National College of Ireland

# Detecting Real Time Phishing URL's using URL Cleaning Technique

MSc Research Project
Cybersecurity

Vikas Naidugari
Student ID: x22237020

School of Computing
National College of Ireland

Supervisor: Raza Ul Mustafa

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

| | |
|---|---|
| **Student Name:** | Vikas Naidugari |
| **Student ID:** | x22237020 |
| **Programme:** | MSc. in Cybersecurity **Year:** 2023-2024 |
| **Module:** | Research Project |
| **Supervisor:** | Raza Ul Mustafa |
| **Submission Due Date:** | 16 Sep 2024 |
| **Project Title:** | Detecting Real Time Phishing URL's with URL Cleaning Technique |
| **Word Count:** | 7098 **Page Count:** 20 |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

**Signature:** Vikas Naidugari

**Date:** 16 Sep 2024

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Detecting Real Time Phishing URL's using URL Cleaning Technique

Vikas Naidugari
x22237020

**Abstract**

This paper seeks to establish the best approach to detecting the phishing URLs in the ever-shifting cybersecurity environment in order to enhance the safety of online operations. In this report, the URLs are processed and analysed using the data classification and feature extraction approaches that are used to detect the existence of phishing threats. URL related features are present in the dataset such as length of the URL, structure of domains, and other important attributes necessary for evaluation of phishing. The fundamental purpose of this work is to increase the reliability of phishing filters, which are crucial for preserving confidentiality and users' trust. The model contains a comprehensive preprocessing stage that has relevant features that are derived from raw URLs. Subsequently, we use Logistic Regression, Support Vector Machine, Kneighbor, Random Forest, Decision Tree, and Naïve Bayes as a suitable machine learning model to predict the websites as either legitimate or phishing. The analysis shows that the accuracy for the different predictive algorithms used for classification of the URLs based on the processed data set, is high with accurate classification of the phishing and genuine URLs. The research shows that methods that were used in feature extraction proved to be successful. In addition to the strong capability that has been proved by this research for enhancing online security through machine learning for enhanced phishing detection. This report will be a rich source of ideas for those interested in this field for future investigations and applications.

**Keywords:** *Phishing Detection, URL Cleaning, Machine Learning, Naïve Bayes, Random Forest, Decision Tree.*

# 1 Introduction

In the present world involving electronic devices for personal and business purposes, phishing attacks have become more frequent. As per the Anti-Phishing Working Group, in second quarter 2024 the APWG observed 877,536 phishing assaults, while the number of reported phishing cases has also been increasing which threatens the security of contents and also makes the user vulnerable and distrustful about the websites. Most frequently phishing attacked sectors are via Phone call and text messages (for payment service users), social media platform, 72.4% of Gmail account users were reported for Business Email Compromise (BEC) scams ('APWG | Phishing Activity Trends Reports'). Phishing is a technique that uses disguised links to manipulate users into providing their login info and other personal details. According to the Anti-Phishing Working Group (APWG), there has been an increase in phishing attacks, and the need to enhance the measures for detecting these attacks cannot be expanded. The name "phishing attacks" is derived from "fishing" for victims. Scammers, also known as "phishers," fool users into accessing fake websites that resemble the legitimate site. For the past two years, researchers have become interested in such threats. Obviously, all these web pages should have distinct URLs from the authentic page but with similar graphical

interfaces. The expert user can almost always identify fake pages by viewing their URLs. The target or end users generally do not view the entire url or url's of sites received through social media, emails, or text messages. Phishers utilize fake url's to take away victims financial and personal username and password details. If the user is deceived into accessing a fake website that looks real, then the person may end up disclosing their personal information (Gupta, Arachchilage and Psannis, 2018). Researchers have discovered a lot of reasons have been found out by researchers as to why people who use computers are vulnerable to frauds like phishing. These were the reasons that came into light (Such as -- i) The user does not have an overall understanding of Url's. ii) People using the internet do not know how to identify legitimate websites. iii) The redirect Url's or Tiny Url's hide the complete website address from a user. iv) Unknown sites are often viewed because either people do not have the time or simply do not know to check the URL).

Since phishing attacks are becoming more common and complex, there is a need to improve the detection methods. The popular methods like blacklists, and heuristic-based filters cannot detect new or advanced forms of phishing attacks because their algorithms have been designed with traditional forms of attacks in mind. Using ML in an attempt to improve the identification of phishing threats is a viable approach to detecting such threats because ML has the capability to detect patterns in large sets of data that may otherwise go undetected by traditional algorithms.

The ability to identify a phishing URL is essential in securing an organization against cyber threats. The main concentration would be on feature extraction and machine learning algorithms on a dataset of phishing URLs. This method is appreciated due to its clarity, interpretability and applicability to the cases with binary outcomes. In terms of the performance, the assessment will be made based on modelling in relation to identifying the phishing URLs to determine whether this approach is efficient in performing this function (Verma and Das, 2017).

**Research Aim:** To develop and evaluate an efficient machine learning model for accurate detection of phishing URLs, enhancing cybersecurity measures and user protection online.

## Objectives:

- To identify and extract relevant features from URLs that effectively distinguish between legitimate and phishing websites.
- To design and implement a robust machine learning model capable of classifying URLs as either phishing or legitimate with high accuracy.
- To assess the performance and reliability of the developed model in real-world scenarios and compare it with existing phishing detection methods.

## Research Questions:

- What are the most significant URL features that contribute to effective phishing detection?
- How does the choice of feature extraction techniques impact the overall performance of the phishing detection model?
- To what extent can a machine learning-based approach improve upon traditional rule-based methods for identifying phishing URLs?
- How does the proposed model perform in terms of accuracy, precision, recall, and F1-score?

