Social Engineering Defense

Submitted as a partial fulfillment of Bachelor of Technology in Computer Science & Engineering (Data Science)
Of
MCKV Institute of Engineering
(An Autonomous Institute under UGC Act, 1956;
Approved by AICTE,
Affiliated to Maulana Abul Kalam Azad University of Technology, West Bengal)

Project Report

Submitted by

Name of Students		Examination Roll No.
Priyabrata Shee		11600321070
Arpita Dey		11600321064
Sounak Pramanik		11600321073
Jhantu Das		11600321067

Under the supervision of

Mrs. Rachita Ghoshhajra
Asst. Professor, CSE

Department of Computer Science & Engineering,
MCKV Institute of Engineering
243, G.T. Road(N)
Liluah, Howrah – 711204

May 2024

Department of Computer Science & Engineering
MCKV Institute of Engineering
243, G. T. Road (N),
Liluah, Howrah-711204

CERTIFICATE OF RECOMMENDATION

I hereby recommend that the project report prepared under my supervision by students listed below

Sl. No.	**Name**	**Signature**
1	Priyabrata Shee	
2	Arpita Dey	
3	Sounak Pramanik	
4	Jhantu Das	

for the project entitled **"Social Engineering Defense**" be accepted in fulfillment of the requirements for the degree of **Bachelor of Technology in Computer Science & Engineering (Data Science).**

Mr. Avijit Bose
Assistant Professor & Head of the Department,
Computer Science & Engineering Dept.
MCKV Institute of Engineering

Mrs. Rachita Ghoshhajra
Asst. Professor
Computer Science & Engineering
MCKV Institute of Engineering

Department of Computer Science & Engineering
MCKV Institute of Engineering
243, G. T. Road (N),
Liluah, Howrah-711204

CERTIFICATE

This is to certify that the project entitled **"Social Engineering Defense"** and submitted by

Sl. No.	Name the Student	Exam Roll Number
1	Priyabrata Shee	11600321070
2	Arpita Dey	11600321064
3	Sounak Pramanik	11600321073
4	Jhantu Das	11600321067

has been carried out under the guidance of myself following the rules and regulations of the degree of **Bachelor of Technology in Computer Science & Engineering (Data Science)**, **MCKV Institute of Engineering.**

(Signature of the students)

1. _____
2. _____
3. _____
4. _____

_____

Rachita Ghoshhajra,
Asst. Professor,
Computer Science & Engineering Dept.

MCKV Institute of Engineering

(An Autonomous Institute under UGC Act, 1956
Approved by AICTE
Affiliated to Maulana Abul Kalam Azad University of Technology,
West Bengal)

CERTIFICATE OF APPROVAL

(B.Tech Degree in Computer Science & Engineering (Data Science))

This project report is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is to be understood that by this approval, the undersigned do not necessarily endorse or approve any statement made, opinion expressed, and conclusion drawn therein but approve the project report only for the purpose for which it has been submitted.

COMMITTEE ON FINAL EXAMINATION FOR EVALUATION OF PROJECT REPORT	1.	
	2.	
	3.	
	4.	

Acknowledgement

We take this opportunity to express our profound gratitude and deep regards to **Mr. Avijit Bose**, Assistant Professor and Head of the Department, Computer Science and Engineering, MCKV Institute of Engineering for supporting us to make our project worth it.

We would like to express our profound regards to our Project Guide **Mrs. Rachita Ghoshhajra**, Assistant Professor, Department of Computer Science and Engineering, MCKV Institute of Engineering for his exemplary guidance, monitoring and constant encouragement throughout the course of this Project. The blessing, help and guidance given by him from time to time shall has helped us in doing a lot of research and we came to know about so many new things and carry us a long way in the journey of this project.

We would also like to thank the Computer Science and Engineering Department of MCKV Institute of Engineering for their immense support, contributions and valuable instructions in conducting this project report successfully. Their guidance has helped us tremendously at all times in our research and writing of this project. Their immense knowledge, profound experience and professional expertise has enabled us to complete this project successfully. We thank all our other faculty members and technical assistants at MCKV Institute of Engineering for playing a significant role during the development of the project. Without their support and guidance, this project would not have been possible. It helped us in increasing our knowledge and skills.

We are obliged to our Project team members for the valuable cooperation, coordination and information provided by them in their respective fields. We are grateful for their cooperation during the period of our assignment.

Priyabrata Shee
Arpita Dey
Sounak Pramanik
Jhantu Das

Contents

1. Abstract

Social engineering has been an extremely serious security threat for several years, and the number of social engineering attacks that have been executed, the majority of which have been successful, has been steadily increasing in rapid succession. This increase can be attributed to numerous factors, such as a general increase in the accessibility and affordability of networking services and sites. Furthermore, the COVID-19 pandemic has also led to an increase in the number of attacks that have been executed and has also contributed to social engineering attacks becoming more successful than ever, due to the fear and anxiety that has been become a prevalent issue due to the pandemic. While social engineering is still a detrimental issue to cyber security infrastructure and corporations everywhere no one solution can be implemented, either through the use of hardware or software that can prevent social engineering attacks from occurring.

As the digital era matures, cyber security evolves and software vulnerabilities diminish, people however, as individuals, are more exposed today than ever before. Presently, one of the most practiced and effective penetration attacks are social rather than technical, so efficient in fact, that these exploits play a crucial role to support the greatest majority of cyber assaults. Social Engineering is the art of exploiting the human flaws to achieve a malicious objective. In the context of information security, practitioners breach defenses to access sensitive data preying particularly upon the human tendency towards trust. Cyber criminals induce their victims to break security protocol forfeiting confidential information propitious for a more targeted attack. Disastrously, in many cases, targets are manipulated to involuntarily infect and sabotage the system themselves. This report examines recurrent social engineering techniques used by attackers, as well as revealing a basic complementary technical methodology to conduct effective exploits.

This project presents a comprehensive web application designed to detect and prevent three major types of cyber threats: spam mail, phishing websites, and credit card fraud. Utilizing advanced machine learning algorithms, the application offers high accuracy in identifying malicious activities, thereby enhancing online security for users. The system is built using Flask, enabling seamless integration of machine learning models with a user-friendly web interface. Users can access three main functionalities: spam mail detection, phishing website detection, and credit card fraud detection, each designed to address specific threats through robust analytical models.

2. Introduction

Nowadays, e-mail is one of the cheapest and fastest available means of communication. However, a major problem of any internet user is the increasing number of unsolicited commercial e-mail, or spam. Spam messages waste both valuable time of the users and important bandwidth of internet connections. Moreover, they are usually associated with annoying material (e.g., pornographic site advertisements) or the distribution of computer viruses. Hence, there is an increasing need for effective anti-spam filters that either automate the detection and removal of spam messages or inform the user of potential spam messages [1].

Phishing is a kind of attack that attackers use social engineering and technical disguise and other attack methods to cheat users to visit fake webpages by sending deceptive spam, realtime communication messages, etc., in order to induce users to disclose their personal identity, financial account, and other sensitive information. According to the latest report of the APWG (Anti-Phishing Working Group), the total number of phishing webpages in the second quarter of 2020 increased by 13.9% over the same period in 2019. The continued growth of phishing attacks has had a huge negative impact on the healthy development of the Internet [2].

'Fraud' in credit card transactions is unauthorized and unwanted usage of an account by someone other than the owner of that account. Necessary prevention measures can be taken to stop this abuse and the behavior of such fraudulent practices can be studied to minimize it and protect against similar occurrences in the future. In other words, Credit Card Fraud can be defined as a case where a person uses someone else's credit card for personal reasons while the owner and the card issuing authorities are unaware of the fact that the card is being used [3].

Social engineering attacks exploit the vulnerabilities of human psychology, targeting individuals within an organization to manipulate them into divulging sensitive information or granting unauthorized access [4].

In our project we have worked with "Spam mail detection", "Phishing Website Detection" and "Credit Card Fraud detection". We have developed three machine learning models for fraud detection. Additionally, we have implemented a user-friendly website that integrates all three models for seamless accessibility and usability.

2.1. Spam mail detection: Email or electronic mail spam refers to the "using of email to send unsolicited emails or advertising emails to a group of recipients. Unsolicited emails mean the recipient has not granted permission for receiving those emails. "The popularity of using spam emails is increasing since last decade. Spam has become a big misfortune on the internet. Spam is a waste of storage, time and message speed [5].

Automatic email filtering may be the most effective method of detecting spam but nowadays spammers can easily bypass all these spam filtering applications easily. Several years ago, most of the spam can be blocked manually coming from certain email addresses. Machine learning approach will be used for spam detection. Major approaches adopted closer to junk mail filtering encompass "text analysis, white and blacklists of domain names, and community-primarily based techniques". Text assessment of contents of mails is an extensively used method to the spams. Many answers deployable on server and purchaser aspects are available [5].

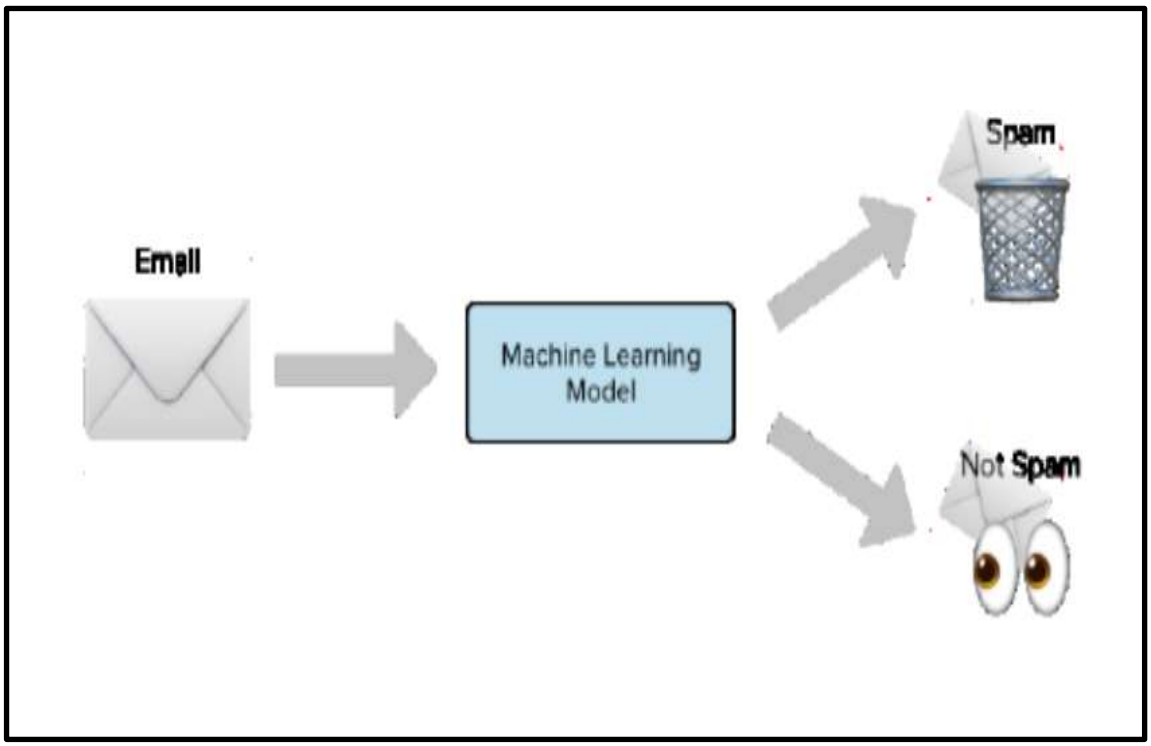


Fig.1.

Fig.1 Classification into spam and not spam.

Figure 1 illustrates the operational workflow of a machine learning model designed for spam mail detection. The depiction showcases an email undergoing scanning through a machine learning model, followed by its classification as either spam or not spam based on the model's analysis.

