MALAYSIAN JOURNAL OF
CYBERSECURITY AND APPLICATIONS

DEVELOPING AN EMAIL DETECTION TOOL INTEGRATING RANDOM FOREST AND NAIVE BAYES ALGORITHMS FOR PHISHING PROTECTION

Justine Kurniadi*, Julia Juremi

Asia Pacific University of Technology & Innovation
Technology Park Malaysia, Bukit Jalil, 57000 Kuala Lumpur, Malaysia

Emails: justine.kurniadi@gmail.com*, julia.juremi@apu.edu.my

ABSTRACT

Phishing attacks pose a significant threat to organizational email security, especially small and medium sized organizations, exploiting human vulnerabilities to steal sensitive information. This study develops a phishing detection tool that integrates Random Forest and Naive Bayes algorithms in a hybrid model to enhance detection accuracy. The tool analyzes email headers, content, and URLs, providing actionable insights to enhance organizational email security. Through dataset training and testing, the hybrid model achieved 96.81% accuracy, outperforming individual models. The proposed solution includes a user-friendly GUI and CLI, with features like URL screenshot previews and report generation. Nevertheless, the project will deliver a user-friendly security tool which strengthens email protection while decreasing phishing risks to safeguard organizational sensitive data. The project supports Sustainable Development Goal (SDG) 9: Industry, Innovation, and Infrastructure through its promotion of safe email communication security for organizations.

Keywords: Email Phishing; Threat Detection; Hybrid Machine Learning; Email Security; URL Analysis.

1. INTRODUCTION

Phishing attacks have become a significant cybersecurity threat to organizations all over the world, especially as the years keep moving forward, phishing attacks will be more likely to occur, whether in unprepared or prepared organizations, by exploiting human vulnerabilities, error, trust, and technical weaknesses to steal sensitive information, such as login credentials, financial details, and confidential documents. Cybercriminals often use fake emails, websites, messages, and phone calls to make their scams look real [1]. Despite many organizations have used the availability of email security solutions or tools to protect their email systems. As technology improves, hackers are also becoming more advanced, finding new ways to bypass security measures, making traditional detection systems less effective. As we know, email communication serves as a critical backbone for organizations while employees use email systems throughout their daily work activities, so phishing attacks represent their primary security threat. Based on recent data, 57% of firms see phishing efforts on a weekly or daily basis. Furthermore, malicious emails make up nearly 1.2% of all emails sent, or over 3.4 billion phishing emails every day [2]. This statistic shows the scope of the issue and emphasizes how crucial strong security measures are to protect organizations.

This paper proposes a phishing detection tool that integrates Random Forest and Naive Bayes algorithms in a hybrid model to improve accuracy and efficiency. The tool evaluates email content, headers, and URLs, generating logs and reports. The study emphasizes the proposed solution's architecture, machine learning model testing, and key diagrams to illustrate the system's workflow.

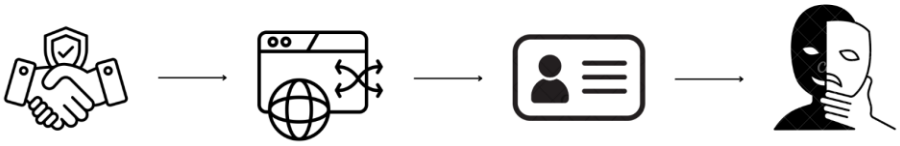

Fig. 1: Four Steps of Phishing Attacks

Phishing attacks begin with four step processes, which include gaining victim trust so that the user acts or performs the desired actions the attacker wants. Then it will redirect victims to a phishing site (if the email contains an URL) which user then provides their sensitive information. The next step is attackers gaining data from victims over redirected forms or websites or as a reply over fake emails. Finally, the attacker can perform required identity fraud using the sensitive information obtained on the previous step [3].

2. RELATED WORKS

This pseudocode on (Fig. 2) details the evaluation of email headers to detect spoofing or authentication issues, contributing to a threat assessment. It loads and parses the email file into headers, initializing a metadata dictionary with default values (false for authentication flags, zero for threat score). It iterates through headers, extracting key fields like Message-ID, From, Return-Path, Received (counting hops for routing analysis), IP addresses, and authentication results (using regex to check SPF, DKIM, DMARC passes/fails). If no authentication passes, it flags as spoofed. The threat score is computed by adding points for issues like spoofing (25 points), reply-to mismatches, or excessive routing hops, with reasons appended to a list. The threat level is determined based on score thresholds (MALICIOUS >=70, SUSPICIOUS 30-69, LEGITIMATE otherwise). A recommendation is generated based on the level and reasons. The function returns the metadata with all computed details or an error if parsing fails.

```
FUNCTION analyze_email_header(file_path):
    TRY:
        LOAD file: OPEN and PARSE email message
        INITIALIZE meta dictionary with defaults (e.g., spf-record: False, threat-score: 0)
        INITIALIZE received_count = 0
        INITIALIZE threat_reasons list
        FOR each header in parsed headers:
            IF header == 'message-id': SET meta['message-id']
            IF header == 'from': SET meta['from']
            IF header == 'return-path': SET meta['return-path']
            IF header == 'received': INCREMENT received_count, IF first: SET sender-client
            IF header == 'x-originating-ip': EXTRACT IP using regex
            IF header == 'authentication-results':
                CHECK regex for spf=pass, dkim=pass, dmarc=pass → SET respective flags
                IF fail patterns: SET spoofed = True
            IF header == 'reply-to': SET meta['reply-to'] and spoofed-mail
            IF header == 'date': SET meta['dt']
            IF header == 'content-type': SET meta['content-type']
        IF NOT (spf or dkim or dmarc): SET spoofed = True
        SET meta['multiple-received'] = received_count
        CALCULATE threat_score:
            IF spoofed: ADD 25, APPEND reason "Authentication failures detected"
            IF reply-to mismatch with from: ADD 20, APPEND reason "Reply-To mismatch"
            IF multiple-received > threshold (e.g., 5): ADD 15, APPEND reason "Unusual routing"
            IF suspicious IP or client: ADD points accordingly
        DETERMINE threat_level:
            IF threat_score >=70: 'MALICIOUS'
            ELSE IF >=30: 'SUSPICIOUS'
            ELSE: 'LEGITIMATE'
        GET recommendation based on level and reasons
        RETURN meta with all details, score, level, reasons
    EXCEPT exception as e:
        RETURN error: "Header analysis failed: {e}"
```

Fig. 2: Pseudocode Details of Email Headers Analyze

2.1 Email Headers and Metadata Analysis

Phishing detection often relies on analyzing email headers and metadata, since attackers commonly manipulate fields to make malicious emails appear genuine. Key headers such as From, Reply-To, Return-Path, Received, Message-ID, X-Mailer, MIME-Version, Content-Type, and X-Originating-IP play important roles in detecting phishing attempts and provide critical indicators of spoofing and authenticity [4]. Security protocols including SPF, DKIM, and DMARC add an additional verification layer to validate the legitimacy of emails [5]. Table 1 summarizes and explains the purpose of these headers and protocols, their phishing indicators, and relevance to detection. By combining header analysis with content-based machine learning, phishing detection systems can provide stronger defenses against email-based threats.

