

**A Project Report on**  
**Detecting Malicious Sites Using ML Algorithms**

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the  
academic requirements for the award of the degree.

**Bachelor of Technology**  
**In**  
**Computer Science and Engineering**

Submitted by

G. NAVEEN  
(20H51A0564)  
I.SREE ANVITA  
(20H51A0566)  
FARHEEN  
(20H51A0591)

Under the esteemed guidance of  
B.Gayathri  
(Assistant Professor)



**Department of Computer Science and Engineering**

**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

(UGC Autonomous)

\*Approved by AICTE \*Affiliated to JNTUH \*NAAC Accredited with A<sup>+</sup> Grade

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

**2020- 2024**

# **CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



### **CERTIFICATE**

This is to certify that the Major Project report entitled "**Detecting Malicious sites**" being submitted by G.NAVEEN(20H51A0564), I.SREE ANVITA(20H51A0566), FARHEEN(20H51A0591) in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree.

**B.Gayathri**  
Assistant Professor  
Dept. Of CSE

**Dr. S. Siva Skandha**  
Associate Professor and HOD  
Dept. of CSE

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project a grand success.

We are grateful to **B.Gayathri, Assistant Professor**, Department of CSE for her valuable technical suggestions and guidance during the execution of this project work.

We would like to thank, **Dr. Siva Skandha Sanagala**, Head of the Department of CSE, CMR College of Engineering and Technology, who is the major driving forces to complete our project work successfully.

We are very grateful to **Dr. Ghanta Devadasu**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana**, Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Dept Name for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary& Correspondent, CMR Group of Institutions, and **Shri Ch Abhinav Reddy**, CEO, CMR Group of Institutions for their continuous care and support.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly or indirectly in completion of this project work.

G.NAVEEN	20H51A0564
I.SREE ANVITA	20H51A0566
FARHEEN	20H51A0591

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	LIST OF FIGURES	iii
	ABSTRACT	iv
<b>1</b>	<b>INTRODUCTION</b>	1
	1.1 Problem Statement	2
	1.2 Research Objective	3
	1.3 Project Scope and Limitations	4
<b>2</b>	<b>BACKGROUND WORK</b>	6
	2.1. Phishing website detection using Email Filtration	7
	2.1.1. Introduction	7
	2.1.2. Merits, Demerits and Challenges	7
	2.1.3. Implementation website detection using Email Filtration	9
	2.2. Malicious Sites Using URL	10
	2.2.1. Introduction	10
	2.2.2. Merits, Demerits and Challenges	10
	2.2.3. Implementation of Malicious Sites Using URL	12
	2.3. Malicious app detection using Deep learning	13
	2.3.1. Introduction	13
	2.3.2. Merits, Demerits and Challenges	13
	2.3.3. Implementation of website using Deep learning	15
<b>3</b>	<b>PROPOSED SYSTEM</b>	16
	3.1. Objective of Proposed Model	17
	3.2. Algorithms Used for Proposed Model	18
	3.2.1. Support Vector Machine	18
	3.2.2. Gaussian Naïve Bayes	19
	3.2.3. Decision Tree	20
	3.2.4. AdaBoost	21
	3.2.5. Random Forest	22
	3.3. Designing	24

	3.3.1.UML Diagram	25
	3.4. Stepwise Implementation and Code	27
<b>4</b>	<b>RESULTS AND DISCUSSION</b>	38
4	Results	39
<b>5</b>	<b>CONCLUSION</b>	47
	5.1 Conclusion and Future Enhancement	48
	<b>REFERENCES</b>	51
	<b>GITHUB LINK</b>	52
	<b>PUBLISHED PAPER</b>	53
	<b>CERTIFICATES</b>	60

### List of Figures

#### FIGURE

NO.	TITLE	PAGE NO.
3.3.1	DESIGNING	24
3.3.2	DESIGNING	24
3.3.1.1	UML DIAGRAM	25
4.1	OVERVIEW OF WEBSITE	40
4.2	LEGITIMATE	41
4.3	PHSIHING	42
4.4	PROJECT DETAILS	43
4.5	FEATURE TABLE	44
4.6	PIE CHART	45
4.7	ACCURACY	46
4.8	BAR GRAPH	46

## **ABSTRACT**

There are number of users who purchase products online and make payment through various websites. There are multiple websites who ask user to provide sensitive data such as username, password or credit card details etc. often for malicious reasons. This type of websites is known as phishing website. In order to detect and predict phishing website, we proposed an intelligent, flexible and effective system that is based on using classification Data mining algorithm. We implemented classification algorithm and techniques to extract the phishing data sets criteria to classify their legitimacy. The phishing website can be detected based on some important characteristics like URL and Domain Identity, and security and encryption criteria in the final phishing detection rate. Once user makes transaction through online when he makes payment through the website our system will use data mining algorithm to detect whether the website is phishing website or not. This application can be used by many E-commerce enterprises in order to make the whole transaction process secure. Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms. With the help of this system user can also purchase products online without any hesitation.

# **CHAPTER 1**

## **INTRODUCTION**



# CHAPTER 1

## INTRODUCTION

### 1.1. Problem Statement

In the digital age, online shopping has become increasingly popular, with millions of users purchasing products and services through various websites every day. However, alongside legitimate online platforms, there exists a significant threat in the form of phishing websites. These malicious websites aim to deceive users into providing sensitive information such as usernames, passwords, or credit card details under false pretenses. Phishing is a form of internet scam where attackers masquerade as trusted entities to trick individuals into divulging personal information or downloading malicious content. Typically, phishing attacks involve sending out deceptive emails or messages that appear to originate from legitimate sources, such as banks, social media platforms, or e-commerce websites. These emails often contain links or attachments that, when clicked, redirect users to fraudulent websites designed to steal their confidential data or infect their devices with malware.

Initially, phishing attempts were carried out through indiscriminate spam campaigns, targeting broad groups of people with generic messages. However, as cybersecurity measures have improved and users have become more cautious, attackers have refined their tactics. Modern phishing attacks are more sophisticated, often employing social engineering techniques to personalize messages and increase their effectiveness. This evolution has made phishing one of the most prevalent and damaging forms of cybercrime today.

Despite advancements in cybersecurity, phishing remains a significant challenge due to several factors. Traditional methods of identifying phishing websites, such as blacklisting known malicious domains, are no longer sufficient to combat the evolving tactics of attackers. Many phishing websites are designed to mimic legitimate sites, making them difficult to distinguish. Additionally, while cybersecurity measures have improved, many users and organizations still lack robust protective measures against phishing attacks. This includes insufficient email filtering systems, outdated software with unpatched vulnerabilities, and . lack of user education and awareness.

## 1.2. Research Objective

The objective of this project is to develop and evaluate machine learning models and deep neural networks capable of predicting phishing websites. To achieve this goal, a dataset comprising both phishing and benign URLs is created. This dataset serves as the foundation for training and testing the predictive models.

The dataset encompasses a diverse range of URLs, including those associated with legitimate websites as well as URLs indicative of phishing attempts. From this dataset, various URL and website content-based features are extracted. These features capture relevant information about the URLs and the content of the associated websites, such as domain characteristics, HTML attributes, and textual content.

Machine learning models and deep neural networks are trained using this dataset and the extracted features. These models are trained to learn patterns and relationships within the data that distinguish between phishing and benign URLs. During the training process, the models adjust their parameters to minimize prediction errors and improve their ability to accurately classify URLs.

Once trained, the performance of each model is evaluated using appropriate evaluation metrics. These metrics provide insights into the models' effectiveness in predicting phishing websites, including measures of accuracy, precision, recall, and F1 score. By comparing the performance of different models, researchers can identify the most effective approaches for phishing website detection.

The project aims to assess the strengths and weaknesses of various machine learning and deep learning techniques in the context of phishing website prediction. By evaluating and comparing the performance of different models, researchers can gain valuable insights into the most promising strategies for mitigating the risks posed by phishing attacks. Ultimately, the goal is to develop robust and accurate predictive models that can help users and organizations identify and avoid potential threats posed by phishing websites.

### 1.3. Project Scope and Limitations:

#### **Project Scope:**

The primary objective of this project is to create a robust and effective system capable of detecting and preventing phishing websites. Phishing websites pose a significant threat to users' sensitive data and personal information, as they are designed to deceive individuals into disclosing confidential details such as usernames, passwords, and financial information. By designing and implementing a comprehensive system to combat phishing, the project aims to enhance cybersecurity measures and safeguard users from falling victim to fraudulent schemes.

#### **To achieve this objective, the project involves several key components:**

**Detection Mechanisms:** The system will incorporate advanced algorithms and techniques to identify potential phishing websites. These mechanisms will analyze various attributes of URLs and website content to detect suspicious patterns and indicators of phishing activity. This may include analyzing domain characteristics, webpage content, and user interaction behavior to differentiate between legitimate and malicious websites.

**Real-time Monitoring:** Implementing real-time monitoring capabilities allows the system to continuously scan web traffic for potential phishing indicators. By monitoring incoming traffic in real-time, the system can promptly detect and respond to emerging threats, minimizing the risk of users encountering phishing websites.

**Alerting and Notification System:** Upon detecting a phishing attempt, the system will trigger alerts and notifications to inform users and system administrators. These alerts may take various forms, such as pop-up messages, email notifications, or SMS alerts, depending on the severity and urgency of the threat. By promptly notifying users and administrators, the system enables swift action to mitigate the risk posed by phishing attacks.

**Data Protection Measures:** Safeguarding users' sensitive data and personal information is a crucial aspect of phishing prevention. The system will implement robust data protection measures to ensure that users' confidential information remains secure. This may include encryption techniques, access controls, and secure storage mechanisms to prevent unauthorized access or disclosure of sensitive data.

**Reporting and Analysis Tools:** Providing reporting and analysis tools allows system administrators to track and understand phishing trends and patterns effectively. By analyzing

data collected from phishing attempts, administrators can gain insights into the tactics and techniques used by attackers, enabling them to refine detection mechanisms and improve the system's effectiveness over time.

## **2.Inclusions:**

- a. Phishing Website Detection: Develop algorithms and mechanisms to identify potential phishing websites.
- b. Real-time Monitoring: Implement real-time monitoring of web traffic for phishing indicators.
- c. Alerting and Notification: Notify users and system administrators when a phishing attempt is detected.
- d. Data Protection: Develop methods to safeguard users' sensitive data and personal information.
- e. Reporting and Analysis: Provide reporting and analysis tools to track and understand phishing trends and patterns

## **3.Exclusions:**

- a. Physical security measures are not within the scope of this project.
- b. This project does not include the development of end-user web browsers or email clients.
- c. Legal and law enforcement actions against phishing perpetrators are not part of the scope.

## **Limitations:**

- 1. User Education:** While the project may include user education and awareness materials, it cannot guarantee that all users will follow recommended security practices. Human error remains a significant factor in phishing incidents.
- 2. User feedback:** Relying solely on automated detection may miss emerging threats. User feedback may be essential to enhance the system's accuracy.

# **CHAPTER 2**

## **BACKGROUND WORK**

## CHAPTER 2

### BACKGROUND WORK

#### 2.1. . Phishing website detection using Email Filtration [1]:

##### 2.1.1. Introduction

Phishing website detection via Email Filters is integral to cybersecurity, shielding users from online fraud. This method employs advanced algorithms to scrutinize emails for phishing indicators like sender details, content, and embedded links. Initially, sender authenticity and domain credibility are verified. Anomalies or suspicious addresses trigger deeper probes. Email content undergoes thorough examination for phishing hallmarks such as urgent language, coercion, or requests for sensitive data. Additionally, embedded links' legitimacy is assessed by comparing sender claims to actual website domains. Machine learning aids in identifying suspicious URL patterns and cross-referencing against phishing domain databases. By integrating rule-based filters and machine learning, the system effectively distinguishes genuine emails from phishing attempts. Rule-based filters flag emails with dubious attributes, while machine learning detects subtle phishing behavior patterns using historical data. Overall, this system reduces user vulnerability to fraud, protects sensitive data, and upholds organizational network integrity.

##### 2.1.2. Merits, Demerits and Challenges

###### Merits:

- **Enhanced Security:** By effectively filtering out phishing emails, the system enhances overall security posture, reducing the risk of unauthorized access to sensitive information or systems. This helps in maintaining data integrity and confidentiality, ensuring the protection of valuable assets
- **Cost-Efficiency:** Automated email filtration systems are cost-efficient and scalable compared to manual inspection or remediation of phishing attacks. They require minimal human intervention, reducing operational costs associated with manual labor and enabling organizations to allocate resources more efficiently.

- **Real-Time Detection:** Advanced algorithms enable real-time detection of phishing attempts, allowing for prompt action to mitigate potential threats before they cause harm. Real-time detection minimizes the window of vulnerability, enhancing the organization's ability to respond effectively to emerging threats.

**Demerits:**

- **False Positives:** Overly aggressive filtering algorithms may inadvertently flag legitimate emails as phishing attempts, leading to potential loss of important communication and user frustration. False positives can undermine user confidence in the filtration system and result in the inadvertent blocking of critical communications.
- **Sophisticated Attacks:** Despite advanced algorithms, highly sophisticated phishing attacks may still evade detection, particularly those employing social engineering techniques or zero-day vulnerabilities. These attacks can be difficult to detect using traditional detection methods, posing a significant challenge to the effectiveness of the filtration system.
- **Overhead:** Email filtration systems impose additional overhead on network and system resources, particularly during peak usage periods. This can result in decreased performance and increased latency, impacting user experience and productivity.

