WEB PHISHING DETECTION USING MACHINE LEARNING

PROJECT REPORT

21AD1513- INNOVATION PRACTICES LAB

Submitted by

DHINESH K - 211422243066

DIVAHAR REDDY A - 211422243069

GOWTHAM P H - 211422243081

in partial fulfillment of the requirements for the award of degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123

ANNA UNIVERSITY: CHENNAI-600 025

October, 2023

BONAFIDE CERTIFICATE

Certified that this project report titled " **WEB PHISHING DETECTION USING MACHINE LEARNING**" is the bonafide work of **DHINESH K ,DIVAHAR REDDY A , GOWTHAM P H**, Register No.**211422243066, 211422243069, 211422243081** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

INTERNAL GUIDE
K.VIJAYAKUMAR M.E.,

Department of AI &DS

HEAD OF THE DEPARTMENT
Dr.S.MALATHI M.E., Ph.D
Professor and Head,
Department of AI & DS.

Certified that the candidate was examined in the Viva-Voce Examination held on ……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

I also take this opportunity to thank all the Faculty and Non-Teaching Staff Members of Department of Computer Science and Engineering for their constant support. Finally I thank each and every one who helped me to complete this project. At the outset we would like to express our gratitude to our beloved respected Chairman, **Dr.Jeppiaar M.A.,Ph.D,** Our beloved correspondent and Secretary **Mr.P.Chinnadurai M.A., M.Phil., Ph.D.,** and our esteemed director for their support.

We would like to express thanks to our Principal, **Dr. K. Mani M.E., Ph.D.,** for having extended his guidance and cooperation.

We would also like to thank our Head of the Department, **Dr.S.Malathi M,E.,Ph.D.,** of Artificial Intelligence and Data Science for her encouragement.

Personally we thank **K.VIJAYAKUMAR M.E.,** Department of Artificial Intelligence and Data Science for the persistent motivation and support for this project, who at all times was the mentor of germination of the project from a small idea.

We express our thanks to the project coordinators **DR. A.Joshi M.E., Ph.D.,** Professor & **Dr.S.Chakaravarthi M.E.,Ph.D.,** Professor in Department of Artificial Intelligence and Data Science for their Valuable suggestions from time to time at every stage of our project.

Finally, we would like to take this opportunity to thank our family members, friends, and well-wishers who have helped us for the successful completion of our project.

We also take the opportunity to thank all faculty and non-teaching staff members in our department for their timely guidance in completing our project.

DHINESH K **DIVAHAR REDDY A** **GOWTHAM PH**

ABSTRACT

With raising in-depth amalgamation of the Internet and social life, the Internet is looking differently at how people are learning and working, meanwhile opening us to growing serious security attacks. The ways to recognize various network threats, specifically attacks not seen before, is a primary issue that needs to be looked into immediately. The aim of phishing site URLs is to collect the private information like user's identity.

Passwords and online money related exchanges. Phishers use the sites which are visibly and semantically like those of authentic websites. Since the majority of the clients go online to get to the administrations given by the government and money related organizations, there has been a vital increment in phishing threats and attacks since some years.

As technology is growing, phishing methods have started to progress briskly and this should be avoided by making use of anti-phishing techniques to detect phishing. Machine learning is a authoritative tool that can be used to aim against phishing assaults. There are several methods or approaches to identify phishing websites.

Once the user enters the URL in the search bar provided, then our model will predict whether the URL is legitimate or a phishing attempt, and if it is a phishing URL, a warning message will be displayed to the user. This approach will help prevent users from falling victim to phishing attacks and safeguard their sensitive information.

CHAPTER 1
INTRODUCTION

1.1. OVERVIEW

Web Phishing is a form of fraud in which the attacker tries to learn sensitive information such as login credentials or account information by sending as a reputable entity or person in email or other communication channels. Typically, a victim receives a message that appears to have been sent by a known contact or organization. The message contains malicious software targeting the user's computer or has links to direct victims to malicious websites to trick them. into divulging personal and financial information, such as passwords, account IDs or credit card details. Phishing is popular among attackers, since it is easier to trick someone into clicking a malicious link which seems legitimate than trying to break through a computer's defense systems. The malicious links within the body of the message are designed to make it appear that they go to the spoofed organization using that organization's logos and other legitimate contents.

1.2. OBJECTIVE

Website Phishing costs internet users billions of dollars per year. Phishers steal personal information and financial account details such as usernames and passwords, leaving users vulnerable in the online space. The COVID-19 pandemic has boosted the use of technology in every sector, resulting in shifting of activities like organizing official meetings, attending classes, shopping, payments, etc. from physical to online space. This means more opportunities for phishers to carry out attacks impacting the victim financially, psychologically & professionally. In 2013, 450 thousand phishing attacks led to financial losses of more than 5.9 billion dollars. As per Checkpoint Research Security Report 2018, 77% of IT professionals feel their security teams are unprepared for today's cybersecurity challenge. The same report indicates that 64% of organizations have experienced a phishing attack in the past year. Detecting phishing websites is not easy because of the use of URL obfuscation to shorten the URL, link redirections and manipulating links in such a way that it looks trustable and the list goes on. This necessitated the need to switch from traditional programming methods to machine learning approaches. The purpose of this project is to develop an application that can predict if visiting a given website is safe or unsafe and let the final decision of opening the website to the user.

1.3 SCOPE OF THIS PROJECT

The scope of the project involves developing a system to detect and warn users about potential phishing URLs. This entails collecting a dataset of known phishing and legitimate URLs, extracting relevant features, and training a machine learning model. The system will integrate with user interfaces, providing real-time predictions and warning messages. Advanced features may include continuous model updates, collaboration with cybersecurity experts, and scalability considerations. The goal is to deliver a robust solution that enhances user security against evolving phishing threats while remaining user-friendly and adaptable to changing cybersecurity landsca

1.4 LITERATURE SURVEY

1.4.1 WEIFENG ZHANG (SCHOOL OF COMPUTER, NANJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS, CHINA), HUA LU (DEPARTMENT OF COMPUTER SCIENCE, AALBORG UNIVERSITY, DENMARK) 2013

1.4.1.1 MERITS:

- Effective and efficient phishing detection method.
- Protects web users from loss of sensitive private information.
- Identifies legitimate web pages similar to suspicious phishing pages.

1.4.1.2 DEMERITS:

- Users can be imprecise and inconsistent in identifying phishing attempts.
- There is a degree of uncertainty in the relation between user input and intent.

1.4.2 GAURAV VARSHNEY, MANOJ MISRA, PRADEEP K. ATREY, A SURVEY AND CLASSIFICATION OF WEB PHISHING DETECTION SCHEMES 2016

1.4.2.1MERITS:

- Provides an overview of various web phishing detection schemes.
- Raises awareness about the deceptive techniques used in phishing attacks.
- Aids in understanding the complexity of phishing as a cybercrime.

1.4.2.2DEMERITS:

- Despite research efforts, there is no complete and accurate solution for thwarting phishing attacks.
- Phishing attacks continue to pose a severe threat to Internet users' security and privacy.

1.4.3 Z. DOU, I. KHALIL, A. KHREISHAH, A. AL-FUQAHA AND M. GUIZANI, SYSTEMATION OF KNOWLEDGE: A SYSTEMATIC REVIEW OF SOFTWARE-BASED WEBPHISHING DETECTION IN IEEE COMMUNICATIONSSURVEYS & TUTORIALS, VOL. 19, NO. 4, PP. 2797-2819, FOURTH QUARTER 2017

1.5.3.1 MERITS:

- Enhances explicit task goals and constraints.

- Aids user comprehension of model uncertainty and confidence.
- Focuses on capturing user intent rather than input.

1.4.3.2 DEMERITS:

- Users may exhibit imprecision and inconsistency.
- Uncertainty often exists in the correlation between user input and intent.

1.4.4 YUXIANG GUAN,FUTAI ZOU, YAO YAO, WEI WANG, AND TING ZHU, WEB PHISHING DETECTION USING A DEEP LEARNING FRAMEWORK 2018

1.4.4.1 MERITS:

- Focuses on utilizing a deep learning framework for detecting phishing websites.
- Introduces original and interaction features for web phishing detection.
- Presents a detection model based on Deep Belief Networks (DBN).
- Achieves approximately 90% true positive rate and 0.6% false positive rate in tests using real IP flows from ISPs.