Table 1: Overview of the Important Email Headers

Header	Purpose	Phishing Indicator	Why It Matters in Detection
From [6]	Show the sender's email address.	Attackers often spoof this field to appear as a trusted sender.	Checking this can reveal domain mismatches that indicate phishing attempts.
Reply-To [7]	Specifies where email replies go.	Phishers use a different reply-to address to redirect user responses to their inbox.	Helps detect cases where emails pretend to be from one sender but direct replies elsewhere.
Return-Path [8]	Defines where undelivered emails are sent.	If the Return-Path address differs from the sender's email, it may be a phishing attempt.	Helps identify emails that don't truly originate from the claimed sender's domain.
Received [9]	Logs the email's journey through mail servers.	If the email claims to be from a US bank but was sent from a server in another country, it is suspicious.	Allow tracking of the real sender's IP and mail servers to detect fraud.
Message-ID [10]	Provides a unique email identifier.	Phishers may use fake or missing Message-IDs to hide their real email source.	Helps verify whether the email was sent from a legitimate email provider.
X-Mailer [11]	Shows the software used to send the email.	Phishing emails may use "PHP Mailer" or outdated email clients, unlike legitimate business emails.	Detects unusual mailing software, often used by attackers.
MIME-Version & Content-Type [12]	Defines email format (text, HTML, attachments).	Emails with hidden images, obfuscated text, or base64-encoded content are often phishing attempts.	Help identify emails that try to bypass security filters.
X-Originating-IP [13]	Displays the original sender's IP address.	If a company email comes from an unexpected foreign IP, it may be fake.	Verifies whether an email truly came from the claimed location.
SPF [14]	Verifies if the sending mail server is authorized by the domain's DNS records.	Failure indicates the sender is not authorized, common in spoofed phishing emails.	Detect unauthorized servers sending emails on behalf of a domain, preventing spoofing.
DKIM [15]	Provides a digital signature to verify the email content hasn't been altered in transit.	DKIM failure or absence suggests tampering or forgery by phishers.	Ensures email integrity and authenticity, helping identify modified or fake messages.
DMARC [16]	Builds on SPF and DKIM to specify actions if authentication fails and provides reporting.	DMARC fail often means the email doesn't align with the domain's policy, indicating potential phishing.	Enforces domain-level protection and reports failures, aiding in the detection of unauthorized emails.

Each header feature was assigned a relative importance weight based on model correlation and information gain analysis. For instance, SPF, DKIM, and DMARC validation contributed the most (≈45% cumulative influence) toward classification outcomes, while "Received" and "From" headers contributed ≈25%, emphasizing that modern phishing detection models must incorporate authentication-based weighting for improved accuracy.

2.2 Email Content Analysis with Machine Learning

Machine learning-based analysis is another crucial method for phishing detection. Traditional rule-based phishing detection systems frequently have trouble adjusting to newer attack methods. A more flexible alternative can be found by machine learning algorithms like Random Forest and Naïve Bayes. A probabilistic model called Naïve Bayes uses past knowledge of phishing patterns to categorize emails. It determines an email's likelihood of being phished by looking at its structure, language, and links. While Random Forest is an ensemble learning technique that uses several decision trees to determine whether an email is safe or phishing. By training these models on phishing datasets, the system can identify patterns and detect new phishing attempts more effectively.

```
IMPORT necessary libraries: pandas, numpy, sklearn (model_selection, feature_extraction, naive_bayes, ensemble, preprocessing),
re, warnings, joblib, google.colab.drive

SUPPRESS warnings

MOUNT Google Drive for file access

LOAD dataset from CSV file path in Drive

PREPROCESS dataset:
    DROP rows with NaN in 'Email Text' or 'Email Type'
    DEFINE function preprocess_text(text):
        REMOVE URLs using regex (http\S+)
        REMOVE non-word characters ([^\w\s])
        CONVERT to lowercase
        REPLACE multiple spaces with single space and strip
        RETURN cleaned text
    APPLY preprocess_text to 'Email Text' column

SET X to 'Email Text' column (features)
SET y to 'Email Type' column (labels)

ENCODE y to numerical values using LabelEncoder

SPLIT X and y into train/test sets (test_size=0.2, random_state=42, stratify=y_encoded)

INITIALIZE TF-IDF Vectorizer (max_features=5000, lowercase=True, stop_words='english', ngram_range=(1,2))

TRANSFORM X_train and X_test using TF-IDF fit_transform on train and transform on test

INITIALIZE classifiers:
    SET nb_classifier to MultinomialNB (alpha=0.1)
    SET rf_classifier to RandomForestClassifier (n_estimators=100, max_depth=30, min_samples_split=5, random_state=42, class_weight='balanced')

CREATE VotingClassifier with nb and rf, using soft voting

FIT VotingClassifier on X_train_tfidf and y_train

CREATE dictionary model_components with:
    'tfidf_vectorizer': the vectorizer
    'voting_classifier': the fitted voting classifier
    'label_encoder': the label encoder

SAVE model_components to joblib file in Drive path
PRINT save confirmation message
```

Fig. 3: Pseudocode Details of Machine Learning Model Training

The pseudocode describes the workflow of training a hybrid phishing email detector. Prior to training, all email data underwent preprocessing to ensure consistency and accuracy. The text content was converted to lowercase, punctuation, stopwords, and URLs were removed. The cleaned text was then vectorized using the TF-IDF to represent weighted numerical values. For model training, a hybrid classifier combining Naïve Bayes and Random Forest was implemented. The Random Forest used 100 estimators, a maximum depth of 30, while the Naïve Bayes model applied the multinomial variant for text-based features. The TF-IDF vectorizer was limited to 5000 features with bi-gram support (n-gram = (1,2)). The two classifiers are trained and used to form a soft-voting ensemble to enhance accuracy. Finally, the vectorizer, trained classifier, and label encoder are bundled into a dictionary and saved as a joblib file for efficient serialization and later reuse. This workflow ensures a reproducible, hybrid model for phishing detection.

2.2.1 Naïve Bayes

Naive Bayes is a probabilistic classifier, which relies on the Bayes Theorem, and is commonly applied in text classification, like spam or phishing detection. It has the assumption of feature independence which simplifies the computation and facilitates the work with large datasets [17]. The algorithm stands out because it processes large datasets efficiently and operates with simplicity which makes it a preferred solution for these tasks. Naïve Bayes performs phishing email detection through a process of probability calculation based on specific features. The Naïve Bayes algorithm evaluates phishing email likelihood through an assessment of features that appear or do not appear in messages [18]. Because Naïve Bayes can handle vast amounts of data quickly and make predictions based on

probabilistic reasoning, it is very effective for phishing detection despite its simplicity. However, in situations where characteristics are correlated, its accuracy may be limited due to its dependence on the independence assumption [19].

2.2.2 Random Forest

Random Forest is an ensemble learning method that builds multiple decision trees during training and combines their predictions to improve accuracy and resilience [20]. Random Forest differs from Naïve Bayes because it does not require feature independence assumptions, and it can detect complex feature relationships. The construction of each decision tree in the forest uses random data subsets and random feature subsets to minimize overfitting and enhance generalization capability. The prediction process concludes by combining the output of all individual trees through a majority voting algorithm. Random Forest identifies phishing emails through the combined decision-making process of numerous decision trees [21]. Random Forest demonstrates exceptional performance for phishing detection because it can analyze complex non-linear feature relationships. The ensemble method in Random Forest produces a strong model with reduced vulnerability to overfitting which makes it an ideal solution for advanced phishing detection [22].

