
# PHISHING WEBSITES DETECTION USING BAGGING ENSEMBLE MACHINE LEARNING

**2023**
**MASTER THESIS**
**COMPUTER ENGINEERING**

**Nuha Abubaker IBRAHEEM**

**Thesis Advisor**
**Assoc. Prof. Dr. Adib HABBAL**

# PHISHING WEBSITE DETECTION USING BAGGING ENSEMBLE MACHINE LEARNING

**Nuha Abubaker IBRAHEEM**

**Thesis Advisor**
**Assoc. Prof. Dr. Adib HABBAL**

**T.C.**
**Karabuk University**
**Institute of Graduate Programs**
**Department of Computer Engineering**

**Prepared as**
**Master Thesis**

**KARABUK**
**July 2023**

I certify that, in my opinion the thesis submitted by Nuha Abubaker IBRAHEEM titled "PHISHING WEBSITE DETECTION USING BAGGING ENSEMBLE MACHINE LEARNING" is fully adequate in scope and quality as a thesis for the degree of Master of Computer Engineering.

Assoc. Prof. Dr. Adib HABBAL .....
Thesis Advisor, Department of Computer Engineering

This thesis is accepted by the examining committee with a unanimous vote in the Department of Computer Engineering as a Master of Science thesis. 20/07/2023.

| Examining Committee Members (Institutions) | Signature |
|---|---|
| Chairman : Assist. Prof. Dr. Emrullah SONUÇ (KBU) | ..... |
| Member : Assoc. Prof. Dr. Adib HABBAL (KBU) | ..... |
| Member : Assist. Prof. Dr. Halil YETGİN (BEU) | ..... |

The degree of Master of Science by the thesis submitted is approved by the Administrative Board of the Institute of Graduate Programs, Karabuk University.

Assoc. Prof. Dr. Zeynep ÖZCAN .....
Director of the Institute of Graduate Programs

*“I declare that all the information within this thesis has been gathered and presented in accordance with academic regulations and ethical principles and I have according to the requirements of these regulations and principles cited all those which do not originate in this work as well.”*

Nuha Abubaker IBRAHEEM

## ABSTRACT

**M. Sc. Thesis**

**PHISHING WEBSITE DETECTION USING BAGGING ENSEMBLE MACHINE LEARNING**

**Nuha Abubaker IBRAHEEM**

**Karabük University**
**Institute of Graduate Programs**
**The Department of Computer Engineering**

**Thesis Advisor:**
**Assoc. Prof. Dr. Adib HABBAL**
**July 2023, 40 pages**

Phishing attacks have become a critical threat to the security of online users. Traditional phishing detection methods often face challenges in accurately identifying malicious websites and emails. In this research, we propose a novel model for phishing detection using a bagging ensemble with random forest, decision tree, gradient boosting, and k-nearest neighbors' algorithms. Those classifiers combine with ensemble learning to improve the overall detection performance, where the dataset is trained and tested by this model. The results showed that this model can handle noisy data and variations in phishing techniques, and also reduce the impact of outliers leading to higher accuracy and overall performance.



**Key Words:** Cyber Security, Phishing Detection, Bagging, Decision Tree, Random Forest, K-nearest Neighbors, Gradient Boosting.

**Science Code :** 92403

# ÖZET

**Yüksek Lisans Tezi**

**TORBALAMA TOPLULUK MAKİNE ÖĞRENMESİ KULLANARAK OLTALAMA WEB SİTELERİNİ TESPİT ETME**

**Nuha Abubaker IBRAHEEM**

**Karabük Üniversitesi**
**Lisansüstü Eğitim Enstitüsü**
**Bilgisayar Mühendisliği Anabilim Dalı**

**Tez Danışmanı:**
**Doç. Dr. Adib HABBAL**
**Haziran 2023, 40 sayfa**

Kimlik avı saldırıları, çevrimiçi kullanıcıların güvenliği için kritik bir tehdit haline gelmiştir. Geleneksel kimlik avı tespit yöntemleri, kötü niyetli web sitelerini ve e-postaları tanımlamada genellikle zorluklarla karşılaşmaktadır. Bu araştırmada, rastgele orman, karar ağacı, gradyan artırma, k-en yakın komşu algoritmaları ile bir torbalama topluluğu kullanarak kimlik avı tespiti için yeni bir model öneriyoruz. Bu sınıflandırıcılar, veri kümesinin bu model tarafından eğitildiği ve test edildiği genel algılama performansını iyileştirmek için topluluk öğrenimi ile birleştirilmektedir. Sonuçlar, bu modelin gürültülü verilerle ve kimlik avı tekniğindeki varyasyonlarla başa çıkabildiğini, ayrıca aykırı değerlerin etkisini azaltarak daha yüksek doğruluk ve genel performans sağladığını göstermiştir.

**Anahtar Kelimeler:** Siber Güvenlik, Kimlik Avı Tespiti, Torbalama, Karar Ağacı, Rastgele Orman, K-en Yakın Komşu, Gradyan Güçlendirme.

**Bilim Kodu** **:** 92403

## ACKNOWLEDGEMENT

First and foremost, I express my heartfelt gratitude to Allah, the Almighty, for His unwavering guidance, mercy, and countless blessings throughout my academic journey. His divine presence and grace have been the source of strength and inspiration. Thanks to the jury members, my teachers, and my supervisor Assoc. Prof. Adib Habbal for his continuous support, patience, and encouragement.

My deepest appreciation to my parents and my lovely sister, whose unconditional love, unwavering belief in my abilities, and constant encouragement have been the driving force behind my accomplishments.

**CONTENTS**

**Page**

## LIST OF FIGURES

**Page**

## LIST OF TABLES

**Page**

## ABBREVIATIONS INDEX

### ABBREVIATIONS

AI : Artificial Intelligence
CUI : Conversational User Interface
DT : Decision Tree
F : False
FN : False Negative
FP : False Positive
GB : Gradient boosting
KNN : k-nearest neighbors
ML : machine learning
NLP : Natural Language Processing
TN : True Negative
TP : True Positive
URL : Uniform Resource Locator

# PART 1

# INTRODUCTION

## 1.1 RESEARCH BACKGROUND

The Internet in our days is considered to be an irreplaceable tool in a wide range of activities and sharing knowledge, which may lead to threatening users and individuals or organizations. A phishing attack is considered one of the most common cyber-crime attacks, it usually happens when an E-mail from a fake website created by phishers to steal the user's credentials and carry out fraudulent activities. Phishing detection is an important aspect of cyber security that aims to identify and prevent fraudulent stealing of individuals' and organizations' sensitive information. Recently, machine learning (ML) algorithms have been defined as powerful tools for facing the challenges associated with phishing detection. Several studies have been done on this topic using ML algorithms, demonstrating their effectiveness in detecting and preventing this cyber security attack. One study by Yuan. Li. [1] focused on using ensemble methods and GB with machine learning algorithms, to improve the accuracy of phishing detection. They collected and tested a large dataset of phishing emails and used feature extraction techniques to show the email content and header information. Their results demonstrated that ensemble methods outperformed individual classifiers, achieving high accuracy in detecting phishing emails. In another investigation research by Md. Belal. et al. [2] explored the use of hybrid machine learning algorithms, combining support vector machines and genetic algorithms, for phishing detection. They proposed a feature selection mechanism to identify the most relevant features from the data set and improve the performance of the model. Their findings demonstrated the efficacy of hybrid algorithms in achieving high accuracy and robustness in phishing detection. In addition to these studies, several other research works have investigated the application of machine learning algorithms such as decision tree (DT), logistic regression, and Bayesian

networks for phishing detection. These studies have focused on improving accuracy and reducing false positives. Furthermore, L. D. et al. [3] employed deep learning techniques for phishing detection They developed a deep neural network model that integrated both text and image-based features extracted from phishing web pages. The model showed promising results, achieving high precision and recall in identifying phishing websites.

Despite these advancements, there is still a need for further research to address emerging challenges in phishing detection, including the detection of targeted and sophisticated attacks [3]. This thesis aims to contribute to the existing body of knowledge by comprehensively evaluating the effectiveness of various machine learning algorithms in phishing detection. By utilizing real-world dataset, conducting rigorous performance evaluations, and exploring feature importance and interpretability, this research seeks to enhance the understanding and capabilities of machine learning-based phishing detection systems.

## 1.2. PROBLEM STATEMENT

While various techniques have been proposed for phishing detection, the effectiveness of individual classifiers is often limited due to the dynamic nature of phishing attacks, therefor Bagging combines the predictions of multiple models by either majority voting (for classification) or averaging (for regression). This aggregation helps to smooth out the individual model's predictions and reduce the impact of outliers, making the ensemble more robust and less prone to overfitting also ensures scalability and leads to higher accuracy. The limited feature set is also a concern, where the models trained on a specific set of features without considering other potentially important indicators of phishing attacks [1], our model contains 30 features combined with URL-Based and Domain-based features.

