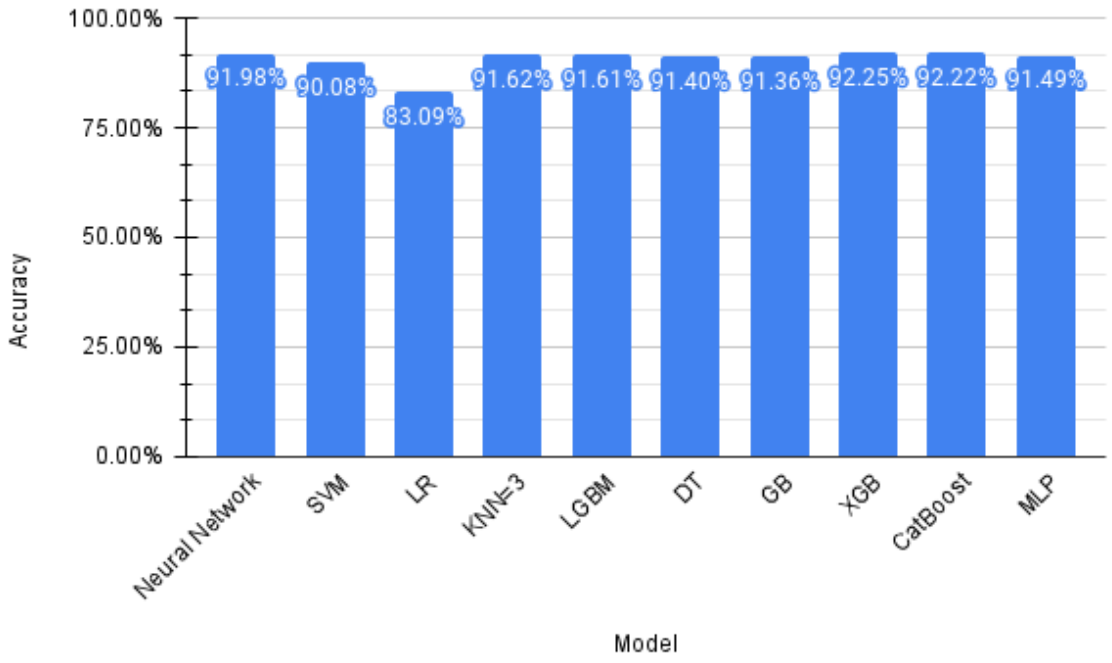


Figure 5.9. Accuracy per model with the new dataset

Figure 5.10 shows accuracy comparison per model with the new dataset when feature selection is used vs. not used. The chart shows that the accuracy value has improved between 0.4% and 2.3%. The highest accuracy value with feature selection was achieved using the Decision Tree model, when 22 features were used, showing an improvement of over 2%. Similar to the results obtained with the UCI dataset, the Decision Tree model achieved the highest accuracy value when using feature selection.