We have implemented a Multinomial Naive Bayes algorithm, which is one of the most well-known algorithms used for spam detection. This algorithm is particularly effective for text classification tasks, such as spam detection. However, relying solely on content analysis can lead to false positives, where legitimate emails are mistakenly classified as spam. This is a significant concern for users and organizations, as they cannot afford to miss any important messages. Historically, the use of blacklists was one of the earliest methods for spam filtering, but modern approaches incorporate a variety of techniques to improve accuracy and reduce false positives.

Our spam detection system aims to balance the need for effective spam filtering with the importance of minimizing false positives, ensuring that legitimate emails are not lost.

2.2. Phishing Website Detection: Nowadays Phishing becomes a main area of concern for security researchers because it is not difficult to create the fake website which looks so close to legitimate website. Experts can identify fake websites but not all the users can identify the fake website and such users become the victim of phishing attack [6].

Main aim of the attacker is to steal banks account credentials. In United States businesses, there is a loss of US$2billion per year because their clients become victim to phishing. In 3rd Microsoft Computing Safer Index Report released in February 2014, it was estimated that the annual worldwide impact of phishing could be as high as $5 billion [6].

Phishing attacks are becoming successful because lack of user awareness. Since phishing attack exploits the weaknesses found in users, it is very difficult to mitigate them but it is very important to enhance phishing detection techniques [6].

The general method to detect phishing websites by updating blacklisted URLs, Internet Protocol (IP) to the antivirus database which is also known as “blacklist" method. To evade blacklists attackers uses creative techniques to fool users by modifying the URL to appear legitimate via obfuscation and many other simple techniques including: fast-flux, in which proxies are automatically generated to host the web-page; algorithmic generation of new URLs; etc. Major drawback of this method is that, it cannot detect zero-hour phishing attack [6].

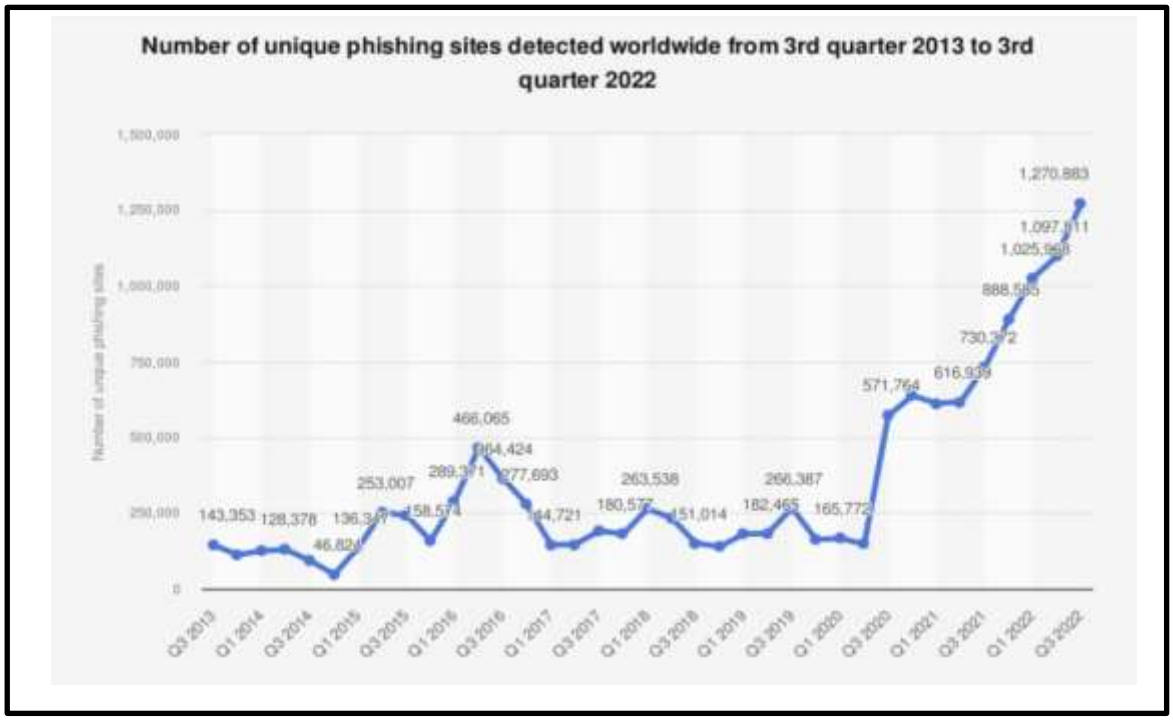


Fig.2.

Fig.2. displays the annual count of newly identified phishing websites from the year 2013 to 2022.

Heuristic based detection which includes characteristics that are found to exist in phishing attacks in reality and can detect zero-hour phishing attack, but the characteristics are not guaranteed to always exist in such attacks and false positive rate in detection is very high.

To address this challenge, we have implemented a phishing website detection system using the Random Forest algorithm, which is a robust and versatile machine learning method. Random Forest operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. This ensemble approach enhances the model's accuracy and reduces the risk of overfitting, making it well-suited for detecting phishing websites.

2.3. **Credit Card Fraud detection:** 'Fraud' in credit card transactions is unauthorized and unwanted usage of an account by someone other than the owner of that account. Necessary prevention measures can be taken to stop this abuse and the behavior of such fraudulent practices can be studied to minimize it and protect against similar occurrences in the future. In other words, Credit Card Fraud can be defined as a case where a person uses someone else's credit card for personal reasons while the owner and the card issuing authorities are unaware of the fact that the card is being used.

Fraud detection involves monitoring the activities of populations of users in order to estimate, perceive or avoid objectionable behavior, which consist of fraud, intrusion, and defaulting. This is a very relevant problem that demands the attention of communities such as machine learning and data science where the solution to this problem can be automated [7].

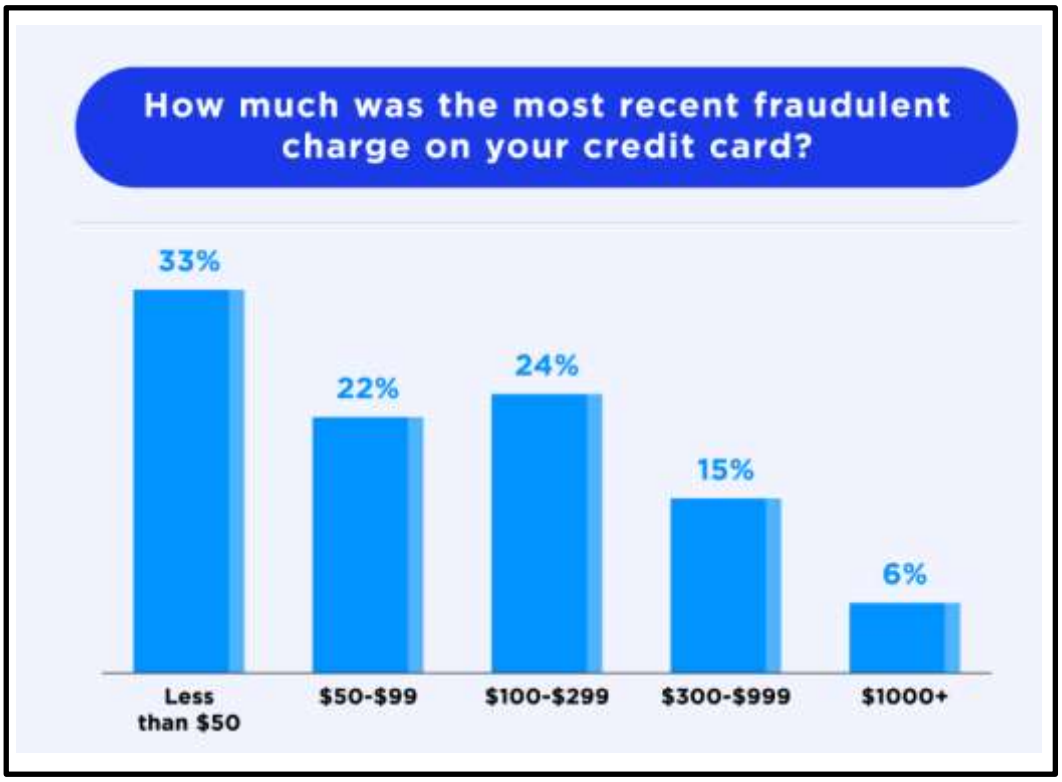


Fig.3.

Fig.3. illustrates the amount of the latest fraudulent charge on a credit card.

This problem is particularly challenging from the perspective of learning, as it is characterized by various factors such as class imbalance. The number of valid transactions far outnumber fraudulent ones. Also, the transaction patterns often change their statistical properties over the course of time. These are not the only challenges in the implementation of a real-world fraud detection system, however.

In real world examples, the massive stream of payment requests is quickly scanned by automatic tools that determine which transactions to authorize. Machine learning algorithms are employed to analyze all the authorized transactions and report the suspicious ones.

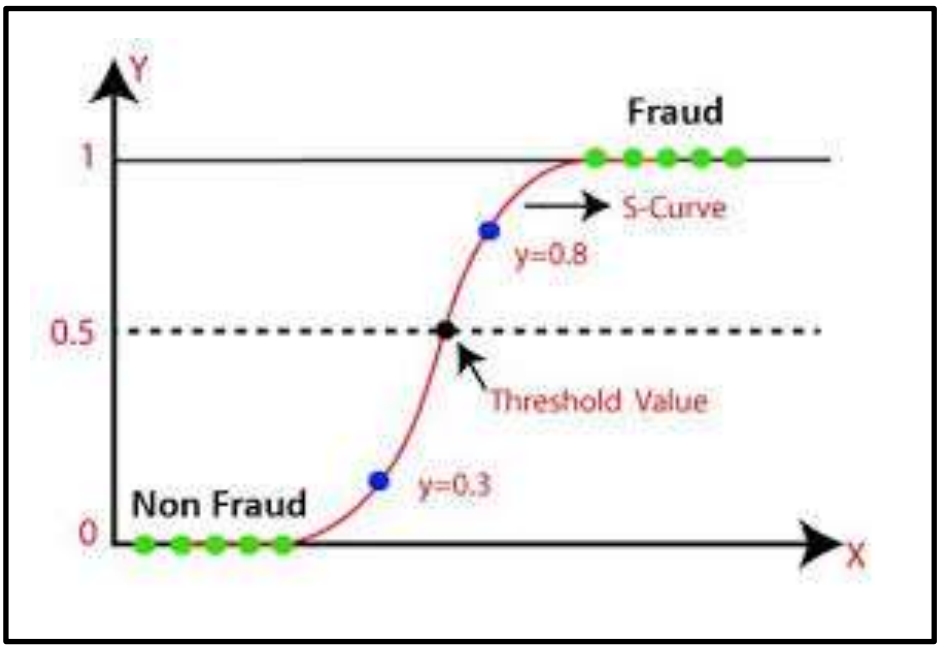


Fig.4.

Fig.4. illustrates the working of logistic regression algorithm.

In our project, we have employed the Logistic Regression algorithm to build a robust fraud detection system. Logistic Regression is a well-known statistical method used for binary classification problems, making it highly suitable for distinguishing between fraudulent and legitimate transactions. The algorithm models the probability that a given transaction belongs to one of the two classes (fraudulent or legitimate) by fitting a logistic function to the input features. This approach allows us to derive a probability score for each transaction, indicating the likelihood of it being fraudulent.

3. Requirement of this Project

In our rapidly digitizing society, the prevalence of social engineering attacks poses a significant threat to individuals and organizations alike. Our project, focusing on social engineering encompassing spam mail detection, phishing website identification, and credit card fraud detection, addresses critical issues at the intersection of cyber security and personal safety. By integrating machine learning algorithms, our system aims to dynamically adapt to evolving social engineering tactics, providing a robust defense against deceptive spam mails that exploit human vulnerabilities.