## 1.3. RESEARCH OBJECTIVES

The main objective of this research is to build a robust ensemble learning models. In particular, we aim to achieve the following specific objectives:

- To design an ensemble-based model that can handle variations and evolution of phishing techniques. This will enhance the performance of machine learning algorithms in phishing detection.
- To build a reliable model that will be able to generalize well to new and unseen instances to ensure scalability.

## 1.4. SCOPE

The scope of this study is to develop a machine learning-based approach for detecting phishing attacks using a bagging ensemble with RF, decision tree, GB, and k-nearest neighbors' algorithms. The study focuses on identifying the most important features of phishing detection and developing an accurate and efficient model for identifying phishing websites. The study also evaluates the performance of the proposed model against existing anti-phishing solutions and provides insights and recommendations for improving the effectiveness of anti-phishing techniques in real-world scenarios.

## 1.5. SIGNIFICANCE OF THE STUDY

In this chapter, a brief introduction to phishing attacks is given. Followed by an introduction to the used technology to detect this kind of attack. Afterward, the problems faced in creating models to prevent these attacks, and the main objectives to achieve during this study. The chapter ends with a description of the scope of the research.

# PART 2

# LITERATURE REVIEW

Various studies and solutions have been published about phishing detection, in this kind of attack the hacker does not necessarily search for vulnerability but the target is unaware users to steal their credentials, for example creating a login page for a well-known website and sending the link of it as an E-mail. Recently lots of researchers tried different approaches according to the level of the hacker using and updating.

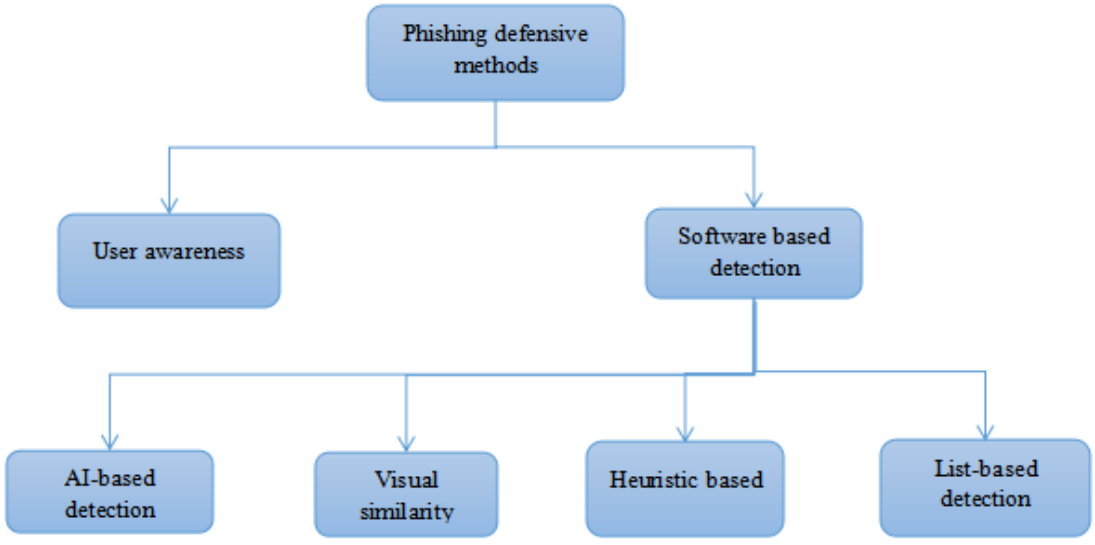


Figure 2.1. Phishing detection main approaches

## 2.1. PHISHING DETECTION MAIN APPROACHES

we classified all these approaches and efforts into two basic categories as shown in Figure 2.1 human-based solutions including awareness courses for the employees and detection software, the other solution is software-based including the used approach to build software and new techniques for phishing detection [4].

### 2.1.1. HEURISTIC-BASED

Heuristic-based is a common approach used in phishing detection. This technique depends on a capsule of rules and heuristics to determine possible phishing attacks. Heuristics are based on characteristics of phishing emails and websites that which is different from legitimate emails and websites. There are examples of the heuristics used in phishing detection [4]:

- **URL analysis:** Phishing websites mostly use URLs to look similar to legitimate websites but it contains few variations. Heuristics can be used to identify URLs that contain typos or diversity of the legitimate website's URL.
- **Content analysis:** Phishing emails often use urgency, fear, or other emotional tactics to persuade the recipient to take action. Heuristics can be used to identify emails with suspicious content, such as urgent requests for personal information or offers that seem too good to be true.
- **Sender analysis:** Phishing emails often come from unknown or suspicious senders. Heuristics can be used to identify emails that come from unknown or suspicious senders.
- **Attachment analysis:** Phishing emails often contain attachments that are designed to look like legitimate links that lead to malicious websites. Heuristics can be used to identify suspicious attachments or links.
- **IP address analysis:** Phishing attacks often start from IP addresses that are known to be associated with phishing attacks. Heuristics can be used to identify emails or websites that originate from suspicious IP addresses.
- **Machine learning:** machine learning algorithms can be trained using heuristics and other features to identify phishing emails and websites. These algorithms can improve over time as they are exposed to more data.
- In summary, heuristic-based approaches for phishing detection use a set of rules and heuristics to identify potential phishing attacks. These approaches can be effective, but they can also produce false positives if

the heuristics are too strict. Therefore, it is important to use a combination of approaches, to detect and prevent phishing attacks.

### 2.1.2. BLACKLISTS

Blacklists are a commonly used technique for phishing detection. Blacklists are essentially lists of known phishing websites or email addresses that have been identified as malicious or suspicious. When an email or website matches a known entry on the blacklist, it is automatically blocked or flagged as suspicious [5]. Modern browsers are embedded and update the list regularly. The browser checks all the websites that users want to visit and compares that list and if the web page is listed there, gives a warning to the user. Figure 2.2 shows an example of that warning.

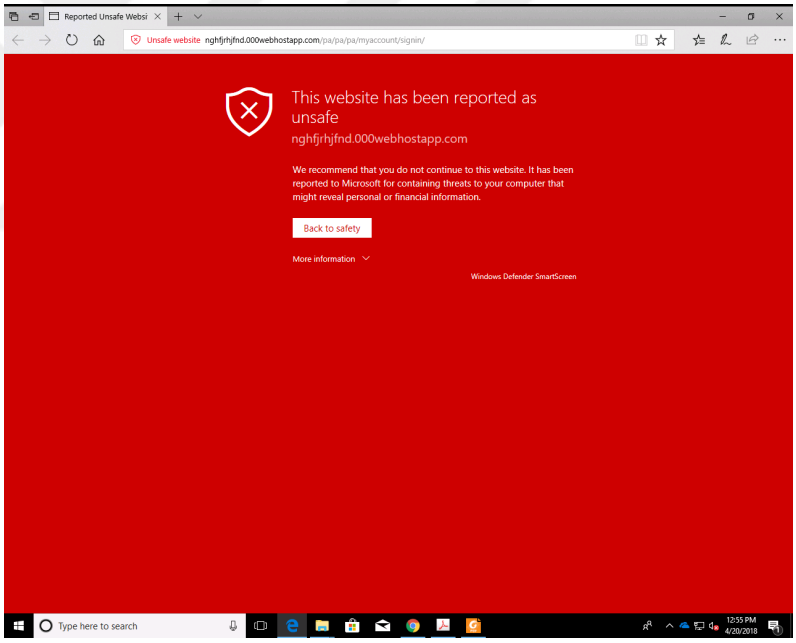


Figure 2.2. Warning of phishing attacks on Microsoft Edge [6]

Blacklists are created by security researchers or organizations that specialize in phishing detection. These organizations use a variety of techniques to identify phishing attacks, including manual analysis of emails and websites, automated scans, and user reports. Once a phishing email or website is identified, it is added to the blacklist. Blacklists are constantly updated as new phishing attacks are identified.

This means that phishing emails and websites that are not yet on the blacklist may still be able to bypass detection. There are two main types of blacklists used in phishing detection: domain-based and IP-based blacklists. Domain-based blacklists contain a list of suspicious domain names or URLs. IP-based blacklists contain a list of suspicious IP addresses. Some blacklists may also include other information such as email addresses, sender names, or patterns of behavior associated with phishing attacks.

Blacklists are only effective if they are up-to-date and comprehensive. New phishing attacks that have not yet been identified may be able to bypass detection. Additionally, blacklists can produce false positives if legitimate emails or websites are mistakenly identified as suspicious.

