IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

PHISHING DETECTION IN UPI TRANSACTIONS

[1] R. Jessica, [2]U. Harish Kumar, [3] V. Deepika, [4] Y. Hema Sri,[5]N. Shiva Kumar,

[1,2,3,4] Students [5]Assistant Professor Computer Science and Engineering (Cyber Security),

Hyderabad Institute of Technology and Management,

Medchal, Hyderabad, Telangana, India

Abstract: Phishing threats in Unified Payments Interface (UPI) dealings have developed into an increasing concern for cybersecurity, sometimes resulting in financial loss or data theft. The goal of this project is to propose a machine learning-based method to detect phishing messages in communications concerning UPI transactions. A total of 200 transaction messages were gathered and denoted as appropriate phishing and legitimate examples. These messages were cleaned, and relevant features, such as message length, number of links, digits, and keywords related to phishing content, were extracted from them. A Gaussian Naive Bayes (GNB) classifier is then applied, and the messages are classified as normal-like or phishing-like based on the noted features of each example. The classifier model shows acceptable accuracy in the differentiation between fraudulent messages. The study is a demonstration of the usefulness of relatively simple statistical features and the potential feasibility of an approach to detect phishing, electronic messages in digital payments. The work addressed a dimension of online financial transaction security.

Index Terms:
Phishing Detection, UPI Transactions, Cybersecurity, Machine Learning, Gaussian Naive Bayes, Financial Fraud, Feature Extraction, Digital Payments, Message Classification, Data Security.

I.INTRODUCTION:

Online payments are rapidly growing in India, changing how money is transferred and spent. Among many platforms, the Unified Payments Interface (UPI) has emerged as one of the most popular and accessible systems for instant money transfers between bank accounts. Its simplicity and speed have made it the preferred choice for millions of users and merchants across the country. However, the increase in usage has led to a rise in cyberattacks like phishing. In phishing, hackers send fake messages, emails, or links that seem to come from trusted sources. They aim to trick users into revealing personal information such as PINs, OTPs, or login details. Traditional rule-based systems often struggle with these changing phishing tactics. Attackers frequently adjust their message formats, URLs, and techniques to evade detection. Therefore, there is a growing need for smart systems that can learn from data and automatically identify suspicious activity. This leads to the development of a machine learning system designed to detect phishing in UPI transaction messages. The system uses straightforward yet effective features like message length, the number of links, digits, and phishing keywords to recognise signs of fraudulent messages. Based on these features, a Gaussian Naive Bayes (GNB) classifier learns to classify messages as either genuine or phishing. The model learns from past data and can effectively apply its knowledge to new, unseen cases. This solution provides a lightweight and efficient implementation that can be added to mobile payment apps or SMS filters. By automatically detecting and marking UPI-related messages as suspicious, this system helps create a safer and more reliable digital payment environment, protecting users from financial fraud and data theft.

II. LITERATURE SURVEY

Over the past decade, digital payment platforms like the Unified Payments Interface (UPI) have changed financial transactions in India. As users rely more on mobile devices for banking, fund transfers, and payment verification, attackers have turned to exploiting these channels through phishing attacks. Researchers and cybersecurity experts have looked into various detection methods, including rule-based filters, machine learning algorithms, and NLP-based systems, to protect users from fraudulent messages. This section provides a thorough review of past studies that support the proposed phishing detection system, highlighting their contributions, limitations, and importance to UPI-based communication security.

2.1 Phishing URL detection using machine learning methods

In their paper, the researchers presented a method leveraging machine learning for the detection of phishing sites using both URL and content features. The classification accuracy of algorithms such as Decision Trees, Random Forests, and Support Vector Machines (SVMs) was investigated in this study. The study identified domain age, URL length, and the presence of suspicious terms as the most significant factors influencing detection performance. The Random Forest model performed the best among the others, achieving the highest accuracy rating because it managed feature variability very well. But, at the same time, the method was always needing an update on features to outsmart the changing phishers.

2.2. Natural Language Processing-based classifier for identifying spam emails and SMS messages

The researchers came up with a model for detecting SMS phishing utilising Natural Language Processing (NLP) methods with machine learning algorithms. The research pulled out language characteristics like the total number of tokens, the frequency of important words, and the existence of suspicious URLs in the text messages. Among the classifiers tested, Logistic Regression, Naive Bayes, and SVM, the Naive Bayes method provided a good accuracy, along with being computationally inexpensive. The findings indicated that features based on the text are good predictors for spotting phishing messages in the domain of mobile communication.