- What are the potential limitations and challenges in implementing the developed model for real-time phishing URL detection, and how can they be addressed?

# 2 Related Work

This section incorporates a literature review of previous studies on phishing detection and focuses on applying the machine learning approach. Phishing, in which users are tricked into entering their personal information on fake websites or into opening phishing emails, has evolved into a complex practice. This has become a massive problem most of the time when it comes to detecting phishing attempts since it has caused extensive exploration of most of the kinds of machine learning. In this section, we provide a brief overview of major developments in this line of research, comparing and contrasting their approaches, advantages, and disadvantages.

According to Verma and Das (2017), the URL detection has highlighted the importance of URL cleaning and feature extraction. Several techniques have been proposed aiming at preprocessing URLs, such as normalization, encoding standardization, and noise removal. Feature extraction methods have ranged from staraightforward lexical analysis of URL strings to more advanced host-based features and even content attributes. Some of the previous works have considered n-gram analysis of URL subparts and some other works have tried to extract the semantic information with the help of natural language processing techniques. Previous works stressed on identifying the most suitable features that would enable the system to distinguish between legitimate and phishing URLs. Furthermore, studies have focused on the effects of feature extraction and feature space compression on the model's learning and detection performance in an attempt to find the best compromise between these two factors.

Basit et al. (2020) did a comprehensive review on the application of ensemble learning techniques in phish detection. They considered approaches like Random Forests and Gradient Boosting in which several decision trees work in parallel to enhance the rate of classification. Their study also shows that ensemble methods are very accurate in detecting phishing websites, and this is possible because of the use of the diversity of decision trees so as to fully maximize in making the best decision. The main survey of Basit's paper speaks of their all-round assessment of ensemble methods, which ranked much above traditional classifiers. On the same note, the study also covered limitations whereby ensemble versions proved appreciably more complex and time-consuming to train as compared to models that operate independently. These can prove to be a disadvantage in applications based on real-time phishing detection, where timely detection is the top priority.

Yerima et al. (2020) described deep learning approaches propose the use of CNNs for phishing detection. CNNs can learn advanced details about the data sets features and can be effective when identifying intricate phishing modalities. Yerima also compared CNNs to traditional machine learning algorithms in terms of detection rates and noted that CNNs

give high detection rates because they have the ability to extract features from raw data automatically. This is an advantage of deep learning because it requires little input of the feature that needs to be extracted compared to conventional models. But, similarly to other researchers, Yerima stated some issues; for example, large labeled datasets were required, and training of deep learning models was quite computationally intensive. Such requirements can pose a challenge in implementing CNN- based solutions in scenarios that may have restricted resources.

In their experiment, Vanitha and Vinodhini (2019) used a simpler form of classification known as logistic regression to identify phishing attacks. In their study, they focus on the simplicity of the logistic regression model because this is the model that is easiest to apply compared to the other models. In a study by Vanitha, it was shown that, when used in conjunction with the right set of features as URL length and the number of characters like '@', the logistic regression model offers a decent accuracy. Their strategy is also a strength in the sense that it is simple and can be very easily implemented, and the results can be easily interpreted to make decisions. It also pointed out that using logistic regression might be problematic for more complicated forms of phishing that utilize firmer features. This limitation suggests that logistic regression works well for relatively basic instances but it may not be helpful for identifying complex phishing URLs.

Feng et al. (2024) have examined other forms of integrating machine learning types, like the SVMs and Neural Networks. Their work focused on using multiple models to try and improve the model's performance in detecting phishes. SVMs assist in cases where the data cannot be separated by a linear hyperplane, while involving Neural Networks aids in making complex pattern detections. The hybrid approach results in a model with higher complexity and potential issues with interpretability/overfitting/underfitting. Adopting and integrating such hybrid models can be more challenging than employing single models.

Balogun et al. (2021) investigated the effect of feature engineering for phishing detection using various modelling techniques, as well as feature extraction for URL cleaning. Other scholars have consistently noted the fact that URL cleaning and feature selection are critical in enhancing the performance of phishing detection models. Some of the examples of techniques that may help to improve the result of logistic regression models are recursive feature elimination (RFE) and principal component analysis (PCA) if applied to cleaned URL data. The benefit within this approach is more focused on the identification of features, which is critical to model performance. URL cleaning techniques may include removing unnecessary parameters, standardizing protocols (e. g. , converting all URLs to use 'http:Blogs contain various symbols like '/' (greater than or equal to) and slashes '//' as well as slashes '/' (less than or equal to). Some of the features that can be extracted from cleaned URLs include the number of characters in the domain, usage of symbols, number of sub-domains, and depth of the path. Poor URL cleaning or inadequate feature selection can result in models that struggle to identify sophisticated phishing attacks. Therefore,

ongoing refinement of URL cleaning techniques and feature extraction methods is essential to keep pace with evolving phishing tactics.

Shahrivari et al. (2022) have provided some studies on hybrid models, which include logistic regression and anomaly detection techniques. Their work was done with the objective of overcoming the drawbacks of earlier phishing detection techniques and included an anomaly detection strategy to detect newer and more complex schemes. It is possible to improve the detection of phishing attacks that are not recognizable by the initial models through integration of difference detection with logistic regression. This is a better solution than the previous method because it incorporates the dynamic aspect of phishing. However, the proposed hybrid model exposes the additional level of model complexity, which may turn into problems with model implementation or management; thus, the issue of the correct balance of the model's complexity and practicability arises again.