1.4.4.2 DEMERITS:

- May require substantial computational resources for implementing deep learning models.
- Deep learning models may require significant amounts of labeled data for training, which can be challenging to obtain in the context of web phishing detection.

1.4.5 MORUF A ADEBOWALE, M ALAMGIR HOSSAIN, INTELLIGENT WEB-PHISHING DETECTION AND PROTECTION SCHEME USING INTEGRATED FEATURES OF IMAGES, FRAMES AND TEXT 2018

1.4.5.1 MERITS:

- Effective and efficient phishing detection method.
- Protects web users from loss of sensitive private information.
- Identifies legitimate web pages similar to suspicious phishing pages.

1.4.5.2 DEMERITS:

- Users can be imprecise and inconsistent in identifying phishing attempts.
- There is a degree of uncertainty in the relation between user input and intent.

1.4.6 LIZHEN TANG AND QUSAY H. MAHMOUD (ONTARIO TECH UNIVERSITY, OSHAWA, ON L1G CANADA), A DEEP LEARNING-BASED FRAMEWORK FOR PHISHING WEBSITE DETECTION 2021

1.4.6.1MERITS:

- Effective and efficient phishing detection method.
- Protects web users from loss of sensitive private information.
- Identifies legitimate web pages similar to suspicious phishing pages.

1.4.6.2DEMERITS:

- Users can be imprecise and inconsistent in identifying phishing attempts.
- There is a degree of uncertainty in the relation between user input and intent.

1.4.7 M.A. WAHEED, BASWARAJ GADGAY, SHUBHANGI DC, VISHWANATH P (2022 IEEE NORTH KARNATAKA SUBSECTION FLAGSHIP INTERNATIONAL CONFERENCE (NK CON)), A MACHINE LEARNING APPROACH FOR DETECTING MALICIOUS URL USING DIFFERENT ALGORITHMS AND NLP TECHNIQUES 2022

1.4.7.1 MERITS:

- Effective and efficient phishing detection method.
- Protects web users from loss of sensitive private information.
- Identifies legitimate web pages similar to suspicious phishing pages.

1.4.7.2 DEMERITS:

- Users can be imprecise and inconsistent in identifying phishing attempts.
- There is a degree of uncertainty in the relation between user input and intent.

1.4.8 M. KATHIRAVAN, V. RAJASEKAR, SHAIK JAVED PARVEZ, V. SATHYA DURGA, M. MEENAKSHI, S. GOWSALYA DETECTING PHISHING WEBSITES USING MACHINE LEARNING ALGORITHM 2023

1.4.8.1 MERITS:

- Effective and efficient phishing detection method.
- Protects web users from loss of sensitive private information.
- Identifies legitimate web pages similar to suspicious phishing pages.

1.4.8.2 DEMERITS:

- Users can be imprecise and inconsistent in identifying phishing attempts.
- There is a degree of uncertainty in the relation between user input and intent.

CHAPTER 2
SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

Web Phishing is one among the most common forms of cybercrime. The Internet Crime Report (2020) by FBI reports it to be the most common attack performed by cybercriminals. Phishing is a subset of a bigger school of thought called “Social Engineering”, which is the manipulation of people into believing the attacker. Phishing is done normally by hosting fake websites, sending fake emails, and conducting fake surveys. Phishing normally results in losing access to the user’s accounts but sometimes, the attacker stays silent to get even more information.

2.1.1 DISADVANTAGE

- It cannot be early predict the phishing in modern sites.
- It take more time for processing
- Many Website features not included for the consideration

2.2 PROPOSED SYSTEM

1) Train a machine learning model that can predict if a website is safe or unsafe and give a quantitative score for the same

2) Build a webpage for the user to give input to the proposed model and use the same to display the result.

2.2.2 ADVANTAGE

- The accuracy of proposed is expected to improve by effectively integrating information from multiple sources for model training.
- Reduce the processing time.
- Fast in classification process, less consuming memory, high accuracy, Evolving with time, online working

CHAPTER 3
SYSTEM SPECIFICATION

3.1 SOFTWARE REQUIREMENT SPECIFICATION

The software requirements specification is produced at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by establishing a complete information description as functional representation of system behavior, an indication of performance requirements and design constraints, appropriate validation criteria.

3.2 SYSTEM SPECIFICATION

3.2.1 HARDWARE REQUIREMENTS:

Processer	-	15
Speed	-	3GHZ
RAM	-	8 GB (min)
Hard Disk	-	500 GB
Key Board	-	Standard Windows Keyboard
Mouse	-	Two or Three Button Mouse
Monitor	-	LCD

3.2.2 SOFTWARE REQUIREMENTS

. Programming language - Python

. Operating system - Windows 10

. IDE - Anaconda, iPython version 3**.**x

3.3 FUNCTIONAL REQUIREMENTS:

Functional requirements are product features that developers must implement to enable the users to achieve their goals. They define the basic system behavior under specific conditions.

For our work the Functional requirements as follows:

Functional Requirement No.	**Functional Requirement (Epic)**	**Sub Requirement (Sub tasks)**
FR-1	User Input	Using web scraping, the URL which needs to be checked is given as input by the user automatically.
FR-2	Feature Extraction	Appropriate Expressions of the URL are extracted using Regular Expressions logic using programming and fed as features
FR-3	Prediction	Models must use classification based ML algorithms like Decision Tree, Random Forest Classifier, SVMs etc.
FR-4	Verify the link provided by the user	User inputs the link to be verified
FR-5	Display the result	If the site link is a phishing site, user must be aware and read the precautions displayed If the site link is legit, exit the application
FR-6	Sharing and clearing the Queries	If any doubts, send query Read FAQs

3.4 NON-FUNCTIONAL REQUIREMENTS:

Nonfunctional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs. For our work the Non - Functional requirements are as follows:

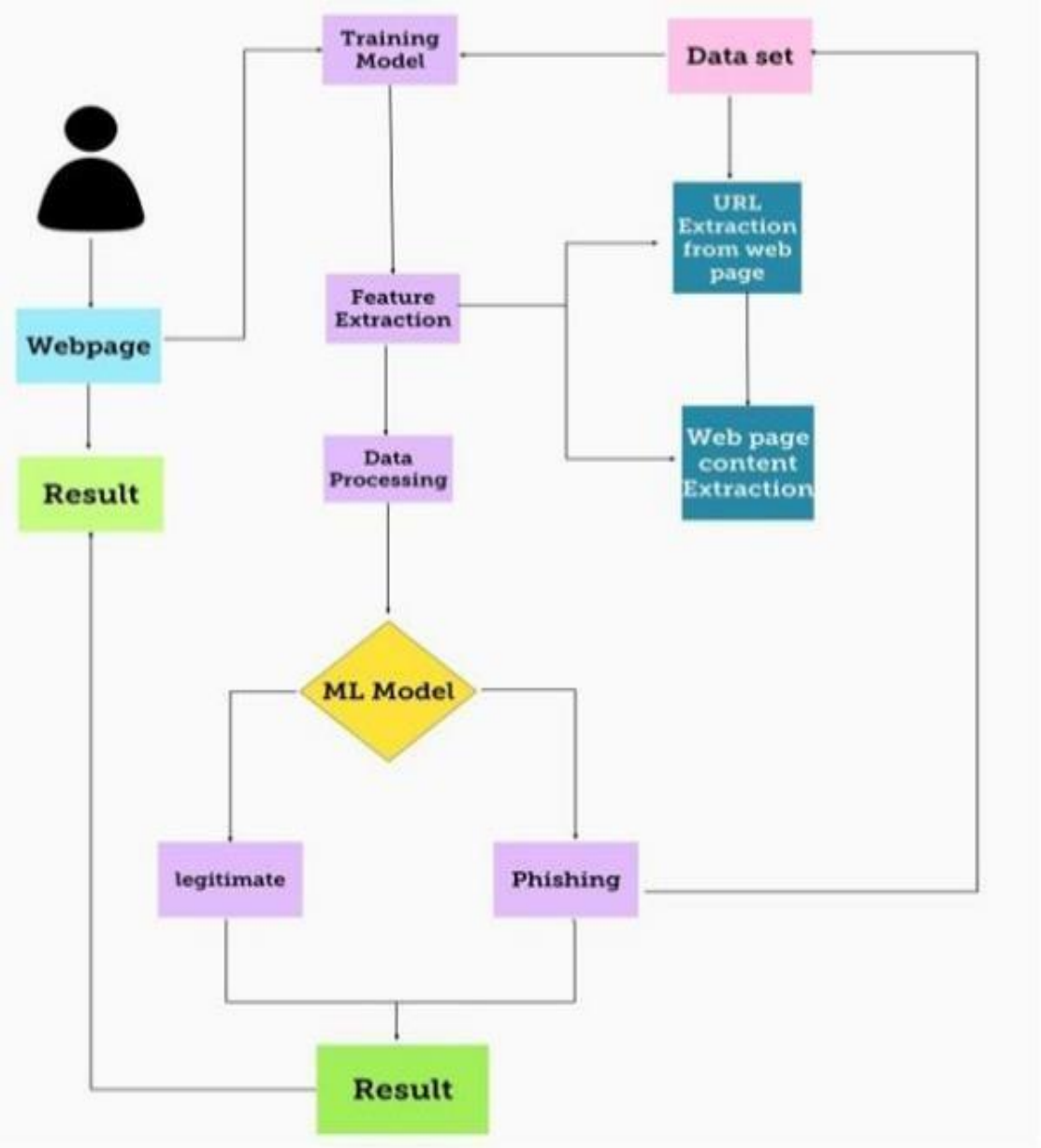
Non-functional Requirement No.	Non-Functional Requirement	Description
NFR-1	Usability	Engage the user about the process to ensure that the functionality can meet design and usability requirements. It relates to overall satisfaction of the user.
NFR-2	Security	Users need to be protected from malicious attacks when using the site.
NFR-3	Reliability	It focuses on preventing failures during the lifetime of the product or system, from commissioning to decommissioning.
NFR-4	Performance	It is the ability of the application to always run acceptably. In time-critical scenarios, even the smallest delay in processing data can be unacceptable.
NFR-5	Availability	Fault tolerance ensuring that the application can meet its availability targets to be resilient
NFR-6	Scalability	It is the ability for the application to scale to meet increasing demands; for example, at peak times or as the system becomes more widely adopted.