In summary, blacklists are a useful technique for phishing detection. They rely on a list of known phishing websites or email addresses to identify potential threats. However, they have some limitations and should be used in conjunction with other phishing detection techniques.

### 2.1.3. VISUAL SIMILARITY

Visual similarity is a technique used in phishing detection to identify websites that are visually similar to legitimate websites but are fake or fraudulent. This technique relies on comparing the visual elements of a website, such as the layout, color scheme, and images, to those of a known legitimate website to identify any differences or inconsistencies [7].

In these below points, we are showing how visual similarity is used in phishing detection:

- **Collection of visual elements:** The first step in using visual similarity for phishing detection is to collect visual elements from legitimate websites. These may include screenshots, HTML code, and CSS files.

- **Creation of a visual similarity model:** The collected visual elements are used to create a model of the legitimate website's visual appearance. This model can be used to compare the visual elements of other websites to identify any similarities or differences.
- **Comparison of websites:** When a new website is encountered, its visual elements are compared to the model of the legitimate website. If there are significant differences or inconsistencies, the website may be flagged as suspicious or fraudulent.
- **Limitations of visual similarity:** Visual similarity can be an effective technique for identifying visually similar phishing websites, but it has some limitations. For example, it may not be able to detect phishing websites that use completely different visual elements or that only use subtle variations from the legitimate website.
- **Advancements in visual similarity:** Recent advancements in machine learning have enabled the development of more sophisticated visual similarity models that can identify more subtle variations and differences between websites. These models use algorithms to identify patterns and features in the visual elements of a website, rather than relying solely on human judgment.

In summary, visual similarity is a technique used in phishing detection to identify websites that are visually similar to legitimate websites but are fake or fraudulent. This technique relies on comparing the visual elements of a website to those of a known legitimate website to identify any differences or inconsistencies. While it has some limitations, visual similarity can be an effective technique when used in combination with other phishing detection methods.

### 2.1.4. AI-BASED APPROACHES

Phishing is a common attack vector used by cyber-criminals to steal sensitive information such as passwords, credit card details, and personal information from unsuspecting victims. Artificial Intelligence (AI) has emerged as a promising technology to help detect and prevent phishing attacks. In this response, we will

provide an overview of AI-based approaches for phishing detection, including the techniques and methods used by these approaches [7].

- **Machine Learning-Based Approaches:** Machine learning (ML) is a popular technique used in AI-based phishing detection systems. In ML-based approaches, the system is trained on a dataset of known phishing attacks and legitimate emails. The system then uses this training data to learn patterns and features that distinguish between phishing and legitimate emails. The most commonly used ML algorithms for phishing detection include decision tree (DT), support vector machines (SVM), k-nearest neighbors (KNN), GB, random forest (RF), and neural networks.
- **Natural Language Processing (NLP)-Based Approaches'** is another popular technique used in AI-based phishing detection systems. In NLP-based approaches, the system analyzes the content and structure of emails to identify suspicious patterns and language that are commonly used in phishing emails. NLP-based approaches can also identify anomalies in email headers and sender information that may indicate a phishing attack.
- **Deep Learning-Based Approaches:** Deep learning is a subset of machine learning that uses neural networks to learn patterns and features from data. In deep learning-based approaches, the system uses a deep neural network to analyze the content and structure of emails to identify phishing attacks. Deep learning-based approaches have shown promising results in detecting complex phishing attacks that traditional machine learning algorithms may struggle to identify.
- **Hybrid Approaches:** Hybrid approaches combine multiple AI techniques, such as machine learning and NLP, to improve the accuracy and effectiveness of phishing detection systems. By combining multiple techniques, hybrid approaches can identify a wider range of phishing attacks and reduce false positives.
- **Rule-Based Approaches:** Rule-based approaches use predefined rules to identify phishing attacks. These rules are based on known

characteristics and patterns of phishing emails, such as misspelled words, suspicious URLs, and requests for personal information. While rule-based approaches can be effective, they may struggle to identify more sophisticated phishing attacks that may not fit predefined patterns.

AI-based approaches for phishing detection use a variety of techniques, including machine learning, natural language processing, deep learning, and hybrid approaches, to analyze the content and structure of emails and identify phishing attacks. These approaches have shown promising results in detecting phishing attacks and can help protect individuals and organizations from the harmful effects of these attacks. Additionally, some researchers have proposed hybrid approaches that combine multiple machine learning techniques to improve detection accuracy. For example, [8] uses a combination of rule-based, logistic regression, and DT algorithms to detect phishing URLs. The model achieved an accuracy of 97.25% on a dataset of 5,000 phishing and non-phishing URLs.

We summarized all these approaches in Table 2.1 below to show a comparison between those technical approaches in specific aspects like the response times and the cost, complexity to implement, FP rates, and how effectively can handle zero-hour attacks:

Table 2.1. Phishing detection main approaches comparison

| Technique | Response Time | Cost | complexity | FP Rate | Mitigation Zero-hour attack |
|---|---|---|---|---|---|
| Black-lists | high | Not costly | Low | Effective | Low |
| Heuristic-based | low | Medium cost | Medium | More effective | medium |
| Visual similarity | low | Medium cost | Medium | More effective | Low |
| Machine Learning | low | Costly | High | More Effective | medium |

there are some limitations to current AI-based phishing detection approaches. Additionally, attackers can employ adversarial techniques to evade detection by AI algorithms, which can limit the effectiveness of these approaches.

In conclusion, existing research on phishing detection using AI has shown promising results, but further research is needed to improve the robustness and effectiveness of these techniques.

## 2.2. PREVIOUS WORK COMPARISON

Much research and related work have been done for phishing attack detection Table 2.2 shows some comparison aspects between these researches in recent years:

Table 2.2. Recent phishing attack detection research comparison

| Ref | year | features | Detection method | Evaluation metrics | strength | weakness |
|---|---|---|---|---|---|---|
| **[9]** | 2021 | √ | √ | √ | Improved performance, robustness, | Computational complexity and reduced interpretability |
| **[10]** | 2019 | √ | √ | × | provide valuable information for phishing detection | May not capture all relevant information, easily manipulated by attackers |
| **[11]** | 2019 | √ | √ | √ | Can capture patterns and structures in web pages | Limited effectiveness |
| **[12]** | 2018 | × | √ | √ | protect users directly on their devices | limitations in processing power and memory |
| **[13]** | 2020 | √ | √ | √ | detect subtle variations in URLs | Not effective against different phishing techniques |
| **[14]** | 2020 | √ | √ | √ | Handles high- | Computational complexity, large |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | dimensional data, Robustness to noise | training data requirements |
| **Our model** | 2023 | √ | √ | √ | Improved performance, robustness, Handles noisy data detect subtle variations in URLs | Large training data requirements |

# PART 3

# THEORETICAL BACKGROUND

The utilization of machine learning algorithms is important for developing robust phishing detection systems. These ML algorithms can analyze and extract different features that is used with the tested dataset, to create defective models for phishing attack. By using the power of ensemble learning and the strengths of each algorithm, to protect organizations and individuals from the ever-evolving landscape of phishing attacks.

## 3.1. MACHINE LEARNING

machine learning is a branch of artificial intelligence that focuses on data and algorithms to imitate the way that humans learn, gradually improving its accuracy and improving their performance on a certain task over time, in a simple programming way. It is a kind of statistical modeling that uses algorithms to set apart the patterns in data and make predictions or decisions based on those patterns.

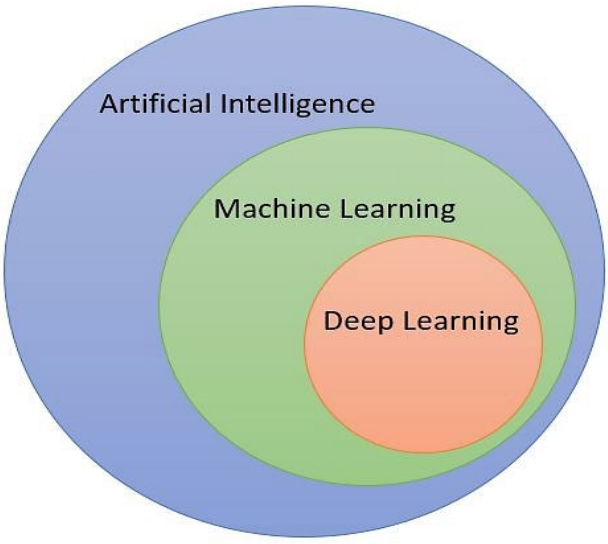


Figure 3.1. Machine learning development and expansion [15]

The machine learning process typically involves data preparation which is collecting, cleaning, and organizing data for analysis, model building: selecting an appropriate algorithm and training it on the prepared data, model evaluation: testing the model's performance on a separate set of data to assess its accuracy and identify areas for improvement, model deployment using the trained model to make predictions or automate decision-making in real-world scenarios [16].