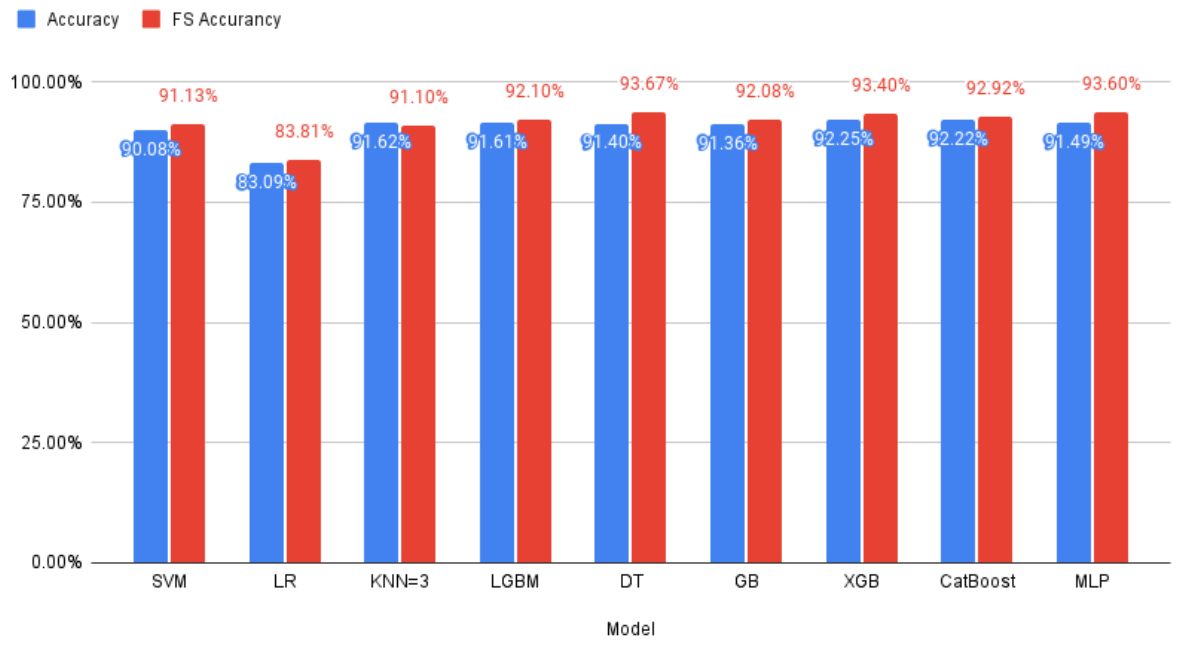


Figure 5.10. Accuracy comparison FS vs. none FS per model with the new dataset

Table 5.11 displays the number of features used and elaborates which features were required to achieve the highest accuracy value for each model using a Sequential Feature Selector, in a Forward Selection mode. LR model received the lowest accuracy (marked in red), the highest accuracy achieved in this model was only 83.81% and was obtained while using 23 features (out of 26). DT, on the other hand, is the best model to

use (marked in green), with the highest accuracy obtained while selecting 22 features (93.67%).

Model	SFS Accuracy	Number of Features	Feature Number [4]
SVM	91.13%	22	'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '12', '13', '14', '15', '17', '18', '21', '22', '23', '24', '25'
LR	83.81%	23	'0', '1', '2', '3', '4', '5', '6', '7', '9', '10', '11', '12', '13', '15', '16', '18', '19', '20', '21', '22', '23', '24', '25'
KNN-3	91.10%	15	'1', '5', '6', '7', '8', '9', '10', '12', '13', '14', '15', '17', '23', '24', '25'
LGBM	92.10%	19	'1', '3', '4', '5', '6', '7', '8', '9', '10', '12', '13', '14', '15', '17', '18', '21', '23', '24', '25'
DT	93.67%	22	'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '12', '13', '14', '15', '17', '18', '21', '22', '23', '24', '25'
GB	92.08%	20	'0', '1', '3', '4', '5', '6', '7', '8', '9', '10', '12', '13', '14', '15', '17', '18', '21', '23', '24', '25'
XGB	93.40%	21	'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '12', '13', '14', '15', '17', '18', '21', '23', '24', '25'
CatBoost	92.92%	22	'1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '12', '13', '14', '15', '16', '17', '19', '20', '21', '23', '24', '25'
MLP	93.60%	21	'1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '17', '18', '21', '23', '24', '25'

Table 5.11. Accuracy and number of selected features per model with the new dataset

Compared to the UCI dataset, the results with the new dataset are less accurate due to the removal of some features, but probably also because the features and their implementation are less suitable for the new websites that were tested.

5.3 Online Machine Learning

5.3.1 Introduction

Online machine learning (also called incremental learning or stream learning) is another type of machine learning model, in which the data is available as a stream in sequential order. Unlike other models mentioned previously, that use batch learning (also called offline learning) techniques where the entire data is available and the models split the dataset into train and test sets, fitting the model on the training batch and predicting accuracy on the test batch, online machine learning perform a sequence of consecutive iterations. On each iteration, the learner first receives an instance and then predicts a label based on the instance data. At the end of the iteration, the learner has determined the correct label. The learner uses the information of each iteration in order to improve his future predictions [21]. In other words, an online machine-learning model can learn from one observation or a small number of observations at a time. It makes the learning stage fast and cheap with the ability to learn new data in real-time as it arrives. Online machine learning is usable for tasks where there are limited computation resources when it is too much to fit the data into the memory, and also when the samples are presented over time. In addition, online machine learning supports model improvement and changes without retraining the entire data required in batch learning. Since online models work on one sample at a time, it means that they are unable to do vectorization.