CHAPTER 4
SYSTEM DESIGN

4.PROJECT DESIGN

4.1 Data Flow Diagrams:

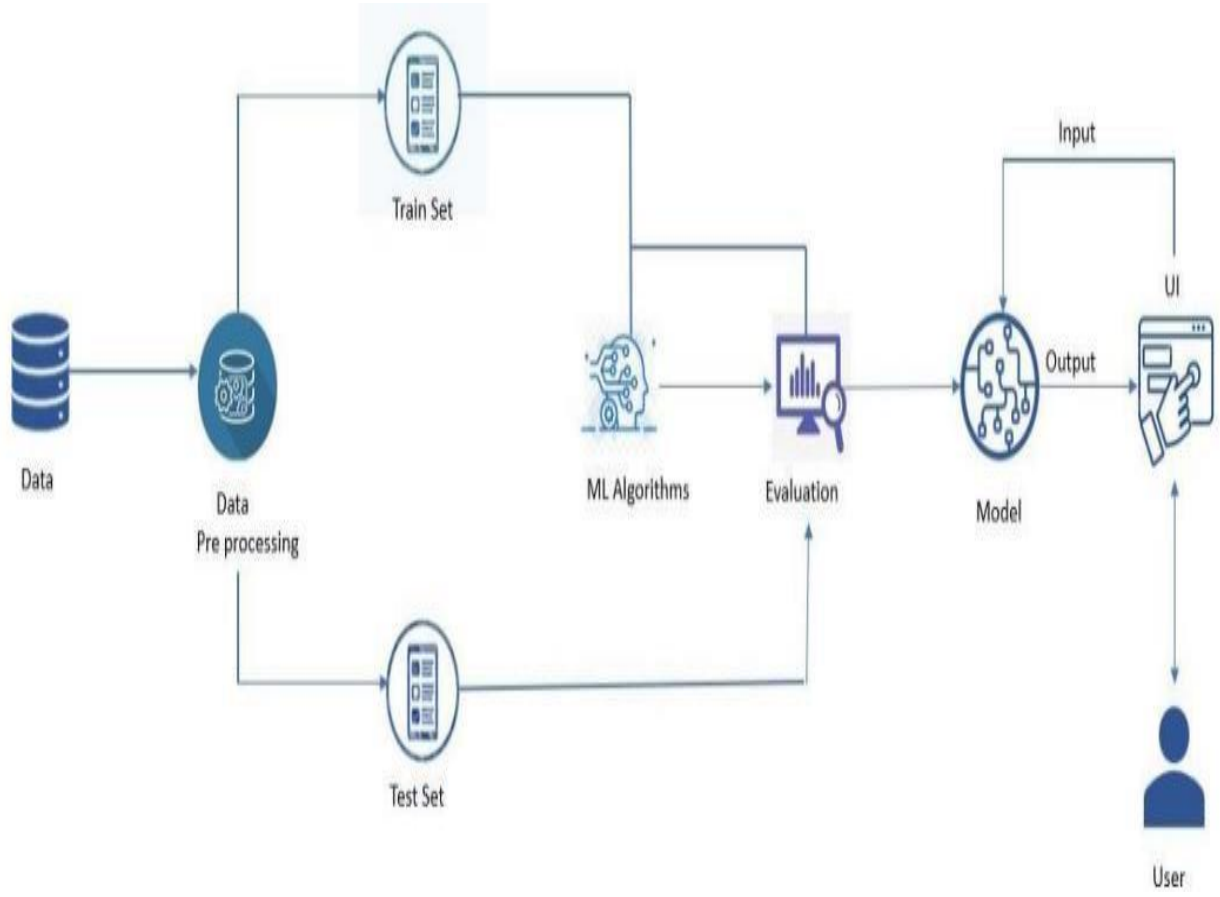
A Data Flow Diagram / DFD is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored. Following is the DFD proposed for the application.



Data Flow Diagram

4.2 Solution and Technical Architecture:

Solution architecture is the process of developing solutions based on predefined processes, guidelines and best practices with the objective that the developed solution fits within the enterprise architecture in terms of information architecture, system portfolios, integration requirements and many more. Following is the proposed technical architecture along with the underlying components and technologies listed out.



TECHNICAL ARCHITECTURE

CHAPTER 5
MODULES

5.MODULES

Project Modules:

Entire project is divided into 3 modules as follows:

1. Research Framework.

2. Training the model using following Machine Learning algorithms

3. Final Prediction Model integrated with frontend

5.1 RESEARCH FRAMEWORK

The steps of this research in which some selected publications were read to determine the research gap and, as a result, the research challenge was defined. Feature selection, classification and phishing website detection were all given significant consideration. It's worth noting that most phishing detection researchers rely on datasets they've created. However, because the datasets utilized were not available online for those who use and check their results, it is difficult to assess and compare the performance of a model with other models. As a result, such results cannot be generalized. For the preparation of this dissertation, I used Python as the primary language. Python is a language that is heavily focused on machine learning. It includes several machine learning libraries that may be utilized straight from an import. Python is commonly used by developers all around the world to deal with machine learning because of its extensive library of machine learning libraries. Python has a strong community, and as a result, new features are added with each release.

Data Collection

The phishing URLs were gathered using the open source tool Phish Tank. This site provides a set of phishing URLs in a variety of forms, including csv, json, and others, which are updated hourly. This dataset is used to train machine learning models with 5000 random phishing URLs.

Data Cleaning

Fill in missing numbers, smooth out creaking data, detect and delete outliers, and repair anomalies to clean up the data.

Data Pre-processing

Data preprocessing is a cleaning operation that converts unstructured raw data into a neat, well-structured dataset that may be used for further research. Data preprocessing is a cleaning operation that transforms unstructured raw data into well-structured and neat dataset which can be used for further research. The data is split into 8000 training samples and 2000 testing samples, before the ML model is trained. It is evident from the dataset that this is a supervised machine learning problem. Classification and regression are the two main types of supervised machine learning issues. Because the input URL is classed as legitimate or phishing, this data set has a classification problem. The following supervised machine learning models were examined for this project's dataset training: Decision Tree, Random Forest, Logistic Regression, Gradient boosting Classifier, Naive Bayes Classifier and Support Vector Machines.