**Challenges:**

- **Sophisticated Phishing Techniques:** As phishing techniques evolve, attackers employ more sophisticated methods to evade detection, making it challenging for traditional algorithms to keep up.
- **False Positives/Negatives:** Over-reliance on rule-based filters can lead to false positives, flagging legitimate emails as phishing attempts, or false negatives, missing actual phishing emails.

### 2.1.3. Implementation of Phishing website detection using Email Filtration:

- **Assessment of Requirements:** Understand organizational requirements, including the volume of emails processed, desired level of security, and budget constraints.
- **Selection of Technology:** Choose appropriate email filtration technology based on organizational needs, considering factors such as scalability, accuracy, and ease of integration.
- **Configuration and Tuning:** Configure email filters and machine learning models based on organizational policies and threat intelligence data. Fine-tune algorithms to minimize false positives and enhance detection accuracy.
- **Deployment:** Deploy the email filtration system within the organization's email infrastructure, ensuring compatibility with existing systems and workflows.
- **Testing and Validation:** Conduct thorough testing and validation of the filtration system to ensure effectiveness and reliability in detecting phishing attempts while minimizing disruptions to legitimate email communication.
- **Ongoing Monitoring and Updates:** Continuously monitor the performance of the email filtration system, applying updates and adjustments as needed to address evolving threats and maintain optimal functionality.
- **User Training:** Provide comprehensive training to users on recognizing and reporting phishing attempts, reinforcing the role of email filtration as part of the organization's overall cybersecurity strategy.



## 2.2. Malicious Sites Using URL [2]:

### 2.2.1. Introduction

Phishing app detection via URL verification involves a detailed analysis of URLs within applications to identify potential threats. Utilizing advanced algorithms, the system examines URL components like domain names, subdomains, paths, and parameters for anomalies or signs of suspicious activity. A crucial aspect is spotting deceptive domain names that mimic legitimate brands, organizations, or entities. By scrutinizing domain names for inconsistencies, the system can flag suspicious URLs for further review. Furthermore, the verification process checks for suspicious redirects, which attackers use to conceal a URL's true destination or bypass security measures. Detecting and analyzing these redirects helps uncover deceptive practices aiming to mislead users or direct them to phishing sites. Overall, this meticulous URL verification aims to prevent phishing attempts within applications, safeguarding users from fraudulent activities. Enhancing cybersecurity posture, organizations can effectively mitigate risks associated with phishing attacks by neutralizing potential threats at the URL level.

### 2.2.2. Merits, Demerits and Challenges

#### Merits:

- **Enhanced Security:** By scrutinizing URLs embedded within applications, this approach enhances overall security posture by identifying and mitigating potential phishing attempts before they can inflict harm on users.
- **Early Detection:** URL verification enables early detection of phishing attempts, allowing organizations to take prompt action to mitigate risks and protect users' sensitive information.
- **Cost-Efficiency:** Automated email filtration systems are cost-efficient and scalable compared to manual inspection or remediation of phishing attacks. They require minimal human intervention, reducing operational costs associated with manual labor and enabling organizations to allocate resources more efficiently.

**Demerits:**

- **False Positives:** Overly aggressive URL verification algorithms may flag legitimate URLs as malicious, resulting in false positives. This can disrupt legitimate user activities and diminish user trust in the detection system.
- **Complexity:** Implementing URL verification mechanisms can be complex, requiring specialized expertise and resources. The complexity of the algorithms and the need for continuous updates to detect evolving phishing tactics can pose challenges to implementation and maintenance.

**Challenges:**

- **Evasion Techniques:** Phishing attackers constantly evolve their tactics to evade detection, making it challenging to accurately identify and mitigate phishing URLs embedded within applications. Attackers may employ obfuscation techniques, such as URL shorteners or encryption, to disguise malicious URLs and bypass verification mechanisms.
- **Scale and Volume:** The sheer volume of applications and URLs to be analyzed presents a significant challenge. With millions of applications available across various platforms, manually verifying URLs for phishing threats is impractical. Automated systems must be capable of processing large volumes of URLs efficiently while maintaining accuracy and reliability.
- **Zero-Day Attacks:** Zero-day phishing attacks, which exploit previously unknown vulnerabilities, pose a significant threat. Traditional detection methods may struggle to identify zero-day attacks, as they lack historical data or signatures to detect emerging threats. URL verification systems must incorporate advanced heuristics and anomaly detection techniques to identify potential zero-day attacks effectively.

### 2.2.3. Implementation of Malicious Sites Using URL

- **Requirement Analysis:** Understand the organization's needs, including the types of applications and platforms used, the volume of URLs to be analyzed, and the desired level of security.
- **Technology Selection:** Choose suitable URL verification technologies and tools based on organizational requirements and constraints. Consider factors such as scalability, accuracy, and compatibility with existing systems.
- **Algorithm Development:** Develop sophisticated algorithms capable of analyzing URLs embedded within applications to identify indicators of phishing attempts. These algorithms should be able to assess the structure, domain, and potential risks associated with each URL.
- **Integration with Application Lifecycle:** Integrate URL verification mechanisms into the application development lifecycle. This may involve incorporating verification checks during the application build process, continuous integration/continuous deployment (CI/CD) pipelines, or app store submission processes.
- **Automated Testing:** Implement automated testing processes to verify the effectiveness of URL verification mechanisms. This ensures that the system accurately identifies phishing URLs and provides timely alerts or notifications to users.
- **Continuous Monitoring:** Continuously monitor applications for newly identified phishing URLs and update verification algorithms to detect evolving threats effectively. Implement mechanisms to collect and analyze real-time data on URL access and user interactions for proactive threat detection.
- **User Education:** Provide users with education and awareness materials to help them recognize and avoid phishing attempts embedded within applications. Educate users about the importance of verifying URLs before clicking on them and provide guidance on identifying suspicious URLs.
- **Feedback Mechanisms:** Implement feedback mechanisms to allow users to report suspicious URLs encountered within applications.

## **2.3. Malicious app detection using Deep learning [3]:**

### **2.3.1. Introduction**

Malicious app detection with Deep Learning utilizes neural network architectures like CNNs or RNNs, trained on diverse datasets of both legitimate and malicious apps. These datasets cover various app types, functionalities, and sources to ensure model robustness. The approach focuses on automatically extracting features from app metadata, permissions, and user interface elements. Unlike traditional methods requiring manual feature engineering, deep learning algorithms learn complex patterns directly from raw data. This adaptability allows them to dynamically respond to new threats without explicit retraining, enhancing effectiveness against evolving cyber threats. After training, the model operates in real-time, analyzing new apps based on extracted features to identify potential malicious behavior or phishing attempts. In summary, deep learning-based malicious app detection offers a proactive and dynamic cybersecurity solution, leveraging advanced machine learning to protect mobile users from the evolving landscape of mobile security threats.

### **2.3.2. Merits, Demerits and Challenges**

#### **Merits:**

- **Automated Feature Extraction:** Deep learning models automatically extract relevant features from app metadata, permissions, and user interface elements, eliminating the need for manual feature engineering. This results in more comprehensive and accurate feature representation, enhancing the model's ability to detect phishing apps.
- **Adaptability:** Deep learning models are highly adaptable and can learn from new data, making them effective at identifying emerging threats and previously unseen phishing app patterns. This adaptability is crucial in combating the rapidly evolving landscape of mobile threats.

**Demerits:**

- **Data Requirements:** Deep learning models require large amounts of labeled data for training, which may be challenging to obtain, especially for rare or novel phishing app patterns. Acquiring diverse and representative datasets can be time-consuming and resource-intensive.
- **Computational Complexity:** Training deep learning models involves significant computational resources, including high-performance GPUs and large-scale computing infrastructure. This computational complexity can pose challenges for organizations with limited resources or infrastructure.
- **Interpretability:** Deep learning models are often considered black boxes, meaning that the internal workings of the model and the reasons behind its predictions are not easily interpretable. This lack of interpretability may hinder trust and understanding of the model's decisions, especially in critical applications such as cybersecurity.

**Challenges:**

- **Data Quality and Quantity:** Acquiring labeled datasets containing both legitimate and malicious apps of sufficient quality and quantity can be challenging. Obtaining diverse and representative data that encompasses a wide range of app types, versions, and sources is essential for effective model training. Additionally, ensuring the accuracy and reliability of the labels assigned to each app in the dataset is crucial for training a robust deep learning model.
- **Feature Extraction:** Extracting relevant features from app metadata, permissions, and user interface elements requires careful consideration and expertise. Identifying the most informative features while filtering out noise and irrelevant information is essential for maximizing the model's performance. The complexity of feature extraction may vary depending on the nature of the app and the specific characteristics indicative of malicious behavior.

### 2.3.3. Implementation

- **Data Collection:** Gather a diverse and representative dataset containing both legitimate and malicious apps. Ensure that the dataset covers a wide range of app types, categories, and sources to facilitate comprehensive training.
- **Data Preprocessing:** Preprocess the app data to extract relevant features, such as metadata, permissions, and user interface elements. Normalize and standardize the data to ensure consistency and compatibility for training the deep learning model.
- **Model Training:** Train a deep learning model, such as a convolutional neural network (CNN) or recurrent neural network (RNN), using the preprocessed app data. Use techniques such as transfer learning or data augmentation to improve model performance and generalization.
- **Evaluation:** Evaluate the trained model using validation data to assess its performance metrics, such as accuracy, precision, recall, and F1 score. Fine-tune the model parameters and architecture as needed to optimize performance.
- **Deployment:** Deploy the trained model into production environments to analyze new apps in real-time. Integrate the model into existing security systems or mobile app stores to automatically identify and flag potential phishing apps for further investigation or mitigation.
- **Monitoring and Maintenance:** Continuously monitor the performance of the deployed model in real-world scenarios. Update the model periodically with new data and retrain it to adapt to evolving threats and maintain optimal detection accuracy.

# **CHAPTER 3**

## **PROPOSED SYSTEM**

## CHAPTER 3

### PROPOSED SYSTEM

#### 3.1. Objective of proposed model:

**Objective:** The primary aim of the project was to develop an intelligent system capable of detecting and predicting phishing websites, ultimately enhancing the security of online transactions.

**Methodology:** Leveraging classification data mining algorithms, the system categorized websites as either legitimate or malicious based on a predefined set of criteria. These criteria primarily focused on URL and domain identity, as well as security and encryption factors.

**Outcome Evaluation:** One key outcome was the performance evaluation of the classification algorithm. The system's effectiveness was compared with traditional classification methods, demonstrating superior performance in detecting phishing websites.

**Advantages of Data Mining Approach:** The system exhibited a higher detection rate for phishing websites by considering multiple characteristics, such as URL and domain identity. This comprehensive assessment allows for better identification of sophisticated phishing attempts that mimic legitimate websites.

**Addressing False Positives:** Minimizing false positives, where legitimate websites are incorrectly classified as malicious, was a major challenge. The project focused on fine-tuning the data mining algorithm and criteria to reduce false positives significantly.

**Balancing Security and Usability:** By effectively mitigating false positives, the system achieves a balance between security and usability. Users can confidently make online purchases without undue hesitation, enhancing their overall experience.



### 3.2. Algorithms used for proposed model:

- Support Vector Machine
- Gaussian Naive Bayes
- Decision Tree
- Random Forest
- AdaBoost

#### 3.2.1 Support Vector Machine:

Support Vector Machine (SVM) is a machine learning algorithm commonly used in phishing detection applications to classify URLs as either malicious (phishing) or legitimate. Here's how SVM decides whether a given URL is malicious:

**Feature Extraction:** Before using SVM, features are extracted from the URL to represent various characteristics that may indicate whether it is phishing or legitimate. These features may include

**URL length:** Phishing URLs often have longer, convoluted URLs.

**Domain age:** Legitimate websites tend to have older domain registrations compared to newly created phishing domains.

**Presence of hyphens and special characters:** Phishing URLs may contain unusual characters or hyphens to mimic legitimate URLs.

**Use of IP address:** Phishing URLs may use IP addresses instead of domain names.

**Domain reputation:** Checking the reputation of the domain using third-party services.

**Presence of suspicious keywords:** Phishing URLs may contain words like "login," "password," "account," etc.

**HTTPS encryption:** Legitimate websites are more likely to have HTTPS encryption, while phishing sites may lack it.

**Training the SVM Model:** Once features are extracted, a dataset containing labeled examples of phishing and legitimate URLs is used to train the SVM model. The SVM learns to distinguish between the features of phishing and legitimate URLs by finding the optimal decision boundary that maximizes the margin between the two classes

**Classification:** When a new URL is encountered, its features are extracted, and the SVM model predicts whether it is phishing or legitimate. The decision is based on which side of the

decision boundary the features fall. If the features are closer to the boundary, the model may assign a lower confidence score, indicating uncertainty.

**Thresholding:** The output of the SVM model is typically a continuous score between 0 and 1, representing the likelihood that the URL is phishing. A threshold is applied to this score to determine the final classification. URLs with scores above the threshold are classified as phishing, while those below are classified as legitimate.

**Evaluation:** The performance of the SVM model in classifying URLs is evaluated using metrics such as accuracy, precision, recall, and F1-score on a validation dataset. The model may be fine-tuned by adjusting hyperparameters to optimize its performance.