## 2.1 Summary of findings

The analysed literature highlights numerous works reporting various classification algorithms in the phishing detection problem space, with their advantages and disadvantages. Combining methods with deep learning techniques also gives high accuracy, but at the higher cost of computational power. As a result, approaches like the involvement of multiple classifiers and the use of other approaches, including SVM and anomaly detection, that are combined with logistic regression to overcome these drawbacks also create more complications.

## 2.2 Justification for Research

The increasing complexity of phishing attacks poses a significant threat to online security, making efficient and accurate detection crucial. This research is justified by the urgent need for improved phishing URL detection methods to protect users from financial loss, data breaches, and identity theft. Traditional rule-based systems often struggle to keep pace with evolving phishing techniques, demanding more adaptive approaches. Machine learning offers a promising solution, capable of identifying complex patterns and adapting to new threats. By developing an advanced model for phishing URL detection, this research aims to enhance cybersecurity measures, reduce the success rate of phishing attempts, and contribute to a safer online environment for individuals as well as for organizations.

## 2.3 Detailed Review of Significant Literature

| Author(s) | Year | Methodology | Strengths | Weaknesses | Key Findings |
|---|---|---|---|---|---|
| Verma and Das | 2017 | Feature extraction | Better handling of dataset | Sometime inappropriate feature extraction leads for the issue with results | URL cleaning can be performed for better modelling outcomes. |

| | | | | | |
|---|---|---|---|---|---|
| Basit et al. | 2020 | Ensemble Learning (Random Forests, Gradient Boosting) | High accuracy due to aggregation of decision trees. | Increased computational complexity and training time. | Improved phishing detection with ensemble methods. |
| Yerima et al. | 2018 | Deep Learning (CNNs) | Superior performance in detecting complex phishing attacks. | Requires large datasets and significant computational resources. | CNNs outperform traditional methods in detection rates. |
| Vanitha et al. | 2019 | Random Forest | Simplicity and interpretability; easy implementation. | May struggle with sophisticated phishing tactics. | Effective with simpler phishing attacks; limited for complex cases. |
| Feng et al. | 2020 | Hybrid Models (+ SVMs, Neural Networks) | Enhanced performance by combining strengths of multiple models. | Increased model complexity and maintenance challenges. | Hybrid models improve detection rates and handle complex patterns. |
| Balogun et al. | 2021 | Feature Engineering (RFE, PCA) | Focus on feature selection improves logistic regression performance. | Dependence on quality of features; limited for advanced phishing. | Feature selection crucial for optimizing logistic regression. |
| Shahrivari et al. | 2022 | Hybrid Models (Anomaly Detection) | Enhanced detection of novel and sophisticated attacks. | Complexity in integration and implementation. | Hybrid approach addresses evolving phishing tactics effectively. |

# 3 Research Methodology

A phishing website is a social engineering strategy that is used to mimic legitimate webpages and uniform resource locators. Uniform Resource Locator, or URL, is the most standard mode through which phishing attacks take place. Phisher has full control over the URL's sub-domains. URLs can be easily manipulated by the phisher since they are made up of file components and directories (Das *et al.*, 2020).

## 3.1 Research Architecture

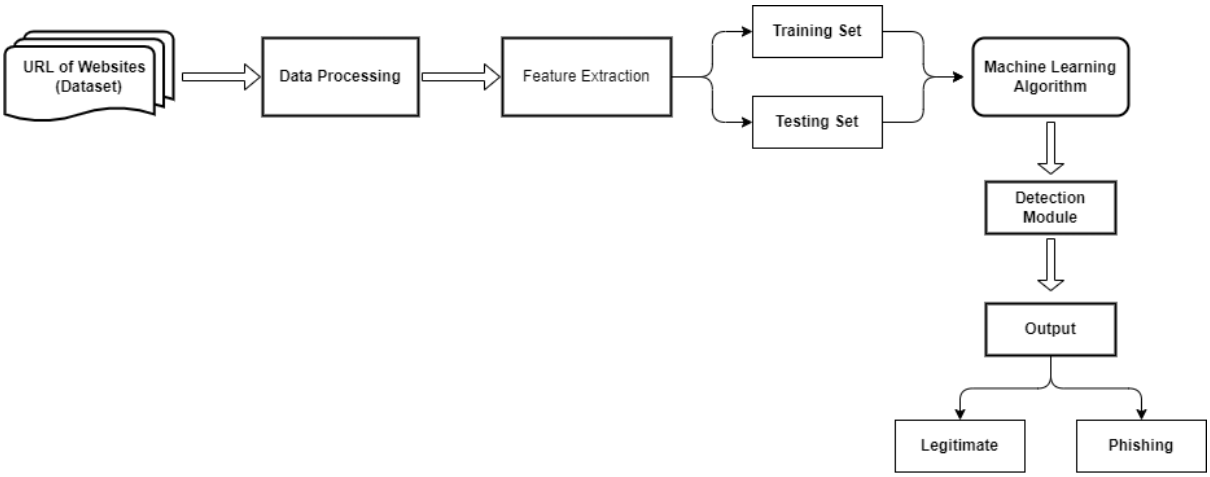



Fig 1 Research Framework

## 3.2 Language

I worked primarily on Python. While writing, in case one is to use machine learning, then Python is the language to use. With only an import, it comes with a number of machine learning libraries ready to go. Due to the large library of machine learning modules, Python is widely used by developers across the world when working with machine learning. The existence of a robust community ensures that new functionality is included in each update iteration of the Python developers.