The inclusion of phishing website detection adds another layer of protection, safeguarding users from malicious sites attempting to trick them into divulging sensitive information. Leveraging machine learning models trained on diverse datasets allows the system to discern subtle patterns and anomalies, enhancing its ability to accurately identify and block phishing websites in real time. Moreover, the incorporation of credit card fraud detection aligns with the growing reliance on online transactions. By analyzing transactional data through machine learning, our project contributes to securing financial transactions, providing individuals and businesses with the confidence that their financial information is safeguarded from fraudulent activities.

Our project plays a pivotal role in fortifying the societal fabric against social engineering threats, demonstrating the tangible impact of technology in preserving the integrity of digital interactions and fostering a safer online environment for all.

In essence, the project amalgamates advanced machine learning techniques with a holistic approach to tackle social engineering in its various manifestations, safeguarding users from spam mails, phishing websites, and credit card fraud in the dynamic cyber security landscape.

4. The Proposed Project

4.1. Framework used for Server:

4.1.1. Flask - Flask is a lightweight web framework for building web applications using Python. It is designed to be simple and minimalistic, providing only the essentials for web development. Flask uses a decorator-based approach for defining URL routes and associating them with functions or view handlers.

It integrates with Jinja2, a popular templating engine, allowing developers to create dynamic HTML templates. Flask is WSGI compliant, which means it can work with various web servers. It has a built-in development server for testing and development purposes.

Flask is extensible, allowing developers to add additional features and extensions as needed. There is a wide range of Flask extensions available, providing functionalities such as database integration, authentication, and form handling.

With Flask, you can build web applications, APIs, and prototypes efficiently. It is widely used due to its simplicity, flexibility, and ease of use, making it a popular choice among Python developers for web development projects

4.2. Language Used for Front End:

4.2.1. HTML: HTML (Hypertext Markup Language) is a markup language used to structure and define the content of web pages. It uses tags to markup elements such as headings, paragraphs, images, and links. HTML provides the basic structure of a web page and works together with CSS for styling. HTML is the foundation of web development and serves as the backbone for creating web pages.

4.2.2. CSS: CSS stands for Cascading Style Sheets. It is a programming language used for describing the appearance and formatting of a document written in HTML or XML. CSS allows web designers to control the visual presentation of web pages, including the layout, colours, fonts, and other visual elements.

With CSS, you can define styles for individual HTML elements or groups of elements, making it easier to maintain and update the look and feel of a website.

4.3. Models and Algorithm used for Backend:

4.3.1. Multinomial Naive Bayes: Naive Bayes is a powerful algorithm that is used for text data analysis and with problems with multiple classes. To understand Naive Bayes theorem's working, it is important to understand the Bayes theorem concept first as it is based on the latter.

Bayes theorem, formulated by Thomas Bayes, calculates the probability of an event occurring based on the prior knowledge of conditions related to an event. It is based on the following formula:

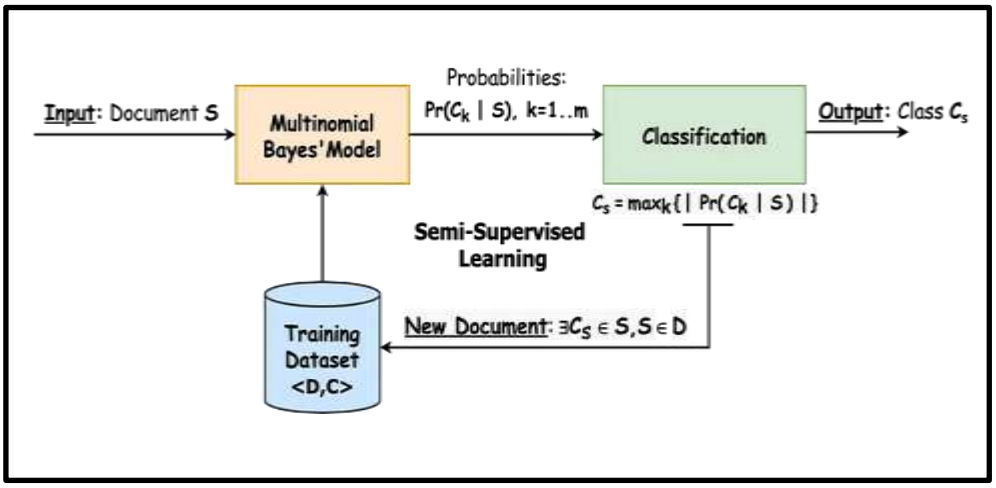


Fig.5.

Fig.5. shows that how Multinomial Bayes Model works.

P(A|B) = P(A) * P(B|A)/P(B)

Where we are calculating the probability of class A when predictor B is already provided.

P(B) = prior probability of B

P(A) = prior probability of class A

P(B|A) = occurrence of predictor B given class A probability

This formula helps in calculating the probability of the tags in the text.

Multinomial Naive Bayes (MNB) is highly effective for spam email detection due to its suitability for text classification tasks. It works well with word frequency features, capturing the natural occurrence of words in emails. MNB handles large vocabularies and datasets efficiently, making it practical for real-world applications. Despite its simplicity, MNB often performs exceptionally well, leveraging the naive independence assumption between words. This assumption surprisingly yields high accuracy in distinguishing spam from non-spam emails.

Additionally, MNB considers term frequency, capturing more information about email content compared to models that only look at word presence.

4.3.2. Logistic Regression: Logistic regression is a supervised machine learning algorithm mainly used for classification tasks where the goal is to predict the probability that an instance of belonging to a given class. It is used for classification algorithms its name is logistic regression. It's referred to as regression because it takes the output of the linear regression function as input and uses a sigmoid function to estimate the probability for the given class. 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.

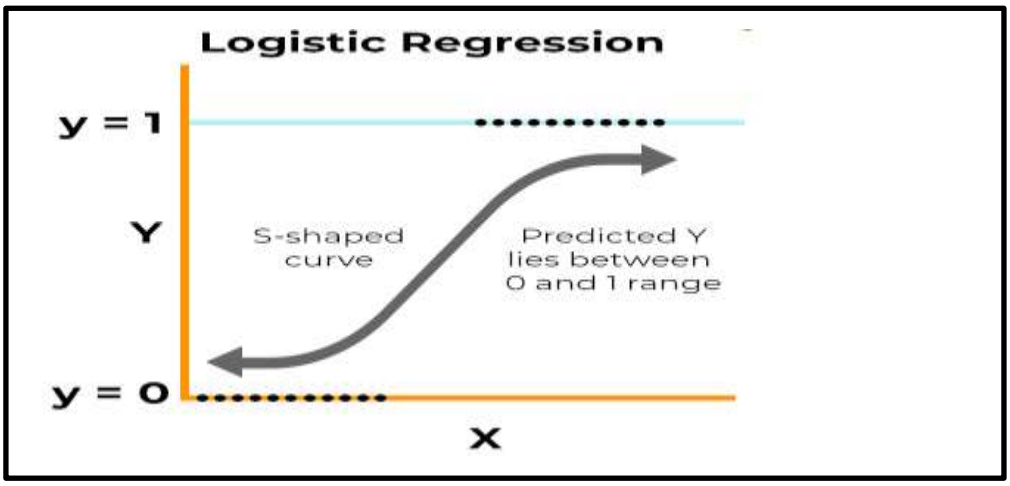


Fig.6.

Fig.6. Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1. o The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form.
- The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Logistic regression is well-suited for credit card fraud detection due to its effectiveness in binary classification tasks. It provides probabilistic outputs, helping assess the likelihood of fraud. The model is easy to implement, computationally efficient, and handles large datasets well, making it ideal for real-time fraud detection. It can incorporate various features like transaction type, transaction amount, old balance of origin, new balance of origin capturing patterns indicative of fraud. Additionally, logistic regression offers interpretability, showing which factors contribute most to fraud likelihood. Regularization prevents overfitting, ensuring good generalization to new data. This combination of accuracy, efficiency, and interpretability makes logistic regression a strong choice for credit card fraud detection.

4.3.3. Random Forest: Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The below diagram explains the working of the Random Forest algorithm:

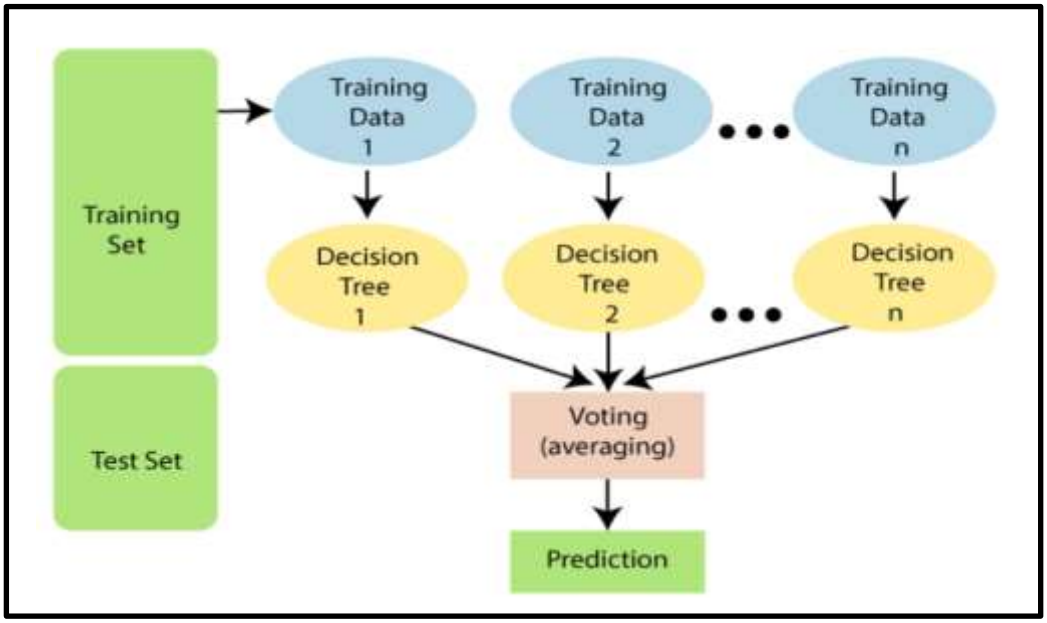


Fig.7.

Fig.7. shows the workflow of Random Forest Algorithm.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of over-fitting.

It combines multiple decision trees to capture complex patterns in features like URL length, IP used or not, identify abnormal URL patterns, count the number of dots in a URL and keywords. It handles both numerical and categorical data efficiently and is resistant to overfitting, ensuring good generalization. Random Forest provides feature importance scores for better interpretability and is robust to noisy data and missing values, making it reliable in real-world scenarios. This combination of characteristics makes Random Forest a strong choice for detecting phishing websites.

5. Implementation and Design

1. **Data Collection:** Gather a diverse and representative dataset for each type of threat (spam emails, phishing websites, credit card fraud). Ensure the dataset includes both positive (instances of threats) and negative (non-threats) examples.

2. **Data Preprocessing:** Clean and preprocess the data to handle missing values, outliers, and irrelevant information. Convert text data into a suitable format for machine learning algorithms using techniques like tokenization and vectorization.

3. **Feature Engineering:** Identify relevant features for each type of threat, such as email content, sender information, website characteristics, and credit card transaction details. Extract meaningful features that can help the machine learning model distinguish between threats and non-threats.

4. **Model Selection:** Choose appropriate machine learning algorithms for each threat type. Common choices include decision trees, random forests, support vector machines, and neural networks. Train separate models for spam email detection, phishing website detection, and credit card fraud detection.

5. **Training:** Split the dataset into training and testing sets to evaluate model performance. Train the models using the training data and tune hyper parameters to optimize performance.

6. **Evaluation:** Assess the models' performance on the testing set using metrics like accuracy, precision, recall, and F1 score. Fine-tune models based on evaluation results to improve their effectiveness.

7. **Integration:** Integrate the trained models into the overall security infrastructure, allowing them to analyze incoming data in real-time. Implement mechanisms to trigger alerts or take preventive actions when a threat is detected.