5.3.2 Online Classification on Phishing Dataset

As we have seen, phishing attacks have increased significantly over time, making it possible to collect a substantial amount of data. Moreover, we have observed that the features have changed over time, and they must be maintained and updated regularly. Based on these facts, online machine learning is a good solution for detecting website phishing attacks. A few online learning tools can be used to achieve this goal. In my experiment, I used the River tool. River is a powerful library for building online machine learning models, it is a result of a merge between creme and scikit-multiflow. It is based on Python and works with Python 3.8 and above. The RIver repository is maintained and improved on a daily basis, it has more than 3,500 commits, 4500 stars, and more than 100 contributors. River doesn't pretend to present the best performance and it mainly focuses on user experience. River has many features such as measuring performance, model selection, model composition, and feature extraction. The dataset used in RIver for this experiment is the same dataset that is described in Chapter 4. Although the dataset is given as a batch I treat it as streaming data since I read each instance at a time.

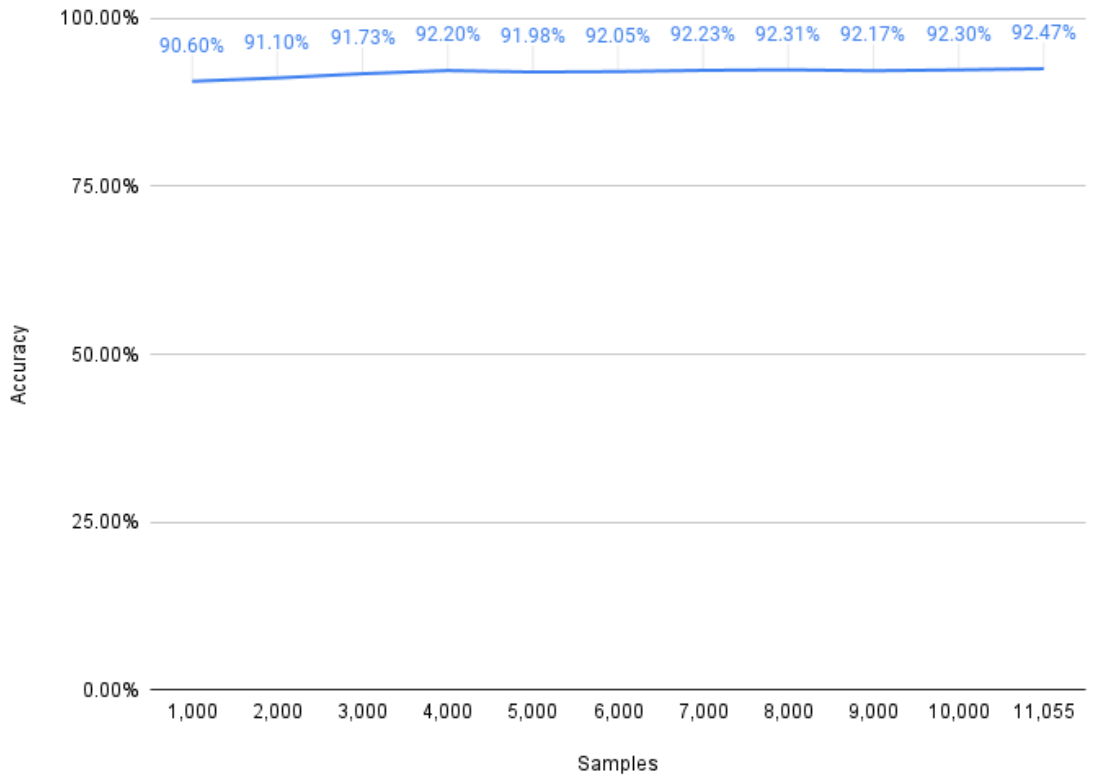


Figure 5.8. Measuring Online Learning

Figure 5.8 presents the model measuring when using River with a Logistic Regression classifier. The measuring performed every 1,000 samples, and shows the accuracy in each stage, while after 5,000 samples the accuracy is 91.98% and after measuring all 11,055 samples the accuracy is 92.47%.