### 3.2. DECISION TREE

Decision tree (DT) is a type of supervised learning algorithm that can be used for classification problems. They are particularly well-suited for phishing detection because they can be used to model the decision-making process of an attacker. In the context of phishing detection, DT can be used to learn a set of rules that can be used to classify an email as legitimate or phishing.

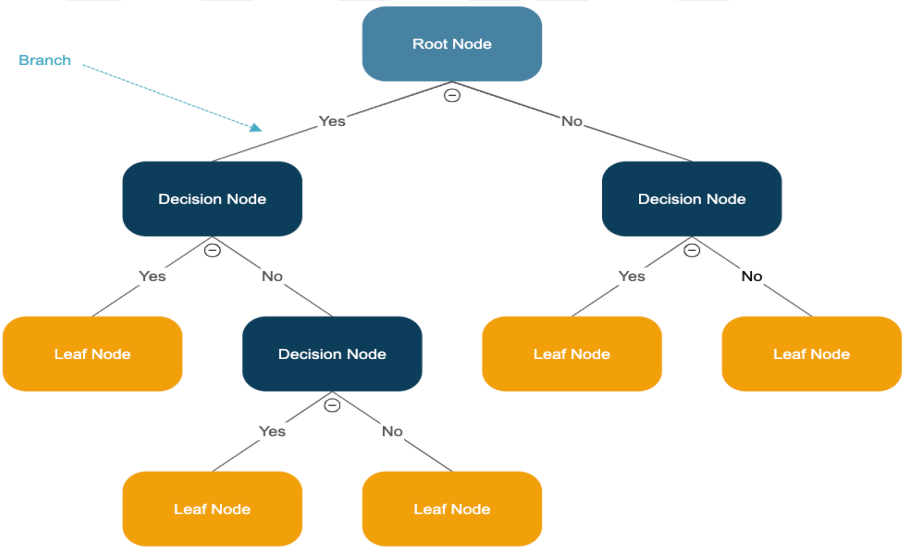


Figure 3.2. DT algorithm [17]

The DT algorithm starts with a root node that represents the entire dataset and then it recursively splits the dataset into smaller subsets based on the features that provide the best information gain. Each internal node of the DT represents a test on a feature, and each leaf node represents a class label. Once a DT has been trained, it can be used to classify new instances by traversing the tree from the root to a leaf node

based on the values of the features of the instance [16]. The code below explains the basic steps to train a dataset using a DT:

```
function BuildDecisionTree(dataset, targetAttribute):
    if all instances in the dataset belong to the same class:
        return a leaf node with the class label
    if the dataset is empty:

        return a leaf node with the majority class label
from the parent node

        bestAttribute = SelectBestAttribute(dataset,
targetAttribute)

       decisionTree = new DecisionTreeNode(bestAttribute)

    partitions = PartitionDataset(dataset, bestAttribute)
    for each partition in partitions:
        value = partition.attributeValue
        subset = partition.dataset
        if subset is empty:
            childNode = new
LeafNode(MajorityClassLabel(dataset, targetAttribute))
        else:

            childNode = BuildDecisionTree(subset,
targetAttribute)

        decisionTree.addChild(value, childNode)

    return decisionTree
```

### 3.3. RANDOM FOREST

Random Forest (RF) is a popular machine learning algorithm that is widely used for both classification and regression tasks. It belongs to the ensemble learning methods, which combine multiple individual models to make more accurate predictions. The RF algorithm creates an ensemble of DT, where each tree is built using a random subset of the training data and a random subset of the input features [16].

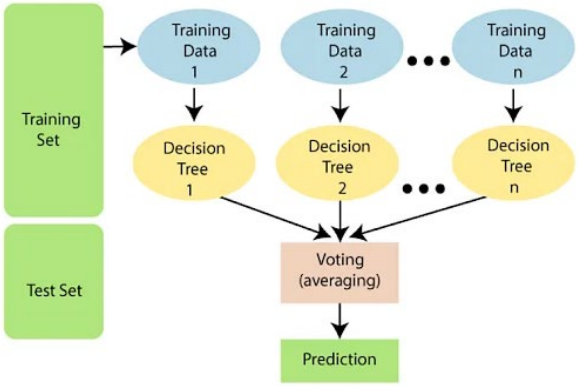


Figure 3.3. RF algorithm [18]

From the given dataset, a random subset of the data points (observations) is selected. This process is called bootstrapping, and it involves sampling with replacement, which means that a data point can be selected multiple times or not at all. The code below explains the basic steps to train a dataset using a RF:

```
function BuildRandomForest(dataset, targetAttribute,
numTrees, numFeatures):
    forest = []
    for i = 1 to numTrees:
        randomFeatures =
RandomSubsetOfFeatures(dataset.attributes, numFeatures)

        bootstrapSample = BootstrapSample(dataset)

features
        tree = BuildDecisionTree(bootstrapSample,
targetAttribute, randomFeatures)

        forest.append(tree)

    return forest
```

### 3.4. GRADIENT BOOSTING

Gradient Boosting (GB) is a popular machine learning algorithm that is used for both classification and regression tasks. It works by combining multiple weak predictive models, typically DT, to create a strong predictive model. GB builds the models in an

iterative manner, where each subsequent model is trained to correct the mistakes made by the previous models. This process is done by minimizing a loss function, such as mean squared error or log loss, through gradient descent. GB has proven to be highly effective in a wide range of applications, including predicting customer churn, fraud detection, and image recognition [19]. The code below explains the basic steps to train a dataset using a gradient boosting:

```
# Import the necessary libraries
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score
  variable
   X, y = load_data()
      X_train, X_test, y_train, y_test = train_test_split(X, y,
      test_size=0.2, random_state=42)

    gb_classifier = GradientBoostingClassifier(n_estimators=100,
    learning_rate=0.1, max_depth=3)

    Gb_classifier.fit(X_train, y_train)
# Make predictions on the testing set
y_pred = gb_classifier.predict(X_test)

# Evaluate the performance of the classifier
  accuracy = accuracy_score(y_test, y_pred)
  precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
# Print the performance metrics
print("Accuracy: ", accuracy)
print("Precision: ", precision)
print("Recall: ", recall)
print("F1 Score: ", f1)
```

### 3.5. K-NEAREST NEIGHBORS

Nearest Neighbors (KNN) is a simple yet powerful algorithm usually used for both

regression and classification tasks. KNN works by comparing a new data point to its K nearest neighbors in the training set, where K is a user-defined parameter. The predicted value or class of the new data point is determined by the majority vote or average of the K nearest neighbors. KNN is a non-parametric algorithm, meaning it does not make any assumptions about the underlying data distribution. It is often used for tasks such as recommendation systems, anomaly detection, and image recognition, KNN is easy to understand and implement [19]. The code below explains the basic steps to train the dataset using KNN:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

 data = pd.read_csv('phishing_dataset.csv')
          X = data.drop('label', axis=1)
          y = data['label']

  X_train,  X_test,  y_train,  y_test  =  train_test_split(X,  y,
  test_size=0.2, random_state=42)

# Create a KNN classifier object
    knn = KNeighborsClassifier(n_neighbors=5)

# Fit the classifier to the training data
    knn.fit(X_train, y_train)

# Predict the labels for the test data
   y_pred = knn.predict(X_test)

# Calculate the accuracy of the model
   accuracy = accuracy_score(y_test, y_pred)
   print("Accuracy:", accuracy)
```

### 3.6. BAGGING ENSEMBLE

Bagging (Bootstrap Aggregating) is an ensemble learning technique that combines multiple machine learning models into a single model that can be used to improve

the performance of DT for phishing detection. The idea behind bagging is to generate multiple decision trees using different random subsets of the training data and then combine the results of these trees to make a final decision. Each decision tree in the bagging ensemble is trained on a different bootstrap sample of the training data, which is obtained by sampling the data with replacement [19]. Bagging work in parallel, can enhance the performance of phishing detection systems by improving accuracy, and robustness, and reducing over-fitting. It enables the detection models to learn from diverse examples and variations of phishing attacks, leading to more effective and efficient detection capabilities. The code below explains the basic steps to train the dataset using a bagging ensemble:

```
function BuildBaggingEnsemble(dataset, targetAttribute,
numModels):
    ensemble = []
    for i = 1 to numModels:

        bootstrapSample = BootstrapSample(dataset)

            model       =       BuildModel(bootstrapSample,
targetAttribute)

                ensemble.append(model)
    return ensemble
```

### 3.7. PHISHING WEBSITES FEATURES

Phishing attacks often involve the use of fake or spoofed URLs that appear to be legitimate but are designed to steal user information [20].

To detect phishing URLs, various features can be analyzed, as shown in Table 3.1 below:

Table 3.1. URL and domain features and their explanation

| NO. | FEATURE | EXPLANATION |
|---|---|---|
| 1 | Using the IP Address | If an IP address is used as an alternative to the domain name in the URL, users can be sure that someone is trying to steal their personal information. |
| 2 | Long URL to Hide the | Phishers can use long URLs to hide the doubtful |

| | | |
|---|---|---|
| | Suspicious Part | part in the address bar. |
| **3** | Misspelled domain names | Look for domain names that contain deliberate misspellings of popular websites. |
| **4** | Subdomain inconsistencies | Check for irregular subdomains that don't match the legitimate website's structure. |
| **5** | Extra hyphens or special characters | Phishing URLs sometimes include additional hyphens or special characters to mimic legitimate URLs. |
| **6** | IP address instead of domain name | Be cautious if the URL contains an IP address instead of a domain name |
| **7** | Long or complex URLs | Phishing URLs can be excessively long or convoluted, with multiple directories and parameters. |
| **8** | Non-standard port numbers | Legitimate websites typically use standard port numbers (e.g., 80 for HTTP, and 443 for HTTPS). Unusual port numbers may indicate a phishing attempt. |
| **9** | Mixed or mismatched protocols | Check for URLs that mix HTTP and HTTPS or use incorrect protocol prefixes. |
| **10** | Fake file extensions | Phishing URLs might use deceptive file extensions to make a malicious file appear harmless (e.g., txt.exe). |
| **11** | URL encoding | Phishers may use URL encoding (%20, %22, %3C, etc.) to obfuscate the true nature of the URL. |
| **12** | Punycode domains | Verify domains that use Punycode to represent non-Latin characters, as they can be used for IDN homograph attacks. |
| **13** | Suspicious top-level domains (TLDs) | Watch for uncommon or suspicious TLDs that deviate from well-known ones (e.g., .cm instead of .com). |
| **14** | Extra subdomains | Be cautious if the URL contains unexpected or unrelated subdomains. |
| **15** | Replicated domain names | Phishing URLs may replicate domain names (e.g., paypa1.com instead of paypal.com). |
| **16** | Unusual domain extensions | Phishing sites often use non-standard or country-specific domain extensions. |
| **17** | Random numbers or letters | URLs with random numbers or letters inserted within the domain or path can be indicators of phishing attempts. |
| **18** | Fake login pages | Phishing attacks often use URLs that imitate legitimate login pages to steal user credentials. |
| **19** | Numbers instead of words in the domain | Phishers may use numbers instead of words to mimic legitimate domains (e.g., 0 instead of o). |
| **20** | Redirects and multiple domains | Pay attention to URLs that involve frequent redirects or multiple domains before landing on |

| | | |
|---|---|---|
| | | the final page. |
| **21** | Overly long URLs with unnecessary parameters | Be wary of URLs that contain excessive parameters or query strings that seem unnecessary. |
| **22** | Suspicious pop-up windows | If a pop-up window prompts you to enter personal information or login credentials, verify the URL before proceeding. |
| **23** | URL mismatch with email content | Check if the URL in an email matches the content and purpose of the email. Phishing emails often contain mismatched URLs. |
| **24** | Misplaced or added slashes | Phishing URLs may include misplaced or additional slashes in the domain or path. |
| **25** | Multi Sub Domains | A domain name that includes more than two dots can be classified as phishing. |
| **26** | URL containing "login" or "banking" | Be cautious of URLs that include common keywords related to login or banking activities, as phishers often exploit these terms. |
| **27** | Suspicious shortened URL services | when clicking on URLs generated by unfamiliar or suspicious URL-shortening services. |
| **28** | URLs with embedded login credentials | Phishing URLs might include login credentials within the URL itself, attempting to trick users into thinking it's a legitimate login page. |
| **29** | URLs with multiple domains separated by hyphens | Phishing URLs may include multiple domain names separated by hyphens to appear legitimate. |
| **30** | Sub Domain | A domain name that includes more than two dots can be classified as phishing. |

### 3.8. DATASET

The dataset created by [21] and shared with CUI (Controlled Unclassified Information) in 2012 consists of 11,000 websites that are specifically related to CUI phishing attacks. This dataset is designed to provide researchers and practitioners with a comprehensive collection of websites that are used in phishing attempts targeting users of command-line interfaces.

The dataset includes a wide range of phishing websites about 6158 phishing instances from PhishTank.com and 4898 legitimate instances from Alexa.com. These websites may utilize various social engineering techniques to deceive users and convince them to provide their information.

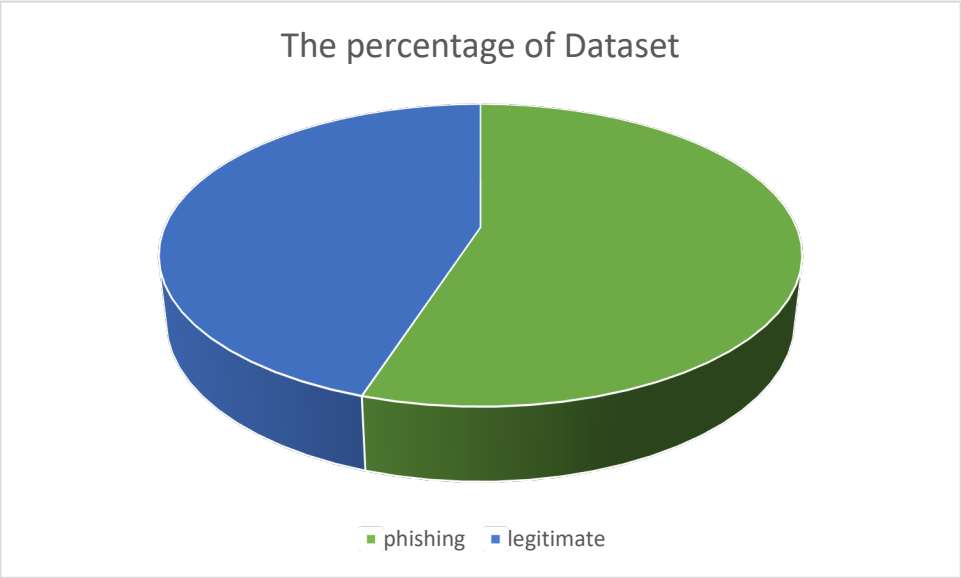


Figure 4.1. Dataset percentage of legitimate and phishing sites [22]

# PART 4

# METHODOLOGY AND IMPLEMENTATION OF OUR PROPOSED MODEL

According to the literature review, there are variant methods and algorithms used to detect phishing attacks, thus this chapter explains the method followed in this thesis to achieve the designed results.

## 4.1. THESIS METHODOLOGY

- Data collection: The first step in this research design is to collect the dataset of legitimate and phishing websites and it should contain different types of phishing attacks.
- Feature extraction: After the data is collected and processed the features of the website should be extracted from the dataset to show the characteristics of legitimate and phishing websites, this step includes feature selection and scaling.
- Model selection: Selecting the appropriate ML algorithm to detect phishing attacks is the next step, RF and DT, KNN, and GB are commonly used for phishing detection, and bagging ensemble will be used to enhance the performance of these algorithms.
- Model training: The selected model is trained on the prepossessed dataset using a training set. The training set is used to optimize the model parameters and improve its accuracy.
- Model evaluation: There is certain matrix will be used to assess the performance of the model including accuracy, precision, recall, and F1 score.

## 4.2. DATA COLLECTION

Data collection plays a crucial role in the development of effective models for phishing website detection. It is essential to ensure that the training data collection process is independent of the attackers' actions to create reliable and unbiased models. The dataset should contain legitimate and phishing websites that include different types of phishing and different features to extract, the web pages that are used in phishing attacks can be used to train machine learning models to detect the attack [23].

## 4.3. FEATURES EXTRACTION

Phishing websites can exhibit various features which are mentioned in the previous chapter that can help in their identification and differentiation from legitimate websites. While the presence of a single feature may not be definitive proof of phishing, a combination of these features can raise suspicion [22]. The features are represented in binary where the numbers show the status of the URL and are used in the dataset in the same format. For example:

```
having_IP_Address { -1,1}
Shortening_Service {1, -1}
having_At_Symbol {1, -1}
double_slash_redirecting { -1,1}
Prefix_Suffix { -1,1}
```

## 4.4. ENSEMBLE LEARNING MODEL

In this chapter we are examining phishing detection using ML algorithms, steps involve gathering a representative dataset of phishing and legitimate websites, comprising features that can help differentiate between the two. The collected dataset needs to be reprocessed to ensure its quality and compatibility with the model, this step may involve removing duplicate or irrelevant instances, handling missing values, and balancing the dataset, if necessary, then extracting features from the per-

processed dataset that can effectively capture characteristics of phishing attacks. These features include URL components, and train dataset using bagging ensemble model with those algorithms. This involves splitting the dataset into training and validation sets, configuring the ensemble model, and fitting the model to the training data. Finlay evaluates the trained model's performance on a separate test dataset or through cross-validation techniques. Assess various metrics such as accuracy, precision, recall, F1-score, and false positive rate to gauge the model's effectiveness in phishing detection, once the model has been evaluated and its performance metrics have been obtained it will be classified either as a phishing website, or a legitimate and the phishing detection process comes to an end. Figure 4.2 explains the basic steps for our proposed model:

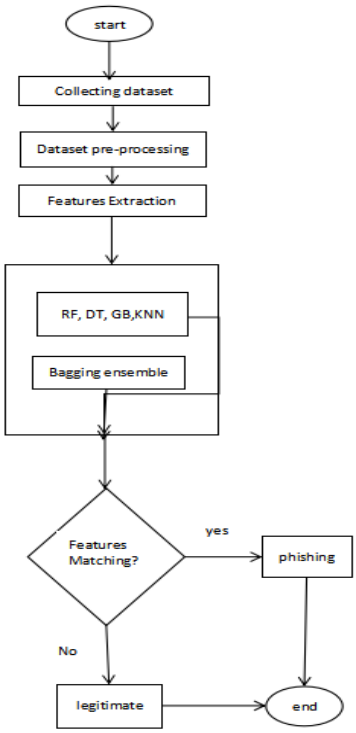


Figure 4.2. Our proposed model flowchart

## 4.5. MODEL TRAINING

The model training phase in phishing detection involves teaching a machine learning model to recognize patterns and make accurate predictions. It begins by providing

the model with a labeled dataset, where each instance is classified as either phishing or legitimate. The model then learns from this data by adjusting its internal parameters using optimization algorithms. The goal is to create a model that can generalize well and accurately classify unseen instances. The model's performance is evaluated using appropriate metrics, and if necessary, the model is fine-tuned to improve its accuracy. Finally, the trained model is deployed in a production environment for real-time phishing detection.

### 4.6. MODEL EVALUATION

Once the ML algorithms are trained, their performance is evaluated using the testing set. There are several metrics extracted through the confusion matrix to evaluate the machine learning models' performance. In this study, we evaluate the presented models by the most common evaluation metrics: Accuracy, Recall, Precision, and F1-score [24]. A confusion matrix is a performance measurement tool used in machine learning classification tasks [25]. It is a table that summarizes the performance of a classification model by displaying the counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). These numbers help evaluate the accuracy and effectiveness of the model, findings will be demonstrated in the next chapter.

### 4.7. EXPERIMENTAL STEPS

This section describes the implementation requirements in software and hardware used to gain the reported results.

- **Software:** starting with Kali Linux as the operating system used in this research, Kali Linux is mostly used for penetration testing and cyber security in general, because it comes with a lot of preinstalled tools that are used to detect various types of cyber-attack.
- **Programming languages:** to implement the classifiers in this research python programming languages are used including Python-based libraries

such as sklearn, numpy which are utilized to complete the process of the implementation phase.

- **Scikit-Learn:** scikit-learn, commonly referred to as sklearn, is a popular machine learning library in Python. It provides a wide range of algorithms and tools for various machine learning tasks, such as classification, regression, clustering, and dimensionality reduction. The library is built on top of other scientific Python libraries, such as NumPy, SciPy, and matplotlib, and offers a unified interface for applying machine learning techniques.
- **Numpy:** Numpy is a powerful numerical computing library in Python that provides efficient data structures and functions for handling large arrays and metrics. Uses of Numpy Python code for data preparation and splitting data.
- **Hardware:** Successful execution of the implantation of the classifiers is a demanding process and requires a high-performing device to finish the execution errors-free, the following are the specifications of the computer in which all the experiments were carried out:
    - CPU: Intel Core i5 $10^{th}$ generation.
    - GPU: NVIDIA GEFORCE RTX 3060 with 6 GB memory.
    - RAM: DDR4 16 GB.

# PART 5

# PERFORMANCE EVALUATION OF OUR PROPOSED MODEL

In this chapter, we discuss the findings and analysis of the built model to detect phishing attacks using ML algorithms and bagging ensemble. We get into the model performance and the results of the evaluation metrics, and also discuss the outcomes from the application of these ML algorithms stand-alone and with bagging ensemble on the dataset. The main objective of this chapter is to evaluate the efficacy of ML algorithms in accurately classifying phishing cases and differentiate them from legitimate ones. We analyze the results obtained through comprehensive evaluations and explore the strengths and weaknesses of each algorithm.

## 5.1. PERFORMANCE ANALYSIS OF CLASSIFIERS

The comprehensive evaluations of ML algorithms with bagging ensembles for phishing detection provide valuable insights into their performance and highlight their respective strengths and weaknesses.

### 5.1.1. PERFORMANCE ANALYSIS WITHOUT ENSEMBLES

This is an analysis of the results obtained and an exploration of the characteristics of each algorithm according to the used dataset. Table 5.1 shows the results of the trained model without bagging ensembles.

As shown in Figure 5.1, the DT model achieved an accuracy of 93%, additionally, it gave a precision of 93%, which means that 93% of the websites predicted as phishing were indeed phishing. The recall value of 93% means that the model correctly identified 93% of the actual phishing websites. The F1 score of 93% reflects a

balance between precision and recall, offering an overall measure of the model's effectiveness.

Table 5.1. Algorithms results without bagging

| ML algorithm | Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|
| **DT** | 93.41 | 93.33 | 93.33 | 93.35 |
| **RF** | 96.12 | 97.23 | 96.21 | 96.65 |
| **KNN** | 94.15 | 95.45 | 92.15 | 94.16 |
| **GB** | 95.88 | 96.01 | 95.82 | 95.90 |

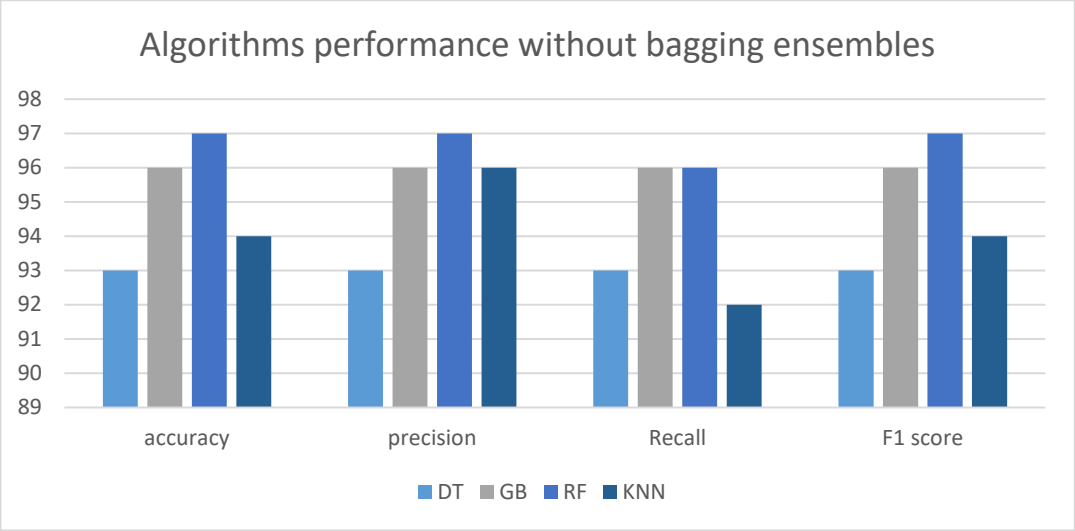


Figure 5.1. Algorithms performance without bagging