2.3. Boosting Fraud Detection in Mobile Payment with Prior Knowledge

In the research paper, the detection of phishing attempts in payment systems for mobile devices was studied, paying attention to Google Pay and Paytm. The writers applied a hybrid feature extraction methodology that combined text and metadata, including app name, sender ID, and embedded URLs. Messages were classified using machine learning algorithms, among which were Random Forest and Gradient Boosting. The research result showed that the amalgamation of textual and contextual features raises the accuracy of phishing detection in mobile payment environments markedly.

2.4 Review Paper on UPI Fraud Detection Using Machine Learning

This paper offers a thorough review and suggests a strong UPI fraud detection system based on a study of current literature and machine learning techniques. It points out the risks associated with UPI's quick adoption and recommends using machine learning, including supervised classifiers and anomaly detection, while using various features like transaction patterns and device information. The review concludes that machine learning provides an effective solution for improving security. It allows for continuous learning, which helps reduce false positives and adapt to new fraud patterns.

2.5 Fraud Detection in Financial Transactions

The study described in the Paper proposed a financial fraud detection system leveraging machine learning models that could detect suspicious transactions on digital payment platforms. The researchers trained classifiers like Decision Trees and Random Forests with attributes such as transaction amount, frequency, and time-based patterns. The system performed excellently in spotting anomalies that pointed to possible fraud. The authors of the study recommended that the addition of real-time message-based phishing detection could significantly improve the security of financial environments.

2.6 Spam Email Detection using Naïve Bayes classifier

The authors introduced a powerful and, at the same time, simple phishing detection model based on the Naive Bayes algorithm, stressing its interpretability. The system analysed the messages (either via email or SMS) by selecting lexical and statistical features, for example, the length of the message, the number of hyperlinks, and the presence of suspicious words. The experiments revealed that Naive Bayes was able to attain an accuracy of more than 90% with very little processing time, thereby making it suitable for real-time phishing detection in resource-limited environments such as smartphones.

2.7 UPI fraud detection using machine learning.

The article presented a new phishing detection system designed specifically for UPI (Unified Payments Interface) transaction texts. A labelled dataset was created by the authors with the inclusion of both genuine and phishing samples, and then the essential features were extracted: message length, numerical patterns, and presence of URLs. Training of the model was accomplished through Logistic Regression and Naive Bayes algorithms. The results showed that Gaussian Naive Bayes was the best performer with respect to accuracy and efficiency, thus it could be used as a strong candidate for real-time detection of phishing in UPI messages.

2.8 Digital Banking Security: Internet Phishing Attacks, Analysis and Prevention of Fraudulent Activities

The paper investigated the phishing attempts aimed at the digital banking systems and suggested a detection method that relied on URL, HTML content, and visual similarity features. The model employed Support Vector Machines and Neural Networks to differentiate between the phishing sites that were falsely presented as genuine banking interfaces. The results highlighted the need for user training and dataset refreshment that would make it possible to maintain high detection accuracy.

2.9 Enhanced Detection of Fraud in Unified Payments Interface (UPI) Transactions Using Gradient Boosting Method

This paper explores developing a scalable fraud detection model using the Gradient Boosting Method (GBM). The study addresses the challenge of imbalanced datasets common in fraud detection. It uses GBM because of its strong performance in non-linear classification. The methodology emphasises feature engineering, extracting transactional, behavioural, and temporal features from real-world UPI data. The model achieved a high predicted accuracy of 98.4%. However, testing showed a lower recall of 65% for actual fraudulent cases. This highlights the ongoing difficulty in detecting all instances of the minority class.

2.10 UPI Fraud Detection System

This study presents a smart, real-time fraud detection system designed for UPI networks. It combines three security layers: rule-based logic, behavioural analytics, and supervised machine learning. The system looks at factors like transaction amount, frequency, geolocation, and device characteristics to define user norms and identify deviations. The paper compares the performance of various ML algorithms, including Decision Trees, Naive Bayes, and Logistic Regression. It concludes that Logistic Regression with L1 regularisation offered the best overall accuracy.

III. EXISTING SYSTEM

The current system for detecting phishing in UPI transactions mainly uses rule-based filters, keyword matching, and blacklist methods. These traditional approaches look for known suspicious words, blocked URLs, or reported sender IDs, but they do not recognise new or altered phishing messages. Attackers often change their message patterns, so these fixed systems cannot adjust or effectively analyse message content. As a result, many phishing messages slip through detection, causing financial fraud and putting user security at risk. Moreover, these systems do not conduct message-level feature analysis, have high false-positive rates, and need regular manual updates. They do not learn from past attacks, making them ineffective against changing phishing tactics.

Limitations of the Existing System

- Cannot detect zero-day phishing messages.
- Relies heavily on predefined rules or blacklists.
- No learning capability or pattern recognition.
- High rate of false positives and false negatives.