Table 2: Comparison and Combination of Naïve Bayes and Random Forest in Phishing Detection

Aspect	Naïve Bayes	Random Forest	Hybrid Approach
Concept	A probabilistic classifier based on Bayes' Theorem, assuming feature independence.	An ensemble learning method that builds multiple decision trees and takes the majority vote for classification.	Combines probabilistic reasoning (Naïve Bayes) with decision tree-based learning (Random Forest) to improve accuracy.
Feature Dependency	Assumes that all features are independent, which may not always be realistic.	Does not assume feature independence, making it more flexible for real-world data.	Leverages Naïve Bayes' efficiency and Random Forest's accuracy in handling complex relationships between features.
Speed and Performance	Fast and efficient, even with large datasets. Works well with text-based classification like email filtering.	Slower than Naïve Bayes due to multiple decision trees but provides higher accuracy.	Allows quick classification using Naïve Bayes and refined results through Random Forest.
Handling of Noise & Outliers	Sensitive to noise and outliers, which can affect classification accuracy.	Robust against noise and outliers due to the averaging effect of multiple trees.	Random Forest compensates for Naïve Bayes' weakness by handling noisy email data effectively.
Accuracy & False Positives	Works well with simple patterns but may produce higher false positives for complex phishing emails.	More accurate than Naïve Bayes but can overfit if too many trees are used.	Combining both models reduces false positives by using Naïve Bayes for quick filtering and Random Forest for deeper analysis.
Interpretability	Easy to understand and implement.	More complex but provide better decision-making insights.	Maintains Naïve Bayes' simplicity while improving classification with Random Forest.

2.3 Comparison with Deep Learning and NLP-Based Phishing Detection

Recent research has investigated deep learning and natural language processing (NLP) methods to improve the accuracy of phishing detection. Deep learning, including Long Short-Term Memory (LSTM) networks and

Convolutional Neural Networks (CNNs), can learn contextual and sequential relationships in emails, detecting phishing attacks that are not just based on keyword analysis [23]. In addition, NLP-based models, such as Bidirectional Encoder Representations of Transformers (BERT), exploit semantic knowledge to identify minor manipulations in phishing messages [24]. These models tend to be more accurate but demand more computational power and larger data sets, which is not as suitable when it comes to lightweight solutions. Conversely, the hybrid Naive Bayes-Random Forest (NB+RF) model introduced in this paper provides a good balance between interpretability, scalability and performance, which is feasible in resource-limited settings. The table presented below outlines the distinguishing features of the hybrid model and deep learning/NLP-based approaches. As shown in Table 3 below, while deep learning and NLP-based models deliver superior accuracy, they often trade-off transparency and computational efficiency. The hybrid NB+RF approach retains interpretability and requires fewer resources, making it an effective compromise for practical phishing detection systems.

Table 3: Comparative Analysis of Hybrid NB+RF vs. Deep Learning and NLP-Based Phishing Detection Models

Aspect	Hybrid (Naïve Bayes + Random Forest)	Deep Learning (LSTM, CNN)	NLP-Based (BERT, Transformer)
Concept	Combines probabilistic reasoning and decision-tree ensemble learning for robust classification.	Uses neural networks to learn complex sequential and spatial text patterns.	Utilizes transformer-based architectures for contextual language understanding.
Feature Dependency	Using TF-IDF feature extraction, assumes partial independence between features.	Learn feature representations automatically without manual feature engineering.	Extracts contextual embedding using self-attention for deep semantic understanding.
Speed and Performance	Fast training and inference, suitable for real-time desktop or local deployment.	Slower training due to deep architecture may require GPU acceleration.	High computational cost and large memory footprint, optimized for cloud environments.
Handling of Noise & Outliers	Random Forest handles noise through averaging, NB is sensitive to imbalanced data.	High tolerance to noise when trained on large datasets, may overfit small data.	Strong generalization ability but may misinterpret rare linguistic cues.
Interpretability	Clear contribution of each feature and decision tree output.	Moderate interpretability, model decisions depend on learned weights.	Low interpretability requires explainable AI (XAI) methods for justification.

3. PROPOSED SOLUTION

The proposed solution system architecture (Fig. 4) is a desktop-based phishing detection toolkit that classifies emails as phishing or legitimate using a hybrid machine learning model. It supports .eml and .txt files, analyzing headers (SPF, DKIM, DMARC), content (via Hybrid Machine Learning Model), and URLs (VirusTotal API integration). Key features include threat scoring, screenshot previews, and report generation in .docx format.

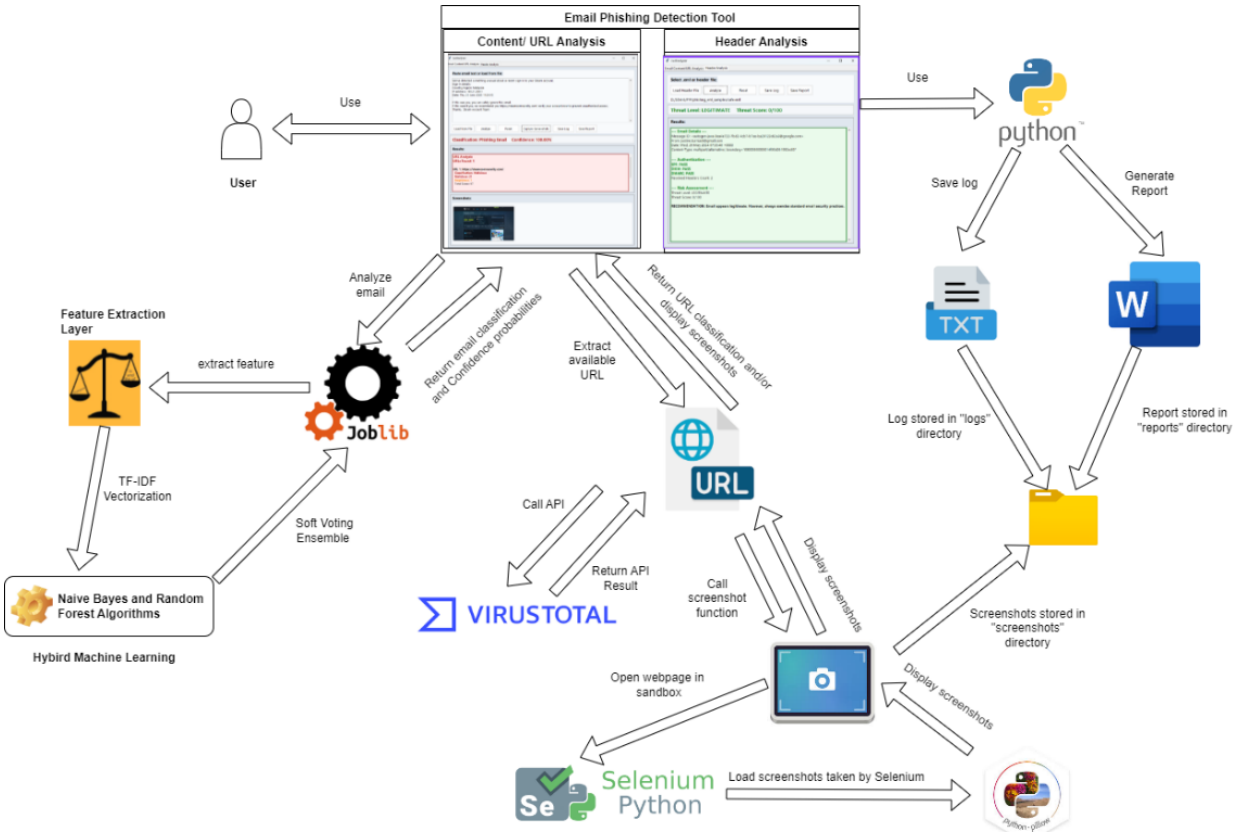

Fig. 4: System Architecture Diagram

The Email Phishing Detection Tool is a Python-based application that has both GUI and CLI interfaces and is aimed at detecting suspicious emails by utilizing three modules, content analysis, URL analysis, and header analysis. In the case of content analysis, the tool is based on a hybrid model that combines Naive Bayes and Random Forest through soft voting, where email text is converted into TF-IDF vectors. Emails containing URLs are scanned with VirusTotal API and questionable pages are previewed with the Selenium WebDriver to be sandboxed. The Python scripts are used to analyze the header, check the SPF, DKIM and DMARC output, determine a threat score and display signs of spoofing.