### 3.2.2 Gaussian Naive Bayes:

**Feature Extraction:** Similar to SVM, before using GNB, features are extracted from the URL to represent various characteristics that may indicate whether it is phishing or legitimate.

These features may include:

URL length: Phishing URLs often have longer, convoluted URLs.

Domain age: Legitimate websites tend to have older domain registrations compared to newly created phishing domains.

Presence of hyphens and special characters: Phishing URLs may contain unusual characters or hyphens to mimic legitimate URLs.

Use of IP address: Phishing URLs may use IP addresses instead of domain names.

Domain reputation: Checking the reputation of the domain using third-party services.

Presence of suspicious keywords: Phishing URLs may contain words like "login," "password," "account," etc.

HTTPS encryption: Legitimate websites are more likely to have HTTPS encryption, while phishing sites may lack it.

Assumption of Feature Independence: GNB makes the assumption that the features are conditionally independent given the class label (phishing or legitimate). Despite this simplifying assumption, GNB often performs well in practice, especially when dealing with high-dimensional data. simplifying assumption, GNB often performs well in practice, especially when dealing with high-dimensional data.

**Training the GNB Model:** Once features are extracted, a dataset containing labeled examples of phishing and legitimate URLs is used to train the GNB model. The algorithm learns the conditional probability distributions of each feature given the class label using a Gaussian (normal) distribution.

**Classification:** When a new URL is encountered, its features are extracted, and the GNB model calculates the probability that it belongs to each class (phishing or legitimate) using Bayes' theorem. The class with the highest probability is assigned to the URL.

**Thresholding:** Similar to SVM, the output of the GNB model is typically a continuous score representing the likelihood that the URL is phishing. A threshold is applied to this score to determine the final classification. URLs with scores above the threshold are classified as phishing, while those below are classified as legitimate.

**Evaluation:** The performance of the GNB model in classifying URLs is evaluated using metrics such as accuracy, precision, recall, and F1-score on a validation dataset. The model may be fine-tuned by adjusting hyperparameters to optimize its performance.

### 3.2.3 Decision Tree:

Decision Tree is a machine learning algorithm that makes decisions by learning simple decision rules inferred from the features of the data. Here's how a decision tree decides whether a given URL is malicious:

**Feature Selection:** Before using a decision tree, features are selected or extracted from the URL. These features can include various characteristics such as:

- URL length: Phishing URLs may have unusual lengths compared to legitimate one.
- Use of IP address: Phishing URLs may use IP addresses instead of domain names.
- Domain reputation: Checking the reputation of the domain using third-party services.

Domain age: Legitimate websites tend to have older domain registrations compared to newly created phishing domains.

- Presence of hyphens: Phishing URLs may contain unusual characters or hyphens to mimic legitimate URLs.
- Presence of suspicious keywords: Phishing URLs may contain words like "login," "password," "account," etc.
- Building the Tree: The decision tree algorithm recursively partitions the feature space

based on the selected features to create a tree-like structure. At each node of the tree, the algorithm selects the feature that best splits the data into homogeneous subsets, i.e., subsets where most URLs are either malicious or legitimate.

**Decision Rules:** The decision tree uses simple decision rules to make predictions about the class (malicious or legitimate) of a given URL. For example, if a URL has a certain feature value (e.g., URL length > 50 characters), it follows a specific path down the tree until it reaches a leaf node.

**Classification:** When a new URL is encountered, At the leaf node, the URL is classified as either malicious or legitimate based on the majority class of the training examples of nodes.

**Evaluation:** The performance of the decision tree in classifying URLs is evaluated using metrics such as accuracy, precision, recall, and F1-score on a validation dataset. The model may be fine-tuned by adjusting hyperparameters to optimize its performance

### 3.2.4 AdaBoost

AdaBoost (Adaptive Boosting) is a machine learning ensemble method that combines multiple weak classifiers to create a strong classifier. Here's how AdaBoost decides whether a given URL is malicious.

**Feature Extraction:** Similar to other machine learning algorithms, before using AdaBoost, features are selected or extracted from the URL. These features represent various characteristics

of the URL, such as its length, domain age, presence of certain keywords.

**Training Weak Classifiers:** AdaBoost starts by training a series of weak classifiers on the dataset. The weak classifiers can be simple models like decision trees with limited depth or linear classifiers.

**Weighting Examples:** During training, AdaBoost assigns weights to each training example based on whether it is classified correctly or incorrectly by the weak classifiers. Initially, all examples have equal weights. However, AdaBoost increases the weights of misclassified examples and decreases the weights of correctly classified examples, focusing more on the difficult-to-classify instances.

**Combining Weak Classifiers:** AdaBoost combines the predictions of the weak classifiers by giving more weight to the predictions of the classifiers with lower error rates. The final

prediction is made by aggregating the weighted votes of all weak classifiers.

**Classification:** When a new URL is encountered, its features are extracted, and each weak classifier. The predictions are weighted according to the performance of the weak classifiers during training, and the final prediction is made by aggregating these weighted votes.

**Thresholding:** Similar to other classification algorithms, AdaBoost may use a threshold to determine the final classification of the URL. For example, if the aggregated weighted votes exceed a certain threshold, the URL may be classified as malicious; otherwise, it may be classified as legitimate.

**Evaluation:** The performance of the AdaBoost classifier in classifying URLs is evaluated using metrics such as accuracy, precision, recall, and F1-score on a validation dataset. The model may be fine-tuned by adjusting hyperparameters (e.g., number of weak classifiers, learning rate) to optimize its performance.

### 3.2.5 Random Forest

Random Forest is a powerful ensemble learning algorithm that builds multiple decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Let's justify why Random Forest was chosen among all the mentioned algorithms for phishing URL detection, emphasizing its potential for higher accuracy.

**Feature Extraction and Training:** Random Forest utilizes a multitude of decision trees, each trained on a random subset of features and samples from the dataset. This diversity allows it to capture complex relationships between the features and the target variable (malicious or legitimate URL) effectively. By considering various aspects of URLs (such as length, presence of hyphens, domain age, etc.), Random Forest can make robust predictions.

**Handling Non-linearity:** URLs can exhibit non-linear relationships between their features and the target class. Random Forest's ability to learn from non-linear relationships makes it well-suited for capturing the intricate patterns present in URLs. For example, it can discern subtle differences between legitimate and phishing URLs that may not be captured by simpler algorithms like Logistic Regression or Naive Bayes.

**Ensemble Learning:** Random Forest leverages the concept of ensemble learning, combining predictions from multiple decision trees to make a final prediction. This ensemble approach

tends to reduce overfitting and improve generalization performance compared to individual decision trees. By aggregating the predictions of multiple trees, Random Forest can achieve higher accuracy and robustness in classifying URLs.

**Reducing Variance:** Decision trees are prone to high variance, meaning they can be sensitive to small changes in the training data. predictions. This property is crucial for maintaining consistent performance across different datasets and scenarios, which is essential for real-world phishing URL detection systems.

**Optimized Hyperparameters:** Random Forest offers various hyperparameters that can be fine-tuned to optimize its performance, such as the number of trees in the forest, the maximum depth of each tree, and the number of features considered at each split. By carefully tuning these hyperparameters, Random Forest can achieve superior performance in terms of accuracy compared to other algorithms.

### 3.3 Designing:

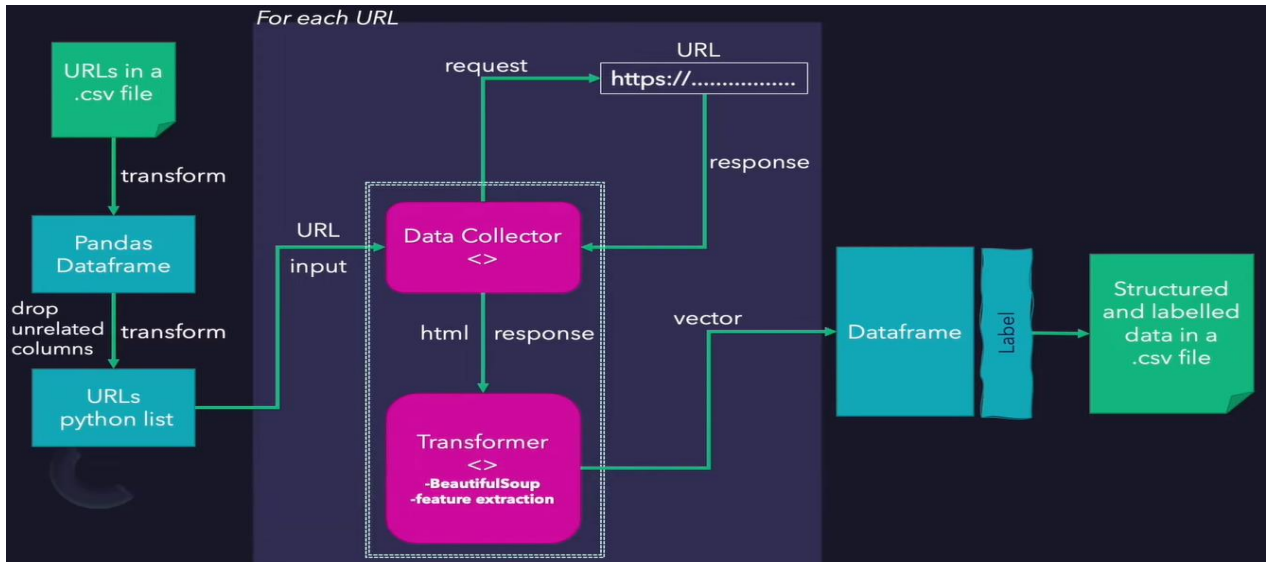


Fig 3.3.1 : Designing

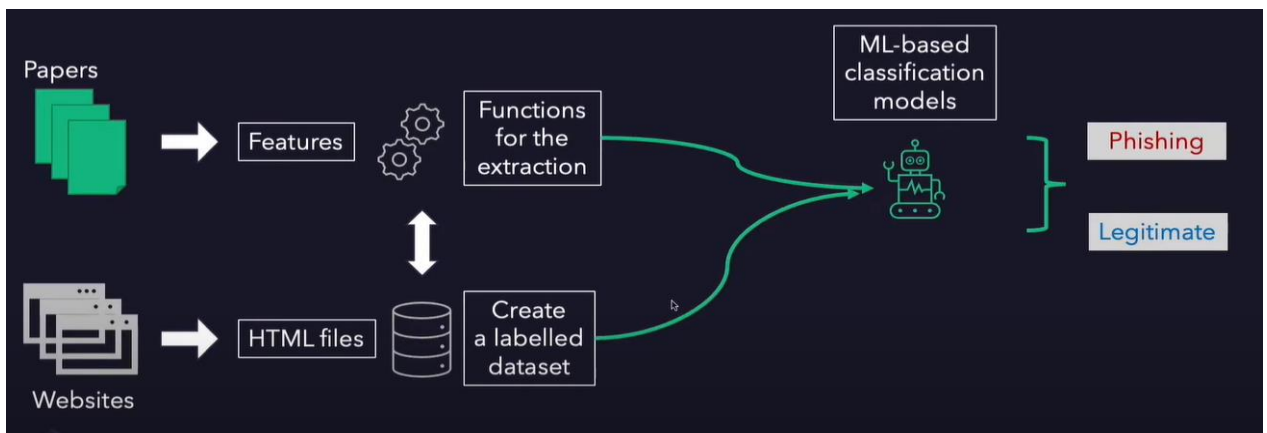
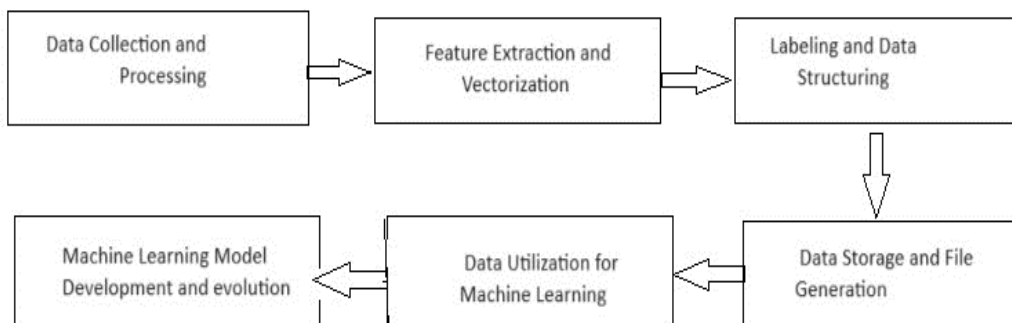


Fig 3.3.2 : Designing

### 3.3.1 UML Diagram:



**Fig 3.3.1.1 UML Diagram**

**Data Collection and processing:** The utilization of CSV files streamlines the extraction of URLs for subsequent analysis. This automated process ensures efficiency and consistency in gathering the requisite data, minimizing errors, and saving valuable time. By parsing the CSV file, URLs can be systematically retrieved, providing a comprehensive dataset for further investigation.

Overall, leveraging a CSV file for data collection enhances the reproducibility and reliability of the analysis process. It provides a structured framework for organizing and managing URLs, laying the foundation for in-depth analysis and exploration of potential threats. This streamlined approach optimizes resource utilization and enables researchers to focus on analyzing the extracted data effectively.