5.2 ADDRESS BASED CHECKING

Below are the categories been extracted from address based

1. Domain of the URL: Where domain which is present in the URL been extracted

2. IP Address in the URL: The presence of an IP address in the URL is checked. Instead of a domain name, URLs may contain an IP address. If an IP address is used instead of a domain name in a URL, we can be certain that the URL is being used to collect sensitive information.

3. "@" Symbol in URL: The presence of the'@' symbol in the URL is checked. When the “@” symbol is used in a URL, the browser ignores anything before the “@” symbol, and the genuine address is commonly found after the “@” symbol.

4. Length of URL: Calculates the URL's length. Phishers can disguise the suspicious element of a URL in the address bar by using a lengthy URL. If the length of the URL is larger than or equal to 54 characters, the URL is classed as phishing in this project.

5. Depth of URL: Calculates the URL's depth. Based on the'/', this feature determines the number of subpages in the given address.

6. Redirection "//" in URL: The existence of"//" in the URL is checked. The presence of the character"//" in the URL route indicates that the user will be redirected to another website. The position of the"//" in the URL is calculated. We discovered that if the URL begins with “HTTP,” the “//” should be placed in the sixth position. If the URL uses “HTTPS,” however, the “//” should occur in the seventh place.

7. Http/Https in Domain name: The existence of "http/https" in the domain part of the URL is checked. To deceive users, phishers may append the “HTTPS” token to the domain section of a URL.

8. Using URL Shortening Services: URL shortening is a means of reducing the length of a URL while still directing to the desired webpage on the "World Wide Web." This is performed by using a “HTTP Redirect” on a short domain name that points to a webpage with a long URL.

9. Prefix or Suffix "-" in Domain: Checking for the presence of a '-' in the URL's domain part. In genuine URLs, the dash symbol is rarely used. Phishers frequently append prefixes or suffixes to domain names, separated by (-), to give the impression that they are dealing with a legitimate website.

5.3 DOMAIN BASED CHECKING

This category contains a lot of features that can be extracted. This category contains a lot of features that can be extracted. The following were considered for this project out of all of them.

1. DNS Record: In the case of phishing websites, the WHOIS database either does not recognize the stated identity or there are no records for the host name.

2. Web Traffic: This function determines the number of visitors and the number of pages they visit to determine the popularity of the website. In the worst-case circumstances, legitimate websites placed among the top100,000, according to our data. Furthermore, it is categorized as "Phishing" if the domain has no traffic or is not recognized by the Alexa database.

3. Age of Domain: This information can be retrieved from the WHOIS database. Most phishing websites are only active for a short time. For this project, the minimum age of a legal domain is deemed to be 12 months. Age is simply the difference between the time of creation and the time of expiry.

4. End Period of Domain: This information can be gleaned from the WHOIS database. The remaining domain time is calculated for this feature by determining the difference between the expiry time and the current time. For this project, the valid domain's end time is regarded to be 6 months or fewer.

5.4 HTML AND JAVASCRIPT BASED CHECKING

Many elements that fall within this group can be extracted. The following were considered for this project out of all of them.

1. IFrame Redirection: IFrame is an HTML tag that allows you to insert another webpage into the one you're now viewing. The “iframe” tag can be used by phishers to make the frame invisible, i.e., without frame borders. Phishers employ the “frame border” attribute in this case, which causes the browser to create a visual boundary.

2. Status Bar Customization: Phishers may utilize JavaScript to trick visitors into seeing a false URL in the status bar. To get this feature, we'll need to delve into the webpage source code, specifically the "on Mouseover" event, and see if it alters the status bar.

3. Disabling Right Click: Phishers disable the right-click function with JavaScript, preventing users from viewing and saving the webpage source code. This functionality is handled in the same way as "Hiding the Link with on Mouseover. “Nonetheless, we’ll look for the “event”event. button==2” in the webpage source code and see if the right click is disabled for this functionality.

4. Website Forwarding: The number of times a website has been redirected is a narrow line that separates phishing websites from authentic ones. We discovered that authentic websites were only routed once in our sample. Phishing websites with this functionality, on the other hand, have been redirected at least four times.

5. Implementation: We'll examine the implementation component of our artefact in this area of the report, with a focus on the description of the developed solution. This is a task that requires supervised machine learning.

5.5 DATASET

We collected the datasets from the open-source platform called Phishing tank. The dataset that was collected was in csv format. There are 18 columns in the dataset, and we transformed the dataset by applying data pre-processing technique. To see the features in the data we used few of the data frame methods for familiarizing. For visualization, and to see how the data is distributed and how features are related to one another, a few plots and graphs are given. The Domain column has no bearing on the training of a machine learning model. We now have 16 features and a target column. The recovered features of the legitimate and phishing URL datasets are simply concatenated in the feature extraction file, with no shuffling. We need to shuffle the data to balance out the distribution while breaking it into training and testing sets. This also eliminates the possibility of over fitting during model training.

5.6 LIBRARIES USED

Pandas: It's a Python-based machine learning library. Pandas is a free and open-source programming language. Pandas is a programming language that is commonly used for dataset loading and data analytics. Pandas is used for machine learning in a variety of domains, including economics, finance, and others. It is extremely user-friendly and can display datasets in a tabular style for easier comprehension.

Sklearn: Sklearn is one of the most essential Python libraries for machine learning. Sklearn includes several tools for statistical classification, modelling, regression, dimensionality reduction and clustering.

NumPy: NumPy is a Python-based machine learning package. In Python, NumPy is used to deal with arrays. NumPy is used for all calculations using 1-d or 2-d arrays. NumPy also has routines for working with linear algebra and the Fourier transform.

Matplotlib: Matplotlib is a library for data visualization. It's a Python open-source module for plotting graphs from model results. These diagrams can aid in comprehending the circumstance of the outcomes. For easier comprehension, several components of the results can be graphically formatted.

5.7 MACHINE LEARNING MODELS

- GRADIENT BOOSTING CLASSIFIER
- RANDOM FOREST CLASSIFIER
- NAIVE BAYES CLASSIFIER
- SUPPORT VECTOR MACHINE

CHAPTER 6
SYSTEM IMPLEMENTATION

6.1 IMPLEMENTATION PROCEDURE:

The implementation procedure for the proposed solution involves gathering a dataset of phishing and legitimate URLs, extracting relevant features, training a machine learning model, and evaluating its performance. This model is then integrated into a user interface where users can input URLs. When a URL is entered, the model analyzes it and, if it detects a phishing attempt with high confidence, displays a warning message to the user. The system undergoes thorough testing before deployment, and continuous monitoring and updates ensure its effectiveness in detecting evolving phishing techniques.

6.2 MACHINE LEARNING MODELS:

GRADIENT BOOSTING CLASSIFIER:

Gradient Boosting is a powerful boosting algorithm that combines several weak learners into strong learners, in which each new model is trained to minimize the loss function such as mean squared error or cross-entropy of the previous model using gradient descent. In each iteration, the algorithm computes the gradient of the loss function with respect to the predictions of the current ensemble and then trains a new weak model to minimize this gradient. The predictions of the new model are then added to the ensemble, and the process is repeated until a stopping criterion is met. In contrast to AdaBoost, the weights of the training instances are not tweaked, instead, each predictor is trained using the residual errors of the predecessor as labels. There is a technique called the Gradient Boosted Trees whose base learner is CART (Classification and Regression Trees).

RANDOM FOREST CLASSIFIER:

Random forest is an ensemble supervised machine learning algorithm made up of decision trees. It is used for classification and for regression as well. In Random Forest, the dataset is divided into two parts (training and testing). Based on multiple parameters, the decision is taken and the target data is predicted or classified accordingly.
Random Forest is a collection of multiple decision trees and the final result is based on the aggregated result of all the decision trees.

LOGISTIC REGRESSION:

Logistic regression is a supervised machine learning algorithm mainly used for binary classification where we use a logistic function, also known as a sigmoid function that takes input as independent variables and produces a probability value between 0 and 1. For example, we have two classes Class 0 and Class 1 if the value of the logistic function for an input is greater than 0.5 (threshold value) then it belongs to Class 1 it belongs to Class 0. It’s referred to as regression because it is the

extension of linear regression but is mainly used for classification problems. The difference between linear regression and logistic regression is that linear regression output is the continuous value that can be anything while logistic regression predicts the probability that an instance belongs to a given class or not.