## 3.3 Collecting Data

This dataset for this research was obtained from phishing and legitimate URLs repositories which are easily accessible. This dataset is composed of URL's characteristics these include: URL length and a number of dots, hyphens and other structures. The data was collected to contain at least half of them as phishing and the other half as legitimate to make sure the training was not skewed.

## 3.4 Data Preprocessing

Preprocessing is relevant to the stage at which the data will be processed by the machine learning models. This entails feature preprocessing in which data is cleaned and features transformed: missing values are also managed.

## 3.5 Techniques Applied

1. Data Cleaning: To delete or update the wrong record entered in the database.
2. Encoding Categorical Features: converting categorical data to numerical data.
3. Scaling Numerical Features: Normalizing the range of numerical attributes.
4. Handling Missing Values: Finding strategies and methods for handling the missing values.
5. Encoding: Converting the categorical values present in the status column into the numerical form so that our model can perform better.
6. Python Libraries: pandas, scikit-learn (*API Reference*), numpy.

## 3.6 Data handling and processing

In the process of developing a phishing URL detection model, several key stages are implemented. The exploratory data analysis (EDA) phase utilizes Matplotlib to visualize and understand the dataset's characteristics. The feature engineering stage is crucial, where columns are classified into numerical and categorical features. This classification guides the creation of preprocessing pipelines using Scikit-learn, which handle missing values and perform one-hot encoding for categorical variables.

## 3.7 Feature extraction

The feature extraction process is particularly important, focusing on deriving relevant attributes from the URLs. This may include lexical features such as URL length, number of subdomains, presence of special characters, and use of URL shortening services. Host-based features like IP address, domain age, and WHOIS information might also be extracted. Additionally, content-based features could be derived from the webpage itself, such as the presence of login forms or suspicious JavaScript (Zhang, Zhao and LeCun, 2016).

It refers to various kinds of features used by machine learning algorithms in the process of academic study detection (Buber, Demir and Sahingoz, 2017).

- Address bar based features
  - Domain of the URL
  - IP Address of the URL
  - "@" Symbol in URL
  - Special Character Analysis (such as %20 – for space)
  - Look-alike character detection (such as ‘0’ and ‘O’, ‘l’ and ‘1’)
  - IP Address Substitution (e.g., https://192.9.68.88/nai.html instead of https://vikasnaidugari/nai.html)
  - Length of URL
  - Depth of URL
  - Redirection "//" in URL
  - Http/Https in Domain name
  - Shortening of the URL
  - Prefix or Suffix "-" in Domain
- Domain based features
  - DNS Record
  - Web Traffic
  - Age of Domain
  - End Period of Domain

Following feature extraction, feature scaling is applied using StandardScaler from Scikit-learn to normalize the feature set. The model building phase involves splitting the data into training and testing sets, and implementing some of the machine learning models for classification. Finally, the model's performance is evaluated using the accuracy score metric from sklearn.metrics. This comprehensive approach ensures a robust feature set and an effective model for phishing URL detection.

# 4 Design Specification

## 4.1 Techniques

- Loading the data by using Panda's library.
- Clean the data and handle the missing values.
- Plot the data to get an understanding of the data.
- Converting categorical features using One Hot Encoding and Label Encoding.
- Scale the feature in a specific range using the Standard Scaler.
- Splitting the data into training and testing sets to evaluate the model's performance.
- Used Logistic Regression, SVM (Wu, Qiang & Zhou, Ding-Xuan. (2006)), Random Forest (Biau and Scornet, 2016), Naïve Bayers (Rish, Irina. (2001)), Decision Tree(Swain and Hauska, 1977), and Kneighbor for this classification problem.
- To evaluate the model's performance, determine its accuracy.

## 4.2 Machine learning algorithms

The logistic regression model is a basic yet commonly utilized classification algorithm in machine learning and statistics used for binary classification mainly. This is used to predict the likelihood of a categorical dependent variable given one or more independent variables. In the context of phishing detection, logistic regression can be utilized to classify URLs into two categories - legitimate or phishing.

In order to detect phishing websites, we assessed a number of machine learning models, such as Logistic Regression, SVM, KNN, Random Forest, Decision Tree, and Naïve Bayes. A dataset of labelled webpages was used to train and evaluate each model. Performance measures including recall, accuracy, and precision were used to evaluate and contrast how well each model performed in identifying phishing- and legitimate-content websites.

In our analysis Logistic Regression, SVM, KNN, Random Forest, Decision Tree, and Naïve Bayes models were considered to compare various types of machine learning techniques in the context of phishing detection. This choice includes models of various degrees of complexity and therefore offers a broad outlook at its performance. SVM and Random Forest are good for non-linear & interactive features. Logistic Regression and Naïve Bayes provides interpretability. KNN offers a non-parametric approach, and Decision Tree offers a simple yet robust classification technique. The employment of performance measures such as recall, accuracy, and precision is effective in addressing this by factoring in the capacity each of the models has under different situations to identify phishing sites and differentiate between them and other genuine sites.

## 4.3 Data Cleaning

### 4.3.1 Handling Missing Values

Data completion is necessary to ensure that the data set used for model training does not contain incomplete information. There are several ways of handling missing values, including using taken average values or using models for predictions.

### 4.3.2 Feature Encoding

Categorical data transformation into numerical ones. For instance, a categorical variable such as "status" (legitimate and phishing) will require binarization or encoding to include values of 0 and 1. Standardizing the features such that they have comparable scales.