The tool enables the user to store the results in the form of logs (.txt) or reports (.docx), which are automatically sorted into separate folders. The tool's uniqueness lies in how it integrates machine learning for textual email classification, API-based URL intelligence, and logic-based header analysis. The modular, layered analysis allows a wider scope of phishing detection and is applicable at an educational institution, organization, or an individual who wants to have protection against phishing emails. The use case diagram (Fig. 5) outlines actors (User) and cases like "Analyze Email Header," "Analyze Content".

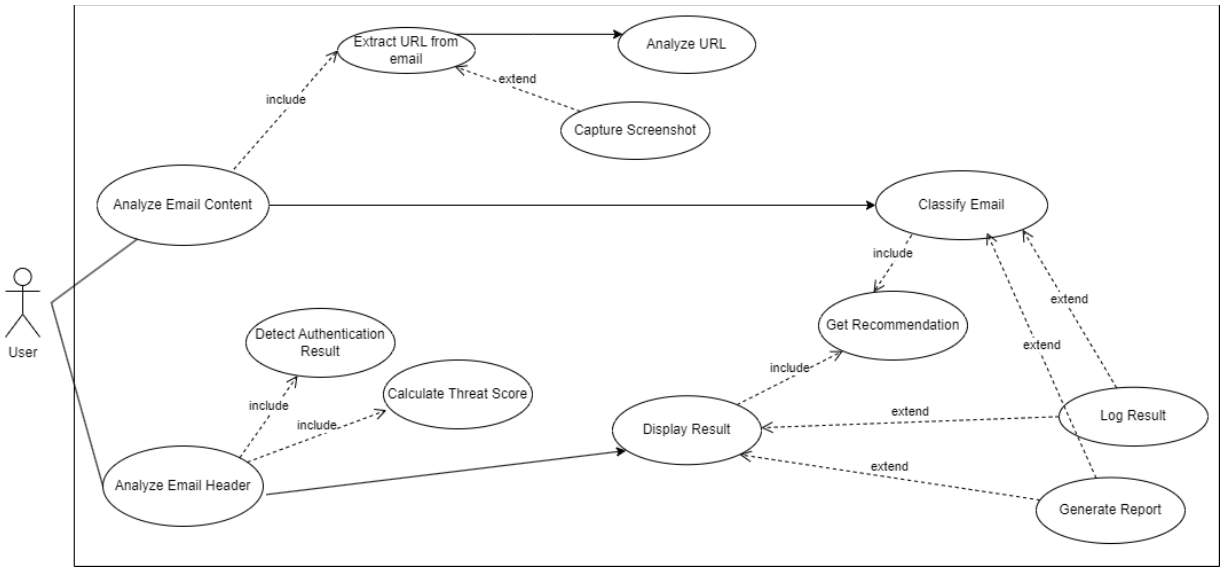

Fig. 5: Use Case Diagram

User Starts the process by interacting with the system. Then Analyze Email Content extracts URLs from the email and extends to further analysis. Screenshots can be captured for the extracted URL to take a safe preview of the URL destination. Lastly it Categorizes the email as malicious or legitimate, including recommendations. Next is to analyze email header which Detect Authentication Result by Identifying authentication issue (SPF, DKIM, DMARC) and Calculate Threat Score in which Computes a threat level based on analysis and displaying the result with recommendations Finally, users can Log Result to record the analysis details and Generate Report to create a detailed report for review. The activity diagram (Fig. 6 and 7) details the workflow of the system.

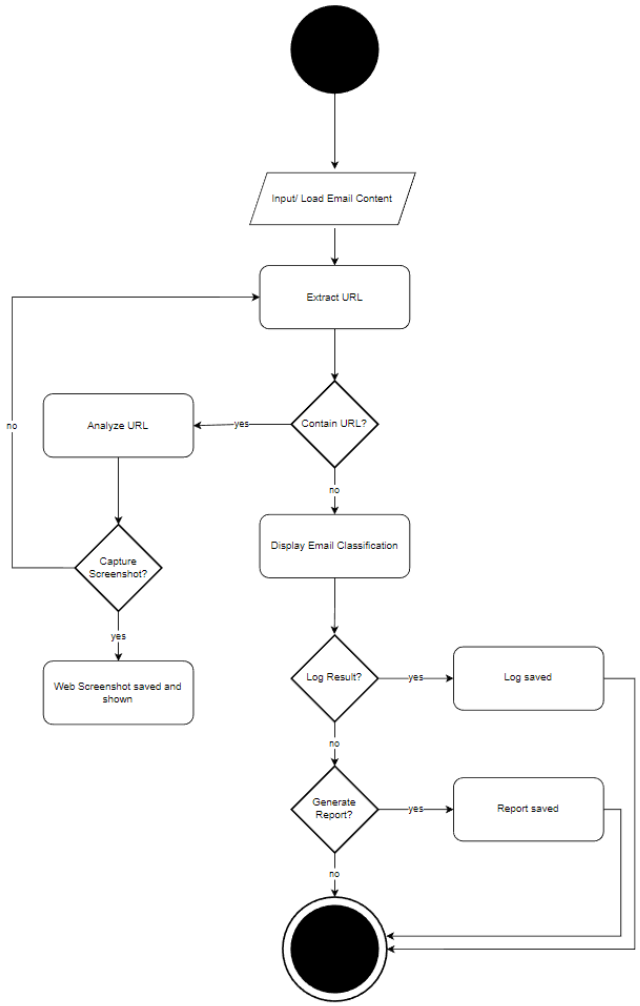

Fig. 6: Activity Diagram

This diagram explains the activity diagram of the operation of the phishing detection tool, where it starts with loading of an email content and ends with generation of logs and reports. Firstly, it involves loading or entering the email contents to the system by the user. After the loading of the content, the tool subsequently tries to harvest or extract any URL present in the email. When URLs are identified, then the tool will go further to process and analyse the URL and specify whether it may be legitimate or suspicious or malicious. Once the analysis has been done, the system will then prompt user to choose whether take a screenshot of the destination site. If yes, then the web screenshot is taken, displayed on the screen, and saved to the screenshot directory.

Once the URL processing is done, or in case there is no URL, the email is then classified as phishing or safe by the system and displaying the result. After that, the users will be notified whether they would like to record the result. In case of yes, the outcome is stored in a log file. Then the tool also examines whether the user is interested to generate a report analysis with recommendation. In case of yes, a report is created and stored. At this stage the process is completed, whereby it goes back to the initial stage to permit the analysis of another email in case desired This structured flow ensures every critical step involved, namely URL extract, email analysis, classification, screenshots, report, and logging, are done in a logical manner and depending on the preference of the user.