**Feature Extraction and vectorization:** By leveraging the requests library in Python, HTTP requests are dispatched to every URL, with Beautiful soup employed to parse the response content. This process enables the extraction of pertinent features from website content. This process enables the extraction of pertinent features from website content, thereby facilitating the creation of structured data conducive to analysis. Features such as metadata, textual content, and structural elements are systematically extracted and encoded into numerical representations, forming a structured feature vector for each URL.



**Labeling and Data Structuring :** The structured data undergoes augmentation with the labels denoting whether a website is classified as phishing (1) or legitimate (0). This crucial step ensures that the data is properly categorized for subsequent analysis, allowing for targeted examination of different website types. By associating each data point with its corresponding label, researchers and machine learning algorithms can differentiate between benign and malicious websites, facilitating accurate detection and classification.

**Data storage and File Generation :** The structured data, now labeled and organized is saved as CSV files (“structured\_data\_legitimate.csv” and “structured\_data\_phishing.csv”). This enables easy access and manipulation of the data for further analysis or sharing with other stakeholders. This meticulous approach ensures data integrity and accessibility, facilitating seamless retrieval and manipulation for further analysis or dissemination to stakeholders. CSV files offer a universally compatible format, enabling easy import into various data analysis tools and platforms. Additionally, the structured naming convention enhances clarity and organization, simplifying navigation and management of datasets. Ultimately, this streamlined process optimizes data storage and retrieval, empowering researchers to conduct in-depth analysis and derive actionable insights efficiently.

**Data Utilization for Machine Learning:** The structured data serves as the basis for training and testing machine learning models. By combining legitimate and phishing data, robust models can be developed to identify and classify potential threats accurately. By amalgamating both legitimate and phishing data, comprehensive datasets are created to develop robust models capable of accurately identifying and classifying potential threats. This approach ensures that the models are trained on diverse and representative samples, enhancing their ability to generalize and effectively detect malicious activities.

**Machine Learning model Development and Evolution:** Five different machine learning models- Support Vector Machine, Gaussian Naïve Bayes, Decision Tree, Random Forest, and AdaBoost are implemented and evaluated. This ensures a comprehensive assessment of various algorithms' effectiveness in detecting phishing websites. Visualizing the performance measures, such as accuracy, precision, and recall, for each model aids in comparative analysis and decision-making. This allows for the identification of the most suitable model for the dataset and problem domain, facilitating informed decision-making in cybersecurity efforts.

### 3.4. Implementation and Code:

#### Stepwise Implementation and code:

- Use csv file to get URLs
- Send a request to each URL and receive a response by requests library of python
- Use the content of response and parse it by BeautifulSoup module
- Extract features and create a vector which contains numerical values for each feature
- Repeat feature extraction process for all content\websites and create a structured dataframe
- Add label at the end to the dataframes | 1 for phishing 0 for legitimate
- Save the dataframe as csv and structured\_data files are ready!
  - Check "structured\_data\_legitimate.csv" and "structured\_data\_phishing.csv"
- After obtaining structured data, you can use combine them and use them as train and test data
- You can split data as train and test like in the machine\_learning.py first part, or you can implement K-fold cross-validation like in the second part of the same file. I implemented K-fold as K=5.
- Then I implemented five different ML models:
  - Support Vector Machine
  - Gaussian Naive Bayes
  - Decision Tree
  - Random Forest
  - AdaBoost
- You can obtain the confusion matrix, and performance measures: accuracy, precision, recall
- Finally, I visualized the performance measures for all models
- We have selected random forest.

**App.py :**

```
import streamlit as st
import machine_learning as ml
import feature_extraction as fe
from bs4 import BeautifulSoup
import requests as re
import matplotlib.pyplot as plt
# col1, col2 = st.columns([1, 3])
st.title('Phishing Website Detection using Machine Learning')
with st.expander("PROJECT DETAILS"):
    st.subheader('Approach')
    st.write('we used _supervised learning_ to classify phishing and legitimate websites. '
            'we benefit from content-based approach and focus on html of the websites. '
            'Also, we used scikit-learn for the ML models.'
    )
    st.write('For this educational project, '
            'we created my own data set and defined features, some from the literature and some
based on manual analysis. '
            'we used requests library to collect data, BeautifulSoup module to parse and extract
features. ')
    st.write('The source code and data sets are available in the below Github link:')
    st.write('_https://github.com/emre-kocyigit/phishing-website-detection-content-based_')

    st.subheader('Data set')
    st.write('I used _"phishtank.org"_ & _"tranco-list.eu"_ as data sources.')
    st.write('Totally 53186 websites ==> **_32120_ legitimate** websites | **_21048_
phishing** websites')
    st.write('Data set was created in December 2023.')

    labels = 'phishing', 'legitimate'

    phishing_rate = int(ml.phishing_df.shape[0]) / (ml.phishing_df.shape[0] +
```

```
ml.legitimate_df.shape[0]) * 100)

legitimate_rate = 100 - phishing_rate

sizes = [phishing_rate, legitimate_rate]

explode = (0.1, 0)

fig, ax = plt.subplots()

ax.pie(sizes, explode=explode, labels=labels, shadow=True, startangle=90,
autopct='%1.1f%%')

ax.axis('equal')

st.pyplot(fig)

st.write('Features + URL + Label ==> Dataframe')

st.markdown('label is 1 for phishing, 0 for legitimate')

number = st.slider("Select row number to display", 0, 100)

st.dataframe(ml.legitimate_df.head(number))

@st.cache

def convert_df(df):

    return df.to_csv().encode('utf-8')

    csv = convert_df(ml.df)

st.download_button(

    label="Download data as CSV",

    data=csv,

    file_name='phishing_legitimate_structured_data.csv',    mime='text/csv',

    ) st.subheader('Features')
```

```
st.write('I used only content-based features. I didn\'t use url-based features like  
length of url etc.' 'Most of the features extracted using find_all() method of  
BeautifulSoup module after parsing html.') st.subheader('Results')
```

```
st.write('I used 7 different ML classifiers of scikit-learn and tested them  
implementing k-fold cross validation.' 'Firstly obtained their confusion matrices, then  
calculated their accuracy, precision and recall scores.' 'Comparison table is below:')
```

```
st.table(ml.df_results)
```

```
st.write('NB --> Gaussian Naive Bayes')
```

```
st.write('SVM --> Support Vector Machine')
```

```
st.write('DT --> Decision Tree')
```

```
st.write('RF --> Random Forest')
```

```
st.write('AB --> AdaBoost')
```

```
st.write('NN --> Neural Network')
```

```
st.write('KN --> K-Neighbours')
```

```
with st.expander('EXAMPLE PHISHING URLs:')
```

```
st.write('_https://rtyu38.godaddysites.com/_')
```

```
st.write('_https://karafuru.invite-mint.com/_')
```

```
st.write('_https://defi-ned.top/h5/#/_')
```

```
st.caption('REMEMBER, PHISHING WEB PAGES HAVE SHORT  
LIFECYCLE!')
```

```
choice = st.selectbox("Based on accuracy selected model",
```

```
[ 'Random ' ] )
```

```
model = ml.nb_model
```

```
if choice == 'Gaussian Naive Bayes':
```

```
model = ml.nb_model

st.write('GNB model is selected!')

elif choice == 'Support Vector Machine':

    model = ml.svm_model

    st.write('SVM model is selected!')

elif choice == 'Decision Tree':

    model = ml.dt_model

    st.write('DT model is selected!')

elif choice == 'Random Forest':

    model = ml.rf_model

    st.write('RF model is selected!')

elif choice == 'AdaBoost':

    model = ml.ab_model

    st.write('AB model is selected!')

elif choice == 'Neural Network':

    model = ml.nn_model

    st.write('NN model is selected!')

else:

    model = ml.kn_model

    st.write('KN model is selected!')url = st.text_input('Enter the URL')

if st.button('Check!'):

    try:
```

```
response = re.get(url, verify=False, timeout=4)

if response.status_code != 200:

    print(". HTTP connection was not successful for the URL: ", url)

else:

    soup = BeautifulSoup(response.content, "html.parser")

    vector = [fe.create_vector(soup)] # it should be 2d array, so I added []

    result = model.predict(vector)

    if result[0] == 0:

        st.success("This web page seems a legitimate!")

        st.balloons()

    else:

        st.warning("Attention! This web page is a potential PHISHING!")

        st.snow()

except re.exceptions.RequestException as e:

    print("--> ", e)
```

### **Machine\_learning.py**

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn import tree
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt

legitimate_df = pd.read_csv("structured_data_legitimate.csv")
phishing_df = pd.read_csv("structured_data_phishing.csv")
df = pd.concat([legitimate_df, phishing_df], axis=0)
df = df.sample(frac=1)
df = df.drop('URL', axis=1)
df = df.drop_duplicates()
X = df.drop('label', axis=1)
Y = df['label']

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=10)

svm_model = svm.LinearSVC()
rf_model = RandomForestClassifier(n_estimators=60)
dt_model = tree.DecisionTreeClassifier()
nb_model = GaussianNB()
nn_model = MLPClassifier(alpha=1)
kn_model = KNeighborsClassifier()
```



```
svm_model.fit(x_train, y_train)
predictions = svm_model.predict(x_test)
tn, fp, fn, tp = confusion_matrix(y_true=y_test, y_pred=predictions).ravel()
accuracy = (tp + tn) / (tp + tn + fp + fn)
precision = tp / (tp + fp)
recall = tp / (tp + fn)
print("accuracy --> ", accuracy)
print("precision --> ", precision)
print("recall --> ", recall)
K = 5
total = X.shape[0]
index = int(total / K)
X_1_test = X.iloc[:index]
X_1_train = X.iloc[index:]
Y_1_test = Y.iloc[:index]
Y_1_train = Y.iloc[index:]
X_2_test = X.iloc[index:index*2]
X_2_train = X.iloc[np.r_[:index, index*2:]]
Y_2_test = Y.iloc[index:index*2]
Y_2_train = Y.iloc[np.r_[:index, index*2:]]
X_3_test = X.iloc[index*2:index*3]
X_3_train = X.iloc[np.r_[:index*2, index*3:]]
Y_3_test = Y.iloc[index*2:index*3]
Y_3_train = Y.iloc[np.r_[:index*2, index*3:]]
X_4_test = X.iloc[index*3:index*4]
X_4_train = X.iloc[np.r_[:index*3, index*4:]]
Y_4_test = Y.iloc[index*3:index*4]
Y_4_train = Y.iloc[np.r_[:index*3, index*4:]]
X_5_train = X.iloc[:index*4]
Y_5_test = Y.iloc[index*4:]
```

```

Y_5_train = Y.iloc[:index*4]
X_train_list = [X_1_train, X_2_train, X_3_train, X_4_train, X_5_train]
X_test_list = [X_1_test, X_2_test, X_3_test, X_4_test, X_5_test]
Y_train_list = [Y_1_train, Y_2_train, Y_3_train, Y_4_train, Y_5_train]
Y_test_list = [Y_1_test, Y_2_test, Y_3_test, Y_4_test, Y_5_test]

def calculate_measures(TN, TP, FN, FP):
    model_accuracy = (TP + TN) / (TP + TN + FN + FP)
    model_precision = TP / (TP + FP)
    model_recall = TP / (TP + FN)
    return model_accuracy, model_precision, model_recall

for i in range(0, K):
    rf_model.fit(X_train_list[i], Y_train_list[i])
    rf_predictions = rf_model.predict(X_test_list[i])
    tn, fp, fn, tp = confusion_matrix(y_true=Y_test_list[i], y_pred=rf_predictions).ravel()
    rf_accuracy, rf_precision, rf_recall = calculate_measures(tn, tp, fn, fp)
    rf_accuracy_list.append(rf_accuracy)
    rf_precision_list.append(rf_precision)
    rf_recall_list.append(rf_recall)
    dt_model.fit(X_train_list[i], Y_train_list[i])
    dt_predictions = dt_model.predict(X_test_list[i])
    tn, fp, fn, tp = confusion_matrix(y_true=Y_test_list[i], y_pred=dt_predictions).ravel()
    dt_accuracy, dt_precision, dt_recall = calculate_measures(tn, tp, fn, fp)
    dt_accuracy_list.append(dt_accuracy)
    dt_precision_list.append(dt_precision)
    svm_predictions = svm_model.predict(X_test_list[i])
    tn, fp, fn, tp = confusion_matrix(y_true=Y_test_list[i], y_pred=svm_predictions).ravel()
    svm_accuracy, svm_precision, svm_recall = calculate_measures(tn, tp, fn, fp)
    svm_accuracy_list.append(svm_accuracy)
    ab_predictions = ab_model.predict(X_test_list[i])
    tn, fp, fn, tp = confusion_matrix(y_true=Y_test_list[i], y_pred=ab_predictions).ravel()
    ab_accuracy, ab_precision, ab_recall = calculate_measures(tn, tp, fn, fp)

```

```

nb_model.fit(X_train_list[i], Y_train_list[i])
nb_predictions = nb_model.predict(X_test_list[i])
tn, fp, fn, tp = confusion_matrix(y_true=Y_test_list[i], y_pred=nb_predictions).ravel()
nb_accuracy, nb_precision, nb_recall = calculate_measures(tn, tp, fn, fp)
nb_accuracy_list.append(nb_accuracy)
nb_precision_list.append(nb_precision)
nn_accuracy_list.append(nn_accuracy)
nn_precision_list.append(nn_precision)
nn_recall_list.append(nn_recall)
kn_model.fit(X_train_list[i], Y_train_list[i])
kn_precision_list.append(kn_precision)
kn_recall_list.append(kn_recall)