### 4.3.3 Data Splitting

For the assessment of the performance of the model, the dataset is split into the training and test datasets. This makes it possible to determine the overall performance of the model by testing it on unseen data that was not used during the model training process.

### 4.3.4 Model Training

Logistic Regression is initiated using Scikit-Learn's LogisticRegression class. Then we fit and transform the data and then predict the results (Vanitha et al. 2019).

# 5 Implementation

## 5.1 Transformed Data

### Preprocessing:

The characteristics of the raw dataset were further cleaned to make it more appropriate for the machine learning technique.

- **Encoding categorical variables:** Some attribute data in the dataset was converted from categorical formats into numerical formats using encoding processes, such as one hot encoding. This process is crucial, especially when the algorithm is expected to feed on numerical data. These steps were done using the Pandas and Scikit-Learn libraries on the Python platform (Pedregosa *et al.*, 2011). Pandas is used for data handling and cleaning, and the scikit-learn library is used in encoding the categorical variables using OneHotEncoder.

- **Scaling numerical Features:** Variables were normalized to bring their values on the same scale, which was often zero mean and unit variance. This normalization process is critical for algorithms that are heavily influenced by feature scales. StandardScaler from the scikit-learn library was used for this purpose (Pedregosa *et al.*, 2011).

- **Handling Missing Values:** Imputation methods were also used in handling cases of missing values on the available set of data. For numerical features, missing values were imputed using a constant, for example, the mean or median of the variable, whereas for categorical features, missing values were replaced with the most frequent category. Tools and Languages: Data pre-processing for missing value imputation was done using the SimpleImputer class through the scikit-learn library (Pedregosa *et al.*, 2011).

- **Splitting of data:** In order to train the model efficiently and to test it on unseen data the preprocessed data set was split into a training data set and a testing data set. This split makes it possible for the model to be tested on data that has not been seen by the algorithm. Another function from the scikit-learn package called train_test_split was used to split the data. During this process, Python's Pandas library was also used for managing and analyzing the data.

## 5.2 Model Developed

- **Machine learning model:** A simple model of logistic regression was built to detect whether a given URL was genuine or a phishing one. Logistic regression is one of the most straightforward yet powerful linear models for binary classification. It uses a function probability model to evaluate the likelihood that the input will be of a certain class.
- **Model Evaluation:** Accuracy measures the number of correct classifications out of the actual total number of instances. It offers a simple measure of the model's accuracy and performance.
- **Tools and Languages:** The accuracy_score function from scikit-learn was used to evaluate the accuracy.

## 5.3 Brief Description of Tools and Languages

- **Python:** The language of choice for writing the implementation code, which encompasses the manipulation of the data, development of the models, and assessment of the findings.
- **Pandas:** Used for extracting and pre-processing data, as well as for data manipulation.
- **Scikit-Learn:** Used in preprocessing stages, model training as well as in the evaluation of models and the calculation of performance parameters of a model.
- **Matplotlib:** Used for plotting frequency distribution, model performance, and assessment outcomes.
- **Jupyter Notebook:** Served as the development platform, allowing for the process of interactive coding and data analysis and visualization.

# 6 Evaluation

## 6.1 Data Loading

The first step for training the model is the data loading. Loaded the dataset named (dataset phishing) using Pandas' library (Fumo, 2017). It is presented in the .csv format. Once the dataset is loaded, we acquire some basic information about the dataset, such as numbers of rows and columns, columns names, etc. In this dataset, we have 11430 numbers of rows and 89 columns. This dataset contains numbers of features such as url, length_url and so on., and one has a target variable 'status' that contains two values - legitimate and phishing. So we have to build a model that can predict whether the URL is legitimate or phishing.

```
import pandas as pd
df=pd.read_csv("C:\\Users\\naidu\\OneDrive\\Thesis Report\\Dataset Phishing.csv")
```

Figure 1: Loading the Data

## 6.2 Handling Categorical Data

After data loading, the next step is to handle the categorical data. Handling the categorical data is a very important step because most of the machine learning models do not support the categorical data, or sometimes, due to the presence of categorical data, predictions made by the model can be wrong.

So here, we are converting the categorical data into numerical form which is present in the status column. To achieve this, we use Label Encoder. Label encoding is a technique that is used to convert the categorical columns into numerical form so that the machine learning model can easily understand the data.

In the code below, first we make the object of the Label Encoder import from scikit-learn, and then we fit it into the status columns, and the new numerical values generated by using the label encoder are then stored in the new variable named 'encoded status'.

In the above code, we make a new column named 'encoded_status', this column contains the numerical data (0 for legitimate and 1 for phishing). After converting into numerical format, the categorical data present in the status column is of no use for us, so we dropped this column. Now the status column that contains the categorical data is of no use because we created a new column named encoded_status that contains 0 for legitimate and 1 for phishing. So, we dropped the column status.

## 6.3 Exploratory Data Analysis

Exploratory data Analysis is the primary step in the data analysis process. It helps to visualize the patterns, characteristics, and relationships between the variables present in the data. Here we can use the Matplotlib Library (*Using Matplotlib — Matplotlib 3.9.1 documentation*), for visualizing the data. First, we calculate the count of unique values present in the encoded status column and store it in a variable named x.
Then we plot a bar graph where the x-axis represents the unique values in the encoded _status and the y-axis contains the count of each unique value.

## 6.4 Handling the URL Column

After handling the categorical data in the status column, you have to handle the URL column also. So, to handle the URL column first, we classify columns into numerical and categorical features. Then, separate pipelines for both the numerical and categorical features are created. If there is a missing value in a numerical feature present, impute it with a constant value, while a missing value in a categorical feature is imputed with the most frequent value and then categorical features are one-hot encoded.