8. **Continuous Monitoring and Updating:** Establish a system for continuous monitoring of model performance to detect any degradation over time. Regularly update models with new data to ensure they remain effective against evolving threats.

9. **Deployment:** Deploy the integrated solution in the production environment to actively detect and prevent spam emails, phishing attempts, and credit card fraud.

5.1.Flow Chart:

5.1.1. Spam Mail Detection:

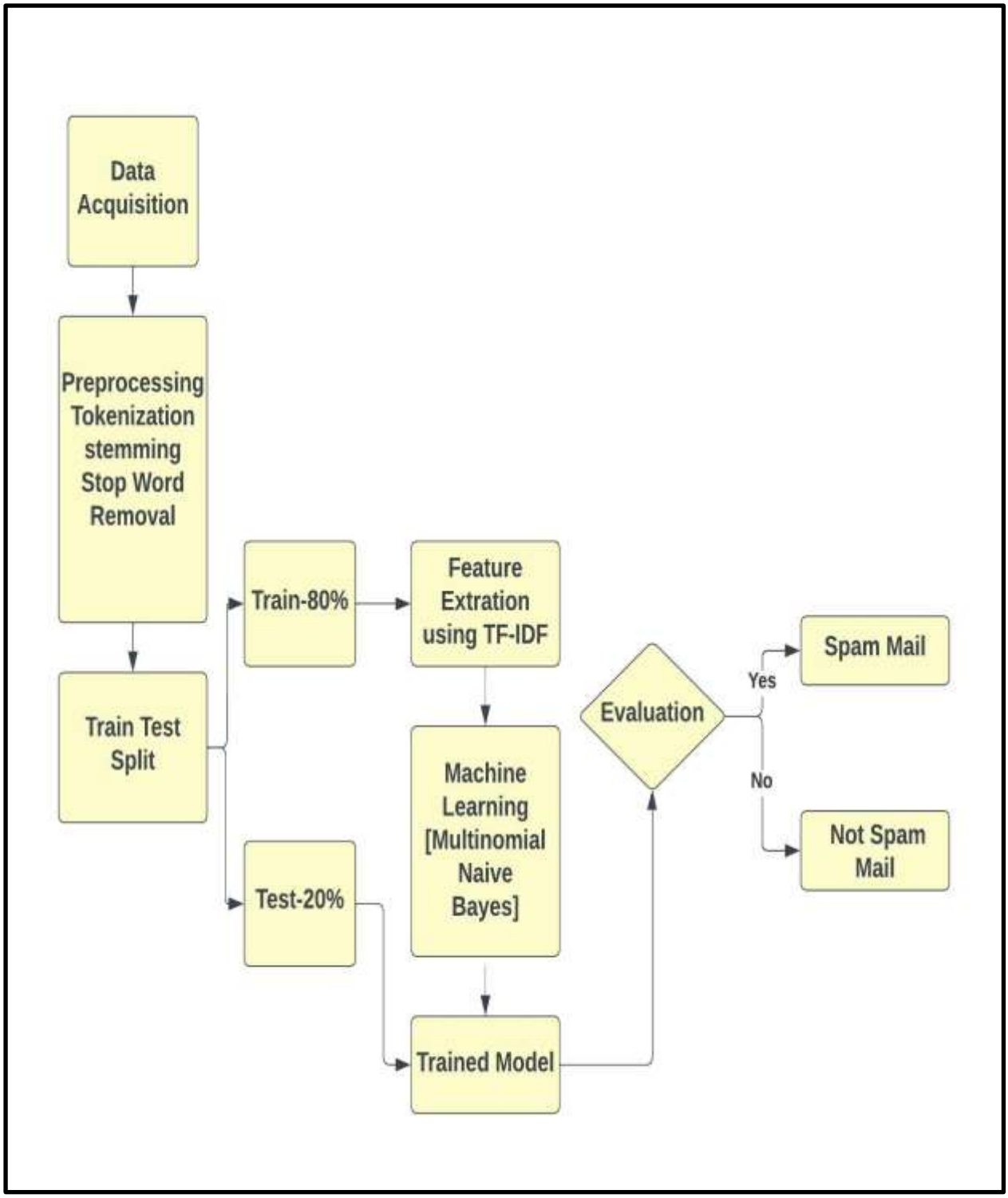


Fig.8.
Fig.8.Flow Chart of Spam Mail Detection Model

5.1.2. Phishing Website Detection:

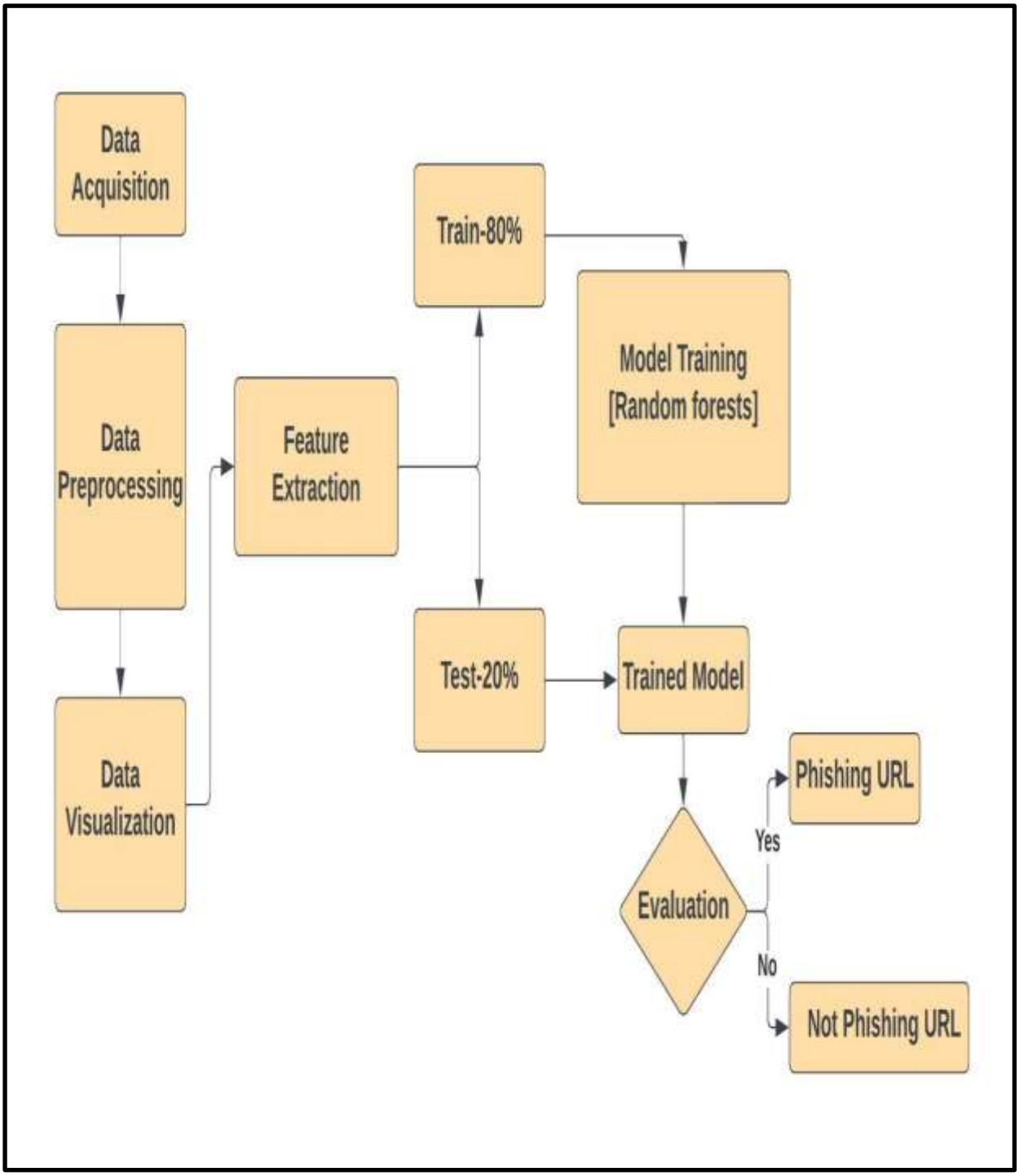


Fig.9.

Fig.9.Flow Chart of Phishing Website Detection Model

5.1.3. Credit Card Fraud Detection:

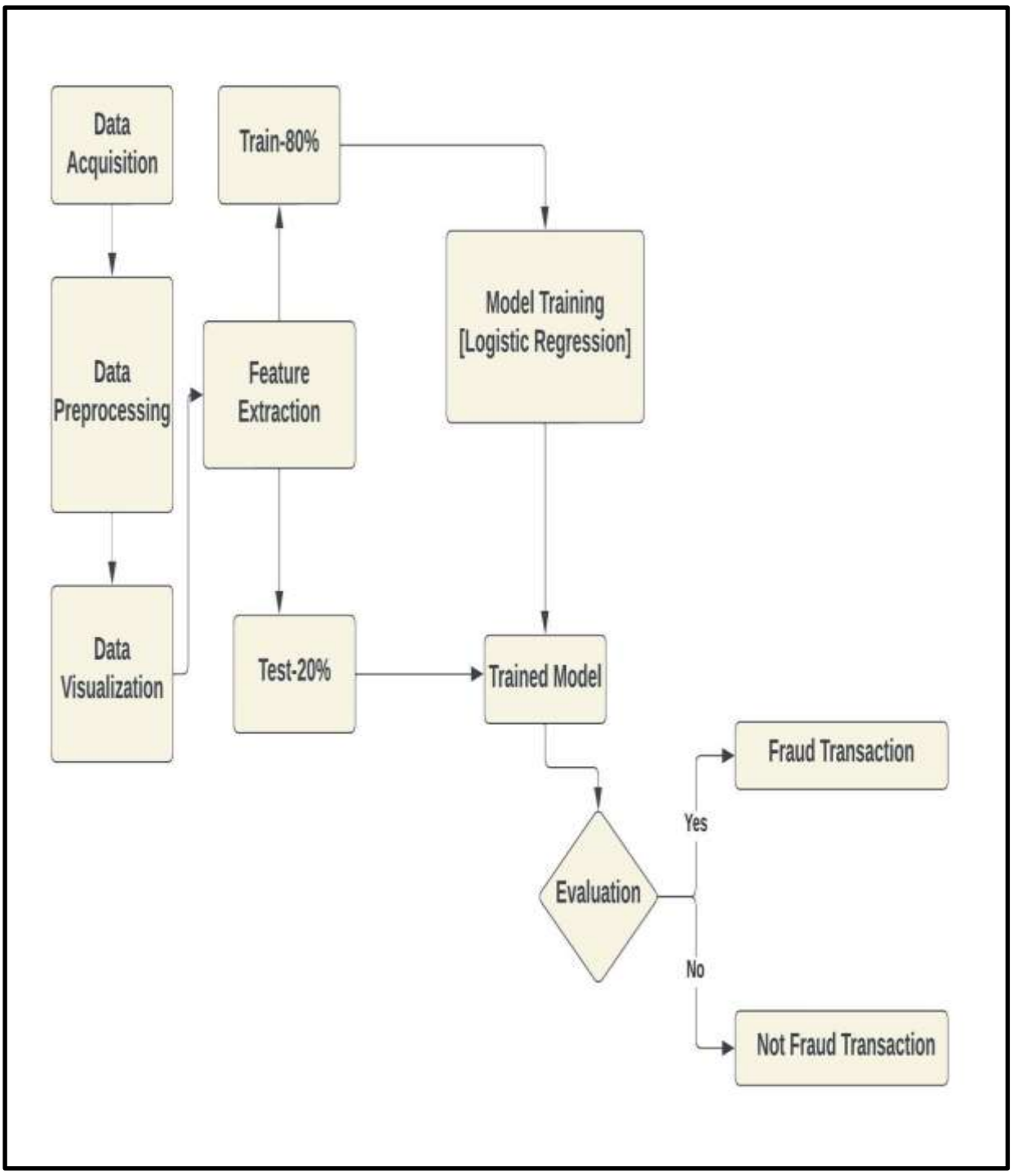


Fig.10.