NAIVE BAYES CLASSIFIER:

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. To start with, let us consider a dataset.

One of the most simple and effective classification algorithms, the Naive Bayes classifier aids in the rapid development of machine learning models with rapid prediction capabilities.

SUPPORT VECTOR MACHINE:

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well it's best suited for classification. The main objective of the SVM algorithm is to find the optimal hyperplane in an N-dimensional space that can separate the data points in different classes in the feature space. The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

	ML Model	Accuracy	f1_score	Recall	Precision
0	Logistic Regression	0.934	0.941	0.943	0.927
1	Support Vector Machine	0.964	0.968	0.980	0.965
2	Naive Bayes Classifier	0.605	0.454	0.292	0.997
3	Gradient Boosting Classifier	0.974	0.977	0.994	0.986

6.3 CREATING FLASK APP

1.Create a flask app. Command Line: from flask import Flask, ,render_template, Response, jsonify.

2.Adding required functions to render the templates.

3.Imported the prediction program to flask app import opencv.

4.Imported train list to the flask app. import trainlist Adding required functions to render the templates.

5.Run the app.py (Flask App).

CHAPTER 7
TESTING

7.1 UNIT TESTING

Unit testing involves testing individual components or units of code in isolation to ensure they function correctly. It focuses on verifying that each unit of code behaves as expected, typically through automated tests. Unit tests are written to cover various scenarios and edge cases, checking inputs, outputs, and internal logic. By isolating each unit of code, developers can quickly identify and fix bugs, improve code quality, and ensure that changes to one part of the codebase do not inadvertently break other parts. Overall, unit testing helps maintain code reliability, facilitates code refactoring, and contributes to the overall stability of the software system.

7.2 INTEGRATION TESTING

Integration testing involves testing how different components or modules of a software system interact with each other. It focuses on ensuring that these components integrate seamlessly and function correctly as a whole. Integration tests verify that data flows correctly between components, that interfaces are properly implemented, and that dependencies are managed correctly. By testing the interactions between modules, integration testing helps uncover issues such as communication errors, data inconsistencies, and integration bugs. It ensures that the system behaves as expected when all its components are combined, ultimately validating its overall functionality and reliability.

7.3 VALIDATION TESTING

Validation testing, also known as acceptance testing, assesses whether a software system meets the requirements and expectations of the end-users and stakeholders. It involves evaluating the software against the user's needs, preferences, and intended use cases. Validation testing verifies that the software satisfies the specified business requirements, functionality, usability, and performance criteria. This type of testing typically occurs towards the end of the development lifecycle, focusing on the overall user experience and ensuring that the software delivers value to its users.

Validation testing aims to confirm that the software fulfills its intended purpose and is ready for deployment in the production environment.

7.4 SYSTEM TESTING

System testing is a comprehensive evaluation of the entire software system as a whole. It verifies that all components, modules, and interactions work together correctly to meet the specified requirements and objectives. System testing assesses the system's functionality, performance, reliability, and compatibility with its intended environment. It includes a wide range of tests such as functional testing, performance testing, security testing, and usability testing. System testing aims to uncover defects, errors, and inconsistencies in the software system before it is deployed to production. By testing the system in its entirety, system testing ensures that the software meets quality standards and is ready for release to end-users.

7.5 RECOVERY TESTING:

Recovery testing evaluates a software system's ability to recover from failures and disruptions. It involves deliberately causing failures, such as system crashes, network outages, or hardware malfunctions, and then observing how the system responds and recovers. The goal of recovery testing is to assess the system's resilience and determine whether it can recover gracefully without data loss or corruption. This type of testing helps identify weaknesses in the system's recovery mechanisms and allows developers to implement appropriate measures to improve fault tolerance and reliability. Ultimately, recovery testing ensures that the software can withstand unexpected events and maintain functionality under adverse conditions.

7.6 SECURITY TESTING:

Security testing involves evaluating a system's defenses against potential threats and vulnerabilities to ensure that sensitive data and resources are protected from unauthorized access, manipulation, or destruction. This process includes identifying weaknesses in software, networks, or infrastructure through techniques such as penetration testing, vulnerability scanning, and code review. The goal is to uncover

security flaws before they can be exploited by attackers and to implement countermeasures to mitigate risks effectively. Security testing is crucial for safeguarding confidential information, maintaining regulatory compliance, and preserving trust among users and stakeholders.

7.7 PERFORMANCE TESTING:

Performance testing assesses the speed, responsiveness, and scalability of a system under various conditions to ensure optimal performance and user experience. It involves measuring factors such as response time, throughput, and resource utilization to identify bottlenecks, weaknesses, or limitations that could affect the system's performance. By simulating different levels of user load or stress, performance testing helps organizations understand how their systems will perform in real-world scenarios and enables them to optimize performance, enhance reliability, and meet user expectations.

7.8 EXPERIMENTAL OUTCOMES:

Experimental outcomes refer to the results obtained from conducting experiments or tests within a scientific or research context. These outcomes provide empirical evidence that can either support or refute hypotheses, theories, or research questions. They often include quantitative data, qualitative observations, or statistical analyses that reveal patterns, trends, or relationships between variables. Interpretation of experimental outcomes is essential for drawing conclusions, making recommendations, or informing future research directions. Additionally, clear and concise presentation of experimental outcomes is crucial for communicating findings to peers, stakeholders, or the broader scientific community.

CHAPTER 8
CODE IMPLEMENTATION

8.CODE IMPLEMENTATION:

The Internet has become an indispensable part of our life, However, It also has provided opportunities to anonymously perform malicious activities like Phishing. Phishers try to deceive their victims by social engineering or creating mockup websites to steal information such as account ID, username, password from individuals and organizations. Although many methods have been proposed to detect phishing websites, Phishers have evolved their methods to escape from these detection methods. One of the most successful methods for detecting these malicious activities is Machine Learning. This is because most Phishing attacks have some common characteristics which can be identified by machine learning methods.

The steps demonstrated in this notebook are:

1. Loading the data
2. Familiarizing with data & EDA
3. Visualizing the data
4. Splitting the data
5. Training the data
6. Comparison of Model
7. Result

#importing required libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn import metrics
import warnings
warnings.filterwarnings('ignore')
```

1. Loading Data:

A collection of website URLs for 11000+ websites. Each sample has 30 website parameters and a class label identifying it as a phishing website or not (1 or -1).

The overview of this dataset is, it has 11054 samples with 32 features. Download the dataset from the link provided.

#Loading data into dataframe