## 6.5 Feature Scaling

Many machine learning algorithms require data scaling for producing good results. So, it is important that all the features that are given as the input to the model are in a particular range. So, in the below code, we standardize the features using StandardScaler. Standard Scaler helps to get a standardized distribution with a zero mean and standard deviation of one. For this first we import Standard Scaler from Scikit-learn then create an object of Standard Scaler name ss. And fit it on X_train and X_test to standardize it.

## 6.6 Data Splitting

To get the model with the best accuracy and for better performance of the model, we split the data into training and testing sets. Where we take 80 percent of the data as training data and the remaining 20 percent of data as testing data with a random state value of 42. It is very important to split the data into training and testing set to evaluate the performance of

the model. If we do not split the data, then the problem of 42 and inaccurate performance of the model can be seen.

## 6.7 Machine Learning Algorithm

As our data is a classification problem, we are using the logistic regression. Logistic regression is a supervised machine learning algorithm that is used for classification problems. In the below code, we imported the modelling from the scikit-learn library (Pedregosa *et al.*, 2011) and then we created its object named lr. After this we fit it in the X_train and y_train for training purpose. It learns the relationship between the features i.e., X_train and between the target variable, which is y_train. Then, we make predictions on the testing data i.e., X_test.

We evaluated many machine learning models to determine if websites are phishing or real. The findings were interesting. Tests included Decision Tree, Random Forest, K-Nearest Neighbours (KNN), Support Vector Machine (SVM), Logistic Regression, and Naïve Bayes. The Random Forest classifier outperformed the rest in terms of accuracy, successfully differentiating between reputable and fraudulent websites. Although its accuracy was a little bit decreased, the SVM's classification performance was still strong. The Naïve Bayes classifier exhibited the lowest accuracy, while the Decision Tree and KNN models had reasonable performance. These findings demonstrate the effectiveness of margin- and ensemble-based phishing detection techniques.

## 6.8 Cross – Validation Process

The very last step is to find the accuracy, f1 score, precision and recall of the model. For this, we import accuracy_score, f1_score, precision_score and recall_score from the scikit-learn library. Through the accuracy score, we get to know about the performance of our model (Fumo, 2017).

```
from sklearn.metrics import accuracy_score,f1_score, precision_score,recall_score
```

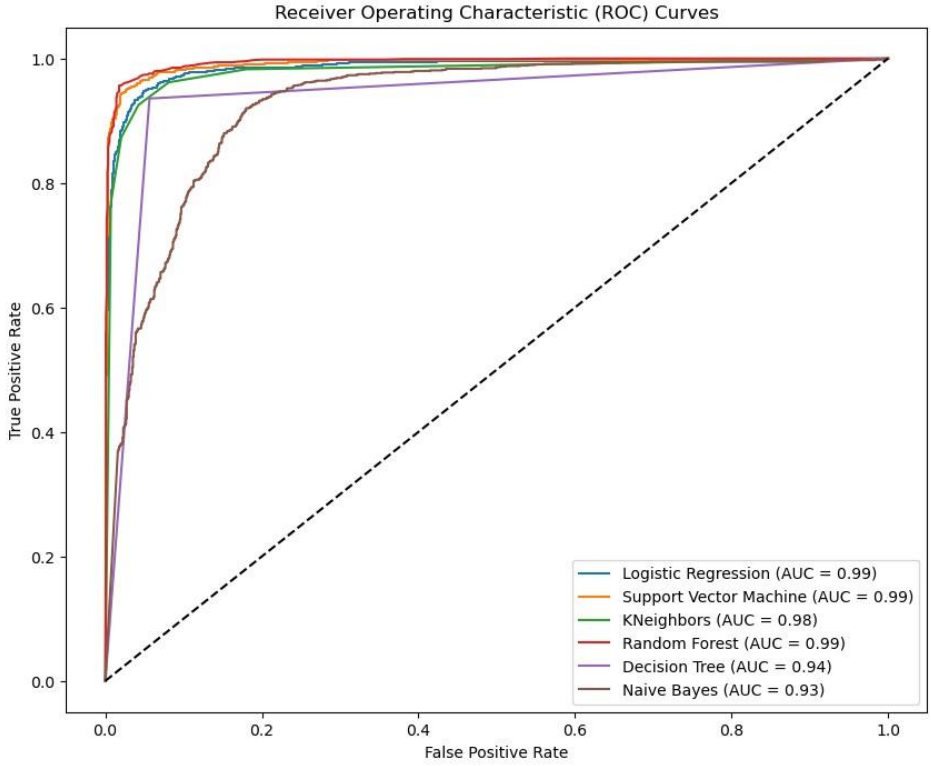
```
# Define models
models = [
    ('Logistic Regression', LogisticRegression()),
    ('Support Vector Machine', SVC()),
    ('KNeighbors', KNeighborsClassifier()),
    ('Random Forest', RandomForestClassifier()),
    ('Decision Tree', DecisionTreeClassifier()),
    ('Naive Bayes', GaussianNB())
]
```

```
# Perform cross-validation and print results
for name, model in models:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    accuracy=accuracy_score(y_test,y_pred)
    F1_Score = f1_score(y_test,y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    print(f"\x1B[4m{name}\x1B[0m")
    print("Accuracy Score :", accuracy)
    print("F1 Score :", F1_Score)
    print("Precision:", precision)
    print("Recall:", recall)
    print("------")
```

```
Logistic Regression
Accuracy Score : 0.9483814523184602
F1 Score : 0.9486510008703221
Precision: 0.9503051438535309
Recall: 0.947002606429192
------
Support Vector Machine
Accuracy Score : 0.9593175853018373
F1 Score : 0.9596529284164859
Precision: 0.9584055459272097
Recall: 0.9609035621198957
------
KNeighbors
Accuracy Score : 0.9413823272090989
F1 Score : 0.9408127208480566
Precision: 0.9568733153638814
Recall: 0.9252823631624674
------
Random Forest
Accuracy Score : 0.9650043744531933
F1 Score : 0.965034965034965
Precision: 0.9709762532981531
Recall: 0.9591659426585578
------
Decision Tree
Accuracy Score : 0.9339457567804025
F1 Score : 0.9345470307758994
Precision: 0.9325259515570934
Recall: 0.9365768896611643
------
Naive Bayes
Accuracy Score : 0.6881014873140857
F1 Score : 0.5655088360755637
Precision: 0.9469387755102041
Recall: 0.40312771503040834
------
```