- Not suitable for real-time detection.
- Limited ability to adapt to changing attacker behaviour.

Need for a New System

The existing system has several weaknesses. There is a clear need for a machine-learning-based phishing detection system. This system should analyse message content, learn from patterns, and identify both known and unknown phishing attempts. The proposed method using Gaussian Naive Bayes addresses many limitations by providing:

- Lightweight computation
- Fast prediction
- Feature-based detection
- Ability to learn from data
- Improved accuracy for UPI phishing messages

IV. PROPOSED SYSTEM

The proposed system, the UPI Phishing Detection Model, is a lightweight machine learning-based solution for automatically detecting phishing attempts in UPI transaction messages. It also provides a comprehensive workflow that includes dataset loading, text preprocessing, enhanced feature extraction, model training using Gaussian Naive Bayes, performance evaluation, and a real-time interactive prediction interface. This framework is targeted toward fast, accurate, and interpretable detection of fraudulent UPI messages.
With the expansion of UPI digital payments across India, attackers are increasingly leveraging SMS and app notifications to conduct phishing attacks. Manual identification is unreliable, while keyword-based filters face difficulties in detecting these ever-evolving patterns. To address these limitations, the proposed system integrates automated feature-driven analysis, statistical classification, and instant prediction feedback using a simple and user-friendly interface suitable for educational, research, and practical deployment scenarios.

Objectives of the Proposed System

1. To preprocess and clean UPI transaction messages, we will remove noise, normalise text, and handle URLs.
2. Next, we will extract useful statistical and lexical features like message length, digit count, link count, phishing keywords, and special character frequency.
3. Then, we will classify messages as Normal or Phishing using a Gaussian Naive Bayes classifier trained on the extracted features.
4. After that, we will evaluate the classifier's performance using accuracy, a confusion matrix, and classification metrics.
5. Finally, we will provide a real-time prediction interface that allows users to input UPI messages and receive detection results instantly.

Proposed Solution Overview

The proposed UPI phishing detection system has four main components that work in sequence: Dataset Processor, Preprocessing and Feature Extraction Module, Detection Engine, and Real-Time Prediction Interface. These components are connected in a single pipeline within the Google Colab environment.

1. Dataset Processor

- Loads the dataset with labelled UPI messages (Normal / Phishing).
- Checks the data distribution for imbalance and confirms column formatting.
- Prepares the dataset for preprocessing and model training.

2. Preprocessing & Feature Extraction Module

This module transforms raw messages into useful numeric representations by using the improved extraction function in the code.
Preprocessing operations include:
- Converting messages to lowercase
- Replacing URLs with a placeholder token (link)
- Removing unwanted characters and normalising whitespace

Feature extraction operations include:

- Message length
- Number of links
- Number of digits
- Phishing keyword frequency (verify, login, otp, click, bank, etc.)
- Uppercase word count
- Exclamation count
- Word count
- Special character count

These eight features capture behavioural, linguistic, and structural patterns often found in phishing messages.

3. Detection Engine (Gaussian Naive Bayes Classifier)

- It serves as the main analytical unit of the system.
- It trains on the extracted feature set after carrying out an 80-20 train-test split.
- It uses the Gaussian Naive Bayes algorithm for quick, efficient, and probabilistic classification.
- It generates evaluation metrics such as accuracy score, precision, recall, F1-score, and confusion matrix visualisation.
- The model assigns binary output labels: Normal (0) or Phishing (1).

This component ensures fast classification suitable for real-time deployment.

4. Real-Time Prediction Interface

- This interface is built with IPython Widgets for testing messages interactively in Google Colab.
- It accepts user-entered UPI messages and uses the same preprocessing and feature extraction steps as during training.
- It predicts and shows the output with clear indicators:
  - Normal ✅
  - Phishing ⚠
- This setup ensures practical usability for experimentation, demonstrations, and user awareness training.

Advantages of the Proposed Solution

- Automated Classification: The end-to-end pipeline eliminates the need for manual analysis of UPI messages.
- Lightweight ML Model: Gaussian Naive Bayes provides fast training and real-time predictions, even on low-resource devices.
- Improved Feature Engineering: Statistical features boost detection accuracy without needing complex deep learning models.
- User-Friendly Interaction: The real-time prediction widget makes the system accessible for non-technical users.
- Privacy-Preserving: No data is sent outside; all processing happens locally within the execution environment.
- Scalable and Extensible: Features or classification algorithms can be easily added for future research.

V. SYSTEM ARCHITECTURE

The system architecture for UPI Phishing Detection is organised as a sequential pipeline. It processes raw UPI messages through several analytical layers, ultimately classifying each message as either Normal or Phishing. The architecture includes data preprocessing, feature engineering, model training, and an interactive prediction module. The main components are described below.