In comparison, the RF algorithm displayed a higher accuracy of 96%. The precision value of 97%, recall value of 96%. The F1 score of 97% demonstrates a strong balance between precision and recall for the RF model. For KNN model achieved an accuracy of 94%, additionally, it gave a precision of 96%, which means that 96% of the websites predicted as phishing were indeed phishing. The recall value of 92% means that the model correctly identified 92% of the actual phishing websites. The F1 score of 94% reflects a balance between precision and recall, offering an overall

measure of the model's effectiveness. GB model achieved an accuracy of 96%, additionally, it gave a precision of 96%. The recall value of 96% means that the model correctly identified 93% of the actual phishing websites. The F1 score of 96% reflects a balance between precision and recall, offering an overall measure of the model's effectiveness.

Overall, the RF model outperformed the DT model in terms of accuracy, precision, recall, and F1 score. It achieved higher values in all these metrics, giving higher performance in detecting phishing websites while minimizing both false positives and false negatives.

### 5.1.2. PERFORMANCE ANALYSIS WITH BAGGING ENSEMBLES

By adding bagging ensemble to the ML algorithm models for phishing detection, we noticed improvements in their performance by the evaluation metrics.

Table 5.2. Algorithms result with bagging

| ML algorithm | Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|
| **DT** | 95.63 | 96.21 | 94.84 | 95.55 |
| **RF** | 97.61 | 97.47 | 96.22 | 96.71 |
| **KNN** | 94.12 | 95.69 | 92.43 | 94.12 |
| **GB** | 95.98 | 96.66 | 96.23 | 95.90 |

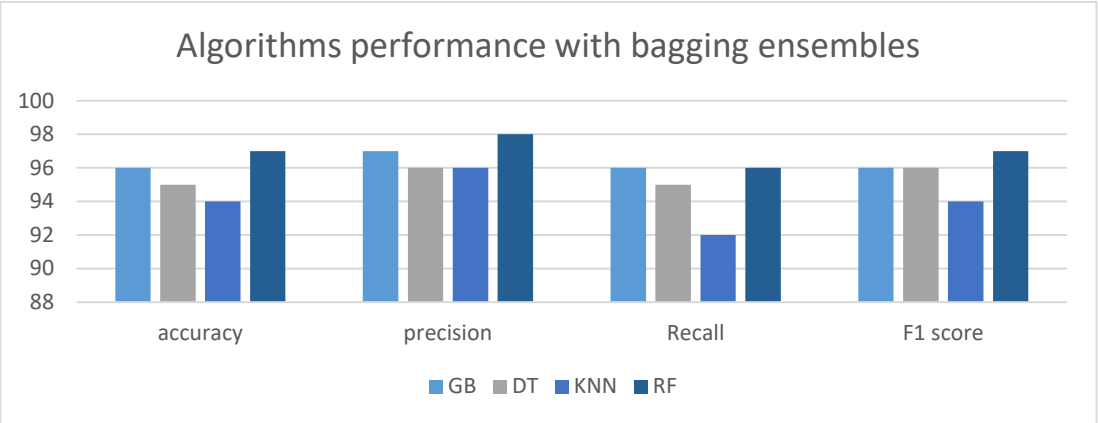


Figure 5.2. Algorithms performance with bagging

The performance of DT with bagging ensemble improved because of the strengths of multiple trees. The standalone trees can focus on different subsets of features or data patterns. The averaging of these predictions helps to reduce the impact of individual tree errors, leading to improved accuracy, precision, recall, and F1 score. The DT model achieved an accuracy of 96%, indicating that it correctly classified 96% of instances, the RF model displayed a higher accuracy of **98%**, implying that it achieved a higher proportion of correct classifications.

However, compared to DT with bagging the improvements might be less for KNN and GB. The performance evaluation of RF with bagging can still yield excellent results, RF already integrates randomness during the sample and the feature selection, which helps reduce overfitting and improve generalization. The benefits of bagging an ensemble, while still valuable may not be as obvious as they are for DT, which can be more apt to overfitting. Figure 5.3 shows the confusion matrix for those algorithms:

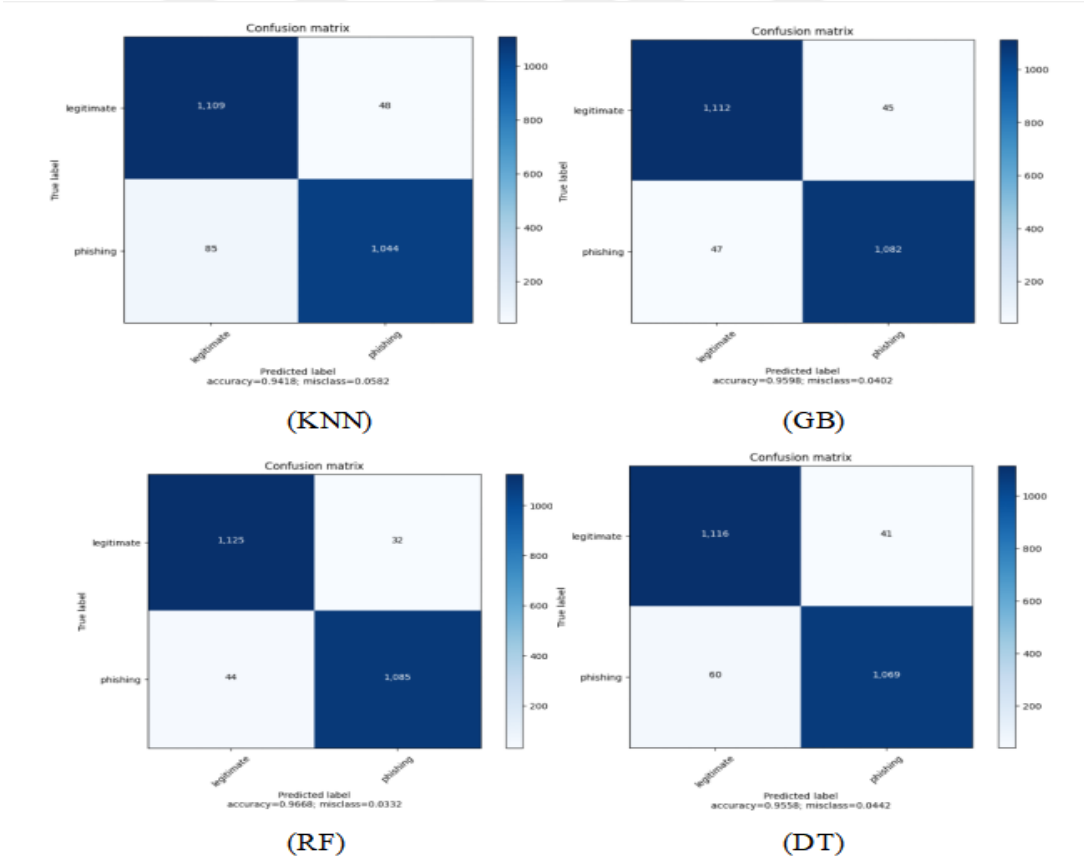


Figure 5.3. Confusion metrics of ML algorithms

## 5.2. EFFECTIVENESS OF THE MODEL:

Comparing this research to previous work like [9], [10], [11], [12] they may have had some limitations. These limitations probably include a limited feature set to extract that wasn't enough to detect phishing websites. Additionally, scalability could be one of these limitations, as in previous methods the ability to handle large dataset or real-time detection was not effective. Generalization concerns also could arise if the models could not adapt and perform well on phishing attack diversity. Some approaches might have been domain or language-specific, lacking the ability to generalize to broader contexts. Furthermore, inadequate evaluation of diverse dataset could limit our understanding of the true performance and robustness of the proposed methods. Overcoming these limitations is crucial to developing more accurate, scalable, adaptable, and effective phishing detection approaches.

Table 5.1. Previous work and proposed model accuracy

| Previous work | Accuracy |
|---|---|
| **[9]** | 96.72% |
| **[10]** | 95.47% |
| **[11]** | 96.85% |
| **[12]** | 96.72% |
| **Our proposed model** | **97.61%** |

Using bagging ensemble with DT and RF can provide several advantages and enhance the effectiveness of these models compared to other solutions in various ways:

- Reduced Variance: Bagging can reduce the variance of individual models by

creating multiple subsets of the data through random sampling with replacement. This helps in reducing overfitting and improving the generalization ability of DT and RF. In contrast, standalone DT are prone to high variance and can easily overfit the training data.

- Improved Accuracy: Bagging ensembles combine predictions from multiple DT or RF models, leading to more accurate overall predictions. By aggregating the outputs of multiple models, the ensemble approach helps to reduce the impact of individual model biases and errors.
- Enhanced Robustness: Bagging ensemble methods, such as those applied to DT and RF, provide robustness against outliers and noisy data. By training models on different subsets of the data, the ensemble can handle diverse patterns and reduce the influence of individual outliers or noisy instances.
- Feature Importance: Bagging ensembles with DT or RF can provide insights into feature importance. By examining the importance assigned to each feature across the ensemble, it becomes possible to identify the most relevant features for classification or regression tasks. This information can be valuable for feature selection or understanding the underlying patterns in the data.

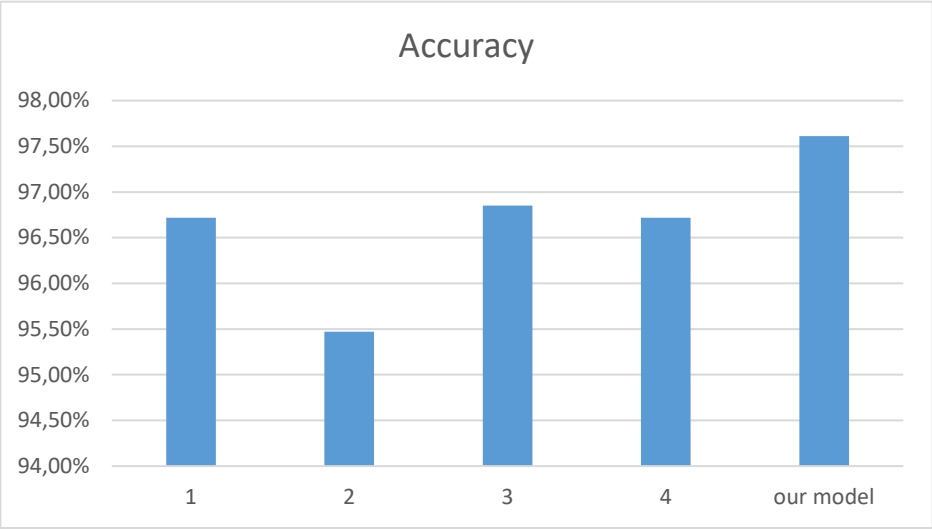


Figure 5.4. Previous work and proposed model accuracy

## 5.3. OPEN CHALLENGES

- Adaptability to new types of phishing attacks: Phishing attacks are constantly evolving, and new techniques and strategies are being developed to bypass anti-phishing solutions.
- Robustness against evasion attacks: Phishers may use evasion techniques to bypass the detection algorithms, such as using obfuscation techniques to hide the phishing content. The proposed approach needs to be robust against such evasion attacks and able to identify and block phishing attempts even when the phishing content is obfuscated.
- Generalization across different languages and cultures: The approach needs to be able to generalize across different languages and cultures, as phishing attacks are not limited to a particular language or culture. It is important to ensure that the approach is effective in detecting phishing attacks in different languages.