```
data = pd.read_csv("phishing.csv")
data.head()
```

2. Familiarizing with Data & EDA:

In this step, few dataframe methods are used to look into the data and its features.

#Shape of dataframe
data**.**shape
(11054, 32)
#Listing the features of the dataset
data**.**columns
#Information about the dataset
data**.**info()
 # nunique value in columns
data**.**nunique()
#droping index column
data = data**.**drop(['Index'],axis = 1)

#description of dataset
data**.**describe()**.**T

data_set.append(9 OBSERVATIONS:

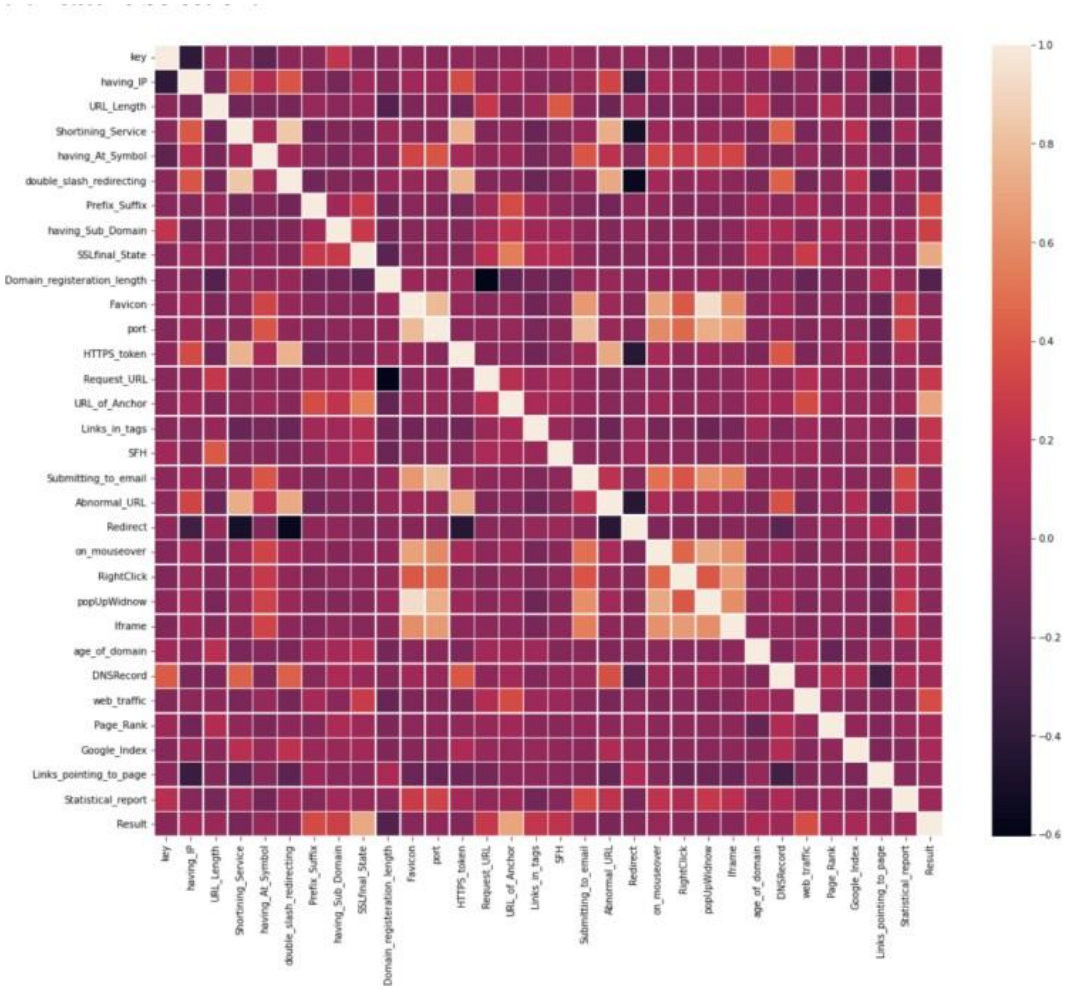
1. There are 11054 instances and 31 fearures in dataset.
2. Out of which 30 are independent features where as 1 is dependent feature.
3. Each feature is in int datatype, so there is no need to use LabelEncoder.
4. There is no outlier present in dataset.
5. There is no missing value in dataset.

3. Visualizing the data:

Few plots and graphs are displayed to find how the data is distributed and the how features are related to each other.

#Correlation heatmap

plt**.**figure(figsize=(15,15))
sns**.**heatmap(data**.**corr(), annot=**True**)
plt**.**show()



FEATURE SELECTION

4.Splitting the Data:

The data is split into train & test sets, 80-20 split.

 # Splitting the dataset into dependant and independant fetature

```
X = data.drop(["class"],axis =1)
y = data["class"]
```

 # Splitting the dataset into train and test sets: 80-20 split

```
from sklearn.model_selection import train_test_split

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

5. Model Building & Training:

Supervised machine learning is one of the most commonly used and successful types of machine learning. Supervised learning is used whenever we want to predict a certain outcome/label from a given set of features, and we have examples of features-label pairs. We build a machine learning model from these features-label pairs, which comprise our training set. Our goal is to make accurate predictions for new, never-before-seen data.

There are two major types of supervised machine learning problems, called classification and regression. Our data set comes under regression problem, as the prediction of suicide rate is a continuous number, or a floating-point number in programming terms. The supervised machine learning models (regression) considered to train the dataset in this notebook are:

1. Logistic Regression
2. Support Vector Classifier
3. Naive Bayes
4. Gradient Boosting

The metrics considered to evaluate the model performance are Accuracy & F1 score.

```
# Creating holders to store the model performance results
ML_Model = []
accuracy = []
f1_score = []
recall = []
precision = []

#function to call for storing the results
def storeResults(model, a,b,c,d):
ML_Model.append(model)
accuracy.append(round(a, 3))
  f1_score.append(round(b, 3))
  recall.append(round(c, 3))
  precision.append(round(d, 3))
```

5.1. Logistic Regression

Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

```
# Linear regression model
from sklearn.linear_model import LogisticRegression
#from sklearn.pipeline import Pipeline
# instantiate the model
log = LogisticRegression()
# fit the model
log.fit(X_train,y_train)
LogisticRegression()
#predicting the target value from the model for the samples

y_train_log = log.predict(X_train)
y_test_log = log.predict(X_test)
```

5.2. Support Vector Machine Classifier

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.

```
# Support Vector Classifier model
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
# defining parameter range
param_grid = {'gamma': [0.1],'kernel': ['rbf','linear']}
svc = GridSearchCV(SVC(), param_grid)
# fitting the model for grid search
svc.fit(X_train, y_train)
GridSearchCV(estimator=SVC(),
             param_grid={'gamma': [0.1], 'kernel': ['rbf', 'linear']})
#predicting the target value from the model for the samples
y_train_svc = svc.predict(X_train)
y_test_svc = svc.predict(X_test)
```

5.3. Naive Bayes : Classifier

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.It is mainly used in text, image classification that includes a high-dimensional training dataset. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.

```
# Naive Bayes Classifier Model
from sklearn.naive_bayes import GaussianNB
from sklearn.pipeline import Pipeline
# instantiate the model
nb=  GaussianNB()
# fit the model
nb.fit(X_train,y_train)
GaussianNB()
#predicting the target value from the model for the samples
y_train_nb = nb.predict(X_train)
y_test_nb = nb.predict(X_test)
```

5.4Gradient Boosting Classifier

Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model. Decision trees are usually used when doing gradient boosting. Boosting algorithms play a crucial role in dealing with bias variance trade-off. Unlike bagging algorithms, which only controls for high variance in a model, boosting controls both the aspects (bias & variance), and is considered to be more effective.

```
# Gradient Boosting Classifier Model
from sklearn.ensemble import GradientBoostingClassifier

# instantiate the model
gbc = GradientBoostingClassifier(max_depth=4,learning_rate=0.7)

# fit the model
gbc.fit(X_train,y_train)
GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
#predicting the target value from the model for the samples
y_train_gbc = gbc.predict(X_train)
y_test_gbc = gbc.predict(X_test)
```

6. Comparision of Models

To compare the models performance, a dataframe is created. The columns of this dataframe are the lists created to store the results of the model.

```
#creating dataframe
result = pd.DataFrame({ 'ML Model' : ML_Model,
                        'Accuracy' : accuracy,
                        'f1_score' : f1_score,
                        'Recall'   : recall,
                        'Precision': precision,
                       })
# dispalying total result
result
```

	ML Model	Accuracy	f1_score	Recall	Precision
0	Logistic Regression	0.934	0.941	0.943	0.927
1	Support Vector Machine	0.964	0.968	0.980	0.965
2	Naive Bayes Classifier	0.605	0.454	0.292	0.997
3	Gradient Boosting Classifier	0.974	0.977	0.994	0.986

CREATING FLASK APP

1.Create a flask app. Command Line: from flask import Flask, ,render_template, Response, jsonify.

2.Adding required functions to render the templates.

3.Imported the prediction program to flask app import opencv.

4.Imported train list to the flask app. import trainlist Adding required functions to render the templates.