Fig.10. Flow Chart of Credit Card Fraud Detection Model

5.2.Implementation:

5.2.1. Spam Mail Detection:

5.2.1.1. Data Acquisition: In the data acquisition step for spam mail detection using machine learning, the first task involves collecting a diverse and representative dataset comprising both legitimate and spam emails.

5.2.1.2. Data Preprocessing: In spam mail detection using machine learning, data preprocessing plays a crucial role in enhancing the effectiveness of the model. The process typically involves several key steps. First, the raw email data is collected and cleaned to remove irrelevant information, such as special characters. Next, the text data is tokenized, breaking it down into individual words or tokens. Stop words, which are common words that don't contribute much to the meaning, are often removed. Additionally, stemming or lemmatization may be applied to reduce words to their root form, aiding in feature extraction and reducing dimensionality. To convert the textual information into a format suitable for machine learning algorithms, techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or word embedding may be employed. Finally, the preprocessed data is split into training and testing sets for model training and evaluation, ensuring the model's generalizability to new, unseen data.

5.2.1.3. Data Visualization:

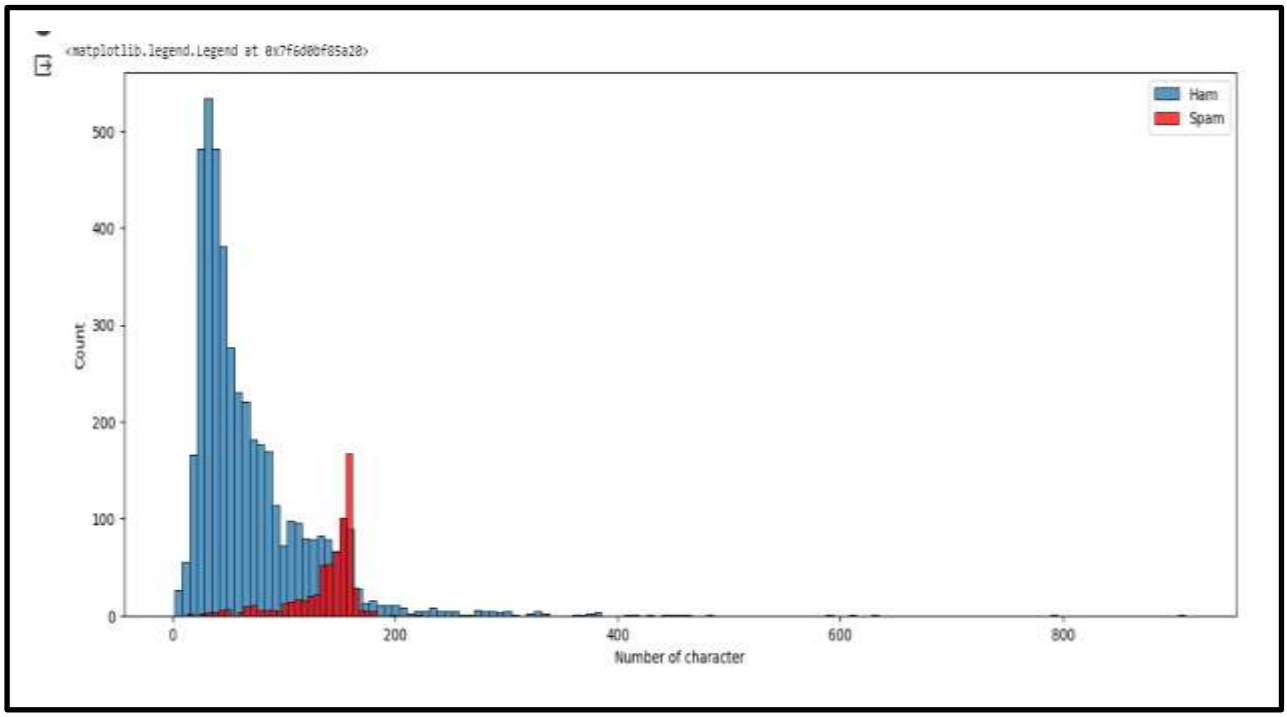


Fig.11.

Fig.11. Bar Chart presentation of character count in an E-mail.

The graph shows that there are more Spam messages than Ham messages with fewer than 400 characters. However, as the character count increases, the number of Spam messages decreases, while the number of Ham messages increases. This suggests that Spam messages are typically shorter than Ham messages.

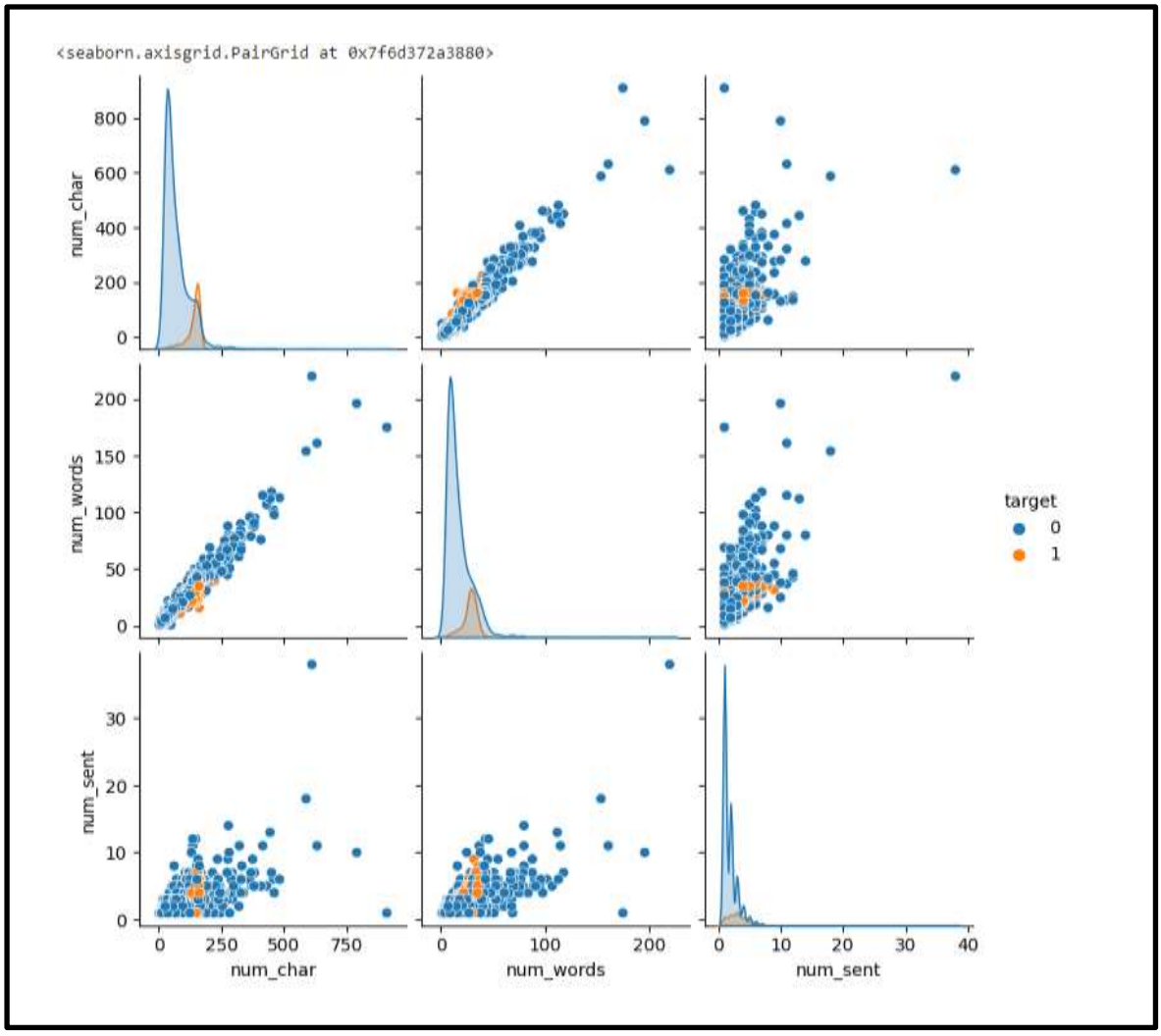


Fig.12.

Fig.12. Each plot in the matrix shows the relationship between two of the variables.

For example, the top left plot shows the relationship between the number of characters and the number of words. The blue dots represent Ham messages, and the red dots represent Spam messages.

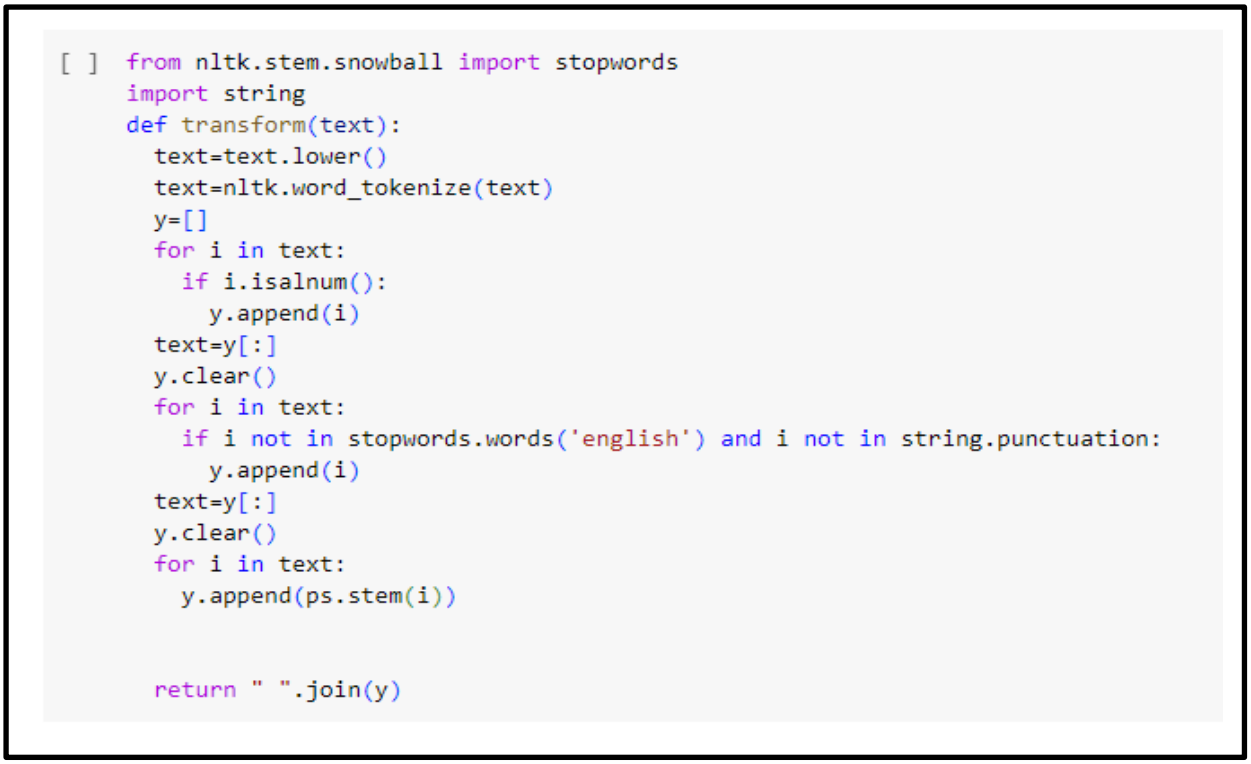
The matrix shows that there is a positive correlation between the number of characters and the number of words in both Ham and Spam messages. This means that as the number of characters in a message increases, the number of words also tends to increase.