RF_accuracy = sum(rf_accuracy_list) / len(rf_accuracy_list)
RF_precision = sum(rf_precision_list) / len(rf_precision_list)
RF_recall = sum(rf_recall_list) / len(rf_recall_list)
print("Random Forest accuracy ==> ", RF_accuracy)
print("Random Forest precision ==> ", RF_precision)
print("Random Forest recall ==> ", RF_recall)

DT_accuracy = sum(dt_accuracy_list) / len(dt_accuracy_list)
DT_precision = sum(dt_precision_list) / len(dt_precision_list)
print("Decision Tree accuracy ==> ", DT_accuracy)
print("Decision Tree precision ==> ", DT_precision)

AB_accuracy = sum(ab_accuracy_list) / len(ab_accuracy_list)
AB_precision = sum(ab_precision_list) / len(ab_precision_list)
AB_recall = sum(ab_recall_list) / len(ab_recall_list)
print("AdaBoost accuracy ==> ", AB_accuracy)
print("AdaBoost precision ==> ", AB_precision)

SVM_accuracy = sum(svm_accuracy_list) / len(svm_accuracy_list)
SVM_precision = sum(svm_precision_list) / len(svm_precision_list)
SVM_recall = sum(svm_recall_list) / len(svm_recall_list)

```

```

print("Support Vector Machine accuracy ==> ", SVM_accuracy)
print("Support Vector Machine precision ==> ", SVM_precision)
print("Support Vector Machine recall ==> ", SVM_recall)
NB_accuracy = sum(nb_accuracy_list) / len(nb_accuracy_list)
NB_precision = sum(nb_precision_list) / len(nb_precision_list)
NB_recall = sum(nb_recall_list) / len(nb_recall_list)
print("Gaussian Naive Bayes accuracy ==> ", NB_accuracy)
print("Gaussian Naive Bayes precision ==> ", NB_precision)
print("Gaussian Naive Bayes recall ==> ", NB_recall)
NN_accuracy = sum(nn_accuracy_list) / len(nn_accuracy_list)
NN_precision = sum(nn_precision_list) / len(nn_precision_list)
NN_recall = sum(nn_recall_list) / len(nn_recall_list)
print("Neural Network accuracy ==> ", NN_accuracy)
print("Neural Network precision ==> ", NN_precision)
print("Neural Network recall ==> ", NN_recall)
KN_accuracy = sum(kn_accuracy_list) / len(kn_accuracy_list)
KN_precision = sum(kn_precision_list) / len(kn_precision_list)
KN_recall = sum(kn_recall_list) / len(kn_recall_list)
print("K-Neighbours Classifier precision ==> ", KN_precision)
print("K-Neighbours Classifier recall ==> ", KN_recall)
data = {'accuracy': [NB_accuracy, SVM_accuracy, DT_accuracy, RF_accuracy, AB_accuracy,
NN_accuracy, KN_accuracy],
        'precision': [NB_precision, SVM_precision, DT_precision, RF_precision, AB_precision,
NN_precision, KN_precision],
        'recall': [NB_recall, SVM_recall, DT_recall, RF_recall, AB_recall, NN_recall,
KN_recall] } index = ['NB', 'SVM', 'DT', 'RF', 'AB', 'NN', 'KN']
df_results = pd.DataFrame(data=data, index=index)
ax = df_results.plot.bar(rot=0)
plt.show()

```

# **CHAPTER 4**

## **RESULTS AND DISCUSSION**

## **CHAPTER 4**

### **RESULTS AND DISCUSSION**

The integration of data mining techniques into cybersecurity operations has emerged as a pivotal asset for organizations combating malicious sites. Utilizing classification algorithms like decision trees, support vector machines, and neural networks, data mining facilitates automatic and accurate detection and classification of threats. Its adaptive nature ensures continuous learning, enabling the identification of emerging threats and previously unseen malicious behavior patterns. Moreover, the scalability of data mining allows efficient analysis of vast and increasingly complex datasets, vital for organizations navigating the evolving cyber threat landscape.

Correlating data from diverse sources, including network logs, endpoint telemetry, and user behavior analytics, data mining provides organizations with a holistic security perspective. This comprehensive understanding not only bolsters their security posture but also informs broader cybersecurity strategies, enabling targeted countermeasures and resource allocation. Despite its advantages, challenges such as data quality, privacy issues, and potential false positives/negatives persist. Nevertheless, with advancements in artificial intelligence and deep learning, coupled with the continuous evolution of data mining techniques, its role in cybersecurity remains indispensable. In conclusion, data mining stands as a potent tool, empowering organizations to proactively identify and mitigate cyber threats, ensuring they remain a step ahead of adversaries in today's dynamic threat landscape.

## Overview of the website :

# Phishing Website Detection using Machine Learning

PROJECT DETAILS ▼

EXAMPLE PHISHING URLs: ▼

Based on accuracy selected model

Random Forest ▼

RF model is selected!

Enter the URL

Check!

Fig 4.1 Overview of website

**Result of the website when it is legitimate (real) :**

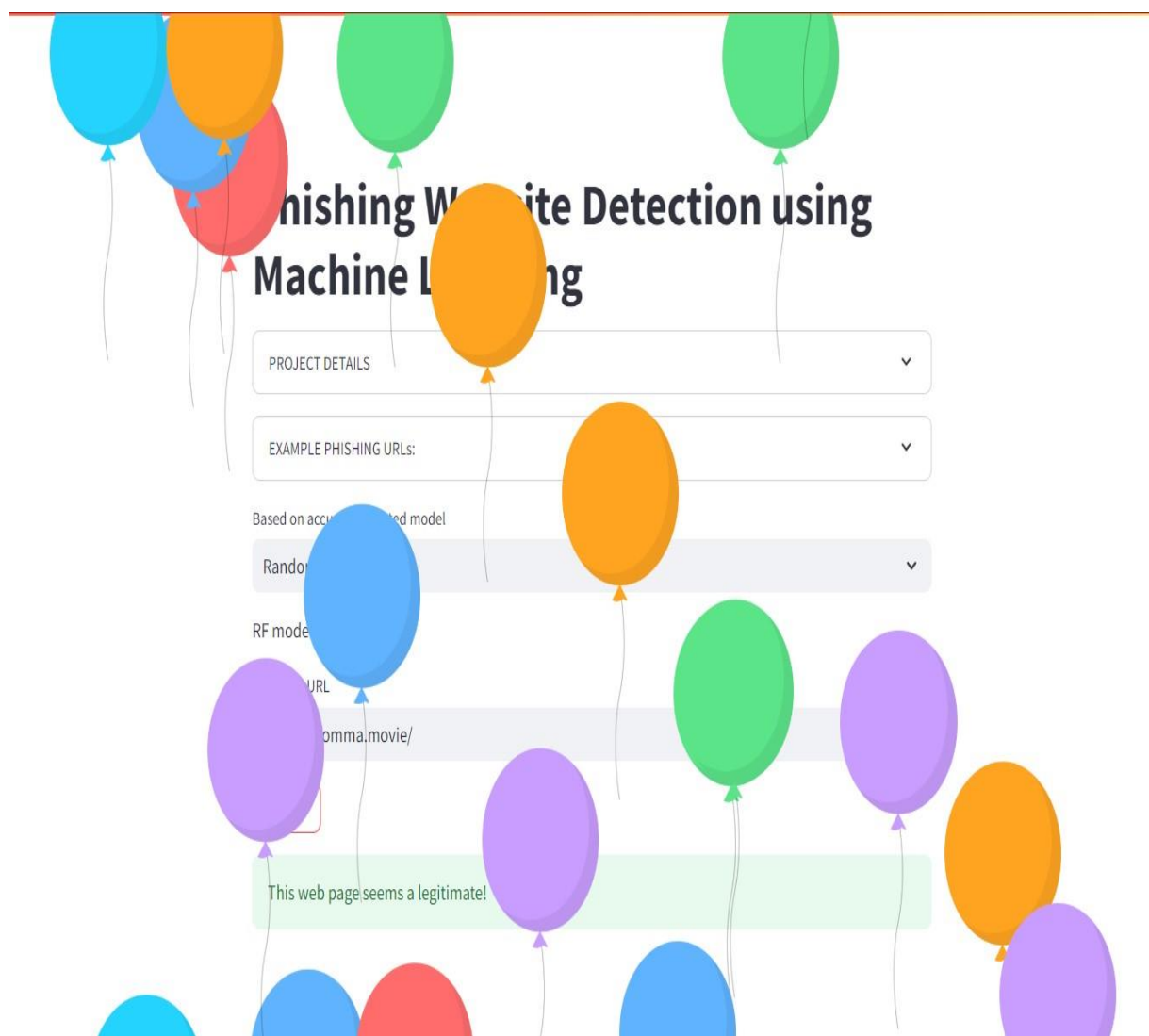
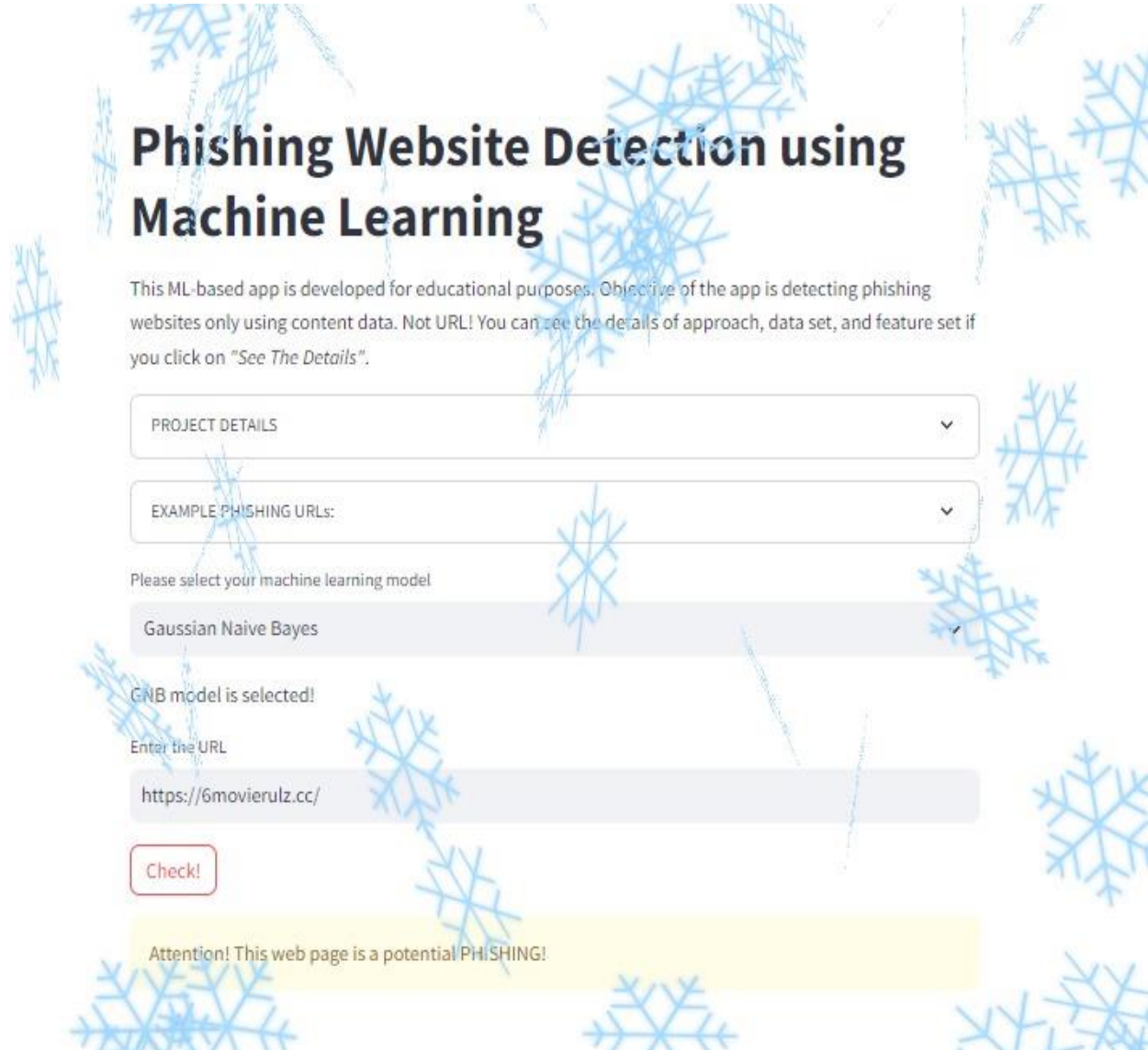


Fig 4.2 Legitimate



**Result of the website when it is Phishing (fake) :**



The screenshot shows a web application titled "Phishing Website Detection using Machine Learning". The interface includes a description of the app's purpose, a dropdown menu for "PROJECT DETAILS", another dropdown for "EXAMPLE PHISHING URLs:", and a section for selecting a machine learning model. The "Gaussian Naive Bayes" model is selected. Below this, there is a text input field for "Enter the URL" containing the address "https://6movierulz.cc/". A red "Check!" button is positioned below the input field. At the bottom, a yellow alert box displays the message: "Attention! This web page is a potential PHISHING!". The entire interface is decorated with a light blue snowflake pattern.

## Phishing Website Detection using Machine Learning

This ML-based app is developed for educational purposes. Objective of the app is detecting phishing websites only using content data. Not URL! You can see the details of approach, data set, and feature set if you click on "See The Details".

PROJECT DETAILS

EXAMPLE PHISHING URLs:

Please select your machine learning model

Gaussian Naive Bayes

GNB model is selected!

Enter the URL

https://6movierulz.cc/

Check!

Attention! This web page is a potential PHISHING!

Fig 4.3 Phishing

**Project Details of the website :**

# Phishing Website Detection using Machine Learning

PROJECT DETAILS

## Approach

we used *supervised learning* to classify phishing and legitimate websites. we benefit from content-based approach and focus on html of the websites. Also, we used scikit-learn for the ML models.

For this educational project, we created my own data set and defined features, some from the literature and some based on manual analysis. we used requests library to collect data, BeautifulSoup module to parse and extract features.

The source code and data sets are available in the below Github link:

<https://github.com/emre-kocyigit/phishing-website-detection-content-based>

## Data set

I used "phishtank.org" & "tranco-list.eu" as data sources.

Totally 53186 websites ==> **32120 legitimate** websites | **21048 phishing** websites

Data set was created in December 2023.

Fig 4.4 Project details

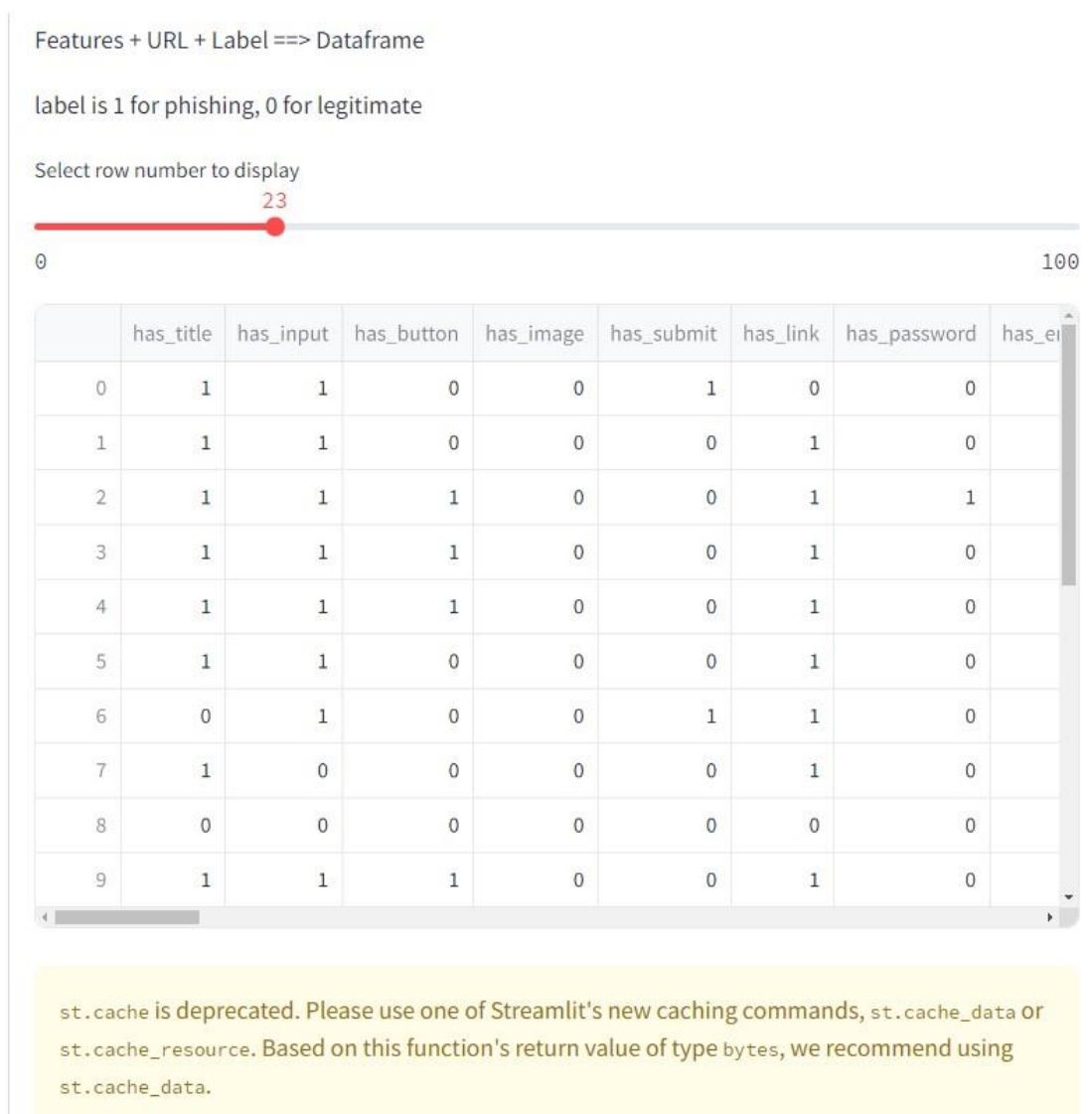


Fig 4.5 Feature Table

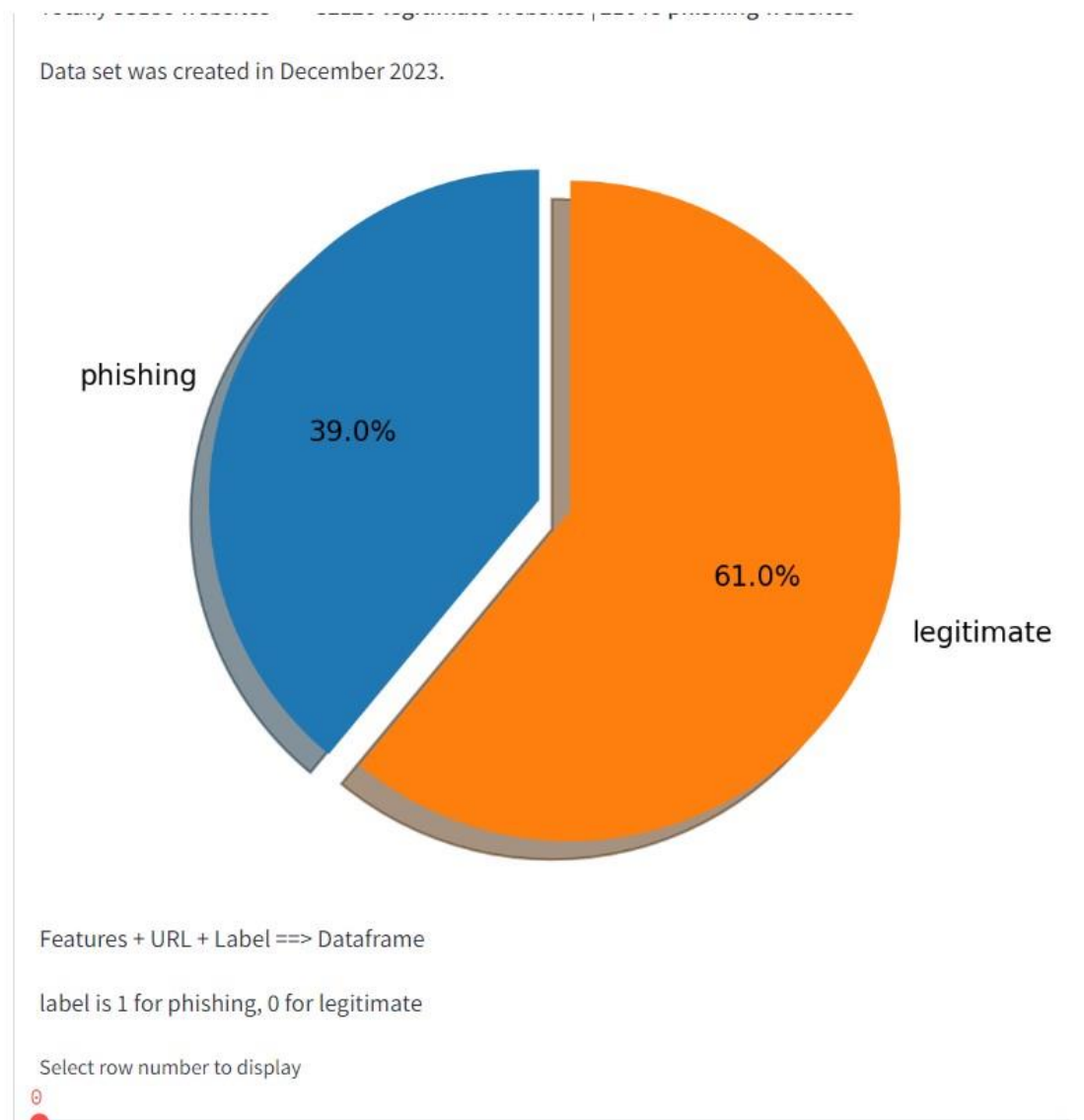


Fig 4.6 Pie chart

## Features

I used only content-based features. I didn't use url-based features like length of url etc. Most of the features extracted using find\_all() method of BeautifulSoup module after parsing html.

## Results

I used 7 different ML classifiers of scikit-learn and tested them implementing k-fold cross validation. Firstly obtained their confusion matrices, then calculated their accuracy, precision and recall scores. Comparison table is below:

	accuracy	precision	recall
NB	0.5291	0.0411	0.9066
SVM	0.9337	0.1981	0.1964
DT	0.9855	0.6870	0.7054
RF	0.9908	0.8525	0.6655
AB	0.9786	0.5685	0.1836
NN	0.9780	<NA>	0.0535
KN	0.9772	0.4065	0.1538

NB --> Gaussian Naive Bayes

SVM --> Support Vector Machine

DT --> Decision Tree

RF --> Random Forest

Fig 4.7 Accuracy

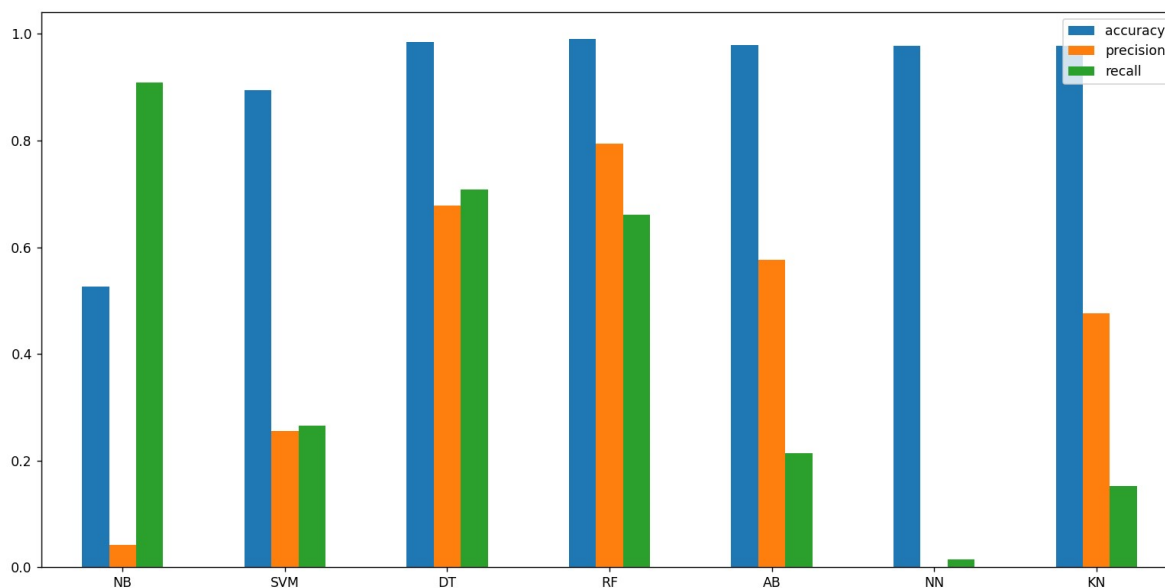


Fig 4.8 Bar Graph