5.Run the app.py (Flask App).

app.py

```
from flask import Flask, request, render_template
import numpy as np
import pandas as pd
from sklearn import metrics
import warnings
import pickle
warnings.filterwarnings('ignore')
from feature import FeatureExtraction

file = open("pickle/model.pkl","rb")
gbc = pickle.load(file)
file.close()


app = Flask(__name__)

@app.route("/", methods=["GET", "POST"])
def index():
    if request.method == "POST":

        url = request.form["url"]
        obj = FeatureExtraction(url)
        x = np.array(obj.getFeaturesList()).reshape(1,30)

        y_pred =gbc.predict(x)[0]
        #1 is safe
        #-1 is unsafe
        y_pro_phishing = gbc.predict_proba(x)[0,0]
        y_pro_non_phishing = gbc.predict_proba(x)[0,1]
        # if(y_pred ==1 ):
        pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)
        return render_template('index.html',xx =round(y_pro_non_phishing,2),url=url )
    return render_template("index.html", xx =-1)


if __name__ == "__main__":
    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <meta name="description" content="This website is develop for identify the safety
of url.">
   <meta name="keywords" content="phishing url,phishing,cyber security,machine
learning,classifier,python">
   <meta name="author" content="VAIBHAV BICHAVE">

   <!-- BootStrap -->
   <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
      integrity="sha384-
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj
7Sk" crossorigin="anonymous">

   <link href="static/styles.css" rel="stylesheet">
   <title>URL detection</title>

</head>

<body>

<div class=" container">
   <div class="row">
      <div class="form col-md" id="form1">
         <h2>PHISHING URL DETECTION</h2>

         <br>
         <form action="/" method ="post">
            <input type="text" class="form__input" name ='url' id="url"
placeholder="Enter URL" required="" />
            <label for="url" class="form__label">URL</label>
            <button class="button" role="button" >Check here</button>
         </form>

   </div>
```

```
    <div class="col-md" id="form2">

        <br>
        <h6 class = "right "><a href= {{ url }} target="_blank">{{ url }}</a></h6>

        <br>
        <h3 id="prediction"></h3>
        <button class="button2" id="button2" role="button"
onclick="window.open('{{url}}')" target="_blank" >Still want to Continue</button>
        <button class="button1" id="button1" role="button"
onclick="window.open('{{url}}')" target="_blank">Continue</button>
    </div>
</div>
<br>
<p>©2021 VAIBHAV BICHAVE</p>
</div>

    <!-- JavaScript -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
        integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRk
fj"
        crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
        integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo
"
        crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
        integrity="sha384-
OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JK
I"
        crossorigin="anonymous"></script>


    <script>

            let x = '{{xx}}';
            let num = x*100;
            if (0<=x && x<0.50){
                num = 100-num;
```

```
        }
        let txtx = num.toString();
        if(x<=1 && x>=0.50){
           var label = "Website is "+txtx +"% safe to use...";
           document.getElementById("prediction").innerHTML = label;
           document.getElementById("button1").style.display="block";
        }
        else if (0<=x && x<0.50){
           var label = "Website is "+txtx +"% unsafe to use..."
           document.getElementById("prediction").innerHTML = label ;
           document.getElementById("button2").style.display="block";
        }

   </script>

</body>

</html>
```

styles.css

```
*,
*::after,
*::before {
  margin: 0;
  padding: 0;
  box-sizing: inherit;
  font-size: 62,5%;
}

body {
  padding: 10% 5%;
  background: #0f2027;
  background: linear-gradient(to right,#2c5364, #203a43, #0f2027);
  justify-content: center;
  align-items: center;
  height: 100vh;
  color: #fff;
}

.form__label {
  font-family: 'Roboto', sans-serif;
```

```
  font-size: 1.2rem;
  margin-left: 2rem;
  margin-top: 0.7rem;
  display: block;
  transition: all 0.3s;
  transform: translateY(0rem);
}

.form__input {
  top: -24px;
  font-family: 'Roboto', sans-serif;
  color: #333;
  font-size: 1.2rem;
  padding: 1.5rem 2rem;
  border-radius: 0.2rem;
  background-color: rgb(255, 255, 255);
  border: none;
  width: 75%;
  display: block;
  border-bottom: 0.3rem solid transparent;
  transition: all 0.3s;
}

.form__input:placeholder-shown + .form__label {
  opacity: 0;
  visibility: hidden;
  -webkit-transform: translateY(+4rem);
  transform: translateY(+4rem);
}


.button {
  appearance: button;
  background-color: transparent;
  background-image: linear-gradient(to bottom, #fff, #f8eedb);
  border: 0 solid #e5e7eb;
  border-radius: .5rem;
  box-sizing: border-box;
  color: #482307;
  column-gap: 1rem;
  cursor: pointer;
  display: flex;
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe
```

```
UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color
Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";
  font-size: 100%;
  font-weight: 700;
  line-height: 24px;
  margin: 0;
  outline: 2px solid transparent;
  padding: 1rem 1.5rem;
  text-align: center;
  text-transform: none;
  transition: all .1s cubic-bezier(.4, 0, .2, 1);
  user-select: none;
  -webkit-user-select: none;
  touch-action: manipulation;
  box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);
}

.button:active {
  background-color: #f3f4f6;
  box-shadow: -1px 2px 5px rgba(81,41,10,0.15),0px 1px 1px rgba(81,41,10,0.15);
  transform: translateY(0.125rem);
}

.button:focus {
  box-shadow: rgba(72, 35, 7, .46) 0 0 0 4px, -6px 8px 10px rgba(81,41,10,0.1), 0px
2px 2px rgba(81,41,10,0.2);
}


.main-body{
  display: flex;
  flex-direction: row;
  width: 75%;
  justify-content:space-around;
}

.button1{
  appearance: button;
  background-color: transparent;
  background-image: linear-gradient(to bottom, rgb(160, 245, 174), #37ee65);
  border: 0 solid #e5e7eb;
  border-radius: .5rem;
  box-sizing: border-box;
```

```
  color: #482307;
  column-gap: 1rem;
  cursor: pointer;
  display: flex;
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe
UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color
Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";
  font-size: 100%;
  font-weight: 700;
  line-height: 24px;
  margin: 0;
  outline: 2px solid transparent;
  padding: 1rem 1.5rem;
  text-align: center;
  text-transform: none;
  transition: all .1s cubic-bezier(.4, 0, .2, 1);
  user-select: none;
  -webkit-user-select: none;
  touch-action: manipulation;
  box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);
  display: none;
}

.button2{
  appearance: button;
  background-color: transparent;
  background-image: linear-gradient(to bottom, rgb(252, 162, 162), #ee3737);
  border: 0 solid #e5e7eb;
  border-radius: .5rem;
  box-sizing: border-box;
  color: #482307;
  column-gap: 1rem;
  cursor: pointer;
  display: flex;
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe
UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color
Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";
  font-size: 100%;
  font-weight: 700;
  line-height: 24px;
  margin: 0;
  outline: 2px solid transparent;
  padding: 1rem 1.5rem;
```

```
  text-align: center;
  text-transform: none;
  transition: all .1s cubic-bezier(.4, 0, .2, 1);
  user-select: none;
  -webkit-user-select: none;
  touch-action: manipulation;
  box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);
  display: none;
}

.right {
  right: 0px;
  width: 300px;
}

@media (max-width: 576px) {
  .form {
    width: 100%;
  }
 }
.abc{
  width: 50%;
}
```

CHAPTER 9
RESULTS AND PERFORMANCE ANALYSIS

9.RESULTS

9.1. Results

A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False. More specifically, True Positives, False Positives, True negatives and False Negatives are used to predict the metrics of a classification report as shown below.