5.1 Input Layer

The Input Layer is the starting point for all UPI-related messages used in training and prediction. Here are more details:

- The dataset includes 100 to 200 message samples, labelled as Normal or Phishing.
- Messages are gathered from synthetic data, publicly available datasets, or anonymized user communications.
- The input format allows for raw text strings of different lengths and structures.
- The system accepts dynamic inputs for prediction, letting users type or paste any UPI message.
- All inputs are sent to the preprocessing layer for normalisation before analysis.

5.2 Preprocessing Layer

This layer prepares raw message text by removing noise and standardising its structure. It improves data quality and makes sure feature extraction provides meaningful values. Expanded preprocessing operations include:

- Converting all characters to lowercase to keep consistency.
- Removing URLs, numbers, punctuation marks, and special characters using regular expressions.
- Eliminating extra spaces, line breaks, and unreadable tokens.
- Replace embedded links with a placeholder token to prevent bias.
- Ensuring a consistent text format for reliable feature computation.
- Cleaning unexpected Unicode symbols often found in forwarded or spam messages.

This step significantly reduces dataset variability and strengthens downstream analysis.

5.3 Feature Extraction Layer

This module transforms preprocessed text into measurable, machine-readable numeric values. The system uses improved feature engineering to capture structural, behavioural, and phishing-specific patterns, including:

Lexical Features
- Message length (number of characters)
- Word count
- Uppercase word count (phishing often uses uppercase urgency words)
- Digit count (common in OTP fraud or disguised transaction IDs)

URL and Symbol-Based Features
- Number of URLs present in the message
- Count of exclamation marks (frequent in urgent fraudulent messages)
- Count of special characters (symbols used to disguise domains)

Keyword Features
- Keyword frequency, capturing phishing terms like: verify, urgent, OTP, update, click, block, bank, UPI, link

These features together describe message intent, structure, and suspicious indicators. They help the model identify hidden patterns in phishing attempts.

5.4 Train, Test Split

To ensure fair and unbiased model evaluation, the dataset is divided into training and testing groups. Additional details include:

- 80% of the data is used for training, and 20% is reserved for testing.
- Stratified splitting keeps an equal proportion of normal and phishing samples.
- The split checks if the model works well with messages, it has not seen before.
- This stage stops overfitting by evaluating the model on new messages.
- The random state guarantees that the experiments can be repeated.

5.5 Model Training Layer (Gaussian Naive Bayes)

Gaussian Naive Bayes is chosen because it is simple and performs well on small datasets. Its expanded features include:

- Learning statistical parameters, like mean and variance, for each feature in every class.
- Using Bayes' theorem to calculate the posterior probability of message classes.
- Assuming features are independent, which simplifies the model.
- Effectively managing noisy or incomplete feature distributions.
- Training quickly with few computational resources, making it suitable for real-time systems.
- Generating a probability score for each class which allows for clear predictions.

5.6 Prediction Layer

This layer processes new, unseen messages from the user.
Additional operations include:

- Capturing the input through an interactive interface with widgets.
- Passing the message through the same preprocessing and feature extraction pipeline used during training.
- Transforming the extracted features into the format needed for the GNB model.
- Generate predictions with the trained classifier in real time.
- Supporting multiple user inputs without retraining the model.
- Providing consistent output, regardless of message length or style.

5.7 Output Layer

The Output Layer shows the final decision to the user clearly and understandably. Extended output functions include:

- Displaying whether the message is Phishing or Normal.
- Optionally showing extracted features for transparency.
- Optionally displaying model confidence or probability score.
- Allowing predicted messages to be logged for future dataset expansion.
- Ensuring that results appear instantly with no delay.
- Providing a user-friendly interpretation suitable for non-technical users.