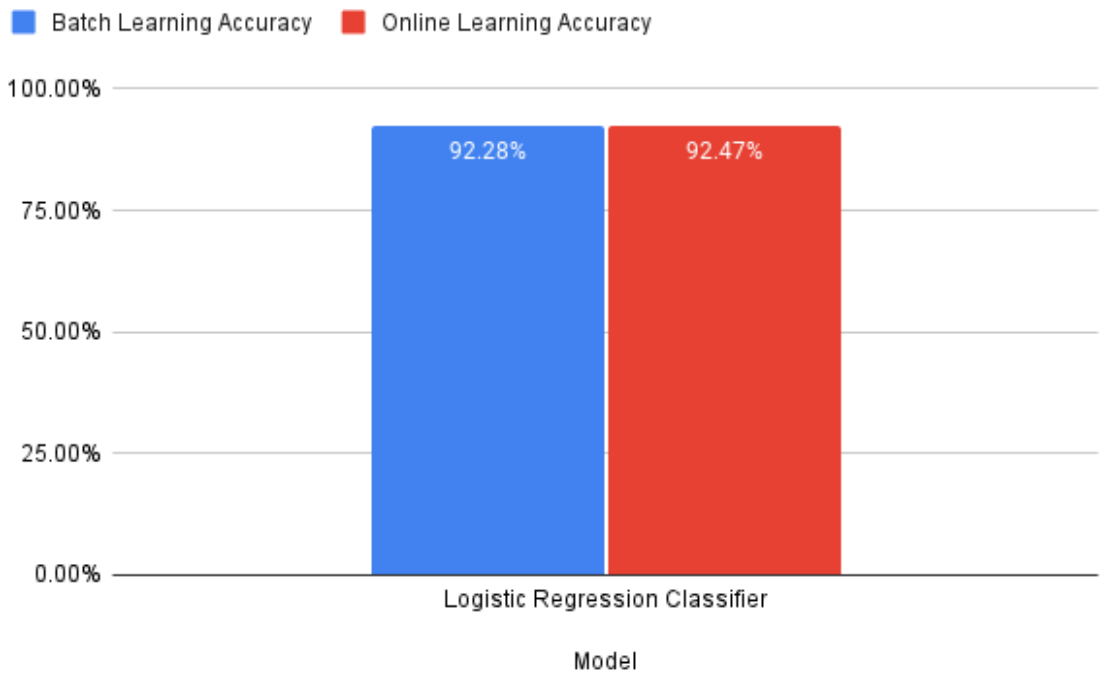


Figure 5.9. Online Learning vs Batch Learning

Figure 5.9 presents the comparison between the accuracy of online learning using the River tool and batch learning using sklearn module, while both of them use a Logistic Regression classifier. The accuracy of batch learning, 92.28%, is a bit lower than the accuracy of online learning, 92.47%, which means it is not a significant factor in choosing online learning over batch learning. The reason that only Logistic Regression was compared, in our case, is the fact that most of the models can infer online but only a few models can learn online.

Online machine learning using the River library, can help measure performance, handle large datasets on limited resources, and modify the model to fit new data on the fly. It supports large-scale data, which is important in light of the increasing number of phishing attacks, the improvement of phishing techniques, and the growing amount of data available.

6. Conclusions and Future Work

This chapter presents conclusions based on the experiments I have done and that were presented in the previous chapter - Chapter 5. It also describes and suggests future work that can be done.