## 5.4. FUTURE WORK

Firstly, exploring the incorporation of domain-specific knowledge or expert systems can enhance the ensemble's ability to detect targeted phishing attacks. Secondly, investigating the utilization of deep learning techniques, such as convolutional neural networks or recurrent neural networks, within the ensemble framework can capture complex patterns in phishing websites. Additionally, integrating dynamic feature updating mechanisms that adapt to evolving phishing techniques and emerging attack vectors can enhance the ensemble's resilience. Furthermore, exploring the use of ensemble diversity measures, such as feature diversity or classifier diversity, can improve the robustness and generalization capabilities of the ensemble. Lastly, considering the integration of explainable AI techniques, such as rule-based systems or feature importance analysis, can enhance the interpretability and trustworthiness of the ensemble model. By focusing on these areas, future research can advance phishing detection using bagging ensemble with RF and DT, leading to more accurate and effective models in combating evolving phishing threats.

## 5.5. CONCLUSION

This research has investigated the effectiveness of ensemble learning and the efficiency of this type of machine learning by showing the strengths of ML algorithms to improve accuracy and strength in identifying phishing websites from the legitimate. However, the performance of the model may vary depending on the quality and relevance of the selected features, highlighting the importance of feature engineering and selection. The findings of the research indicate that using bagging with random forests tends to show greater improvements in performance evaluation compared to using it with the other ML algorithms due to the built-in mechanisms for reducing overfitting and improving generalization, which makes the additional impact of bagging somewhat less significant. Compared to other solutions, such as using standalone ML algorithms, bagging ensembles generally offer improved accuracy, robustness, and better generalization. Nevertheless, it is essential to interpret these findings with caution. The effectiveness of the model heavily relies on the quality and diversity of the training data, the chosen features, and the evaluation metrics employed.

In conclusion, while phishing detection using bagging ensemble with ML algorithms shows promise, addressing challenges related to dataset quality, feature selection, and comprehensive evaluation is crucial to further enhance the model's effectiveness and ensure its practical applicability in real-world phishing detection scenarios.

## REFERENCES


1. Wang, Jingguo, Yuan Li, and H. Raghav Rao. "***Coping responses in phishing detection: an investigation of antecedents and consequences***." Information Systems Research 28.2 (2017): 378-396.

2. Bin Heyat, Md Belal, et al. "***A novel hybrid machine learning classification for the detection of bruxism patients using physiological signals.***" Applied Sciences 10.21 (2020): 7410.

3. Jones, L. D., et al. "***Artificial intelligence, machine learning and the evolution of healthcare: A bright future or cause for concern.***" Bone & joint research 7.3 (2018): 223-225.

4. Zuraiq, AlMaha Abu, and Mouhammd Alkasassbeh. "***Phishing detection approaches.***" 2019 2nd International Conference on New Trends in Computing Sciences (ICTCS). IEEE, 2019.

5. Patil, Srushti, and Sudhir Dhage. "***A methodical overview on phishing detection along with an organized way to construct an anti-phishing framework.***" 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS). IEEE, 2019.

6. Internet: Microsoft learn blacklist phishing detection, **https://learn.microsoft.com/**

7. Jain, Ankit Kumar, and Brij B. Gupta. "***A machine learning based approach for phishing detection using hyperlinks information***." Journal of Ambient Intelligence and Humanized Computing 10 (2019): 2015-2028.

8. Cui, Qian, et al. "***Phishing attacks modifications and evolutions***." Computer Security: 23rd European Symposium on Research in Computer Security, ESORICS 2018, Barcelona, Spain, September 3-7, 2018, Proceedings, Part I 23. Springer International Publishing, 2018.

9. Lakshmanarao, A., P. Surya Prabhakara Rao, and MM Bala Krishna. "***Phishing website detection using novel machine learning fusion approach.***" 2021 international conference on artificial intelligence and smart systems (ICAIS). IEEE, 2021.

10. O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "***Machine learning based phishing detection from URLs,***" Expert Systems with Applications, vol. 117, pp. 345–357, 2019.

11. J. Mao, J. Bian, W. Tian, S. Zhu, T. Wei, A. Li, and Z. Liang, "***Phishing page detection via learning classifiers from page layout feature***," Journal on Wireless Communications and Networking, 2019.

12. A. K. Jain and B. B. Gupta, "***Towards detection of phishing websites on client-side using machine learning based approach,***" Telecommunication Systems, vol. 68, no. 4, pp. 687–700, 2018.

13. Shahrivari, Vahid, Mohammad Mahdi Darabi, and Mohammad Izadi. "***Phishing detection using machine learning techniques.***" arXiv preprint arXiv:2009.11116 (2020).

14. Wang, Shan, et al. "***Deep learning-based efficient model development for phishing detection using random forest and BLSTM classifiers."*** Complexity 2020 (2020): 1-7.

15. A. Niakanlahiji, B.-T. Chu, and E. Al-Shaer, "***Phishmon: A machine learning framework for detecting phishing webpages***" in IEEE International Conference on Intelligence and Security Informatics, 2018, pp. 220–225.

16. Save, Prajal, et al. "***A novel idea for credit card fraud detection using decision tree.***" International Journal of Computer Applications 161.13 (2017).

17. Internet:" decision tree algorithm", ***https://www.smartdraw.com/***

18. Internet:" Random Forest algorithm", ***https://medium.com/***

19. Murty, M. Narasimha, and V. Susheela Devi." **Introduction to pattern recognition and machine learning**". Vol. 5. World Scientific, 2015.

20. Manjeri, Akshay Sushena, et al. "***A machine learning approach for detecting malicious websites using URL features.***" 2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA). IEEE, 2019.

21. Internet: "phishing_dataset", ***https://www.kaggle.com/dataset/eswarchandt/phishing-website-detector.***

22. Alswailem, Amani, et al. "***Detecting phishing websites using machine learning.***" 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS). IEEE, 2019.

23. Safi, Asadullah, and Satwinder Singh. "***A systematic literature review on phishing website detection techniques***." Journal of King Saud University-Computer and Information Sciences (2023).

24. Machado, Lisa, and Jayant Gadge. "***Phishing sites detection based on C4. 5 decision tree algorithms***." 2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA). IEEE, 2017.

25. Alkawaz, Mohammed Hazim, et al. "***A comprehensive survey on identification and analysis of phishing website based on machine learning methods.***" 2021 IEEE 11th IEEE Symposium on Computer Applications &

## RESUME

Nuha Abubaker IBRAHEEM finished her elementary education in Sudan. She completed her high school education at KRC Private High School, after that, she started an undergraduate program at Omdurman Islamic University, Department of Computer Science in 2012. Then in 2020, To complete an M.Sc. education, she moved to Karabuk University.