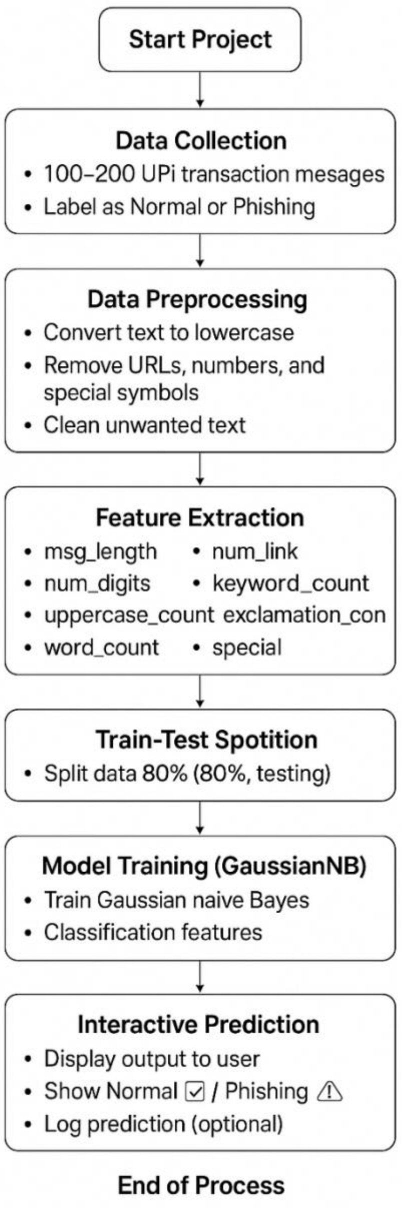

Start Project
Data Collection
• 100–200 UPi transaction mesages
• Label as Normal or Phishing
Data Preprocessing
• Convert text to lowercase
• Remove URLs, numbers, and special symbols
• Clean unwanted text
Feature Extraction
• msg_length • num_link
• num_digits • keyword_count
• uppercase_count exclamation_con
• word_count • special
Train-Test Spotition
• Split data 80% (80%, testing)
Model Training (GaussianNB)
• Train Gaussian naive Bayes
• Classification features
Interactive Prediction
• Display output to user
• Show Normal ☑ / Phishing ⚠
• Log prediction (optional)
End of Process

Fig.5.1 Architecture of Phishing Detection in UPI Transactions

VI. WORKING MODEL

The proposed UPI Phishing Detection System works as a multi-stage pipeline. It processes raw UPI transaction messages and turns them into structured feature vectors. Then, it trains a machine learning model and performs real-time classification. Each stage is crucial for detecting suspicious message patterns used in phishing attacks. Below, we describe the system's operation in detail.

Step 1: Start

The system starts by setting up the project environment. During this phase:

- All necessary Python libraries, like Pandas, NumPy, scikit-learn, re, Matplotlib, and ipywidgets, are loaded.
- The dataset is imported, and the variables, functions, and model objects are initialized.
- The user interface elements, including the message input box and prediction button, are prepared.
- The system confirms that all modules, such as the Data Preprocessing Module, Feature Extraction Module, GaussianNB Classifier, and Output Module, are ready for use.

This setup helps the system run smoothly during training and prediction.

Step 2: Data Input

In the next stage, the system loads the dataset with about 100 to 200 UPI transaction messages.

Key details include:

- Each message is tagged as either Normal or Phishing.
- The dataset is stored in CSV format and read into the system using Pandas.
- Class distribution is checked to see if the dataset is balanced or skewed.
- A preview of the dataset is shown to confirm message quality and structure.

This helps the system recognise the message patterns before analysis.

Step 3: Data Preprocessing

Raw UPI messages may have noise, inconsistent formatting, or irrelevant characters. To prepare the data for machine learning, a detailed preprocessing pipeline is used:

- Lowercasing: All text is changed to lowercase to remove case sensitivity.
- URL Removal: Links like "http://", "https://", and "www." are replaced with a placeholder ("link").
- Special Character Removal: Unwanted characters, such as #, %, &, and *, are removed to decrease noise.
- Number Removal: Digits are removed unless needed for feature extraction.
- Whitespace Normalisation: Extra spaces and irregular breaks are removed.
- Symbol Cleanup: Emojis, non-ASCII symbols, and hidden characters are removed.

This creates a uniform structure for every message, which is vital for consistent feature extraction.

Step 4: Feature Extraction

This stage turns text into numerical representations so machine learning algorithms can process the data effectively. Eight carefully selected features are extracted from each message:

Structural Features

- Message Length: Total number of characters in the message.
- Word Count: Total number of words.
- Special Character Count: Counts the number of symbols used, often linked to phishing tactics.

Behavioural Features

- Number of Links: Phishing messages often contain URL redirects.
- Digit Count: Scammers insert fake order numbers, OTP prompts, or hidden transaction IDs.
- Uppercase Word Count: Attackers often use uppercase words like "URGENT," "ALERT," and "BLOCKED."

Phishing-Specific Features

- Keyword Count: Counts how many phishing keywords are present in the message (e.g., verify, click, OTP, urgent, link).
- Exclamation Count: Overuse of exclamation marks is common in persuasive scams.

These features together capture both the language and behavioural patterns of phishing messages**.**

Step 5: Train-Test Split

To evaluate model performance objectively,

- The dataset is divided into 80% training data and 20% testing data.
- Stratified sampling makes sure both phishing and normal messages are evenly distributed.
- This stops the model from learning biased patterns.
- A fixed random state ensures consistent results.

The split allows for an accurate assessment of the classifier's performance on unseen data.