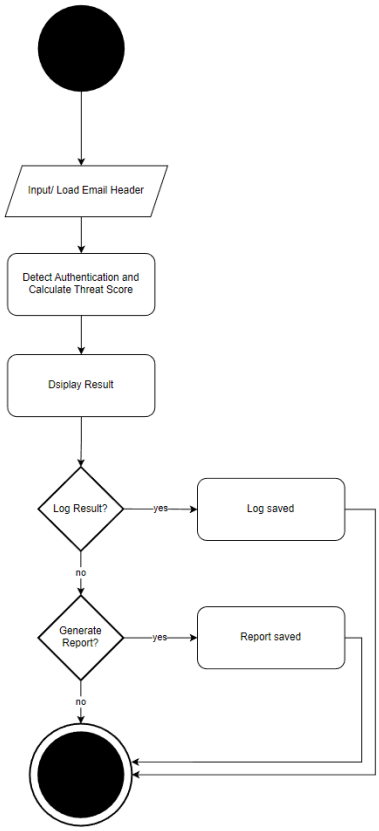

Fig. 7: Activity Diagram

The given activity diagram shows the workflow on analysing an email header of the phishing detection system. The process begins when the user inputs or loads the email header into the system. When the header is loaded, the system runs authentication checks along with procedures as SPF, DKIM, and DMARC, and at the same time calculates a threat score based on the findings obtained. Once the analysis is done, the result is reflected on the system, and the user gets an idea whether the email is legit or suspicious or malicious. At this stage, the system or program will inquire on whether the user desires to log the outcome. When the answer is yes, the outcome is stored in a log file to be referred later or seen by the security team. In case one does not want to log the result, or when the log has been created, the system now asks whether the user wants to create a report or not. When the answer is yes, the system prepares a report with the results and recommendations and saves them. Regardless of the decision to log or generate a report, the flow then ends or can be restarted for analysing another email. Such patterns of work allow users to analyse the email header and notes on possible suspicious activity efficiently and then further analyse and report.

4. RESULTS AND DISCUSSION

4.1 Machine Learning Model Testing

The dataset used for training and testing was obtained from Kaggle repository [25] titled “Phishing Emails Dataset” (From the authors Subhadeep Chakraborty). It contains a combination of phishing and legitimate emails collected from diverse real-world sources. A temporal distribution analysis indicates that phishing samples was last updated in 2023, reflecting evolving patterns in phishing attacks during that period.

Security and Ethical Considerations:
The data is publicly available and does not contain any personal identifiable information. The data set was only experimented academically and it conforms to the ethical standards of data-handling. Email samples were anonymized and no user-specific or organizational metadata was stored. Using a public dataset prevents privacy violations and

ensures reproducibility for future studies. Operationally, phishing detection tools have to strike a balance between false positives and false negatives. False positives are harmful because they disrupt the proper communication and lead to alert fatigue, but false negatives enable malicious emails to pass through the defensive measures. Naive Bayes and Random Forest hybrid model used in this paper was optimized to reduce the two outcomes by balancing the precision and recall, which offers a reliable but ethically accountable method of detection to be used in the real world.

Evaluation Metrics (Accuracy, Precision, Recall, F1-Score)

The results of the evaluation compare the work of three machine learning models, Naive Bayes, Random Forest and a Hybrid Model on the dataset of 3,727 emails (since this number comes from the test set only, with Training Set = 80% ≈ 14,920. Testing Set = 20% ≈ 3,730. Total 100% = 18,650), 1,462 phishing emails and 2,265 safe emails.

```
Naive Bayes Accuracy: 95.33%
Naive Bayes Classification Report:
              precision    recall  f1-score   support

Phishing Email       0.96      0.92      0.94      1462
    Safe Email       0.95      0.98      0.96      2265

      accuracy                           0.95      3727
     macro avg       0.96      0.95      0.95      3727
  weighted avg       0.95      0.95      0.95      3727
```

Fig. 8: Naïve Bayes Evaluation

Naive Bayes Classifier obtained an overall accuracy of 95.33% with high precision (0.96) for phishing emails and high recall (0.98) for safe emails. It had a macro average F1-score of 0.95 meaning that it was performing at a balanced level on both classes, and the weighted average F1-score was 0.95 as well, which shows that the model was also consistent with the class imbalance.

```
Random Forest Accuracy: 93.80%
Random Forest Classification Report:
              precision    recall  f1-score   support

Phishing Email       0.87      0.99      0.93      1462
    Safe Email       0.99      0.91      0.95      2265

      accuracy                           0.94      3727
     macro avg       0.93      0.95      0.94      3727
  weighted avg       0.94      0.94      0.94      3727
```

Fig. 9: Random Forest Evaluation

In contrast, the Random Forest model registered a marginally low accuracy of 93.80%. It scored very highly in the phishing emails (recall of 0.99), meaning it caught almost anything that pretended to be phishing, whereas its precision in phishing was a little lower (0.87), or it had more false positives. The macro and weighted F1-scores were 0.94, showing overall strong, though slightly less balanced, performance compared to Naive Bayes.

```
Hybrid Model Accuracy: 96.81%
Hybrid Model Classification Report:
              precision    recall  f1-score   support

Phishing Email       0.95      0.97      0.96      1462
    Safe Email       0.98      0.97      0.97      2265

      accuracy                           0.97      3727
     macro avg       0.96      0.97      0.97      3727
  weighted avg       0.97      0.97      0.97      3727
```

Fig. 10: Hybrid Model Evaluation

Finally, the Hybrid Model, that probably incorporates best ideas of both base models, is even more accurate (96.81%). It produced a good trade-off between high precision (0.95 on phishing, 0.98 on safe) and high recall (0.97 on both classes), to give macro and weighted F1-scores of 0.97. This implies that the hybrid technique was not only biased towards a low false positive and false negative rate, but it was also effective in addressing the issue of class imbalance. Overall, the Hybrid Model demonstrated the highest overall and per-class performance, which indicates that it is the most reliable one in terms of finding phishing emails in the given case.

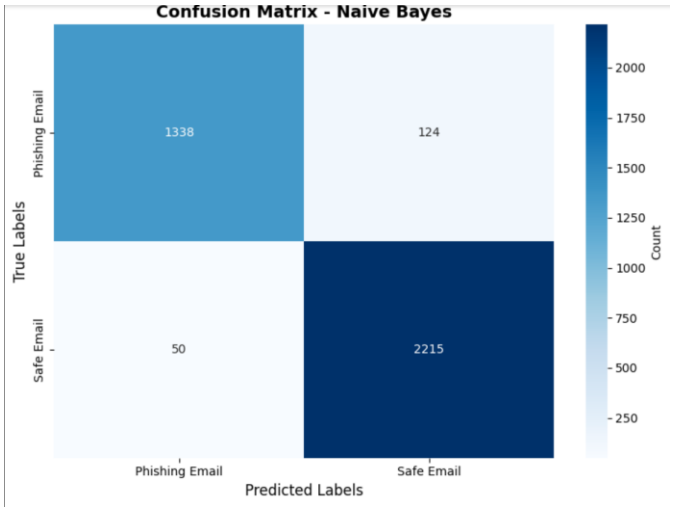

Fig. 11: Naïve Bayes Confusion Matrix

The Naive Bayes classifier classified 1,338 phishing emails and 2,215 safe emails correctly (true positives and true negatives respectively). But it failed to identify 124 phishing emails as unsafe (false negatives) and 50 safe emails as phishing (false positives). The fact that it does a decent job, however, the comparatively high false negative rate shows that it will most likely fail to detect all phishing threats, which could be quite dangerous in real-life scenarios.

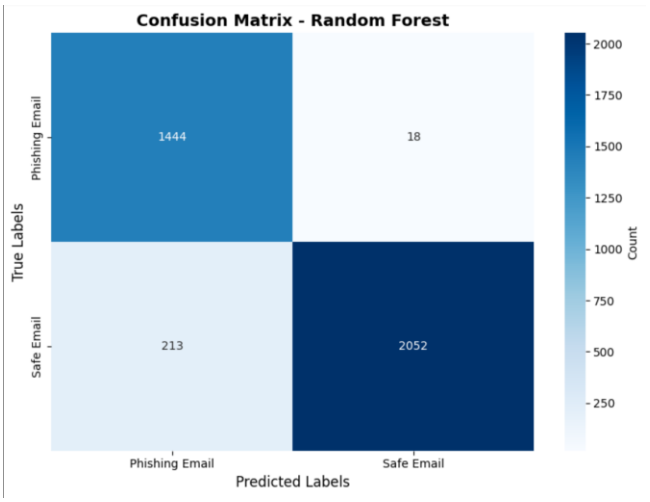

Fig. 12: Random Forest Confusion Matrix

Random Forest model had shown a different trend. It was able to detect 1,444 phishing messages and 2,052 legitimate messages. It has however misclassified 18 phishing emails as safe (very few false negatives) and 213 safe emails as phishing (more false positives). A more aggressive model will more likely flag a phishing email as phishing but will also flag more safe emails as phishing.