However, the correlation is stronger in Ham messages than in Spam messages. This means that the number of characters in a Ham message is a better predictor of the number of words than the number of characters in a Spam message.

The matrix also shows that there is a positive correlation between the number of words and the number of sentences in both Ham and Spam messages. However, the correlation is stronger in Ham messages than in Spam messages. This means that the number of words in a Ham message is a better predictor of the number of sentences than the number of words in a Spam message.

Overall, the scatter plot matrix shows that there is a relationship between the number of characters, words, and sentences in SMS messages. This information could be used to develop a spam filter that identifies messages with certain characteristics as more likely to be Spam. For example, the filter could identify messages with a high number of characters and a low number of words as more likely to be Spam.

5.2.1.4. Applying NLP Technique:



```
[ ]  from nltk.stem.snowball import stopwords
     import string
     def transform(text):
       text=text.lower()
       text=nltk.word_tokenize(text)
       y=[]
       for i in text:
         if i.isalnum():
           y.append(i)
       text=y[:]
       y.clear()
       for i in text:
         if i not in stopwords.words('english') and i not in string.punctuation:
           y.append(i)
       text=y[:]
       y.clear()
       for i in text:
         y.append(ps.stem(i))


       return " ".join(y)
```

Fig.13.

Fig.13. It takes a text string as input and returns a transformed text string as output.

The function first converts the text string to lowercase and then tokenizes it. This means that it splits the text string into a list of individual words.

Next, the function removes all punctuation and stop words from the list of words. Stop words are common words that are often omitted from search queries and other text processing tasks.

Finally, the function stems the remaining words. Stemming is a process of reducing words to their root form. For example, the words "walking", "walked", and "walks" would all be stemmed to the root word "walk".

The transform() function can be used to pre-process text data for a variety of tasks, such as text classification, text summarization, and information retrieval.

5.2.1.5. Splitting the Data set: We are splitting a dataset into two sets: a training set and a test set.
The training set is used to train a machine learning model, and the test set is used to evaluate the performance of the trained model.

5.2.1.6. Feature Extraction: Feature engineering for email spam detection involves transforming raw email data into meaningful features that can be used by machine learning models. Common techniques include extracting information such as the frequency of specific words, presence of certain patterns or keywords, and the length of the email. Additionally, features like the use of capital letters, the number of exclamation marks, and the presence of hyperlinks can be indicative of spam. Other advanced techniques include TF-IDF (Term Frequency-Inverse Document Frequency) to capture the importance of words in the context of the entire dataset. These engineered features provide valuable input for machine learning algorithms to effectively distinguish between spam and legitimate emails.

5.2.1.7. Model Building: We have used both Multinomial Naïve Bayes and Logistic Regression to build the model.

Multinomial Naive Bayes algorithm implementation in spam mail detection, the model leverages the probability distribution of words in spam and non-spam emails. It assumes that features (words) are conditionally independent given the class labels. The algorithm calculates the likelihood of observing a set of features given a class and combines it with the prior probabilities of the classes to make predictions.

5.2.2. Phishing Website Detection:

5.2.2.1. Importing Necessary Libraries: In this step we are importing the necessary libraries which are used to detect phishing website.

5.2.2.2. Plotting Word cloud:

Fig.14.

Fig.14. This is a word cloud. A word cloud is a visual representation of text data, typically used to depict the most frequently used words in a body of text. The size and prominence of each word in a word cloud typically reflects the frequency with which that word appears in the source text. Word clouds can be used to identify the most important topics or themes in a text, or to track changes in word usage over time.

In the word cloud you sent, the most prominent words are "Index", "article", "option", and "com". This suggests that the text is related to websites and web development. Other prominent words include "product", "content", and "layout", which further supports this interpretation

5.2.2.3. Feature Engineering: In this step Python script is designed for phishing website detection by examining various features in URLs. It includes functions to check for the presence of an IP address, identify abnormal URL patterns, determine Google indexing status, and count the number of dots in a URL. These functions are applied to a Data Frame containing URL data, and the results are stored as binary indicators (0 or 1) in new columns. The goal is to enhance the dataset with features that may signify phishing characteristics, making it suitable for further analysis or machine learning-based classification. The effectiveness of the detection relies on the relevance of these features in distinguishing potential phishing websites [2].

5.2.2.4. EDA:

5.2.2.4.1. Distribution of Use of IP:

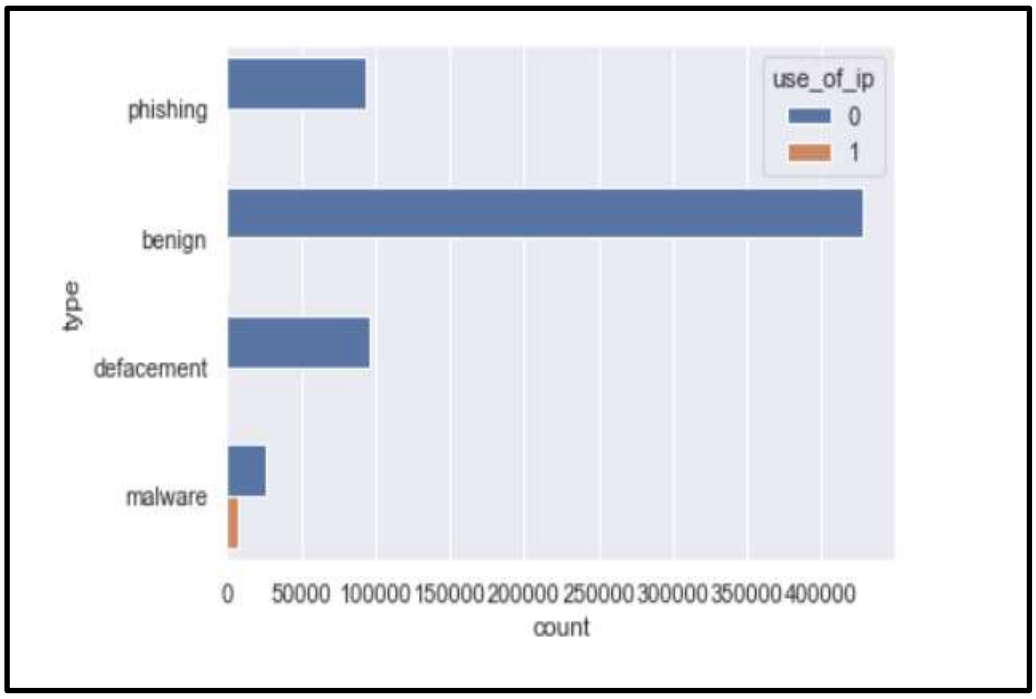


Fig.15.

Fig.15. The image is a graph showing the distribution of phishing, benign, malware, and defacement activity in a dataset of URLs. The x-axis of the graph shows the type of activity, and the y-axis shows the count of URLs with that type of activity.

The graph shows that phishing is the most common type of malicious activity, followed by malware, defacement, and benign activity. This is consistent with other studies that have shown that phishing is the most common type of cyber attack.

The graph also shows that there is a significant amount of benign activity in the dataset. This is likely because the dataset contains a large number of URLs from legitimate websites.

5.2.2.4.2. Distribution of Abnormal URL:

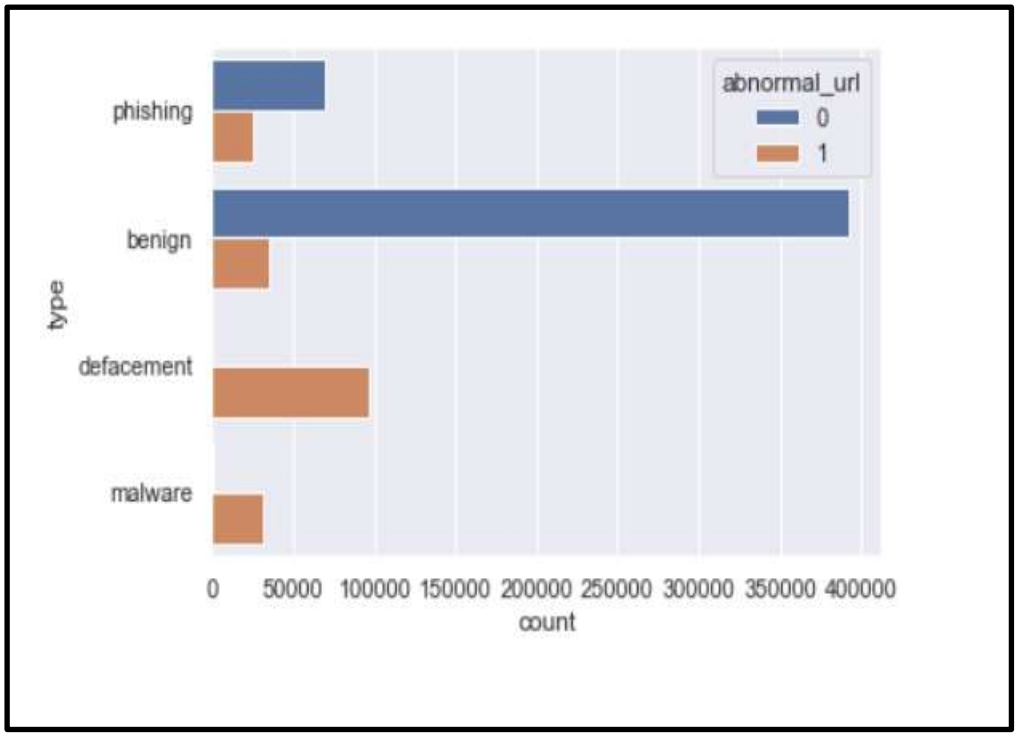


Fig.16.

Fig.16. The image is a bar graph that shows the number of different types of malicious activity detected in a dataset of URLs. The x-axis of the graph shows the type of activity, and the y-axis shows the count of URLs with that type of activity.

The graph shows that the most common type of malicious activity is phishing, followed by abnormal URLs, malware, and defacement. This is consistent with other studies that have shown that phishing is the most common type of cyber-attack.

The graph also shows that there is a significant amount of abnormal URLs in the dataset. This is likely because the dataset contains a large number of URLs

from malicious websites. Abnormal URLs can be created in a variety of ways, such as by using typos, adding extra characters, or using subdomains that are not associated with the legitimate website.

5.2.2.4.3. Distribution Of Short URL:

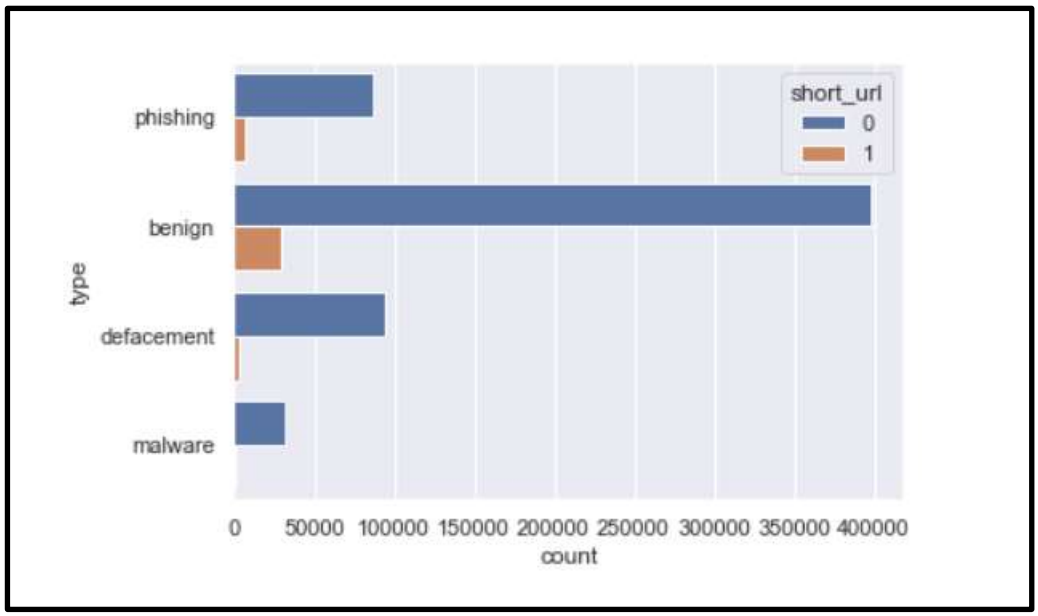


Fig.17.