# CHAPTER 5

## CONCLUSION

## **CHAPTER 5**

### **CONCLUSION**

Leveraging data mining techniques for detecting malicious sites offers a robust and versatile approach to cybersecurity. By analyzing vast datasets containing website characteristics, user behaviors, and history of instances of malicious activities, data mining algorithms can effectively identify patterns and indicators of potential threats. The utilization of classification algorithms, such as decision trees, support vector machines, or neural networks, enables the automated detection and classification of malicious sites with high accuracy. Furthermore, data mining facilitates proactive defense against evolving threats by continuously learning from new data and updating classification criteria accordingly. Its adaptive nature allows for the detection of emerging threats and the identification of previously unseen patterns of malicious behavior. Additionally, the insights gleaned from data mining analysis can inform broader cybersecurity strategies and decision-making processes within organizations. By identifying common attack patterns, trends, and vulnerabilities, organizations can strengthen their security posture, implement targeted countermeasures, and allocate resources more effectively to mitigate risks. Moreover, the scalability of data mining enables organizations to analyze large volumes of data efficiently, even in the face of increasing data volumes and complexity this scalability ensures that organizations can effectively scale their cybersecurity efforts to meet the growing challenges posed by cyber threats. In conclusion, this project represents a data mining that offers a potent and versatile approach to detecting malicious sites, leveraging advanced analytics and machine learning to uncover hidden threats and vulnerabilities. By harnessing the power of data, organizations can enhance their cybersecurity defenses, protect sensitive information, and safeguard against financial losses and reputational damage. In today's rapidly evolving threat landscape, data mining remains a cornerstone of effective cybersecurity strategies, providing organizations with the tools they need to stay one step ahead of cyber adversaries.

**Future Enhancement:**

**Advanced Machine Learning Algorithms:**

Develop machine learning algorithms that can analyze and detect subtle patterns in phishing websites, even those mimicking legitimate sites. Implement real-time learning capabilities to adapt to new phishing tactics and variations rapidly.

**Behavioral Biometrics Authentication:**

Integrate behavioral biometrics, such as keystroke dynamics and mouse movement patterns, into authentication processes to enhance user verification. Use these biometrics to identify irregularities in user behavior that may indicate a phishing attempt.

**Multi-Factor Authentication (MFA):**

Promote the widespread adoption of MFA for online transactions and account logins.

Implement adaptive MFA that adjusts the level of authentication based on the perceived risk of the transaction or login attempt.

**Improved Email Filtering and AI-driven Analysis:**

Develop AI-driven email filtering systems capable of identifying and blocking sophisticated phishing emails with high accuracy. Integrate natural language processing (NLP) to analyze email content and identify deceptive or malicious intent.

**User Education and Awareness Campaigns:**

Launch comprehensive awareness campaigns to educate users about the latest phishing tactics and how to recognize and avoid them. Provide regular updates and training sessions to keep users informed about emerging threats and protective measures.

**Collaborative Threat Intelligence Sharing:**

Establish a collaborative platform where organizations can share threat intelligence and indicators of compromise (IoCs) in real-time. Foster collaboration between public and private sectors to enhance collective defense against phishing and other cyber threats.



# REFERENCES

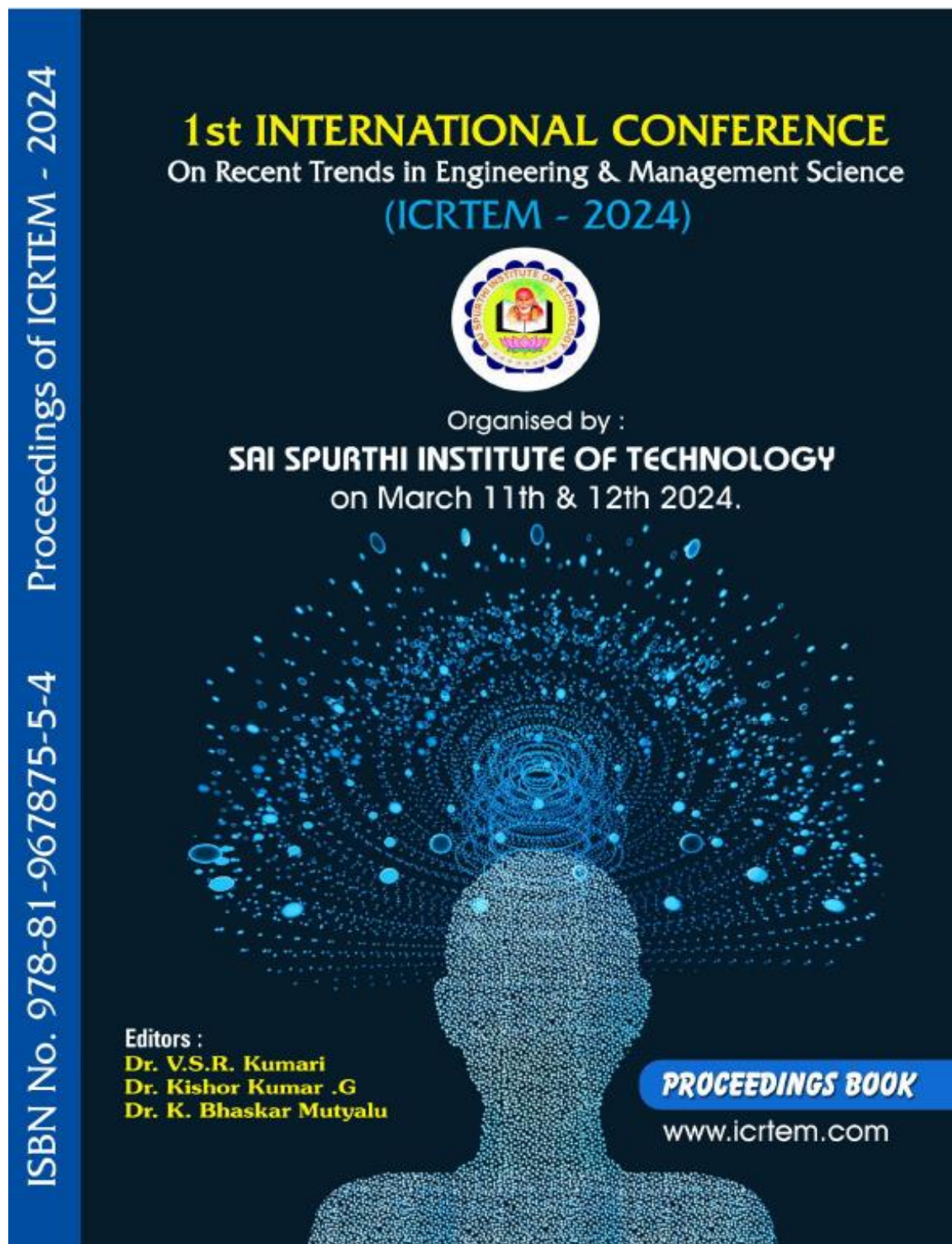
## REFERENCES

- [1] Naim, Or, Doron Cohen, and Irad Ben-Gal. "Malicious website identification using design attribute learning." *International Journal of Information Security* (2023):111. <https://link.springer.com/10.1007/s10207-023-00686-y>
- [2] Shantanu, B. Janet and R. Joshua Arul Kumar, "Malicious URL Detection: A Comparative Study," 2021 International Conference on Artificial Intelligence and Smart Systems(ICAIS),Coimbatore India,2021,pp.11471151,doi:10.1109/ICAIS50930.2021.9396014. keywords:{Uniformresource locators;Training;Phishing;Predictivemodels;Malware;Randomforests;Web search;Malicious URL;Machine learning;Phishing;Spamming;Malware;Spoofing}, <https://ieeexplore.ieee.org/document/9396014>
- [3] L. Wei, W. Luo, J. Weng, Y. Zhong, X. Zhang and Z. Yan, "Machine Learning-Based Malicious Application Detection of Android," in *IEEE Access*, vol. 5, pp. 25591-25601, 2017, doi: 10.1109/ACCESS.2017.2771470. keywords: {Smart phones;Androids;Humanoid robots;Security;Machine learning algorithms;Computerscience;Training;Maliciousapplications of Android;machine learning;system function}, <https://ieeexplore.ieee.org/document/8101455>
- [4] Aalla, Harsha Vardhan Sai, Nikhil Reddy Dumpala, and M. Eliazer. "Malicious URL prediction using machine learning techniques." *Annals of the Romanian Society for Cell Biology* (2021): 2170-2176.<https://www.annalsofrscb.ro/index.php/journal/article/view/4752>.
- [5][https://www.researchgate.net/publication/351676213\\_Android\\_Malware\\_Detection\\_Using\\_Deep\\_Learning](https://www.researchgate.net/publication/351676213_Android_Malware_Detection_Using_Deep_Learning)

## GITHUB LINK

<https://github.com/Naveen5453/maliciousdetection/tree/main>

## PUBLICATION PAPER



## DETECTING MALICIOUS SITES:

<sup>#1</sup> Farheen, <sup>#2</sup> G. Naveen, <sup>#3</sup> I. Sree Anvita, <sup>#4</sup> Ms. B. Gayathri

<sup>1,2,3</sup> UG Student, Department of CSE, CMR College of Engineering & Technology, Hyderabad, Telangana, India

<sup>4</sup> Assistant Professor, Department of CSE, CMR College of Engineering & Technology, Hyderabad, Telangana, India

*Corresponding Author: Farheen, farheenfarheen1109@gmail.com*

**Abstract**—Several users purchase products online and make payments through various websites. Multiple websites ask users to provide sensitive data such as usernames, passwords credit card details, etc. often for malicious reasons. This type of website is known as a phishing website. In order to detect and predict phishing websites, we proposed an intelligent, flexible, and effective system that is based on using a classification Data mining algorithm. We implemented classification algorithms and techniques to extract the phishing data set criteria to classify their legitimacy. The phishing website can be detected based on some important characteristics like URL and Domain Identity, and security and encryption criteria in the final phishing detection rate. Once the user makes a transaction online when he makes payment through the website our system will use a data mining algorithm to detect whether the website is phishing website or not. This application can be used by many E-commerce enterprises to make the whole transaction process secure. The data mining algorithm used in this system provides better performance as compared to other traditional classification algorithms. With the help of this system, users can also purchase products online without any hesitation.

**Keywords**— *Phishing website detection, Datamining algorithm, Classification techniques, Online Transaction security, E-commerce Enterprises, URL and Domain identity, Security and encryption criteria.*

### I. INTRODUCTION

There is a growing trend of users purchasing products online and conducting financial transactions across various websites. However, this convenience is accompanied by the looming threat of phishing websites, which maliciously request sensitive information such as usernames, passwords, and credit card details. Phishing websites, often disguised as legitimate platforms, exploit unsuspecting users for malicious purposes. This cybercrime involves the dissemination of deceptive messages that appear to originate from trustworthy sources. Messages typically contain URLs or attachments that, when interacted with, can lead to the theft of personal information or the installation of harmful malware onto the recipient's device. Traditionally, phishing attacks were executed through mass spam campaigns, indiscriminately targeting large groups of individuals. Today, phishing has emerged as one of the most potent and damaging forms of cyber-attacks due to the lack of robust identification techniques and protective measures.

As the prevalence of phishing continues to rise, there is a pressing need for advanced strategies to combat this pervasive threat. Addressing this challenge requires the development and implementation of sophisticated detection and prevention mechanisms. By leveraging innovative technologies such as machine learning algorithms and artificial intelligence, organizations can enhance their ability to identify and thwart phishing attempts effectively.

### II. RELATED WORK

In the quest for innovation and efficiency, modern projects frequently rely on existing solutions as fundamental building blocks for development. This approach not only recognizes the expertise and advancements of those who came before us but also nurtures a collaborative ecosystem where

ideas can evolve and confront new challenges. In our project, we wholeheartedly embrace this ethos, conscientiously integrating elements from existing solutions to enrich our endeavor. These existing solutions serve as guiding lights, offering insights and frameworks that shape the direction of our project.