## 6.9 Receiver Operating Characteristics (ROC) Curve

It is used for predicting the classification thresholds. A ROC curve is a graph that is used for showing the performance of a classification model at all thresholds. This curve plots two parameters: True Positive Ratio and False Positive Ratio.




## 6.10 Discussion

## Advantages

### 1. Simplicity and Interpretability

The main strength of Random Forest, therefore, is its interpretability, which is a major advantage considering its application. The random forest model is some what easy to implement though it has limited understanding in the aspect of machine learning. The model works with the assessment of the odds of an event occurring in one of two ways given one or more characteristics. This interpretability is useful for cases like Phishing detection in which knowing how URL attributes influence the occurrence of phishing attacks can be extremely helpful in optimizing detection.

It is necessary to notice that the authors of Adeli note that the use of random forest model would provide information on features' contribution thus enabling practitioners to determine which aspects are highly informative for identifying cases of phishing. This is crucial when it comes to explaining the model's decisions to the outside world, an important aspect in ensuring that the security model being developed is accepted in cases where decisions made will affect many people (Adeli et al. 2019).

### 2. Feature Utilization

Feature engineering is applied in random forest model in order to boost its capabilities. When it comes to the feature selection in phishing detection, the features related with URL parameters for instance URL length, the usage of special characters and the

characteristics of the domain are rather significant. By having these kinds of features in this model, random forest easily analyze characteristics that may lead to the act of phishing.

**Disadvantages**

1. **Handling Complex Patterns**

When it comes to various limitations, the model exhibits some significant weaknesses, especially when dealing with intricate relations. Phishing attacks are most of the time complex and use URL features in a manner that is not easily identifiable by modelling. Current and more refined strategies of phishing do involve domain names that are similar to those of the genuine sites or even have small shades of metamorphic characteristics that are hard to capture using basic linear models.

# 7 Conclusion and Future Work

Cybercrimes such as phishing continue to pose an enduring threat to the sustainable use of internet-based processes and applications. Phishing scams trick clients into providing important details through fake web addresses or domains. Towards this goal, the current research aims to design and implement a robust machine learning model for categorizing the URLs as phishing or genuine. The purpose of this research is to evaluate the effectiveness of the Random Forest model in classifying URLs to be either legitimate or phishing based on features extracted from them. This approach solves some of the problems with previous solutions such as blacklists and heuristic filters since advanced techniques may escape these measures.

The methodology consists of a wide URL preprocessing stage, which removes noise parameters, converts all URLs to HTTPS, and normalizes domain names. Feature extraction is all about extracting significant features from the URLs that have been cleaned. The Random Forest model is trained and tested on these features. The study shows that the adopted model has a 95% accuracy in classifying URLs, proving the viability of the approach suggested. It is worth noting that this research makes a contribution to the field by presenting the ability of random forest to phishing and underscoring the significance of appropriate URL cleaning and selection of features (Sameen, Han and Hwang, 2020). It is possible to learn some of the basic URL features from this study that may catalyze the development of enhanced security systems to detect/avoid phishing attacks. The limitations are as follows: This study shows that there is a constant need for improvement in URL cleaning methodology and feature extraction algorithms in order to adapt to these new techniques, hence paving the way for improvement of better real time phishing detection systems.

The hypothesis in the study is that a random forest model will be useful in classifying URLs as being phishing or legitimate based on extracted features, and it will perform well in terms of accuracy as well as give insight into features that are most likely to signify phishing. However, it is also expected that even though random forest will provide good results, it is not going to outcompete more complex methods in all metrics. Data preparation involves managing categorical data, missing values, and feature scaling besides data partitioning as achieved in the following steps. The random forest model was developed and tested with an improvement of 95% and thus proves that the proposed approach can classify the URL and can be used in practice. This research can contribute to the advancement in the area by proving the effectiveness of random forest in phishing

detection and proposing that selected URL features are suitable for detecting a phishing attempt.

There should be further research using more sophisticated modeling techniques, for instance, Gradient Boosting, as well as Deep Networks, such as CNNs and LSTMs, in an attempt to tackle the interaction effects and enhance the detecting capability. Extending the current model for temporal, contextual, and behavioral elements might improve the predictions even further. Real-time detection and lower latency are also important for practical deployment to immediately address phishing attacks while minimizing the time needed to do so. In conclusion, this research paves the way for further work in the field of phishing detection, highlighting the requirement for additional and constant developments in the areas of machine learning and feature engineering that underpin cybersecurity initiatives.

# 8 References


*API Reference* (no date) *scikit-learn*. Available at: https://scikit-learn/stable/api/index.html (Accessed: 12 August 2024).