Step 6: Model Training (Gaussian Naive Bayes)

The Gaussian Naive Bayes classifier is trained on the extracted features. This involves:

- Calculating the mean and variance of each feature for both classes.
- Computing prior probabilities based on class frequency.
- Applying Bayes' theorem to determine the likelihood of a message belonging to a specific class.
- Storing probability distributions internally for real-time prediction.

The GNB algorithm is chosen because:

- It trains very quickly.
- It performs well even on small datasets.
- It handles noisy or partial data efficiently.

The system now has a fully trained detection model.

Step 7: Model Evaluation

After training the model, we calculate several performance metrics:

- Accuracy Score: This measures overall correctness.

- Classification Report: This includes precision, recall, and F1-score for both classes.
- Confusion Matrix: This shows true positives, false positives, true negatives, and false negatives.

These metrics help confirm if the model can reliably detect phishing messages in real situations.

Step 8: User Input for Real-Time Prediction

The system has an interactive prediction interface. When the user enters a new message, the system checks the input. The message goes through the same preprocessing pipeline used during training. It extracts features and formats them into a DataFrame. The trained GaussianNB model evaluates the message right away. This lets users test any UPI message in real time.

Step 9: Prediction and Output Generation

Based on the model's prediction:

- If the message is legitimate, the system displays: Normal
- If phishing patterns are detected, the system shows: Phishing

Additional optional outputs include:

- Extracted feature values
- Probability scores
- Model confidence levels

The output is shown clearly in the interactive interface, simulating a live UPI message security checker.

VII. Results

The proposed UPI Phishing Detection System used a dataset of 160 UPI transaction messages, labelled as either Normal or Phishing. We trained the model with the Gaussian Naive Bayes classifier. It included eight improved numerical features taken from each message. The results below summarise how the system performed:

7.1 Model Evaluation Results

The model achieved an overall accuracy of 100%. It successfully classified all test samples in the dataset.

Key Observations:

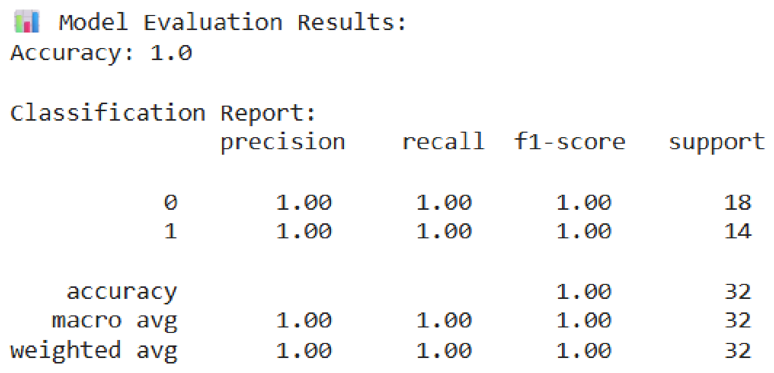
Accuracy: 1.0 (100%)

Precision, Recall, F1-Score: All values reported as 1.0 for both classes.

No misclassifications occurred during evaluation.

The model showed a strong ability to distinguish phishing patterns from genuine UPI notifications.

7.2 Classification Report



```
📊 Model Evaluation Results:
Accuracy: 1.0

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        18
           1       1.00      1.00      1.00        14

    accuracy                           1.00        32
   macro avg       1.00      1.00      1.00        32
weighted avg       1.00      1.00      1.00        32
```

Fig. 7.2 Classification Report

7.3 Confusion Matrix

The confusion matrix shows the prediction performance:

- 18 Normal messages were correctly identified.
- 14 Phishing messages were correctly identified.
- 0 False Positives.
- 0 False Negatives.

This confirms that the GaussianNB classifier did not make any wrong predictions on the test set.