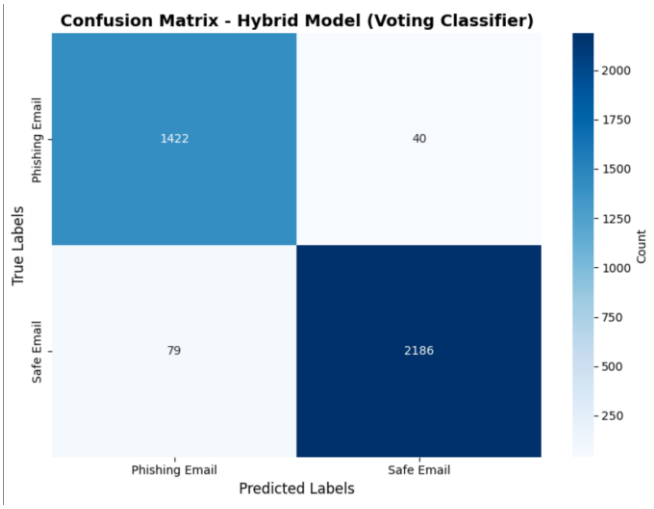

Fig. 13: Hybrid Model Confusion Matrix

A good balance was realized in the Hybrid model. It successfully detected 1,422 phishing emails and 2,186 safe emails and the only 40 phishing emails were incorrectly detected as safe and 79 safe emails were incorrectly detected as phishing. This means that the hybrid model has a low false positive rate and a low false negative rate and hence it is more reliable.

Comparison and conclusion:

- Naive Bayes has less false positives and a greater number of false negatives.
- Random Forest greatly decreases false negatives and does the same in false positives.
- Hybrid Model has improved trade-off of low false positives and false negatives, which means a more balanced and reliable classification.

Thus, the Hybrid Model proves the best of the three models since it shows the best balance in terms of effectively detecting both the phishing and safe emails with minimal misclassification. This qualifies it especially to be used in real life email security systems where precision and recall are crucial.

4.2 Interface Result

The GUI provides an accessible interface for email analysis, enabling users to classify email content, check headers, and assess URLs. Results are displayed with corresponding threat levels, and users may generate logs or reports for further action.

To avoid redundancy, only representative screenshots are included:

- **Email Content Classification Result** = shows phishing vs legitimate classification with confidence levels.
- **Email Header Analysis Result** = displays threat score, authentication checks (SPF, DKIM, DMARC) and risk assessment.

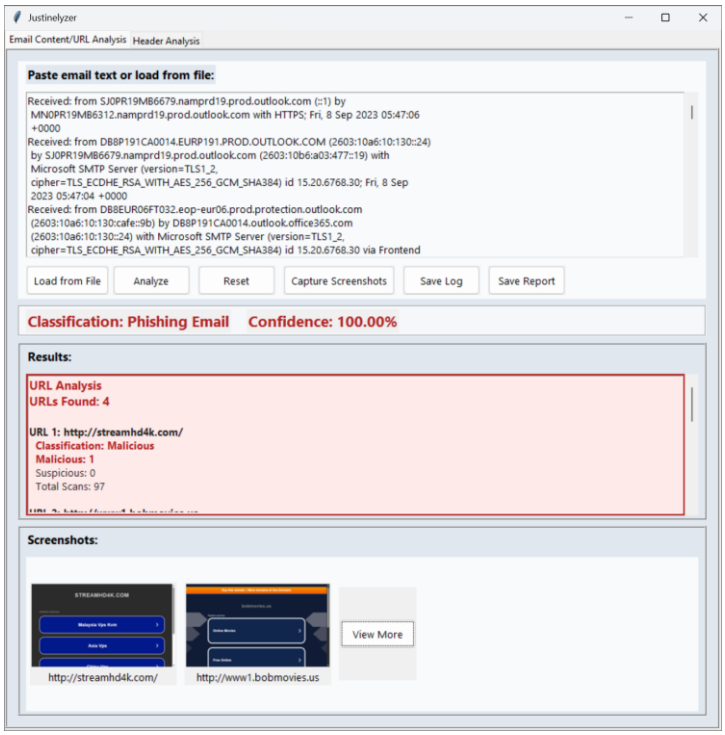

Fig. 14: Email Content Classification with URL

This above GUI shows the analysis phase. When the user has provided an input and presses the Analyze button, then the system initiates phishing detection using a hybrid machine learning model and the system will try to extract any URL. If found, then it will utilize a threat intelligence to analyze the URL. Based on the analysis, the email is labeled as “Malicious,” “Suspicious,” or “Legitimate” with a confidence level of 100%. Furthermore, when the Capture Screenshots button is clicked, the system will then save and load the web preview in the Screenshots field, and if there are more than two screenshots, they are organized under a View More button for better navigation.

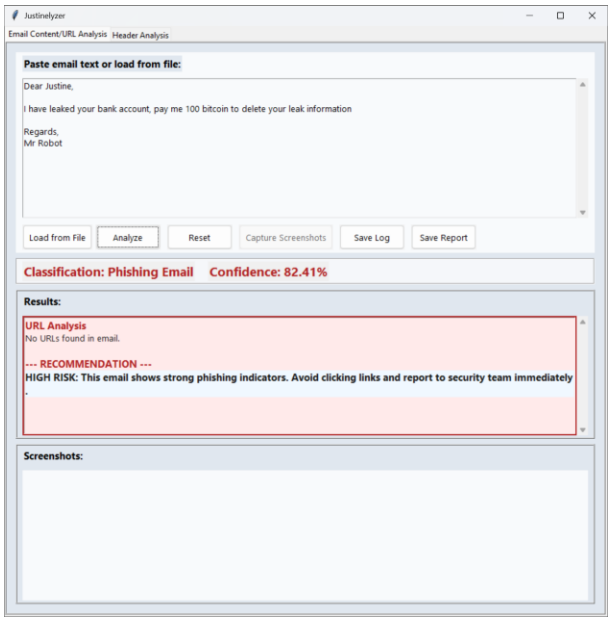

Fig. 15: Email Content Classification without URL

If the user tries to analyze email content that contains no URLs, the system will instead depend on its hybrid machine learning models to classify the content, in addition, it will display a background color following the classification, such as either red color or green color for “Phishing Email” and “Legitimate Email” respectively. Alongside the classification, it will also show the confidence level or probability score associated with the result, and recommendation to boot.