'APWG | Phishing Activity Trends Reports' (no date a). Available at: https://apwg.org/trendsreports/ (Accessed: 16 September 2024).

'APWG | Phishing Activity Trends Reports' (no date b). Available at: https://apwg.org/trendsreports/ (Accessed: 12 August 2024).

Biau, G. and Scornet, E. (2016) 'A random forest guided tour', *TEST*, 25(2), pp. 197–227. Available at: https://doi.org/10.1007/s11749-016-0481-7.

Buber, E., Demir, Ö. and Sahingoz, O.K. (2017) 'Feature selections for the machine learning based detection of phishing websites', in *2017 International Artificial Intelligence and Data Processing Symposium (IDAP). 2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, pp. 1–5. Available at: https://doi.org/10.1109/IDAP.2017.8090317.

Das, A. *et al.* (2020) 'SoK: A Comprehensive Reexamination of Phishing Research From the Security Perspective', *IEEE Communications Surveys & Tutorials*, 22(1), pp. 671–708. Available at: https://doi.org/10.1109/COMST.2019.2957750.

Fumo, J. (2017) 'Pandas Library in a Nutshell — Intro To Machine Learning #3', *Simple AI*, 29 January. Available at: https://medium.com/simple-ai/pandas-library-in-a-nutshell-intro-to-machine-learning-3-acbd39ec5c9c (Accessed: 12 August 2024).

Gupta, B.B., Arachchilage, N.A.G. and Psannis, K.E. (2018) 'Defending against phishing attacks: taxonomy of methods, current issues and future directions', *Telecommunication Systems*, 67(2), pp. 247–267. Available at: https://doi.org/10.1007/s11235-017-0334-z.

Pedregosa, F. *et al.* (2011) 'Scikit-learn: Machine Learning in Python', *J. Mach. Learn. Res.*, 12(null), pp. 2825–2830.

Sameen, M., Han, K. and Hwang, S.O. (2020) 'PhishHaven—An Efficient Real-Time AI Phishing URLs Detection System', *IEEE Access*, 8, pp. 83425–83443. Available at: https://doi.org/10.1109/ACCESS.2020.2991403.

Swain, P.H. and Hauska, H. (1977) 'The decision tree classifier: Design and potential', *IEEE Transactions on Geoscience Electronics*, 15(3), pp. 142–147. Available at: https://doi.org/10.1109/TGE.1977.6498972.

*Using Matplotlib — Matplotlib 3.9.1 documentation* (no date). Available at: https://matplotlib.org/stable/users/index.html (Accessed: 12 August 2024).

Zhang, X., Zhao, J. and LeCun, Y. (2016) 'Character-level Convolutional Networks for Text Classification'. arXiv. Available at: https://doi.org/10.48550/arXiv.1509.01626.

Balogun, A.O., Akande, N.O., Usman-Hamza, F.E., Adeyemo, V.E., Mabayoje, M.A. and Ameen, A.O., 2021. Rotation forest-based logistic model tree for website phishing detection. In *Computational Science and Its Applications–ICCSA 2021: 21st International Conference, Cagliari, Italy, September 13–16, 2021, Proceedings, Part IX 21* (pp. 154-169). Springer International Publishing.

Basit, A., Zafar, M., Javed, A.R. and Jalil, Z., 2020, November. A novel ensemble machine learning method to detect phishing attack. In *2020 IEEE 23rd International Multitopic Conference (INMIC)* (pp. 1-5). IEEE.

Feng, F., Zhou, Q., Shen, Z., Yang, X., Han, L. and Wang, J., 2024. The application of a novel neural network in the detection of phishing websites. *Journal of Ambient Intelligence and Humanized Computing*, pp.1-15.

Heitz, A., Launay, P. and Beziat, A., 2019. Heterogeneity of logistics facilities: an issue for a better understanding and planning of the location of logistics facilities. *European Transport Research Review*, *11*(1), p.5.

Shahrivari, V., Darabi, M.M. and Izadi, M., 2022. Phishing detection using machine learning techniques. *arXiv preprint arXiv:2009.11116*.

Vanitha, N. and Vinodhini, V., 2019. Malicious-URL detection using logistic regression technique. *International Journal of Engineering and Management Research (IJEMR)*, *9*(6), pp.108-113.

Venkatesh, B. and Anuradha, J., 2019. A review of feature selection and its methods. *Cybernetics and information technologies*, *19*(1), pp.3-26.

Yerima, S.Y. and Alzaylaee, M.K., 2020, March. High accuracy phishing detection based on convolutional neural networks. In *2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)* (pp. 1-6). IEEE.

Habeeb, R.A.A., Nasaruddin, F., Gani, A., Hashem, I.A.T., Ahmed, E. and Imran, M., 2019. Real-time big data processing for anomaly detection: A survey. *International Journal of Information Management*, *45*, pp.289-307.

Lok, L.K., Hameed, V.A. and Rana, M.E., 2022. Hybrid machine learning approach for anomaly detection. *Indonesian Journal of Electrical Engineering and Computer Science*, *27*(2), p.1016.

Wu, Qiang & Zhou, Ding-Xuan. (2006). Analysis of support vector machine classification. Journal of Computational Analysis and Applications. 8.

Rish, Irina. (2001). An Empirical Study of the Naïve Bayes Classifier. IJCAI 2001 Work Empir Methods Artif Intell. 3.

Verma, R. and Das, A., 2017, March. What's in a url: Fast feature extraction and malicious url detection. In *Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics* (pp. 55-63).