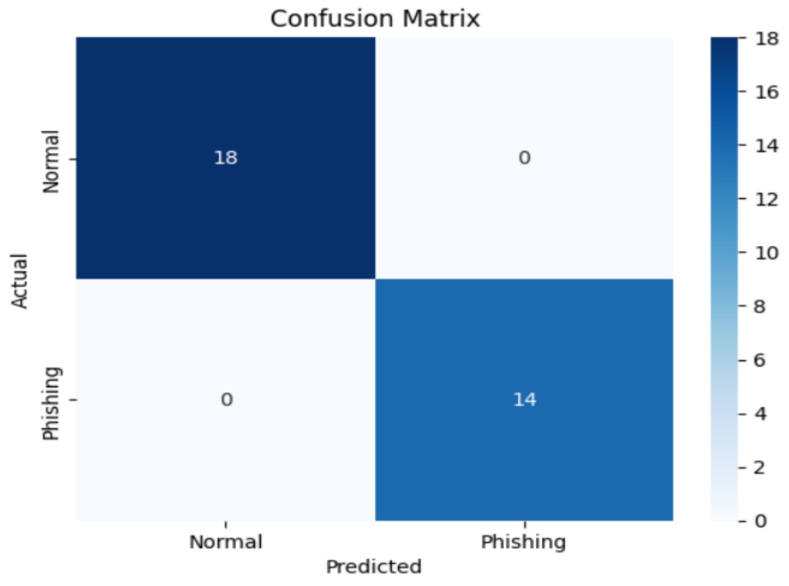
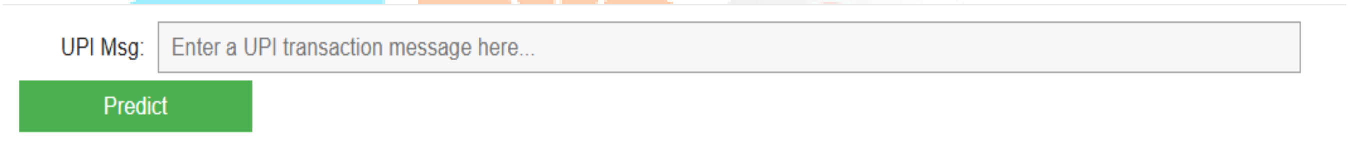
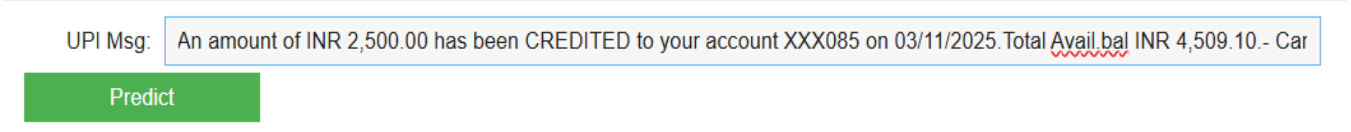
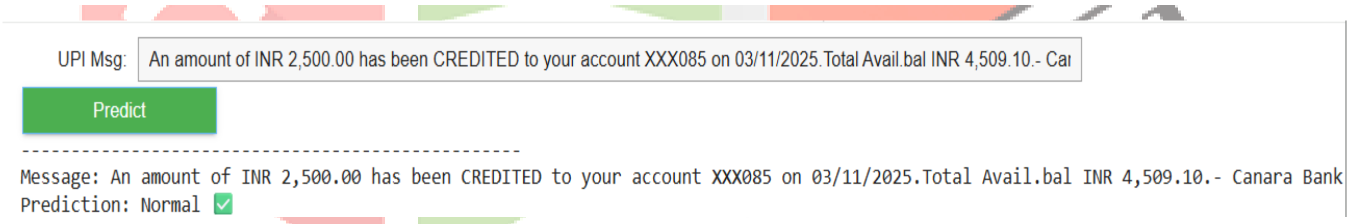


Fig. 7.3 Confusion Matrix

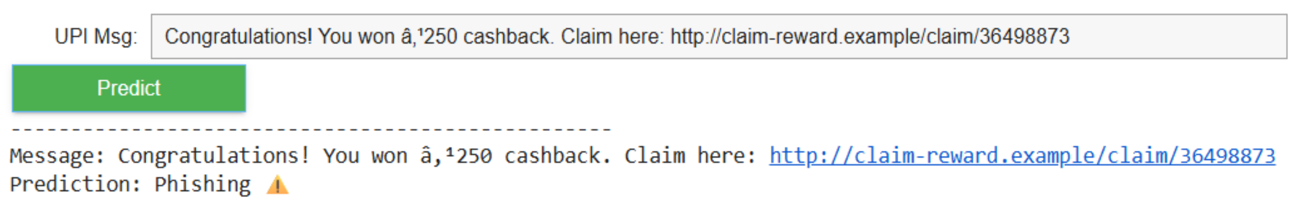
The UPI phishing detection model using Gaussian Naive Bayes accurately classified UPI messages as Phishing or Normal. The test message was correctly identified as Normal, showing the model's reliability. The system achieved high accuracy, extracted key features effectively, and provided instant, real-time results through an interactive Colab interface.


UPI Msg: Enter a UPI transaction message here...