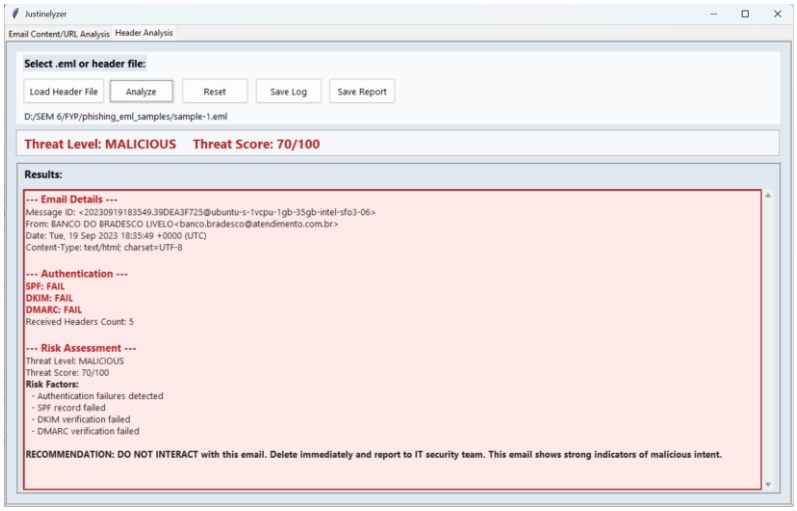

Fig. 16: Email Headers Result Malicious

The above GUI shows the email headers results, the loaded email was recognized by system, and its Threat Level had been set to MALICIOUS with Threat Score 70/100 (it will classify as malicious if threat score >= 70). This category is determined by a number of authentication failures such as SPF, DKIM and DMARC that play an important role in stamping out the authenticity of the sender. Metadata such as sender address, date, and message ID is represented in the section named the “Email Details” but it is clarified in the “Risk Assessment” section why the email is considered malicious with authentication failures being one of the causes. The urgency and the seriousness of the threat are shown by the background in red color. It will also give advice based on the threat level.

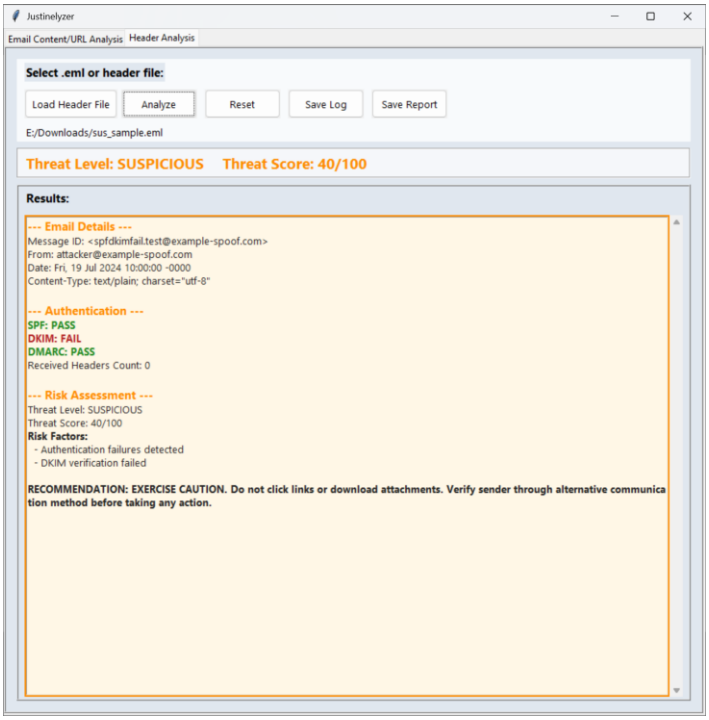

Fig. 17: Email Headers Result Suspicious

The second result shows that the email is marked as suspicious with the Threat Score of 40/100 (it will classify as suspicious if threat score >= 30). Some authentication strategies succeeded (SPF and DMARC), whereas the DKIM test was not successful, showing incompleteness but a non-fraudulent nature. This ambiguous outcome is present in the risk assessment which identifies it as possibly dangerous but not conclusive enough to be worth the label of being outright malicious. The background in yellow/orange color is implemented to make it visually to warn the user and to be careful. The tool contains the ability to identify the grey-zone threats when one part of the verification may pass, but still there are concerns that must be avoided since the system recommends the user look to other forms of verifying the sender and not allowing access to the links or downloading the attachment.

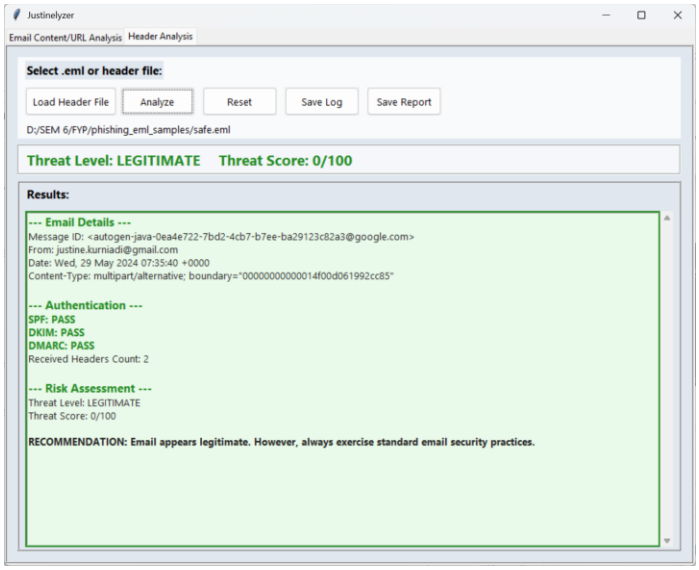

Fig. 18: Email Headers Result Legitimate

Lastly, the email shows as legitimate with a Threat Score of 0/100 (it will classify as suspicious if threat score < 30). All three fundamental validation tests, SPF, DKIM, and DMARC, have passed, which is very solid evidence of authenticity. The box shows the presence of a legitimate source and correct format in the e-mail on the Email Details. The background, which is green, gives a sign of safety and there is nothing that signifies there is any malicious action involved. Nevertheless, the tool suggests to the user to observe the security standards of emailing, which is a security-conscious and conservative approach to design. This demonstrates the tool's balance between usability and cybersecurity by not only identifying threats but reinforcing safe behavior.

4.3 Discussion

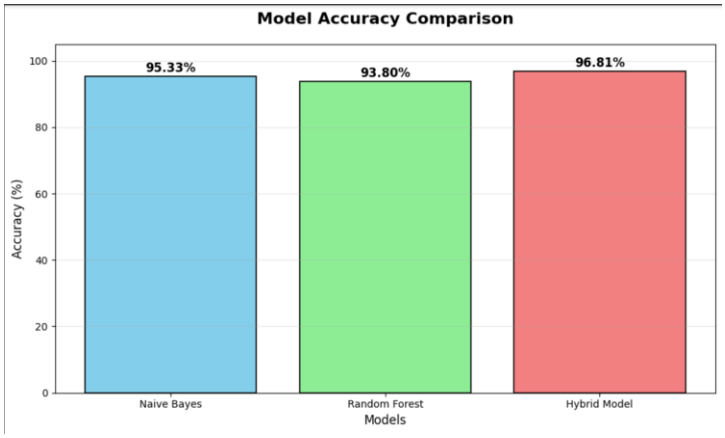

Fig. 19: Machine Learning Model Accuracy Comparison

The comparative study of three models indicates different strengths and trade-offs of the models in terms of their performance on the phishing detection task. The accuracy of the Naive Bayes model was 95.33%, which means that the model was correct in more than 95% of the scenarios on whether an email was phishing or safe. This model is characterized by its ease and efficiency especially in the text-based classification problem like spam or phishing. A slightly lower accuracy of 93.80% was obtained with the Random Forest model which is an ensemble of the decision trees. Although it is strong and powerful overall, it is possible that it was a bit slowed down by the type of dataset or the significance of the text feature associations in this case. In contrast, the Hybrid Model recorded the highest percentage of accuracy of 96.81%. Since this model brings together the advantages of two classifiers (Naive Bayes and Random Forest) and use their complementary nature to give improved generalization and enhanced classification results. In practical terms, the hybrid model's superior balance makes it ideal for organizational environments, where false positives can lead to alert fatigue and false negatives to data breaches. Limitations include dependency on high-quality training data and potential vulnerability to evolving phishing techniques, such as AI-generated content. Overall, these results validate the hybrid approach's efficacy, emphasizing its role in enhancing email security while highlighting the need for adaptive retraining in dynamic threat landscapes.