6.1 Conclusions

Phishing is one of the most popular techniques for stealing sensitive personal data such as passwords, usernames, bank accounts, credit card info, etc. This research presents and describes the main dataset used in many studies, it also explains the existing problems in the dataset, how it can be improved, and based on that introduces a new dataset and the tool for collecting the features information for the new dataset. The research also shows different machine learning tools and models that can help protect users from phishing attacks, and how the accuracy of these models can be improved using feature selection techniques on different models, e.g. Decision Tree model shows an accuracy of 96.56% when using the whole dataset with 30 features, while with feature selection technique the number of features has been reduced to 23 and the accuracy improved to 98.96%, this means that some features are irrelevant and can cause “noise” for the Decision Tree model. In addition to that, the research also introduces online machine learning techniques and the powerful open-source tool “River”, which allows on-demand data changes in real-time, measuring performance, and detecting phishing URLs using a large-scale dataset with limited hardware resources.

6.2 Future Work

The UCI dataset needs to be maintained and improved by dropping irrelevant features based on feature selection along with searching for new strong features. The results for the new dataset were less accurate compared to the UCI dataset, this needs to be examined. Another aspect that should be explored is feature importance, and whether there is a correlation between feature selection and feature importance. Other online learning tools need to be explored and compared. During the experiments, the sorted dataset produced significantly better accuracy with *River*, which requires further investigation.

References

[1] “Phishing Activity Trends Reports.” APWG, https://apwg.org/trendsreports/.

[2] Jain, A.K., Gupta, B.B. (2018). PHISH-SAFE: URL Features-Based Phishing Detection System Using Machine Learning. In: Bokhari, M., Agrawal, N., Saini, D. (eds) Cyber Security. Advances in Intelligent Systems and Computing, vol 729. Springer, Singapore. https://doi.org/10.1007/978-981-10-8536-9_44.

[3] Tan, C. L., Chiew, K. L., Wong, K., & Sze, S. N. (2016). PhishWHO: Phishing webpage detection via identity keywords extraction and target domain name finder. *Decision Support Systems*, *88*, 18–27.

[4] M. A. Adebowale, K. T. Lwin, and M. A. Hossain, “Intelligent phishing detection scheme using deep learning algorithms,” J. Enterp. Inf. Manag., vol. ahead-of-print, no. ahead-of-print, Jun. 2020, doi: 10.1108/JEIM-01-2020-0036.

[5] O.K. Sahingoz, E. Buber, O. Demir, B. Diri, “Machine learning based phishing detection from URLs”, Expert Syst Appl, 117 (2019), pp. 345-357

[6] Scholkopf, B., Burges, C.J., & Vapnik, V.N. (1995). Extracting Support Data for a Given Task. *Knowledge Discovery and Data Mining*.

[7] Scholkopf, Bernhard, et al. "Comparing support vector machines with Gaussian kernels to radial basis function classifiers." *IEEE transactions on Signal Processing* 45.11 (1997): 2758-2765.

[8] Vapnik, V. N. (1998). Statistical Learning Theory. Wiley-Interscience.

[9] Shmilovici, Armin. “SUPPORT VECTOR MACHINES.” Data Mining and Knowledge Discovery Handbook, Springer, 2005, pp. 257-276. Springer Link, https://link.springer.com/chapter/10.1007/0-387-25465-X_12.

[10] Ezukwoke, Kenneth, and Samaneh Zareian. “(PDF) LOGISTIC REGRESSION AND KERNEL LOGISTIC REGRESSION A comparative study of logistic regression and

kernel logistic regression for binary classification." ResearchGate, 14 December 2019, https://www.researchgate.net/publication/337932960_LOGISTIC_REGRESSION_AND_KERNEL_LOGISTIC_REGRESSION_A_comparative_study_of_logistic_regression_and_kernel_logistic_regression_for_binary_classification.

[11] Rokach, Lior, and Oded Maimon. "Decision Trees." Data Mining and Knowledge Discovery Handbook, Springer, 2005, pp. 165-192, https://link.springer.com/chapter/10.1007/0-387-25465-X_9.

[12] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2017). CatBoost: Unbiased boosting with categorical features. ArXiv. /abs/1706.09516

[13] Dorogush, Anna Veronika, et al. "CatBoost: gradient boosting with categorical features support." arXiv.org, 24 Oct 2018, https://arxiv.org/pdf/1810.11363.pdf.

[14] "When to Choose CatBoost Over XGBoost or LightGBM [Practical Guide]." Neptune.ai, https://neptune.ai/blog/when-to-choose-catboost-over-xgboost-or-lightgbm.