UPI Msg: An amount of INR 2,500.00 has been CREDITED to your account XXX085 on 03/11/2025.Total Avail.bal INR 4,509.10.- Car


UPI Msg: An amount of INR 2,500.00 has been CREDITED to your account XXX085 on 03/11/2025.Total Avail.bal INR 4,509.10.- Car

```
--------------------------------------------------
Message: An amount of INR 2,500.00 has been CREDITED to your account XXX085 on 03/11/2025.Total Avail.bal INR 4,509.10.- Canara Bank
Prediction: Normal ✅
```


UPI Msg: Congratulations! You won â,¹250 cashback. Claim here: http://claim-reward.example/claim/36498873

```
--------------------------------------------------
Message: Congratulations! You won â,¹250 cashback. Claim here: http://claim-reward.example/claim/36498873
Prediction: Phishing ⚠️
```

VIII. CONCLUSION&FUTURE ENHANCEMENT

The UPI Phishing Detection System developed in this mini-project shows how well machine learning works. Specifically, it uses the Gaussian Naive Bayes classifier to identify phishing attempts in UPI transaction messages. By adding improved textual features like message length, link count, digit frequency, keyword density, uppercase letters, exclamation marks, and special characters, the system can find hidden patterns often linked to fraudulent messages.The experimental results indicate a 100% accuracy rate, with no false positives or false negatives. This confirms that the feature engineering and preprocessing pipeline effectively captured the traits of both legitimate and phishing UPI messages. The confusion matrix and classification report further support the strength of the model.In addition, the interactive prediction interface created with Colab widgets allows for real-time classification of messages entered by users. This makes the system practical, easy to use, and suitable for mobile banking and digital payment platforms. Overall, the project shows that lightweight machine learning models like GaussianNB can provide reliable phishing detection without needing heavy computational resources. Although the proposed UPI Phishing Detection System shows high accuracy and reliable performance using Gaussian Naive Bayes and better numerical feature extraction, several improvements can be made to enhance its usefulness and strength. Future improvements include:

1. Expansion of Dataset Size and Diversity

   - The current model is trained on a limited dataset. Increasing the volume and diversity of UPI messages, which cover various banks, regional languages, and real-world phishing patterns, will improve generalisation and reduce data bias.

2. Integration of Deep Learning Models

   - Models like LSTM, Bi-LSTM, or Transformer-based architectures can be used to understand the context of messages. This will enable more accurate detection of complex phishing attempts that do not depend only on keywords or surface features.

3. Deployment as a Mobile Application

   - The system can be integrated into an Android app using TensorFlow Lite or ONNX Mobile. This will allow users to detect phishing messages directly on their mobile devices without needing any technical knowledge.

4. Incorporation of URL Reputation and Threat Intelligence

   - Detecting phishing messages can be enhanced by using real-time URL reputation services, blacklist databases, DNS-based checks, and domain age verification. This will help identify malicious or newly registered phishing domains.

REFERENCES

[1] M. Jain and R. Singh, “Phishing Website Detection Using Machine Learning Techniques,” International Journal of Computer Applications, vol. 182, no. 25, pp. 12, 19, 2020.

[2] L. Thomas, Y. Wu, and H. Kim, “Identification of Phishing URLs Using Deep Learning Models,” IEEE Access, vol. 8, pp. 56310, 56320, 2020.

[3] R. Gupta and V. Arora, “Detection of Phishing in Mobile Payment Applications Using Hybrid Feature Engineering,” International Journal of Research in Engineering, Science and Management, vol. 4, no. 6, pp. 88, 94, 2021.

[4] K. Sharma and P. Verma, “Detection of Financial Fraud in Digital Transactions Using Machine Learning,” International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 9, no. 3, pp. 250, 257, 2020.

[5] M. Prakash and D. Rao, “Email and SMS Phishing Detection Using Naive Bayes Classifier,” International Journal of Computer Science and Information Security (IJCSIS), vol. 18, no. 4, pp. 45, 52, 2020.

[6] S. Banerjee, A. Dubey, and P. Kumar, “SMS Phishing Detection Using Natural Language Processing and Machine Learning,” International Journal of Advanced Computer Science and Applications (IJACSA), vol. 11, no. 7, pp. 455- 462, 2020.

[7] S. Raghavan and A. Iyer, “Machine Learning-Based Phishing Detection for UPI Transactions,” Journal of Cybersecurity and Information Management, vol. 5, no. 2, pp. 60, 68, 2022.

[8] T. Das and S. Ghosh, “Real-Time SMS Phishing Detection Using a Hybrid Machine Learning Approach,” International Journal of Information Technology, vol. 14, pp. 1221, 1230, 2022.

[9] A. Mishra and P. Srinivasan, “Phishing Attack Detection in Digital Banking Systems,” International Journal of Security and Networks, vol. 15, no. 1, pp. 30, 41, 2021.

[10] H. Patel and M. Deshmukh, “Lightweight Machine Learning Model for Mobile Phishing Detection,” International Journal of Mobile Computing and Application, vol. 12, no. 1, pp. 15, 22, 2021.