#### A. Phishing website detection using Email Filtration:

Phishing website detection using Email Filters involves employing advanced algorithms to filter incoming emails for signs of phishing attempts. These algorithms analyze various attributes of the email, such as sender information, content, and embedded links, to identify suspicious patterns indicative of phishing. Through a combination of rule-based filters and machine-learning techniques, the filtration system can accurately differentiate between legitimate emails and phishing attempts.

#### B. Implementation of malicious sites using URL verification:

Phishing app detection using URL verification involves scrutinizing URLs embedded within applications to identify potential phishing attempts. This process employs sophisticated algorithms to analyze the structure and authenticity of URLs, looking for indicators of malicious intent such as deceptive domain names or suspicious redirects. By verifying the legitimacy of URLs, this approach deceptive links embedded within applications.

#### C. Malicious app detection using Deep learning:

Detecting phishing apps with deep learning training neural network models on datasets containing both legitimate and malicious, these models automatically extract features from app metadata, permissions, and user interface elements. Unlike traditional methods, deep



learning requires no manual feature, enabling it to adapt to new threats. Once trained, the model analyzes new apps in real-time, identifying potential phishing apps by examining extracted features.

#### D. Visual similarity bases malicious site detection.

Visual similarity-based malicious site detection utilizes advanced algorithms to compare the visual appearance of website layouts, logos, colors, and other visual elements, this approach can detect fraudulent websites attempting to mimic legitimate ones. The process involves extracting visual features from website screenshots or HTML code and comparing them against a database of known malicious sites. Machine learning techniques may be employed to improve accuracy and efficiency in identifying visual similarities. Once a potential threat is detected, appropriate actions such as warning users or blocking access can be taken. Continuous against evolving threats, providing users with enhanced protection against phishing and other malicious activities.

### III. METHODS AND DISCUSSIONS

#### A. Data Collection and Processing:

The utilization of CSV files streamlines the extraction of URLs for subsequent analysis. This automated process ensures efficiency and consistency in gathering the requisite data, minimizing errors, and saving valuable time. By parsing the CSV file, URLs can be systematically retrieved, providing a comprehensive dataset for further investigation.

Automation facilitates scalability, allowing for the extraction of URLs from large datasets with ease. Moreover, utilizing a standardized format like CSV enhances interoperability, enabling seamless integration with other data processing tools and workflows.

Overall, leveraging a CSV file for data collection enhances the reproducibility and reliability of the analysis process. It provides a structured framework for organizing and managing URLs, laying the foundation for in-depth analysis and exploration of potential threats. This streamlined approach optimizes resource utilization and enables researchers to focus on analyzing the extracted data effectively.

#### B. Feature Extraction and Vectorization:

By leveraging the requests library in Python, HTTP requests are dispatched to every URL, with Beautiful soup employed to parse the response content. This process enables the extraction of pertinent features from website content. This process enables the extraction of pertinent features from website content, thereby facilitating the creation of structured data conducive to analysis. Features such as metadata, textual content, and structural elements are systematically extracted and encoded into numerical representations, forming a structured feature vector for each URL.

This approach ensures that essential information is captured and standardized, laying the groundwork for subsequent analysis and modeling. Through automated feature extraction and vectorization, the complexity of website content is distilled into a format suitable for machine learning algorithms, enabling robust analysis and detection of phishing websites.

#### C. Labelling and Data Structuring:

The structured data undergoes augmentation with

Labels denoting whether a website is classified as phishing (1) or legitimate (0). This crucial step ensures that the data is properly categorized for subsequent analysis, allowing for targeted examination of different website types. By associating each data point with its corresponding label, researchers and machine learning algorithms can differentiate between benign and malicious websites, facilitating accurate detection and classification.

#### D. Data Storage and File Generation:

The structured data, now labeled and organized is saved as CSV files ("structured\_data\_legitimate.csv" and "structured\_data\_phishing.csv"). This enables easy access and manipulation of the data for further analysis or sharing with other stakeholders. This meticulous approach ensures data integrity and accessibility, facilitating seamless retrieval and manipulation for further analysis or dissemination to stakeholders. CSV files offer a universally compatible format, enabling easy import into various data analysis tools and platforms. Additionally, the structured naming convention enhances clarity and organization, simplifying navigation and management of datasets. Ultimately, this streamlined process optimizes data storage and retrieval, empowering researchers to conduct in-depth analysis and derive actionable insights efficiently.

#### E. Data Utilization for Machine Learning:

The structured data serves as the basis for training and testing machine learning models. By combining legitimate and phishing data, robust models can be developed to identify and classify potential threats accurately. By amalgamating both legitimate and phishing data, comprehensive datasets are created to develop robust models capable of accurately identifying and classifying potential threats. This approach ensures that the models are trained on diverse and representative samples, enhancing their ability to generalize and effectively detect malicious activities.

#### F. Model Selection, Evaluation, and Performance Detection:

Five different machine learning models- Support Vector Machine, Gaussian Naïve Bayes, Decision Tree, Random Forest, and AdaBoost are implemented and evaluated. This ensures a comprehensive assessment of various algorithm's effectiveness in detecting phishing websites.

Visualizing the performance measures, such as accuracy, precision, and recall, for each model aids in comparative analysis and decision-making. This allows for the identification of the most suitable model for the dataset and problem domain, facilitating informed decision-making in cybersecurity efforts.

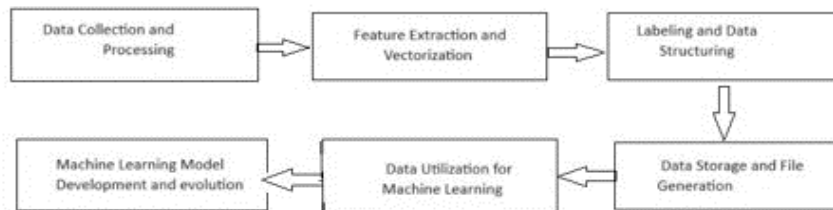


Fig Architecture of the Model

#### IV. RESULTS AND DISCUSSIONS

The exploration of existing solutions sheds light on the diverse approaches and methodologies available to enhance the capabilities of integration of multiple models.

##### A. Malicious website detection using Email filter.

###### Approach:

Incoming emails are thoroughly inspected for signs of phishing, including suspicious sender addresses, unfamiliar content, and embedded links. The email content is for phishing indicators such as an urgent request for personal information, grammatical errors, and alarming messages designed for immediate action. Embedded links are checked to verify their legitimacy.

Links may be flagged or redirected to a safe sandbox environment for further analysis; attachments are scanned for malware or suspicious scripts that could compromise the recipient system; additionally, users may be educated about phishing techniques and encouraged

To report suspicious emails for further investigation.

In phishing website detection through email filtration, a multi-layered approach is employed to ensure comprehensive protection against malicious emails. First, incoming emails undergo a thorough inspection, meticulously sender addresses, content, and embedded links for any signs of suspicious activity.

Content analysis is then conducted, focusing on identifying phishing indicators such as urgent requests for personal information, grammatical errors, or alarming messages aimed at eliciting an immediate response. Embedded links within emails are subjected to verification to ascertain their legitimacy, with any suspicious links flagged or redirected to a secure sandbox environment for further analysis. Additionally, attachments accompanying emails are meticulously examined for malware or suspicious scripts that could potentially compromise the recipient system.