[15] UCI Machine Learning Repository, "Phishing Websites Dataset" [URL]: https://archive.ics.uci.edu/ml/datasets/phishing+websites

[16] Rachmani, Asaf. "arachmani/ML-Phishing-Detection." GitHub, github.com/arachmani/ml-phishing-detection.

[17] "Mlxtend.feature selection - mlxtend." GitHub Pages, https://rasbt.github.io/mlxtend/api_subpackages/mlxtend.feature_selection/#sequentialfeatureselector.

[18] Rachmani, Asaf. "arachmani/ML-Phishing-Detection." GitHub, github.com/arachmani/phishing-websites-features.

[19] Vincent, James, and Alex Castro. "Amazon is retiring Alexa — no, not that one."

The Verge, 9 December 2021, https://www.theverge.com/2021/12/9/22825744/amazon-retiring-alexa-web-ranking-sevice.

[20] Rachmani, Asaf. “arachmani/ML-Phishing-Detection.” GitHub, github.com/arachmani/ml-phishing-detection/tree/main/datasets.

[21] Shalev-Shwartz, Shai, and Shai Ben-David. “Online Learning.” Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, 2014, pp. 287-306, https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf.

תקציר

בשנים האחרונות, שירותים מקוונים הפכו לחלק בלתי נפרד מחיינו. הם הופכים הכל לנגיש יותר ומסייעים לארגונים ולצרכנים להשיג את המטרות שלהם בצורה מהירה וקלה. רוב השירותים המקוונים הקיימים כיום משתמשים באתרי אינטרנט המכילים נתונים רגישים כגון סיסמאות, פרטי כרטיסי אשראי, מידע רפואי וכו'. לנתונים הרגישים הללו יש ערך רב עבור פושעי סייבר שיכולים להשתמש בהם בדרכים מזיקות. אחד מאיומי האבטחה המקוונים העיקריים איתם מתמודד עולם הסייבר כיום הוא אתרי דיוג. בהתקפת פישינג (דיוג), התוקפים יוצרים אתר דמה, על ידי העתקת התנהגות של אתר לגיטימי, שולחים כתובות URL לקורבן היעד באמצעות מיילים, הודעות טקסט או מדיה חברתית, ומבקשים מהקורבן לספק נתונים רגישים. אחת הדרכים למנוע התקפות אלו היא ללמוד לזהות אתרי דיוג. כמובן שניתן לזהות אתר כזה לאחר שהוא פעל וכבר הצליח לגרום נזק למשתמשים, אך המטרה הסופית היא להיות מסוגל לזהות אוטומטית אתרים חשודים כאלה לפני שהם גורמים נזק כלשהו. למטרה זו ניתן להשתמש בלמידת מכונה וסיווג כתובות אתרים ובכך לסייע במניעת התקפות דיוג. עבודת גמר זו מציגה שיטות לזיהוי התקפות דיוג באמצעות טכניקות למידת מכונה. הגישה מציגה מודלים ואלגוריתמים נבחרים של למידת מכונה כגון:

Support Vector Machines, Logistic Regression, K-Nearest Neighbors, Decision Tree, Gradient Boosting, Light Gradient Boosting Machine, eXtreme Gradient Boosting, Categorical Boosting, Neural Networks, Multilayer Perceptrons, online machine learning

ומתאר את מערך הנתונים ותכונותיו. הניסויים מתמקדים בעיקר בהפחתת התכונות של מערך הנתונים ושמירה על דיוק גבוה תוך שימוש במתודולוגיית בחירת תכונות, הצגת הבעיות במערך הנתונים הקיים ויצירת מערך נתונים מעודכן.

מכללה האקדמית תל אביב-יפו

בית הספר למדעי המחשב

זיהוי תקיפות פישינג באמצעות למידת מכונה

חיבור זה הוגש כחלק מהדרישות לקבלת התואר "מוסמך" - M.Sc
במכללה האקדמית תל-אביב-יפו

על ידי

אסף רחמני

העבודה הוכנה בהדרכתו של
פרופ' עדי שרייבמן