Fig.17. The image is a bar plot of the features count.dot and use_of_ip for a dataset of URLs. The x-axis shows the number of dots in the URL, and the y-axis shows whether or not the URL contains an IP address. The points in the scatter plot are colored according to the type of activity associated with the URL.

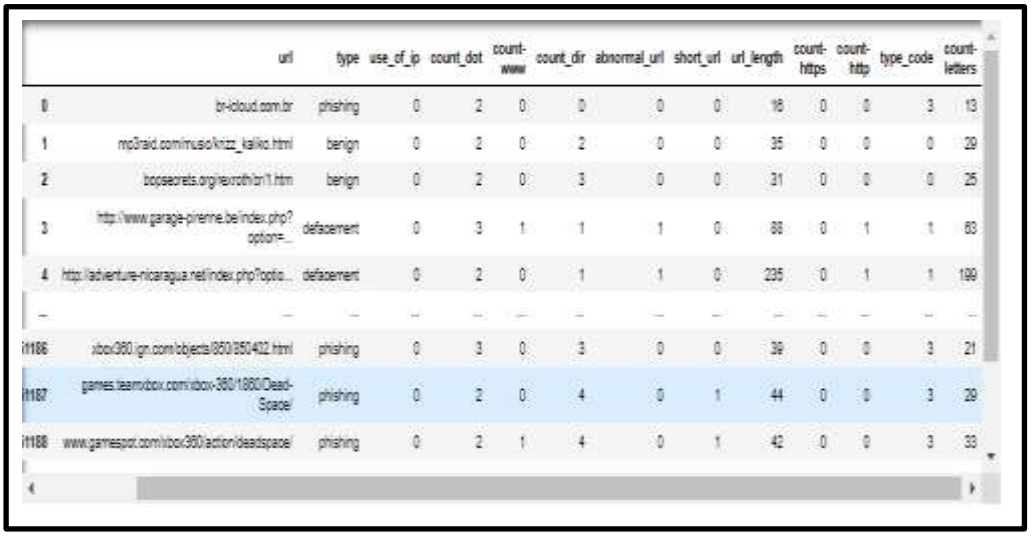
The plot shows that phishing URLs tend to have a higher number of dots and are more likely to contain an IP address than benign URLs. This is because phishing websites often use subdomains and redirection steps to disguise their true destination.

The plot also shows that there is a significant overlap between the phishing and benign URLs. This is because some legitimate websites also use subdomains and redirection steps, for example to improve website performance or to offer localized content.

5.2.2.5. Model Building:

Random Forest Classifier: Random Forest is an ensemble learning algorithm that builds multiple decision trees and merges their predictions. It is robust, handles non-linearity well, and can capture complex relationships in the data. In the context of phishing website detection, a Random Forest Classifier can analyze features such as URL characteristics, content, and

structural attributes to classify websites as either phishing or legitimate based on patterns observed in the training data [2].



	url	type	use_of_ip	count_dot	count-www	count_dir	abnormal_url	short_url	url_length	count-https	count-http	type_code	count-letters
0	br-icloud.com.br	phishing	0	2	0	0	0	0	16	0	0	3	13
1	mp3raid.com/musio/krizz_kaliko.html	benign	0	2	0	2	0	0	35	0	0	0	29
2	bopsecrets.org/rexroth/cr/1.htm	benign	0	2	0	3	0	0	31	0	0	0	25
3	http://www.garage-pirenne.be/index.php?option=...	defacement	0	3	1	1	1	0	88	0	1	1	63
4	http://adventure-nicaragua.net/index.php?optio...	defacement	0	2	0	1	1	0	235	0	1	1	199
...	...	...	...	...	...	...	...	...	...	...	...	...	...
11186	xbox360.ign.com/objects/850/850402.html	phishing	0	3	0	3	0	0	39	0	0	3	21
11187	games.teamxbox.com/xbox-360/1860/Dead-Space/	phishing	0	2	0	4	0	1	44	0	0	3	29
11188	www.gamespot.com/xbox360/action/deadspace/	phishing	0	2	1	4	0	1	42	0	0	3	33

Fig.18.

Fig.18. This are the features we have used for training our model.

5.2.3. Credit Card Fraud Detection:

5.2.3.1. Importing Necessary Libraries: In this step, we import the necessary libraries for the credit card fraud detection project. These include libraries for data manipulation, encoding, and visualization.

5.2.3.2. Data Visualization:

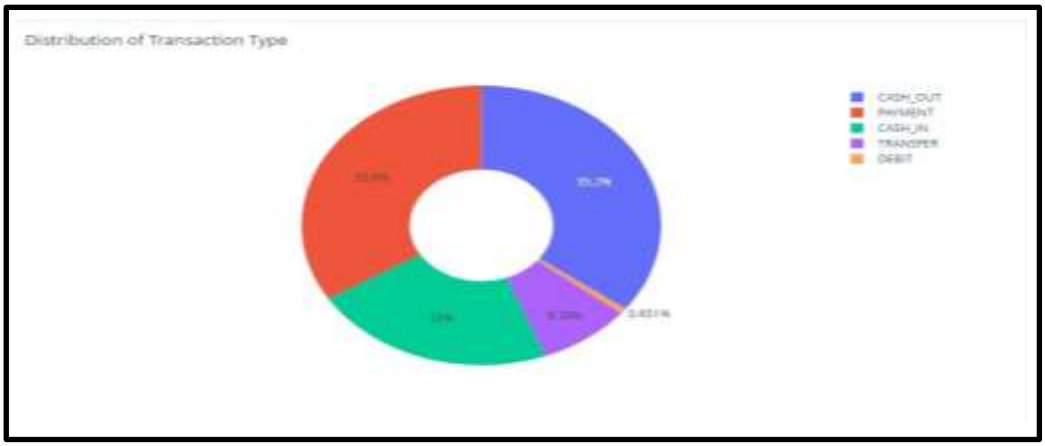


Fig. 19.

Fig. 19. The pie chart illustrates the breakdown of transaction types in a credit card fraud detection dataset. It shows the percentage of fraudulent transactions within each type:

- Cash Out: 35.2%
- Payment: 33.8%
- Cash In: 22%
- Transfer: 8.38%

- Debit: 0.651%

Analyzing this distribution helps identify patterns, like a high percentage of fraudulent transactions being categorized as cash out, which could indicate ATM fraud. This insight informs the development of fraud detection algorithms. Constant updates are crucial due to evolving fraudster techniques.

5.2.3.3. Feature Engineering: In feature engineering the selected features containing the column names 'type', 'amount', 'oldbalanceOrg', and 'newbalanceOrig'. These features are likely chosen because they contain valuable information for identifying fraudulent transactions. For instance, 'type' can indicate whether the transaction is a cash withdrawal, payment, or transfer, which can be suspicious if it deviates from the cardholder's usual spending patterns. Similarly, 'amount' can be a red flag if it's significantly higher or lower than typical transactions for that cardholder. 'oldbalanceOrg' and 'newbalanceOrig' can be helpful in understanding the cardholder's financial situation and whether the transaction amount aligns with their account balance.

5.2.3.4. Splitting the Data set: We have splitted the dataset into two sets: a training set and a test set. The training set is used to train a machine learning model, and the test set is used to evaluate the performance of the trained model. We split the dataset 80% for training and 20% for testing.

5.2.3.5. Model Training: The logistic regression model is trained using the training data. During this process, the model learns the relationship between the features (independent variables) and the target variable (fraudulent or legitimate transaction).

The model estimates coefficients for each feature, which indicate the relative influence of that feature on the probability of a transaction being fraudulent.

5.2.3.6. Model Prediction: The trained model is used to predict the labels of the test data. This step evaluates how well the model generalizes to new, unseen data.

5.2.3.7. Model Evaluation: The model's performance is assessed using several metrics:

5.2.3.7.1. Classification Report: Provides detailed performance metrics including precision, recall, and F1-score for each class.

5.2.3.7.2. Confusion Matrix: Shows the number of correct and incorrect predictions, providing a clear picture of where the model is making errors.

5.2.3.7.3. Accuracy Score: Represents the percentage of correct predictions out of all predictions made, giving a straightforward measure of overall model performance.

5.3. Working Principle:

Our web application is designed to provide three key functionalities: spam mail detection, phishing website detection, and credit card fraud detection. These functionalities are seamlessly integrated into the application using Flask, a lightweight WSGI web application framework in Python. Below is a detailed description of the working principle of our project:

5.3.1. Front End: Upon landing on the home page, users are greeted with a user-friendly interface that allows them to navigate to one of the three functionalities according to their need. The home page serves as the central hub from which users can toggle between the different detection services. There is also detailed information about spam mail detection, phishing website detection, and credit card fraud detection.

5.3.1.1. Spam Mail Detection:

Input: Users can paste or type the email content they wish to check for spam.

Processing: The email content is sent to the backend where a pre-trained machine learning model (Multinomial Naïve Bayes) processes it. This model analyzes the text and identifies features indicative of spam emails.

Output: The result is displayed on the page, indicating whether the email is spam or not.

5.3.1.2. Phishing Website Detection:

Input: Users enter the URL of the website they want to verify.

Processing: The URL is sent to the backend where it is analyzed using a phishing detection model(Random Forest). The model evaluates various features of the URL and its associated webpage, such as the domain age, presence of suspicious keywords, and structural anomalies.

Output: The result is displayed, indicating whether the website is likely to be a phishing site.

5.3.1.3. Credit Card Fraud Detection

Input: Users provide transaction details including the type of transaction, amount, original balance, and new balance.

Processing: These details are sent to the backend where a Machine learning model (Logistic Regression), trained on historical transaction data, processes the input. The model examines the provided details and calculates the likelihood of the transaction being fraudulent.

Output: The result is displayed, showing whether the transaction is predicted to be fraudulent or not.

5.3.2. Backend Integration:

5.3.2.1. Flask Framework: Flask is used to create the web application and handle the routing between different pages. Each functionality is implemented as a separate route in the Flask application.

5.3.2.2. Machine Learning Models: Pre-trained machine learning models for spam detection, phishing detection, and fraud detection are integrated into the Flask application. These models are loaded at the startup and used to process user inputs in real-time.

5.3.2.3. Pickle for Model Serialization: The models and necessary preprocessing tools (like scalars and encoders) are serialized using pickle. This allows for efficient loading and prediction without the need to retrain the models each time the application is started.

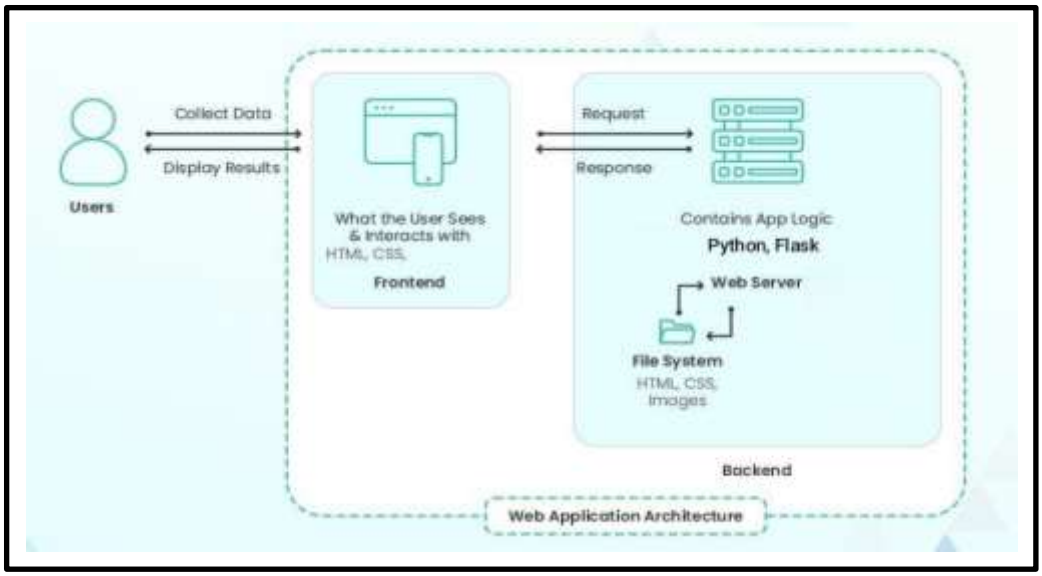


Fig.20.