LOGISTIC REGRESSION:

	precision	recall	f1-score	support
-1	0.94	0.91	0.92	976
1	0.93	0.95	0.94	1235
accuracy			0.93	2211
macro avg	0.93	0.93	0.93	2211
weighted avg	0.93	0.93	0.93	2211

SUPPORT VECTOR MACHINE:

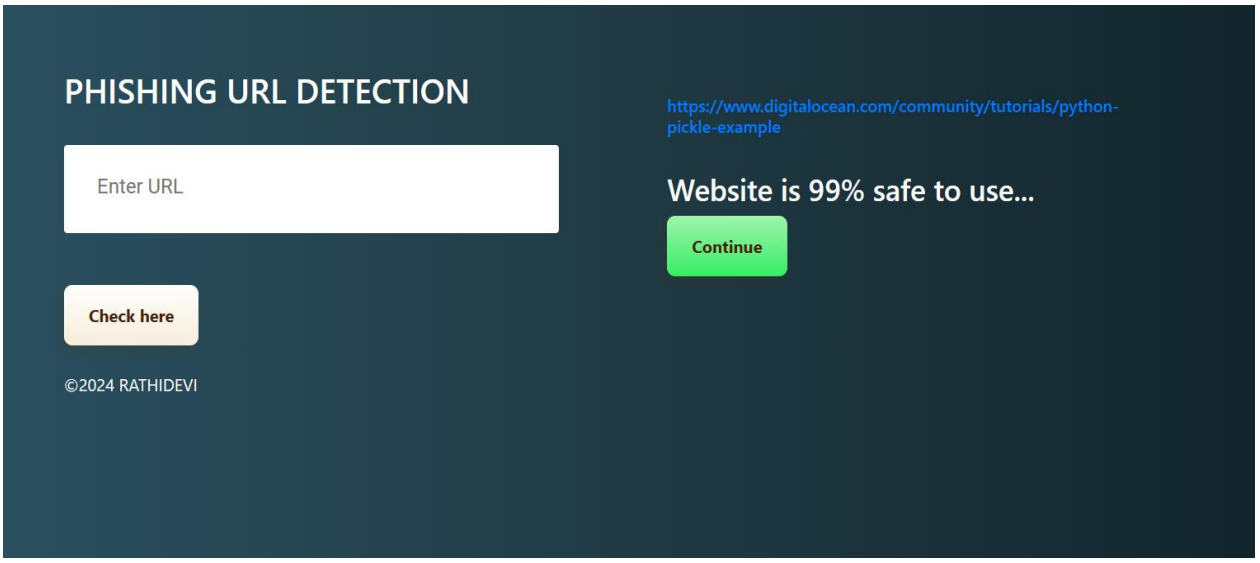
	precision	recall	f1-score	support
-1	0.97	0.94	0.96	976
1	0.96	0.98	0.97	1235
accuracy			0.96	2211
macro avg	0.97	0.96	0.96	2211
weighted avg	0.96	0.96	0.96	2211

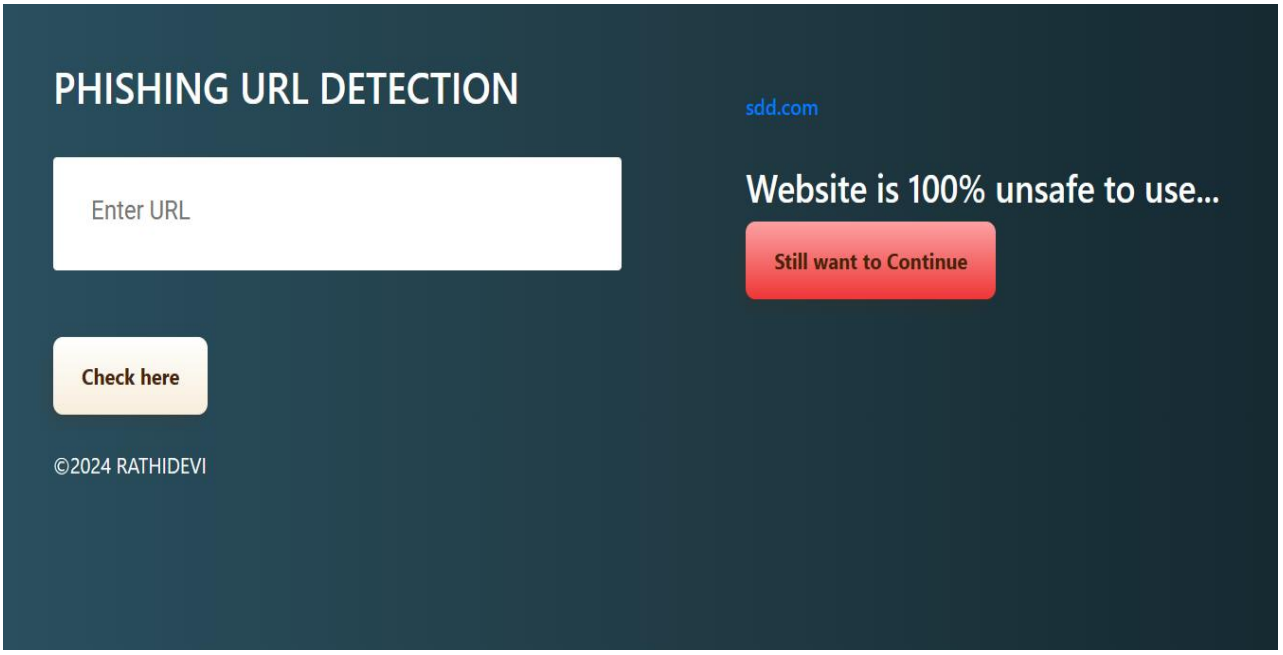
NAIVE BAYES CLASSIFIER

	precision	recall	f1-score	support
-1	0.97	0.94	0.96	976
1	0.96	0.98	0.97	1235
accuracy			0.96	2211
macro avg	0.97	0.96	0.96	2211
weighted avg	0.96	0.96	0.96	2211

GRADIENT BOOSTING CLASSIFIER

	precision	recall	f1-score	support
-1	0.99	0.96	0.97	976
1	0.97	0.99	0.98	1235
accuracy			0.97	2211
macro avg	0.98	0.97	0.97	2211
weighted avg	0.97	0.97	0.97	2211


PHISHING URL DETECTION
Enter URL
Check here
©2024 RATHIDEVI
https://www.digitalocean.com/community/tutorials/python-pickle-example
Website is 99% safe to use...
Continue


PHISHING URL DETECTION
Enter URL
Check here
©2024 RATHIDEVI
sdd.com
Website is 100% unsafe to use...
Still want to Continue

CHAPTER 10
CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION:

The most important way to protect the user from phishing attacks is education about malicious software and awareness. Internet users must be aware of all the security tips which are given by experts. Every user must be trained to blindly follow the links to the websites where they have to send their sensitive information. It is essential to check the URL before entering the websites. Given the dataset of Phishing websites, we have explored a variety of Machine learning architectures and Deep learning architectures to assess the website's trust ability and protection level. The correlation values for each feature was obtained and the top 12 features were used for training.

Models used:

1. Support Vector Machine Classifier
2. Logistic Regression
3. Naive Bayes Classifier
4. Gradient Boosting Classifier

Highest accuracy was obtained while using the Gradient Boosting Classifier. Thisproduced us with a mean accuracy of 97% which is a massive improvement and covers all the basic requirements to make an abstract model of an anti-phishing website. Integrate the trained machine learning model into the Flask web application to perform real-time classification of user-provided URLs. Once the user enters the URL in the search bar provided, then our model will predict whether the URL is safe or unsafe message will be displayed to the user. This approach will help prevent users from falling victim to phishing attacks and safeguard their sensitive information.

FUTURE SCOPE:

In the future, this anti-phishing web app can be improved by integrating advanced machine learning techniques like deep learning, updating the model with real-time threat intelligence, expanding to cover other communication channels like emails, and incorporating user feedback mechanisms for continuous enhancement.

REFERENCE:

[1] W.Zhang,H.Lu,B.Xu,H.Yang (2013) “Web phishing detection based on page spatial layout similarity”

[2] Gaurav Varshney, Manoj Misra, Pradeep K. Atrey (2016) “A Survey and Classification of Web phishing detection schemes

[3] Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha and M. Guizani(2017) “Systemation of Knowledge: A systematic review of software-based web phishing detection”

[4] Yuxiang Guan,Futai Zou, Yao Yao, Wei Wang, and Ting Zhu (2018) “Web Phishing Detection Using a Deep Learning Framework”

[5] M.A. Adebowale,K.T. Lwin,E. Sánchezb,M.A. Hossain(2018) “Intelligent Web-Phishing Detection and Protection Scheme using integrated Features of Images, Frames and Text”

[6] LIZHEN TANG AND QUSAY H. MAHMOUD, (Senior Member,
Department of Electrical, Computer, and Software Engineering, Ontario Tech University, Oshawa (2021) “A Deep Learning-Based Framework for Phishing Website Detection”

[7] M.A. Vahid; Bhasvaraj Gadegai; Shubhangit DC; Vishwanath P (2022) “Machine Learning Approach to Detect Malicious URLs Using Different Algorithms and NLP Techniques”

[8] M. Kathiravan; V. Rajasekar; Shaik Javed Parvez; V. Sathya Durga; M.Meenakshi; S. Gowsalya (2023) Detecting Phishing Websites using Machine Learning Algorithm