###### Benefits:

Proactive Defense, Real-Time Protection, Efficiency  
Customization scalability.

##### B. Detection of Malicious sites using Deep learning:

###### Approach:

In the process of detecting malicious sites using a machine-learning approach, several key steps are involved. First, feature extraction is conducted, where relevant attributes such as URL structure, domain age, SSL certificate status, website content, and hosting information are extracted from website data. These features provide valuable insights into the characteristics of the websites and serve as inputs for the machine learning model.

Next, a dataset comprising labeled examples of both legitimate and malicious websites is prepared for training data. This dataset is carefully curated to represent a diverse range of website types and threat scenarios, ensuring for the model to learn from.

Subsequently machine learning algorithms such as Decision trees, random forests, or neural networks are trained on the prepared dataset. During this phase, the algorithms learn to identify patterns and distinguish between legitimate and malicious websites based on the extracted features.

###### Benefits:

###### Automated Detection:

Machine learning enables the automated detection of malicious websites, reducing the need for manual intervention and enabling continuous monitoring of web traffic.

###### Scalability:

Machine learning algorithms can handle large volumes of data, making them scalable for analyzing vast amounts of web traffic efficiently.

###### Improves Accuracy:

With access to a wide range of features and patterns, machine learning achieves high accuracy in distinguishing between sites.

### C. Detecting Malicious sites using URLs

#### Approach:

The URL of each website is analyzed to identify suspicious patterns or indicators commonly associated with malicious sites. This includes examining the structure, length, presence of hyphens or numbers, and similarity to known phishing domains.

The reputation of the domain hosting the website is assessed using threat intelligence feeds, blacklists, and historical data. Domains with a history of hosting malicious content or engaging in suspicious activities are flagged as potentially malicious.

The presence and validity of SSL certificates are checked to ensure secure communication between the user browser and the website. Lack of an SSL certificate or expired certificates may indicate a higher risk of malicious intent.

The behavior of the website, such as redirects, pop-ups, and user interaction monitored to detect suspicious activity indicative of phishing or other malicious behavior.

**Benefits:** Early detection, Real-Time protection, Scalability, Cost-Effective, User Awareness.

#### Comparison:

**URL Analysis:** Offers quick and lightweight protection against known threats based on URL characteristics but may be less effective against sophisticated or new threats.

**Email Filtration:** Provides effective protection against phishing attempts in email communications but may miss threats that bypass filtering rules.

**Machine learning:** Offer comprehensive and adaptive protection against various types of threats advanced algorithms and continuous learning to detect emerging threats effectively.

each method has its strengths and weaknesses and the most effective approach is using data mining algorithms in this method we will get an accurate result at first it will consider the URL and calculate its accuracy of it by using training and testing in this it will calculate the accuracy of the all the algorithms it will consider the best out of all the algorithms and give the result accordingly.

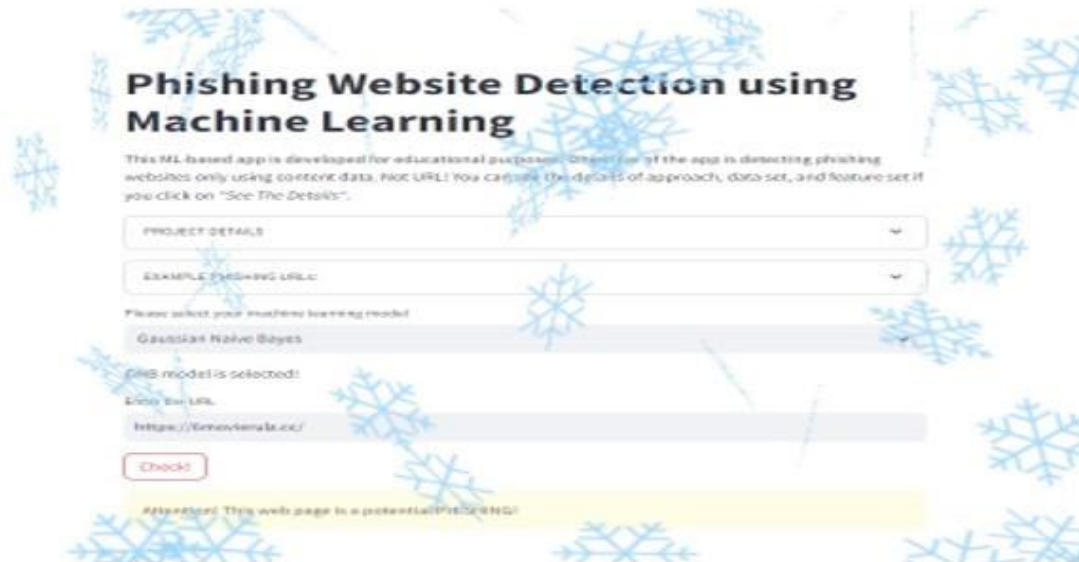


Fig. Detection image 1



### Phishing Website Detection using Machine Learning

This ML based app is developed for educational purposes. Objective of the app is detecting phishing websites only using content data. Not URL! You can see the details of approach, data set, and feature set if you click on "See The Details".

PROJECT DETAILS

EXAMPLE PHISHING URL:

Please select your machine learning model

Support Vector Machine

SVM model is selected!

Enter the URL

https://www.google.com/maps

Check!

This web page seems a legitimate!

Fig 2: Detection image 2

#### V. CONCLUSION

Leveraging data mining techniques for detecting malicious sites offers a robust and versatile approach to cybersecurity. By analyzing vast datasets containing website characteristics, user behaviors, and history of instances of malicious activities, data mining algorithms can effectively identify patterns and indicators of potential threats. The utilization of classification algorithms, such as decision trees, support vector machines, or neural networks, enables the automated detection and classification of malicious sites with high accuracy.

Furthermore, data mining facilitates proactive defense against evolving threats by continuously learning from new data and updating classification criteria accordingly. Its adaptive nature allows for the detection of emerging threats and the identification of previously unseen patterns of malicious behavior.

Additionally, the insights gleaned from data mining analysis can inform broader cybersecurity strategies and decision-making processes within organizations. By identifying common attack patterns, trends, and vulnerabilities, organizations can strengthen their security posture, implement targeted countermeasures, and allocate resources more effectively to mitigate risks.

Moreover, the scalability of data mining enables organizations to analyze large volumes of data efficiently, even in the face of increasing data volumes and complexity this scalability ensures that organizations can effectively scale their cybersecurity efforts to meet the growing challenges posed by cyber threats.

Furthermore, the integration of data mining techniques into cybersecurity operations fosters a proactive and holistic approach to threat detection and operations fosters a proactive and holistic approach to threat detection and mitigation. By correlating data from multiple sources, including network traffic logs, endpoint telemetry, threat intelligence feeds, and user behavior analytics, organizations can gain a comprehensive understanding of their security posture,

Moreover, the continuous evolution of data mining techniques holds promise for future advancements in cybersecurity. As algorithms become more sophisticated and data sets grow in size and complexity, the potential for uncovering novel insights and detecting emerging threats will only continue to expand. Innovations in artificial intelligence, deep learning, and anomaly detection algorithms are poised to further enhance the capabilities of data mining for malicious site

In conclusion, this project represents a data mining that offers a potent and versatile approach to detecting malicious sites, leveraging advanced analytics and machine learning to uncover hidden threats and vulnerabilities. By harnessing the power of data, organizations can enhance their cybersecurity defenses, protect sensitive information, and safeguard against financial losses and reputational damage. In today's rapidly evolving threat landscape, data mining remains a cornerstone of effective cybersecurity strategies, providing organizations with the tools they need to stay one step ahead of cyber adversaries.

### VI. REFERENCES

- [1]Naim, Or, Doron Cohen, and Irad Ben-Gal. "Malicious website identification using design attribute learning." *International Journal of Information Security* (2023):111.  
<https://link.springer.com/10.1007/s10207-023-00686-y>
- [2]Kulp, Philip H., and Nikki E. Robinson. "Graphing Website Relationships for Risk Prediction: Identifying Derived Threats to Users Based on Known Indicators." *Proceedings of the Future Technologies Conference (FTC) 2020, Volume 2*. Springer International Publishing, 2021.[http://link.springer.com/10.1007/978-3-030-63089-8\\_34](http://link.springer.com/10.1007/978-3-030-63089-8_34).
- [3]Xia, Wei, et al. "Old Wine in A New Bottle: A Homogeneous Fraud Sites Discovery Framework." *2021 7th International Conference on Computer and Communications (ICC)*. IEEE, 2021.<https://ieeexplore.ieee.org/document/9674211/>
- [4] Aalla, Harsha Vardhan Sai, Nikhil Reddy Dumpala, and M. Eliazer. "Malicious URL prediction using machine learning techniques." *Annals of the Romanian Society for Cell Biology* (2021): 2170-2176.<https://www.annalsofscb.ro/index.php/journal/article/view/4752>.

## Publication Certificates