5. CONCLUSION

This paper has managed to build and test a hybrid phishing detection tool that combines Naive Bayes and Random Forest algorithms with an accuracy of 96.81 percent in a dataset of 18,650 emails. The thoroughness of analysis of the headers, contents and URLs, combined with GUI/CLI interfaces, previews of screenshots and report creation, offers a comprehensive, user-friendly solution that organization can use to reduce the risk of phishing. The proposed system enhances organizational email security, which is in line with SDG 9 that advocates innovative resilient digital infrastructure. It also contributes a novel, integrated framework that unites three analytical layers: email header authentication, content classification using a hybrid Naïve Bayes and Random Forest model, and URL intelligence through API integration within a single, user-friendly desktop tool. The tool eventually improves security of sensitive organizational information and minimizes the risks associated with phishing. Future directions may include expanding this framework to be directly connected to enterprise email servers (Gmail) via RESTful APIs and real-time email phishing detection of web or mobile applications. The use of deep learning models such as LSTM or BERT might also

enhance contextual knowledge of phishing messages. Also, the ability to retrain and analyze behavior might be introduced to be able to respond to zero-day and AI-generated phishing attacks, which would guarantee the resilience of the model in the long term and preserve the accuracy of detection in the changing threat environment.

ACKNOWLEDGEMENT
The authors declare that there is no financial support or conflict of interest related to this research.

REFERENCES

[1] M. Kosinski, "What is Phishing?," IBM, May 17, 2024. [Online]. Available: https://www.ibm.com/think/topics/phishing. [Accessed: Dec. 20, 2025].

[2] H. Ozsahan and D. Worthington, "50+ Phishing Attack Statistics for 2024," Jumpcloud, Mar. 12, 2024. [Online]. Available: https://jumpcloud.com/blog/phishing-attack-statistics. [Accessed: Dec. 04, 2025].

[3] G. Varshney, R. Kumawat, U. Tupakula, C. Gupta, and V. Varadharajan, "Anti-phishing: A comprehensive perspective," Expert Syst. Appl., vol. 238, pp. 1–20, Mar. 2024.

[4] CCHosting, Inc., "Understanding an Email Header," Chemicloud, May 12, 2021. [Online]. Available: https://chemicloud.com/kb/article/understanding-an-email-header/. [Accessed: Dec. 19, 2025].

[5] Cloudflare, Inc., "What are DMARC, DKIM, and SPF?," Cloudflare, 2025. [Online]. Available: https://www.cloudflare.com/learning/email-security/dmarc-dkim-spf/. [Accessed: Dec. 13, 2025].

[6] K. Shen, C. Wang, M. Guo, X. Zheng, C. Lu, B. Liu, Y. Zhao, S. Hao, H. Duan, Q. A. Xin, Q. Pan, and M. Yang, "Weak Links in Authentication Chains: A Large-scale Analysis of Email Sender Spoofing Attacks," in Proc. 30th USENIX Security Symp., Aug. 2021, pp. 1–18. [Online]. Available: https://www.usenix.org/system/files/sec21-shen-kaiwen.pdf

[7] Groundhogg, "What Are Email Headers," Groundhogg, May 30, 2023. [Online]. Available: https://help.groundhogg.io/article/738-what-are-email-headers. [Accessed: Oct. 18, 2025].

[8] Alibaba Cloud, "How to Analyze Message Header?," Aug. 24, 2023. [Online]. Available: https://www.alibabacloud.com/help/en/alibaba-mail/latest/how-to-analyze-message-header-information. [Accessed: Dec. 20, 2025].

[9] S. Firake, P. Soni, and D. B. B. Meshram, "Phishing E-mail Analysis," Int. J. Comput. Sci. Emerg. Technol., vol. 2, no. 1, pp. 092–102, Feb. 2011.

[10] C. Beaman and H. Isah, "Anomaly Detection in Emails using Machine Learning and Header Information," arXiv preprint arXiv:2203.10408, Mar. 2022. [Online]. Available: https://arxiv.org/abs/2203.10408

[11] W. Packard, "Understanding Email Headers," ClickDimensions, Feb. 5, 2025. [Online]. Available: https://support.clickdimensions.com/hc/en-us/articles/360031909032-Understanding-Email-Headers. [Accessed: Nov. 13, 2025].

[12] A. S., "Email Header: What Is It + Examples," Hostinger, Dec. 2, 2024. [Online]. Available: https://www.hostinger.my/tutorials/email-headers/. [Accessed: Nov. 03, 2025].

[13] B. B. Meshram, V. Mendhe and M. K. S. , "Tracing the Invisible Threads: A Deep Dive into Email Security & Forensics," *International Journal of Enhanced Research in Science, Technology & Engineering,* vol. 13, no. 1, pp. 24-36, January 2024.

[14] S. Nightingale, "Email authentication mechanisms: DMARC, SPF, and DKIM," *NIST Technical Note* 1945, National Institute of Standards and Technology, Feb. 2017.

[15] K. Shen, C. Wang, M. Guo, X. Zheng, C. Lu, B. Liu, Y. Zhao, S. Hao, H. Duan, Q. Pan, and M. Yang, "Weak Links in Authentication Chains: A Large-scale Analysis of Email Sender Spoofing Attacks," arXiv preprint arXiv:2011.08420, Nov. 2020. [Online]. Available: https://arxiv.org/abs/2011.08420

[16] E. Liu, G. Akiwate, M. Jonker, A. Mirian, G. Ho, G. M. Voelker, and S. Savage, "Forward Pass: On the Security Implications of Email Forwarding Mechanism and Policy," arXiv preprint arXiv:2302.07287, Apr. 2023. [Online]. Available: https://arxiv.org/abs/2302.07287

[17] E. Kavlakoglu, "What are Naïve Bayes classifiers?," IBM, 2024. [Online]. Available: https://www.ibm.com/think/topics/naive-bayes.

[18] A. McCallum and K. Nigam, “A comparison of event models for Naive Bayes text classification,” *AAAI Technical Report* WS-98-05, AAAI Press, 1998.

[19] Y. S. Murti and P. Naveen, “Machine learning algorithms for phishing email detection,” *Journal of Logistics, Informatics and Service Science*, vol. 10, no. 2, pp. 249–261, Feb. 2023.

[20] J. Kolla, S. Praneeth, M. S. Baig and K. G. Reddy, "A comparison study of machine learning techniques for phishing detection," *Journal of Business and Information Systems ,* vol. 4, no. 1, June 2022.

[21] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010950718922.

[22] A. A. Akinyelu and A. O. Adewumi, "Classification of Phishing Email Using Random Forest Machine Learning Technique," J. Appl. Math., vol. 2014, Article ID 425307, 9 pages, Apr. 2014, doi: 10.1155/2014/425307.

[23] Z. Alshingiti, R. Alaqel, J. Al-Muhtadi, Q. E. U. Haq, K. Saleem, and M. H. Faheem, “A deep learning-based phishing detection system using CNN, LSTM, and LSTM-CNN,” *Electronics*, vol. 12, no. 1, Art. no. 232, Jan. 2023, doi: 10.3390/electronics12010232.

[24] K. Iyer, "Natural language processing for phishing detection: Leveraging AI to spot deceptive content in real time," Int. J. Curr. Sci., vol. 14, no. 4, pp. 265–272, Oct. 2024, doi: 10.56975/ijcsp.v14i4.302418.

[25] S. Chakraborty, "Phishing Email Detection," Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/subhajournal/phishingemails/data [Accessed: Dec. 18, 2025].