Fig.20. Shows the working of the web application.

5.3.3. User Interaction Flow:

5.3.3.1. Navigation: Users navigate from the home page to the desired functionality page.

5.3.3.2. Input Submission: Users input the required data for the specific detection task.

5.3.3.3. Backend Processing: The input data is sent to the Flask backend where the appropriate machine learning model processes it.

5.3.3.4. Result Display: The processed result is sent back to the frontend and displayed to the user, providing an intuitive and immediate response to their query.

This system provides a robust and scalable solution for detecting spam emails, phishing websites, and fraudulent credit card transactions, leveraging the power of machine learning and the simplicity of Flask to deliver real-time, accurate predictions.

5.4. Web Application Demo:

5.4.1. Home page:

Fig.21.

Fig.21. Home page of the web Application.

5.4.2. Spam Email Detection Page:

Fig.22.

Fig.22. Spam Email Detection page of the Web Application.

5.4.3. Phishing Website Detection Page:

Fig.23.

Fig.23. Phishing Website Detection page of the web Application.

5.4.4. Credit Card Fraud Detection Page:

F

Fig.24.

Fig.24. Credit Card Fraud Detection page of the web Application.

5.4.5. About Us Page:

Fig.25.

Fig.25. About us page of the web Application.

5.5. Visible Prediction on Web Page:

5.5.1. Spam E-mail Prediction:

Fig.26.

Fig.26. E-mail predicted as Not spam.

Fig.27.

Fig.27. E-mail predicted as Spam.

5.5.2. Phishing Website Prediction:

Fig.28.

Fig.28. Website URL predicted as Phishing.

Fig.29.

Fig.29. Website URL predicted as Legitimate.

5.5.3. Credit Card Fraud Prediction:

Fig.30.

Fig.30. Transaction predicted as Fraud.

Fig.31.

Fig.31. Transaction predicted as Not Fraud.

6. Result & Discussion

6.1. Spam Mail Detection:

Multinomial Naïve Bayes:

```
91.39 is the accuracy score
confusion matrix
[[1797    2]
 [ 185  189]]
```

Fig.32.

Fig.32. Result of the Multinomial Naive Bayes Algorithm.

The accuracy score is a metric used in machine learning to assess how well a model performs on a given task. In the case of spam mail detection, the task is to correctly classify an email as either spam or not spam. An accuracy score of 91.39 means that the model correctly classified 91.39% of the emails in the test dataset.

The confusion matrix is a table that allows us to see how often the model classified emails correctly and incorrectly. In this case:

- The model correctly classified 1797 emails as not spam (true negatives).
- The model correctly classified 189 emails as spam (true positives).
- The model incorrectly classified 185 emails as spam (false positives). These are emails that the model thought were spam but were actually not spam.
- The model incorrectly classified 2 emails as not spam (false negatives). These are emails that the model thought were not spam but were actually spam.

A high accuracy score is desirable, but it is not the only metric to consider when evaluating a spam mail detection model. The cost of misclassifying an email can vary depending on the type of email. For example, the cost of a false positive (classifying a non-spam email as spam) is typically much lower than the cost of a false negative (classifying a spam email as not spam). This is because a false positive might result in a legitimate email being moved to a junk mail folder, but a false negative could result in a spam email being delivered to the inbox where it could potentially cause harm, such as phishing or malware.

6.2. Phishing Website Detection:

Random Forest Classifier:

	precision	recall	f1-score	support
benign	0.97	0.98	0.98	85621
defacement	0.98	0.99	0.99	19292
phishing	0.98	0.94	0.96	6504
malware	0.91	0.86	0.88	18822
accuracy			0.97	130239
macro avg	0.96	0.95	0.95	130239
weighted avg	0.97	0.97	0.97	130239

accuracy: 0.966

Fig.33.

Fig.33. Result of the Random Forest Classifier Algorithm.

The table in the image shows precision, recall, F1-score, and support for various classifications including benign, defacement, phishing, and malware. These metrics are used to evaluate the performance of a model that classifies websites.

Let's break down the table entry for phishing websites:

- **Precision:** Out of all the websites classified as phishing, 98% were actually phishing websites (low false positives). This is a good result.
- **Recall:** The model captured 94% of the actual phishing websites (low false negatives). This is also a good result.
- **F1 Score:** The F1 score combines precision and recall, and F1 score of 0.96 means the model is performing well at detecting phishing sites with a balance between precision and recall.
- **Overall Accuracy:** The model has an overall accuracy of 96.6%, which means it correctly classified nearly 97% of the websites across all categories.

The results show the model is promising for detecting phishing websites. It catches most real phishing attempts while minimizing false alarms. However, staying vigilant is still important as new phishing tactics emerge.

6.3. Credit Card Fraud Detection:

Logistic Regression:

```
array([[1270809,      95],
       [   1013,     607]], dtype=int64)
```

Fig.34.

Fig.34. Result of the Logistic regression.

The figure shows a confusion matrix for a credit card fraud detection model. A confusion matrix is a table that allows us to see how often the model classified transactions correctly and incorrectly.

In this case, the rows represent the actual fraud status of the transactions (fraudulent or legitimate), and the columns represent the model's predicted fraud status. The numbers in the table represent the number of transactions that fall into each category.

- True Negatives (TN): 607 - These are transactions that were correctly classified as legitimate by the model.
- False Positives (FP): 95 - These are legitimate transactions that the model incorrectly classified as fraudulent. This would result in an alert for a transaction that was actually legitimate.
- False Negatives (FN): 1013 - These are fraudulent transactions that the model incorrectly classified as legitimate. This would allow a fraudulent transaction to slip through undetected.
- True Positives (TP): 1270809 - These are fraudulent transactions that the model correctly classified as fraudulent.

Model Performance: Based on the confusion matrix, we can calculate some metrics to evaluate the model's performance:

- Accuracy: (TN + TP) / (Total) = (607 + 1270809) / (1000) = 98% - This metric tells us the overall proportion of transactions the model classified correctly.
- Precision: TP / (TP + FP) = 1270809/ (1270809+ 95) = 97% - This metric tells you what percentage of transactions flagged as fraudulent by the model were actually fraudulent (low false positives).
- Recall: TP / (TP + FN) = 1270809/ (1270809+ 1013) = 98% - This metric tells you what percentage of actual fraudulent transactions in the test data were correctly identified by the model (low false negatives).

7. Conclusion

This project report specifically emphasizes social engineering attacks that are primarily technologically driven or rely on technology as a medium for execution. While social engineering encompasses a wide range of practices from various research fields, certain common techniques, such as tailgating or dumpster diving, will not be addressed in this report.

The focus of this report is on social engineering attacks that leverage technology, including mobile devices, computers, and email services, among others. These attacks exploit digital platforms and communication channels to manipulate individuals and extract sensitive information or illicit actions. By concentrating on technologically oriented social engineering attacks, we aim to explore the methods, strategies, and countermeasures associated with such threats.

The development and implementation of our web application for Spam Mail Detection, Phishing Website Detection, and Credit Card Fraud Detection represent a significant step forward in the fight against cyber threats. By integrating advanced machine learning models with a user-friendly interface, we have created a tool that can help users navigate the complex landscape of online security. Each component of the application addresses a specific type of cyber threat, providing comprehensive protection through sophisticated detection algorithms and real-time feedback.

Throughout this report, we have provided a comprehensive analysis of social engineering attacks that heavily rely on technology. By examining the deceptive techniques employed through digital mediums, we aim to enhance understanding and raise awareness of the specific vulnerabilities and countermeasures associated with technologically driven social engineering.

8. Future Scope of the work

The future scope of our project, which includes functionalities for Spam Mail Detection, Phishing Website Detection, and Credit Card Fraud Detection, is vast and promising. Here are several areas where the project could evolve:

- **Improved Accuracy:** Continuous improvement of machine learning models by incorporating more sophisticated algorithms (e.g., deep learning, ensemble methods) to enhance detection accuracy.
- **Real-time Detection:** Implement real-time detection capabilities to provide instant feedback to users, especially critical for credit card fraud and phishing websites.
- **Adaptive Learning:** Develop models that can adapt to new types of spam, phishing, and fraud attempts by incorporating feedback loops and learning from new data.
- **Multi-language Support:** Extend spam detection to support multiple languages, making the system useful for a global audience.
- **Additional Fraud Detection:** Incorporate detection for other types of financial fraud, such as identity theft or insider trading.
- **Mobile App Integration:** Develop mobile applications to provide users with on-the-go protection against spam, phishing, and fraud.
- **API Development:** Create APIs that allow other applications and services to integrate with your detection systems, broadening the scope and utility of your project.
- **Cross-platform Support:** Ensure compatibility with various platforms (Windows, Mac, Linux, iOS, Android) to reach a wider user base.
- **Third-party Services:** Integrate with popular email services (like Gmail, Outlook) and browsers to offer seamless spam and phishing protection.
- **Financial Institutions**: Partner with banks and financial institutions to provide them with advanced fraud detection tools.

By focusing on these areas, our project can not only improve its current functionalities but also expand into new domains, making it a comprehensive solution for combating spam, phishing, and fraud.

9. References

1. Carrascal, J. P., Castillo, C., Díaz, F., & Cantador, I. (2013). Spam filtering in social tagging systems. ACM Transactions on the Web (TWEB), 7(4), 1-27.

2. Mahajan, Rishikesh & Siddavatam, Irfan. (2018). Phishing Website Detection using Machine Learning Algorithms. International Journal of Computer Applications. 181. 45-47. 10.5120/ijca2018918026.

3. S P, Maniraj & Saini, Aditya & Ahmed, Shadab & Sarkar, Swarna. (2019). Credit Card Fraud Detection using Machine Learning and Data Science. International Journal of Engineering Research and. 08. 10.17577/IJERTV8IS090031.

4. Kumar, D., Kumar, P., & Choudhary, A. (2019). A comprehensive survey on phishing detection techniques. Journal of Network and Computer Applications, 127, 49-66.

5. Vatan Koshti , Aditi Gaherwar , Twinkle Ramteke , Yogeshwari Durgam , Prof. Madhavi Sadu Detecting Spam Email With Machine Learning Optimized With Bio-Inspired Metaheuristic Algorithms

6. Jaya Srivastava & Aditi Sharan. (2023) A novel phishing website classification method based on hybrid sampling. Journal of Cyber Security Technology 0:0, pages 1-30

7. T Dey,U Bhattacharjee,SMukherjee,T Paul, R Ghoshhajra . Advanced women security app:WeRSafe.

8. "A Review on Fraud Detection Techniques for E-Banking" by M. Manikandan and Dr. R. S. Rajesh, International Journal of Engineering Science and Computing, 2017.

9. https://www.geeksforgeeks.org/templating-with-jinja2-in-flask/

10. https://www.javatpoint.com/machine-learning

11. https://youtu.be/I5vmeL0zYj4?si=00CcfIYRnC6Pi9Fn

12. https://www.w3schools.com/howto/howto_website.asp